



Running Containers on Amazon EKS

모듈 9: Amazon EKS에서 보안 관리



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.



| 강사용 노트

| 이 모듈을 완료하는 데 약 **75**분 정도 걸립니다.

| 수강생용 노트

이 모듈은 **Amazon EKS** 클러스터에서 보안 관리입니다.



Running Containers on Amazon EKS

모듈 9 개요

- 클라우드 보안 기초
- 인증 및 권한 부여
- IAM 및 RBAC 관리
- RBAC Service Account를 사용한 Pod 권한 관리



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.



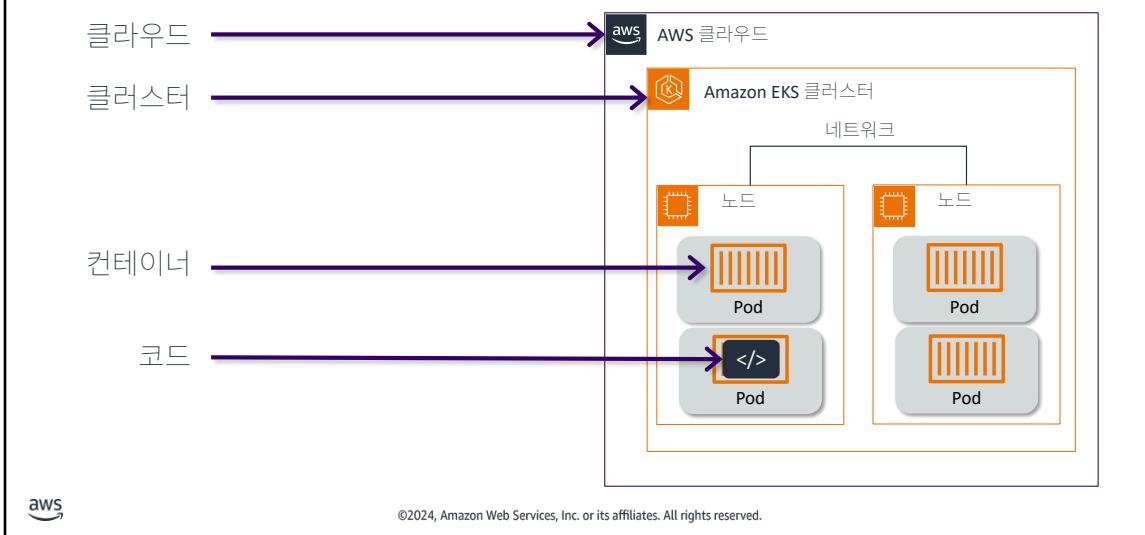
Running Containers on Amazon EKS

클라우드 보안 기초



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

클라우드 네이티브 보안의 ‘4C’



~Alt text

~2개의 노드가 있고 각각 2개의 Pod를 실행하는 Amazon EKS 클러스터입니다.

| 강사용 노트

| 이 모듈에서는 ‘4C’의 모든 측면을 다루지는 않고 **Amazon EKS**와 관련된 부분에 중점을 둘 것입니다.

| 수강생용 노트

클라우드 네이티브 보안의 **4C**는 클라우드, 클러스터, 컨테이너, 코드입니다. 이는 심층 방어를 제공하는 클라우드 보안을 위한 계층형 모델로, 각 계층은 다음 가장 바깥쪽 계층을 기반으로 구축됩니다. 코드 계층은 강력한 기반(클라우드, 클러스터, 컨테이너) 보안 계층의 혜택을 받으며 각 수준에서의 경계가 필수적입니다. 예를 들어, 코드 수준에서 보안을 해결하여 기반 계층의 불량한 보안 표준을 보완할 수는 없습니다.

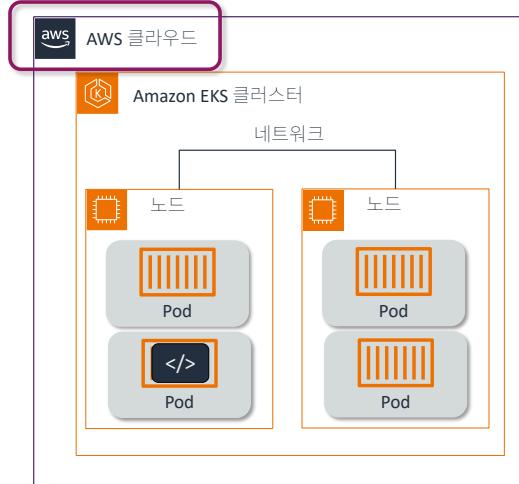
클라우드 계층 보안

고려 사항

- 클라우드 제공업체 보안
- 인프라 보안

솔루션

- AWS에서 많은 측면을 관리
- 공동 책임 모델을 준수



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Running Containers on Amazon EKS

AWS 공동 책임 모델에 대한 이해



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

보안은 공동 책임



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

보안과 규정 준수는 **AWS**와 고객이 함께 분담할 공동 책임에 속합니다. **AWS**는 이른바 클라우드 자체의 보안을 책임집니다. 고객은 클라우드 내에 배치된 모든 요소의 보안을 책임집니다.

AWS 공동 책임 모델



~ALT text

~**고객**: AWS 공동 책임 모델의 고객 책임이 강조 표시된 다이어그램입니다.

~**AWS**: AWS 공동 책임 모델의 AWS 책임이 강조 표시된 다이어그램입니다.

| 수강생용 노트

AWS는 호스트 운영 체제, 가상화 계층부터 서비스 운영 시설의 물리적인 보안에 이르는 구성 요소를 운영하고, 관리하고, 제어합니다. 즉 AWS는 AWS 리전, 가용 영역, 엣지 로케이션 등의 AWS 클라우드에서 제공되는 모든 서비스를 실행하는 글로벌 인프라를 보호할 책임이 있습니다. AWS 글로벌 인프라에는 시설, 네트워크, 하드웨어와 이러한 리소스의 프로비저닝 및 사용을 지원하는 운영 소프트웨어(호스트 OS, 가상화 소프트웨어 등)가 포함됩니다.

이 인프라의 보호는 AWS의 최우선 과제입니다. AWS 데이터 센터나 사무소를 직접 방문하여 어떻게 보호되고 있는지 살펴볼 수는 없습니다. 대신 AWS는 서드 파티 감사자를 통해 다양한 컴퓨터 보안 표준 및 규정 준수 여부를 확인한 여러 보고서를 제공합니다.

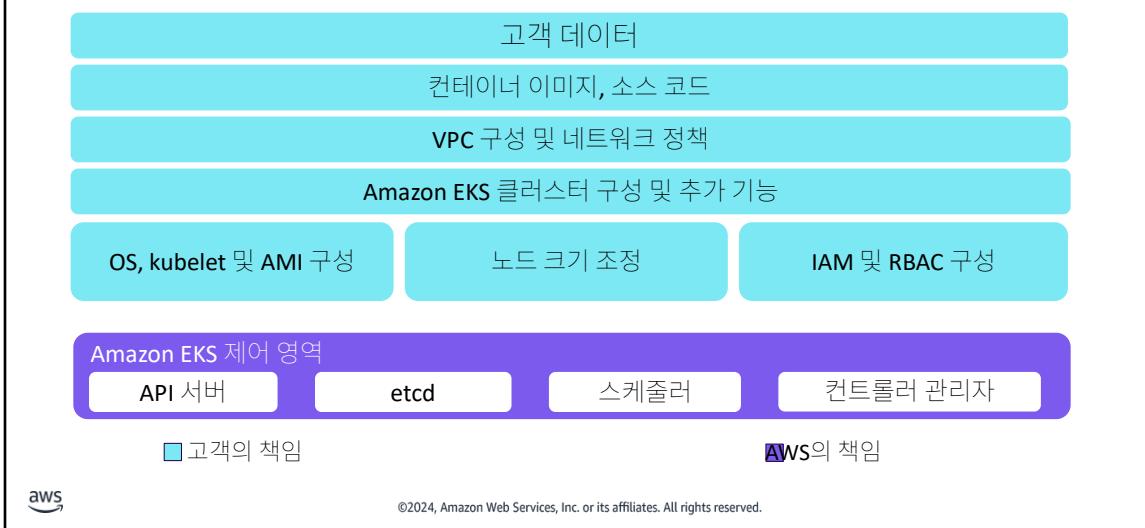
AWS 서비스를 사용하는 고객은 고객 콘텐츠를 완벽하게 제어할 수 있으며, 다음과 같은 중요한 콘텐츠 보안 요구 사항을 관리해야 합니다.

- 고객이 AWS에 저장하기로 선택한 콘텐츠
- 콘텐츠와 함께 사용하는 AWS 서비스
- 콘텐츠가 저장되는 국가
- 콘텐츠의 형식과 구조 및 마스킹, 익명, 암호화 여부

- 콘텐츠에 액세스할 수 있는 사용자 및 이러한 액세스 권한을 부여, 관리, 취소하는 방법

고객은 자체 데이터, 플랫폼, 애플리케이션, **Identity and Access Management**, 운영 체제 보호를 위해 구현할 보안 시스템을 결정할 수 있습니다. 이는 고객이 사용하는 AWS 서비스에 따라 공동 책임 모델이 변경된다는 것을 의미합니다.

Amazon EKS 기본 공동 책임 모델



~ALT text

~고객: Amazon EKS에 대한 AWS 공동 책임 모델의 고객 책임이 강조 표시된 다이어그램입니다.

~AWS: Amazon EKS에 대한 AWS 공동 책임 모델의 AWS 책임이 강조 표시된 다이어그램입니다.

|수강생용 노트

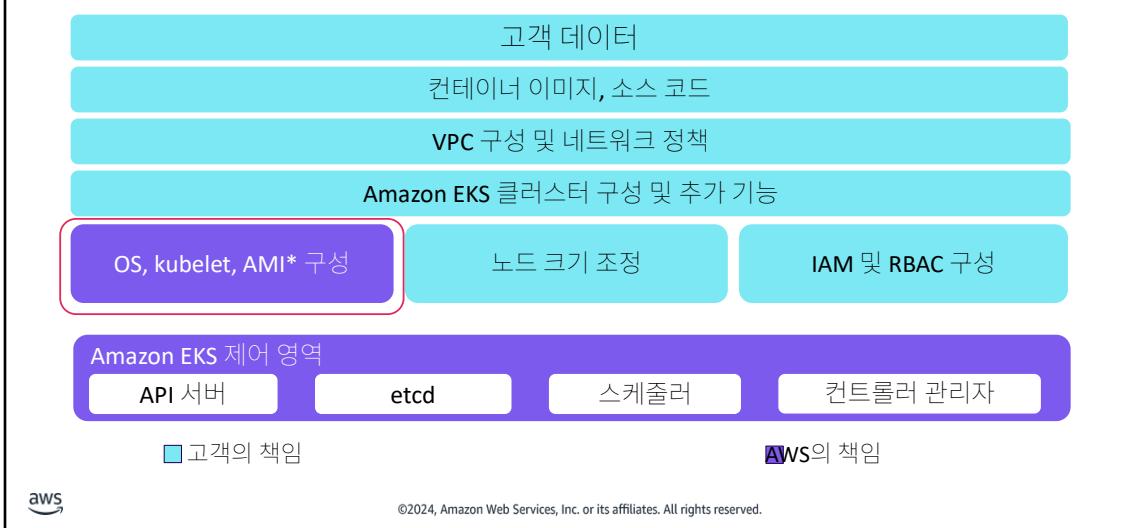
보안 및 규정 준수는 Amazon EKS를 사용할 때 공동 책임 대상으로 간주됩니다. 일반적으로 AWS는 클라우드 ‘자체’의 보안을 책임지고 사용자는 클라우드 ‘내부’의 보안을 책임집니다.

AWS는 Amazon EKS와 함께 Amazon EKS 관리형 Kubernetes 제어 영역을 관리합니다. 여기에는 Kubernetes API 서버 노드, etcd 데이터베이스 및 AWS가 안전하고 신뢰할 수 있는 서비스를 제공하는데 필요한 기타 인프라가 포함됩니다.

Amazon EKS의 소비자는 다음을 포함한 데이터 영역에 대한 책임이 있습니다.

- AWS Identity and Access Management(IAM)
- Pod 보안
- 런타임 보안
- 네트워크 보안
- 직접 개발한 컨테이너 이미지의 코드 보안

관리형 노드 그룹을 이용한 공동 책임 모델



~ALT text

~고객: 관리형 노드에 대한 공동 책임 모델의 고객 책임이 강조 표시된
다이어그램입니다.

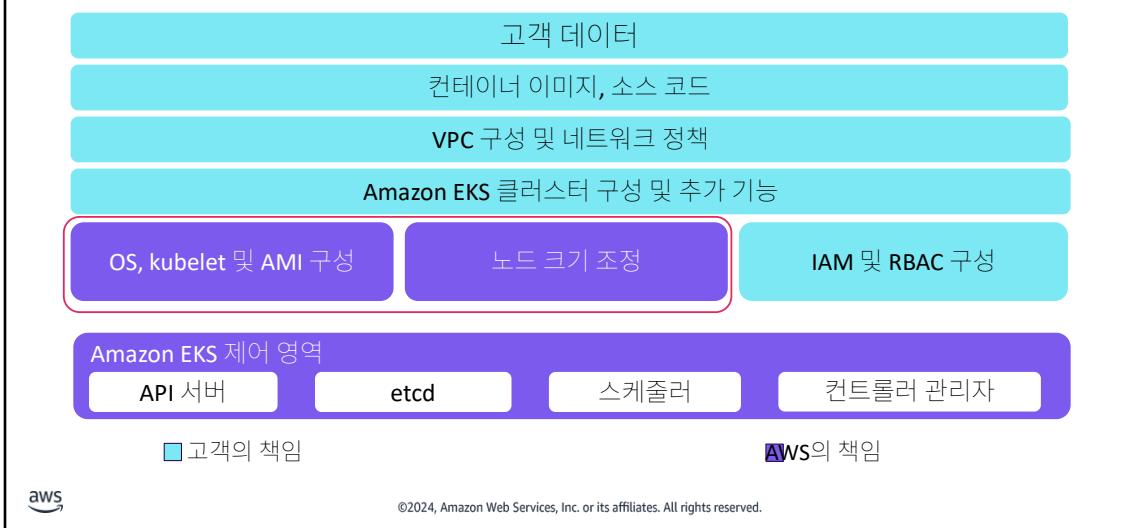
~AWS: 관리형 노드에 대한 공동 책임 모델의 AWS 책임이 강조 표시된
다이어그램입니다.

|수강생용 노트

Amazon EKS는 관리형 노드 그룹의 Common Vulnerabilities and Exposures(CVE) 및 보안 패치에 공동 책임 모델을 적용합니다. 사용자가 셀프 매니지드 작업자에서 관리형 노드 그룹 및 Fargate로 전환하는 경우 AWS는 인프라 보안과 관련해 추가적인 책임을 집니다. AWS는 사용자가 관리형 노드 그룹을 사용할 때 기본 인스턴스의 보안 및 유지 관리를 책임집니다.

*AWS는 버그나 문제가 보고된 경우 Amazon EKS에 최적화된 AMI의 패치 버전을 구축할 책임이 있습니다. 사용자는 이러한 AMI 패치 버전을 관리 노드 그룹에 배포할 책임이 있습니다. 관리형 노드에서 사용자 정의 AMI를 실행하는 경우 사용자는 AMI 패치 버전을 구축하고 배포할 책임이 있습니다.

Fargate를 이용한 공동 책임 모델



~ALT text

~고객: Fargate에 대한 공동 책임 모델의 고객 책임이 강조 표시된 다이어그램입니다.

~AWS: Fargate에 대한 공동 책임 모델의 AWS 책임이 강조 표시된 다이어그램입니다.

|수강생용 노트

사용자가 Fargate를 사용하는 경우 AWS는 Pod를 실행하는 데 사용하는 기본 런타임을 보호할 책임이 있습니다.

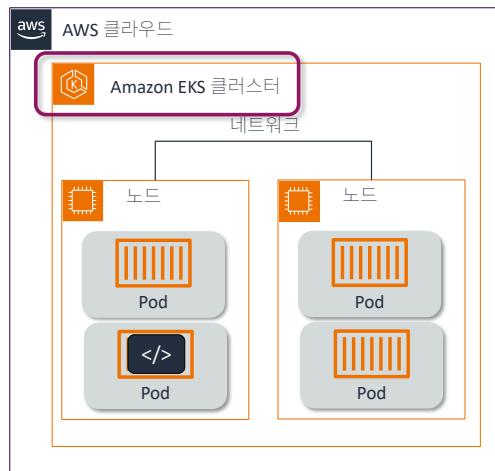
클러스터 계층 보안

고려 사항

- 최소 권한 모델을 준수
- 의심스러운 활동을 탐지
- 잠재적 위협을 식별

솔루션

- 클러스터에 대한 액세스 관리
- Amazon GuardDuty를 사용하여 감사 로그 모니터링 및 노드 런타임 모니터링을 구성



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

~Alt text

~2개의 노드가 있고 각각 2개의 Pod를 실행하는 Amazon EKS 클러스터입니다.

~

|수강생용 노트

클러스터 계층은 광범위하며 중요한 고려 사항이 많이 있습니다. 이 모듈에서는 AWS 및 Kubernetes API에 대한 액세스 보호에 대해 알아봅니다.

Amazon GuardDuty



Amazon
GuardDuty



EKS 감사 로그 모니터링

- EKS 감사 로그를 분석하여 **Amazon EKS** 클러스터 제어 영역 활동을 지속적으로 모니터링합니다.
- 위협 인텔리전스를 사용하여 알려진 악의적 행위자가 액세스하는 클러스터를 탐지하고 보고합니다.
- 기계 학습(**ML**) 모델을 사용하여 의심스러운 활동을 식별합니다.

런타임 모니터링

- 컨테이너 런타임 활동에 대한 가시성을 추가합니다.
- 전체 조직의 모든 **Amazon EKS** 클러스터를 지속적으로 모니터링합니다.
- 위협 탐지를 사용하여 알려진 컨테이너 공격 기법에 맞게 조정된 충실도가 높은 시스템 수준 이벤트를 활용합니다.

©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

|수강생용 노트

Amazon GuardDuty에서 EKS 보호

- 위협 인텔리전스는 알려진 악의적 행위자 또는 **Tor** 노드에서 액세스하는 **Amazon EKS** 클러스터에 대한 검색 결과, 잘못된 구성이 나타낼 수 있는 악명 사용자가 수행한 API 작업, **Amazon EKS** 클러스터에 대한 무단 액세스를 초래할 수 있는 잘못된構성을 탐지하고 보고합니다.
- **GuardDuty**는 기계 학습(**ML**) 모델을 사용하여 기본 **Amazon Elastic Compute Cloud(Amazon EC2)** 호스트에 대한 루트 수준 액세스 권한이 있는 컨테이너의 의심스러운 시작과 같은 권한 상승 기법과 일치하는 패턴을 식별할 수 있습니다.

자세한 내용은 <https://docs.aws.amazon.com/guardduty/latest/ug/kubernetes-protection.html>를 참조하십시오.



Running Containers on Amazon EKS

인증 및 권한 부여



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

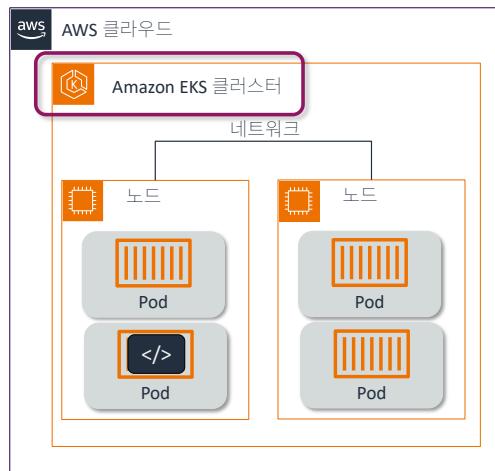
클러스터에 대한 액세스 관리

고려 사항

- AWS 및 Kubernetes API에 대한 액세스를 관리
- AWS 서비스에 대한 애플리케이션 액세스를 관리

솔루션

- AWS IAM을 사용한 인증
- Kubernetes RBAC를 사용한 권한 부여
- Kubernetes Service Account의 IAM 역할



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

~Alt text

~2개의 노드가 있고 각각 2개의 Pod를 실행하는 Amazon EKS 클러스터입니다.

~

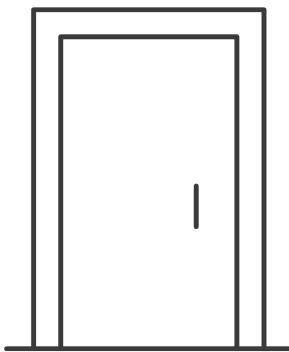
|수강생용 노트

Amazon EKS 클러스터를 생성하면 Amazon EKS 제어 영역에서 클러스터를 생성하는 IAM 보안 주체에게 클러스터의 역할 기반 액세스 제어(RBAC) 구성에 system:masters 권한이 자동으로 부여됩니다. 이 보안 주체는 표시되는 구성에는 나타나지 않으므로 원래 클러스터를 생성한 보안 주체를 추적해야 합니다. 추가 IAM 보안 주체에게 클러스터와 상호 작용할 수 있는 권한을 부여하려면 Kubernetes 내에서 aws-auth ConfigMap을 수정하고 Kubernetes rolebinding 또는 clusterrolebinding을 aws-auth ConfigMap에서 지정하는 이름을 사용하여 생성합니다.

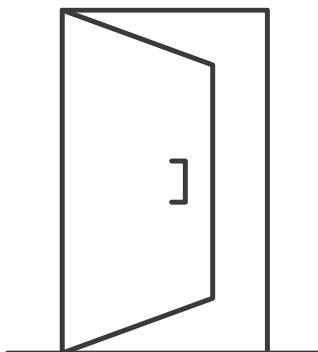
자세한 내용은 <https://docs.aws.amazon.com/eks/latest/userguide/add-user-role.html>를 참조하십시오.

복습: 인증 및 권한 부여

인증
누구입니까?



권한 부여
들어올 수 있습니까?



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

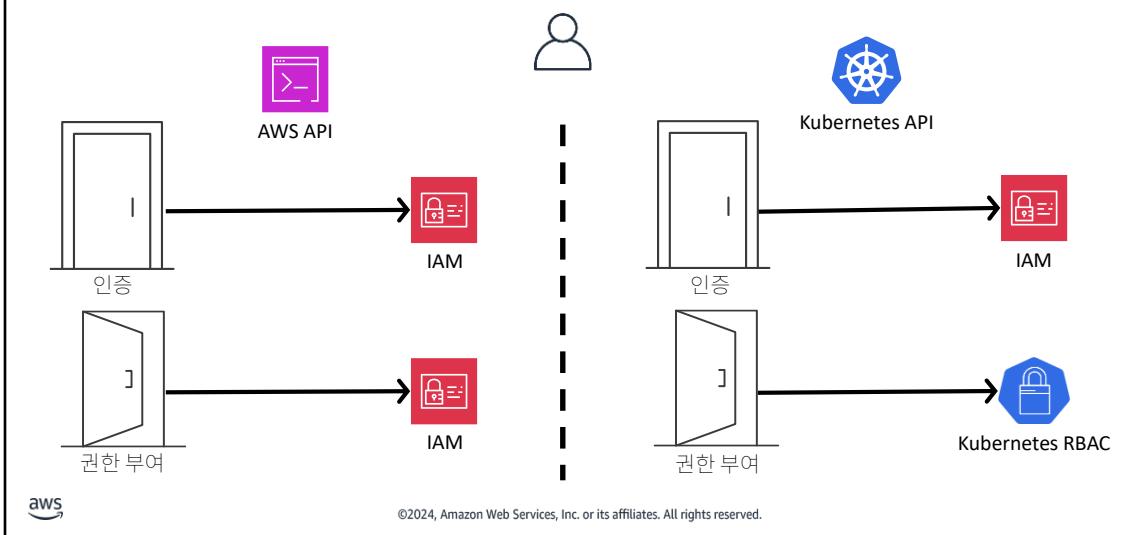
| 수강생용 노트

이 과정의 앞부분에서 인증과 권한 부여의 기본적인 차이점을 배웠습니다.

- **인증** – 인증은 누구인지를 식별합니다. 사용자는 본인이 맞는지 증명해야 합니다. **Amazon EKS**를 사용할 때 이 기능은 주로 **IAM**에서 처리합니다.
- **권한 부여** – 권한 부여는 인증된 사용자가 무엇을 할 수 있는지를 나타냅니다. 시스템에서 사용자가 시도하는 작업을 허용해야 하는지 결정합니다. **Kubernetes** 명령의 경우 **Amazon EKS**를 사용하면 주로 **Kubernetes RBAC**(역할 기반 액세스 제어)에서 이 기능을 처리합니다. AWS 명령의 경우 **IAM**에서 인증과 권한 부여를 모두 처리합니다.

자신의 환경을 잘 보호하려면 **IAM**과 **RBAC**가 함께 작동하는 방식을 이해해야 합니다.

누구의 API를 사용합니까?



~ALT text

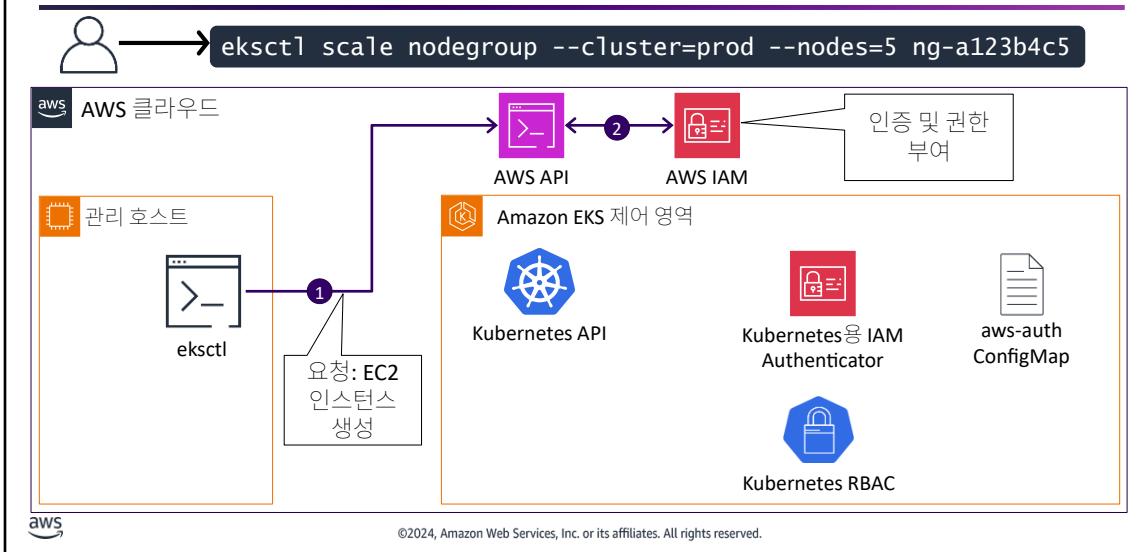
~AWS API 또는 Kubernetes API 사용에 따른 인증 및 권한 부여를 보여주는
다이어그램입니다.

|수강생용 노트

Amazon EKS에서 인증 및 권한 부여는 IAM과 Kubernetes RBAC가 처리합니다.

- AWS API를 호출하는 경우 IAM에서 인증 및 권한 부여를 관리합니다.
- Kubernetes API를 호출하는 경우 IAM에서 주로 인증을 관리하고 Kubernetes RBAC에서 주로 권한 부여를 관리합니다.

예: AWS 권한



~Alt text

~두 단계로 구성된 아키텍처 디어그램, 노트에 자세히 설명되어 있습니다.

~

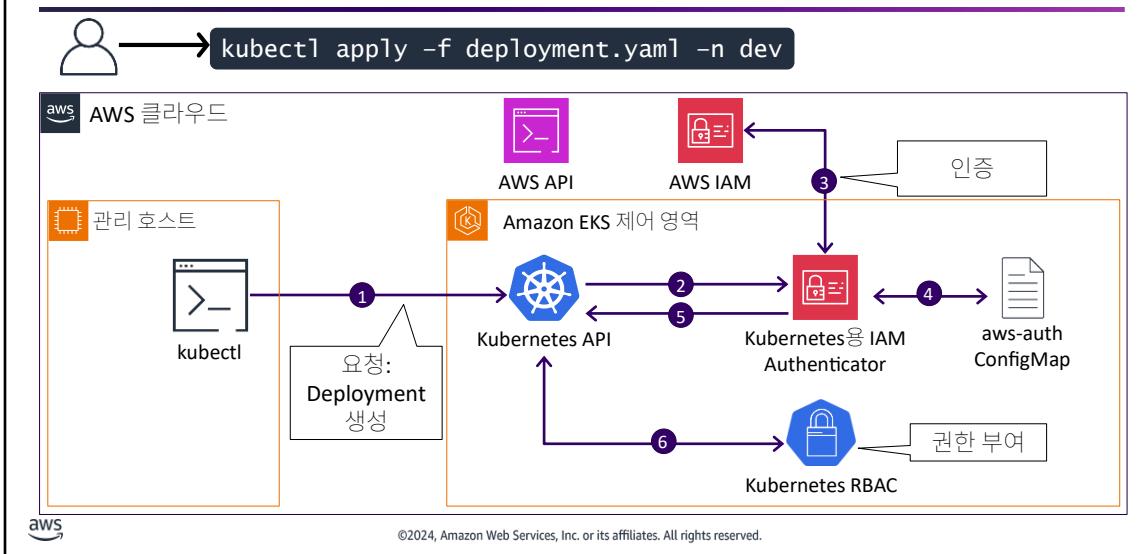
|수강생용 노트

태스크를 수행하는 데 필요한 권한을 결정할 때는 먼저 요청을 처리할 **API**를 고려해야 합니다. **AWS API**는 AWS 클라우드에 있는 리소스를 관리합니다. **Kubernetes API**는 클러스터별 리소스를 관리합니다. 단일 태스크가 실제로는 두 **API**가 모두 동원되는 작은 태스크의 연속일 때도 있습니다. 이 예에서 관리자가 클러스터에 새 노드를 추가하려 합니다. 이 태스크에는 여러 단계가 수반됩니다. 하지만 쉽게 이해할 수 있도록 이 예에서는 2개의 단계만 중점적으로 살펴봅니다.

새 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 생성은 AWS API에 요청해야 합니다. AWS API를 사용하는 경우 IAM에서 인증 및 권한 부여를 관리합니다. Amazon EKS에서의 동작은 모든 기본 AWS 서비스에서의 동작과 동일합니다.

[이미지 설명: eksctl 명령은 관리 호스트의 관리자가 실행합니다. 그러면 EC2 인스턴스 생성 요청이 AWS API로 전송됩니다. 거기서 요청이 AWS IAM으로 전송되어 요청자를 인증하고 요청 승인 여부를 결정합니다. 설명 끝]

예: Kubernetes 권한



~Alt text

~여섯 단계로 구성된 아키텍처 다이어그램, 노트에 자세히 설명되어 있습니다.

~

| 수강생용 노트

Kubernetes 클러스터에 새 Deployment를 생성하려면 Kubernetes API에 요청을 전송해야 합니다. Kubernetes에서는 RBAC를 이용해 권한을 부여할 수 있지만, 인증할 수는 없습니다. Deployment 생성 요청이 이루어지면 IAM 자격 증명이 Kubernetes 호출과 함께 전달됩니다(예시의 1단계). Kubernetes API가 인증을 위해 Kubernetes용 AWS IAM Authenticator로 자격 증명을 확인합니다(2단계 및 3단계).

Kubernetes용 IAM Authenticator가 aws-auth ConfigMap에서 AWS 자격 증명과 Kubernetes 자격 증명이 일치하는지 확인하고(4단계), IAM이 인증된 Kubernetes 자격 증명을 다시 Kubernetes API로 전송합니다(5단계). 그런 다음 Kubernetes가 RBAC에서 해당 자격 증명에 원하는 작업을 수행할 권한이 있는지 확인합니다(6단계).

클러스터에서 EKS 제어 영역 로깅을 사용하면 Kubernetes용 IAM Authenticator에서 로그 이벤트를 볼 수 있습니다.

OIDC 인증

The screenshot shows the AWS Management Console interface for an EKS cluster named 'MyCluster'. The 'Configuration' tab is active. In the 'Authentication' section, there is a button labeled 'Associate Identity Provider' which is highlighted with a purple border.

~ALT text

~EKS 클러스터에 대한 **AWS Management Console** 인증 탭에서 자격 증명 공급자를 연결하는 버튼이 강조 표시된 스크린샷입니다.

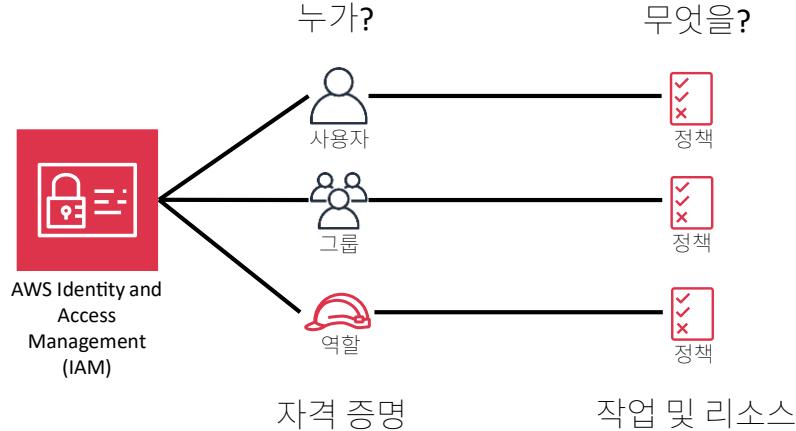
|수강생용 노트

Amazon EKS를 사용하면 AWS IAM을 사용하는 것 외에도 OIDC 자격 증명 공급자를 통해 클러스터(Kubernetes 버전 1.16 이상)에 대한 사용자 액세스를 관리할 수 있습니다.

클러스터로 이동하여 OIDC 자격 증명 공급자를 클러스터에 연결하고 구성 탭을 선택한 다음 인증 탭을 선택하고 ID 공급자 연결을 선택합니다. 그런 다음 **RoleBinding**과 **ClusterRoleBinding**을 생성하고 Kubernetes RBAC를 사용하여 권한 부여를 구성합니다.

클러스터용 OIDC 사용에 대한 자세한 내용은 ‘OpenID Connect 자격 증명 제공자에서 클러스터 사용자 인증’을 참조하십시오(<https://docs.aws.amazon.com/eks/latest/userguide/managing-auth.html>).

복습: IAM



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

~ALT text

~IAM에 저장된 리소스를 보여주는 다이어그램, 노트에 자세히 설명되어 있습니다.

| 수강생용 노트

IAM을 사용하면 AWS 리소스에 대한 액세스를 안전하게 제어할 수 있습니다. 이 다이어그램은 IAM에 저장된 일부 리소스를 보여줍니다.

- 자격 증명이란 정책을 연결할 수 있는 IAM 리소스입니다.
 - IAM 사용자란 주로 특정 개인 또는 그룹을 지칭하는 고유한 자격 증명입니다.
 - IAM 그룹이란 IAM 사용자 모음입니다. IAM 그룹을 사용하면 개별 사용자의 권한을 관리할 필요가 없어 사용자 모음의 권한을 더욱 쉽게 관리할 수 있습니다.
 - IAM 역할은 AWS 서비스를 요청하기 위한 권한 집합을 정의합니다. IAM 역할은 특정 사용자나 그룹과 연결되지 않습니다. 특정 태스크를 수행하기 위한 권한이 필요할 때 신뢰할 수 있는 엔터티(예: IAM 사용자, 애플리케이션 또는 다른 AWS 서비스)에서 IAM 역할을 담당합니다.
- IAM 정책은 사용자, 그룹, 역할과 연결되어 AWS 리소스에 대한 권한을 할당합니다. 기본적으로 IAM 자격 증명에는 권한이 없습니다. IAM 자격 증명에 원하는 권한은 IAM 정책을 사용하여 부여해야 합니다. IAM 정책에는 특정 리소스에 대해 허용되거나 허용되지 않는 작업 목록이 포함됩니다.

복습: IAM 정책

The screenshot shows the AWS IAM Management Console. On the left, the navigation pane is visible with options like Dashboard, Access management, Policies, and Access reports. The main area is titled 'Summary' for the 'eksAdmin' policy. It displays the Policy ARN (arn:aws:iam::429430136407:policy/eksAdmin) and a 'Delete policy' button. Below this, there are tabs for Permissions, Policy usage, Policy versions, and Access Advisor. Under the Permissions tab, there's a 'Policy summary' button, a JSON editor, and an 'Edit policy' button. A 'Filter' search bar is present. The main content area shows 'Allow (2 of 235 services)' with 'Show remaining 233'. Two entries are listed: EC2 with 'Full access' to 'All resources' and EKS with 'Full access' to 'All resources'. The bottom of the screen has a copyright notice: '©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.'

~ALT text

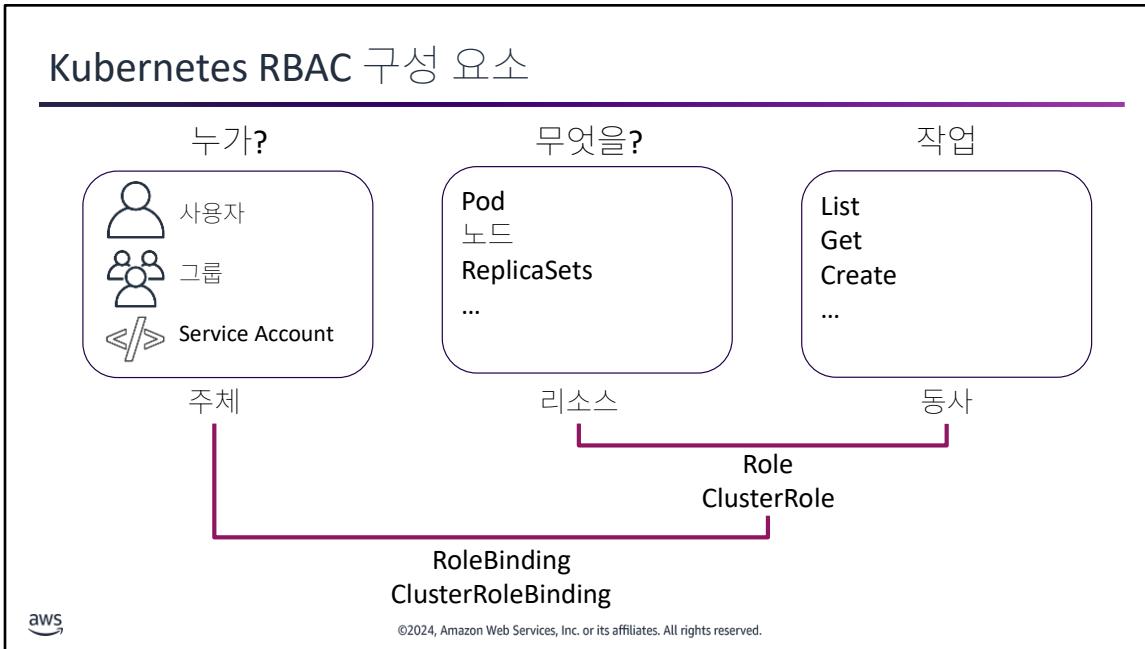
~IAM 정책의 예를 보여 주는 AWS Management Console의 스크린샷, 노트에 자세히 설명되어 있습니다.

|수강생용 노트

IAM 정책은 AWS 리소스에 대한 권한을 제어합니다. 이 예의 정책은 사용자 또는 역할과 같은 자격 증명에 연결되어 나열된 AWS 리소스를 관리할 권한을 부여할 수 있습니다. IAM 정책은 클러스터를 관리하는 사람뿐 아니라 클러스터에도 중요합니다. Amazon EKS의 클러스터와 노드 모두 AWS 리소스를 요청하는 데 필요한 권한을 부여해줄 IAM 역할이 필요합니다. 요청할 권한이 없으면 클러스터와 노드가 제대로 작동할 수 없습니다.

이 정책은 예일 뿐입니다. 이 정책은 많은 사용자에게 필요한 것보다 많은 광범위한 권한을 Amazon EC2와 Amazon EKS 서비스에 부여합니다. 권한을 구성할 때는 최소 권한의 원칙을 따르는 것이 좋습니다. 자세한 내용은 **AWS Identity and Access Management** 사용 설명서의 ‘최소 권한 부여’(<https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#grant-least-privilege>)를 참조하십시오.

Kubernetes RBAC 구성 요소



~ALT text

~Kubernetes RBAC 구성 요소를 보여주는 다이어그램, 노트에 자세히 설명되어 있습니다.

|수강생용 노트

RBAC의 구성은 IAM 같은 다른 인증 서비스와 유사하지만 용어와 기능은 약간 다릅니다.

- 주체는 Kubernetes API에 액세스하려는 사용자, 사용자 그룹, Service Account를 의미합니다.
- 리소스는 클러스터에서 사용할 수 있는 Kubernetes API 객체를 의미합니다. 예로는 Pod, 노드, ReplicaSet 등이 있습니다.
- 운영(동사)은 리소스에서 실행할 수 있는 작업을 의미합니다. **get, watch, create, patch, delete**를 예로 들 수 있습니다. 리소스마다 동사가 다르지만 모두 CRUD(create, read, update, delete) 작업입니다.
- 역할 또는 ClusterRole에는 특정 리소스에서 수행할 수 있는 작업 집합을 지정하는 규칙이 포함되어 있습니다. 이 규칙은 추가만 가능합니다. 'deny' 규칙은 존재하지 않습니다. 역할과 ClusterRole의 차이점은 범위입니다.
 - 역할은 Namespace가 지정된 리소스입니다. 특정 Namespace에 존재하는 리소스에 대한 권한을 설정합니다.
 - ClusterRole은 클러스터 범위의 리소스입니다. ClusterRole이 할 수 있는 기능은 다음과 같습니다.
 - Namespace 또는 모든 Namespace에서 Pod와 같이 Namespace가 지정된 리소스에 대한 권한을 정의합니다.

- 노드와 같이 **Namespace**가 지정되지 않은 리소스에 대한 권한을 정의합니다.
- **RoleBinding** 또는 **ClusterRoleBinding**은 역할에 정의된 권한을 주체 또는 주체 목록에 부여합니다.
 - **RoleBinding**은 동일한 **Namespace**에서 **Role**을 참조하는 **Namespace**가 지정된 리소스입니다. 혹은 **RoleBinding**은 **ClusterRole**을 참조하고 해당 **ClusterRole**을 **RoleBinding**의 **Namespace**에 바인딩할 수 있습니다.
 - **ClusterRoleBinding**은 클러스터의 모든 **Namespace**에 **ClusterRole**을 바인딩하는 클러스터 범위의 리소스입니다.

Kubernetes 권한 부여에 대한 자세한 내용은 Kubernetes 온라인 설명서의 ‘Authorization Overview’(<https://kubernetes.io/docs/reference/access-authn-authz/authorization/>)를 참조하십시오.

RBAC 구성 요소에 대한 자세한 내용은 Kubernetes 온라인 설명서의 ‘Using RBAC Authorization’(<https://kubernetes.io/docs/reference/access-authn-authz/rbac/>)을 참조하십시오.

예: RBAC 역할

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: default
  name: pod-creator
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["get", "list", "create"]
```



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

이 예에서는 기본 Namespace의 Pod에서 특정 작업에 대한 액세스 권한을 부여하는 역할을 보여 줍니다. **apiGroups**의 NULL 항목("")은 핵심 API 그룹을 나타냅니다.

예: RoleBinding을 사용하여 RBAC 역할에 사용자 바인딩

```
apiVersion: rbac.authorization.k8s.io/v1
# 이를 통해 'student'는 'default' Namespace에서 Pod로 작업할 수 있습니다.
# 'default' Namespace에 'pod-worker'라는 Role이 필요합니다.
kind: RoleBinding
metadata:
  name: create-pods
  namespace: default
subjects:
- kind: User
  name: student
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: pod-creator
  apiGroup: rbac.authorization.k8s.io
```



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

이 예에서는 기본 Namespace의 사용자 **student**에게 **pod-creator** 역할을 부여하는 **RoleBinding**을 보여 줍니다. 이를 통해 **student**는 **default** Namespace에서 Pod를 나열하고 가져오고 생성할 수 있습니다.

RoleBindings의 **name** 필드는 대소문자를 구분합니다.

예: RBAC ClusterRole

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: secret-reader
rules:
- apiGroups: [""]
  resources: ["secrets"]
  verbs: ["get", "watch", "list"]
```



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

ClusterRole을 사용하면 RBAC 역할과 동일한 사용 권한을 부여할 수 있습니다.

ClusterRole은 클러스터 범위이기 때문에 다음에 대한 액세스 권한을 부여하는 데도 사용할 수 있습니다.

- 노드와 같은 클러스터 범위의 리소스
- 리소스가 아닌 엔드포인트
- Pod와 같이 Namespace 전반에서 Namespace가 지정된 리소스. 예를 들어, ClusterRole을 사용하여 특정 사용자가 `kubectl get pods --all-namespaces`를 실행하도록 허용할 수 있습니다.

이 예에서는 Secret에 대한 읽기 액세스 권한을 부여하는 ClusterRole을 보여 줍니다.

Namespace를 지정하지 않는다는 점을 제외하면 RBAC 역할과 유사합니다.

ClusterRole은 하나의 특정 Namespace(RoleBinding에 의해 주체에 바인딩된 경우)에 적용되거나 모든 Namespace(ClusterRoleBinding에 의해 주체에 바인딩된 경우)에 적용될 수 있습니다.

예: RoleBinding을 사용하여 RBAC ClusterRole에 사용자 바인딩

```
apiVersion: rbac.authorization.k8s.io/v1
# 이를 통해 'student'는 'development' Namespace의 Secret을 읽을 수 있습니다.
# 'secret-reader'라는 ClusterRole이 필요합니다.
kind: RoleBinding
metadata:
  name: read-secrets
  # 'development' Namespace 내에서만 권한을 부여합니다.
  namespace: development
subjects:
- kind: User
  name: student
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: secret-reader
  apiGroup: rbac.authorization.k8s.io
```



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

또한 RoleBinding은 ClusterRole을 참조하여 해당 ClusterRole에 정의된 권한을 RoleBinding의 Namespace 내 리소스에 부여할 수 있습니다. 이렇게 하면 클러스터 전반에서 공통 역할 집합을 정의한 다음 여러 Namespace에 다시 사용할 수 있습니다.

이 예에서는 RoleBinding이 ClusterRole을 참조하더라도 student는 development Namespace에서만 Secret을 읽을 수 있습니다. 그 이유는 (메타데이터에 있는) RoleBinding의 Namespace가 development이기 때문입니다.

예: ClusterRoleBinding을 사용하여 RBAC ClusterRole에 그룹 바인딩

```
apiVersion: rbac.authorization.k8s.io/v1
# 이를 통해 'admins' 그룹의 모든 사용자가 Namespace의 Secret을 읽을 수 있습니다.
kind: ClusterRoleBinding
metadata:
  name: read-secrets-global
subjects:
- kind: Group
  name: admins
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: secret-reader
  apiGroup: rbac.authorization.k8s.io
```



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

이 ClusterRoleBinding의 예에서는 그룹 관리자의 모든 사용자가 Namespace의 Secret을 읽을 수 있습니다. ClusterRole은 이전 예시와 정확히 동일하지만 이 예시에서는 ClusterRoleBinding을 사용하므로 전체 클러스터에 적용됩니다.

ClusterRoleBinding의 name 필드는 대소문자를 구분합니다.

지식 확인



aws

IAM 및 RBAC 용어

이 활동에서는 AWS IAM 및 Kubernetes RBAC에서 허용되는 작업 및 리소스를 나열하는 객체를 식별합니다.

©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

IAM에서 허용되는 작업 및 리소스 목록은 무엇입니까?

IAM

Kubernetes RBAC

IAM 사용자

RBAC 주체

IAM 역할

RBAC 역할

IAM 정책

RBAC 그룹

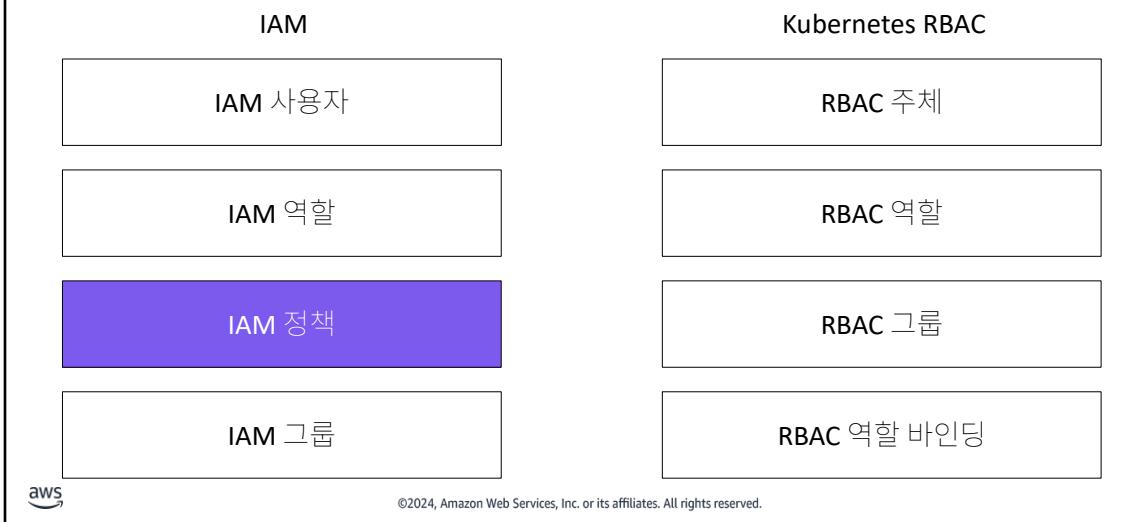
IAM 그룹

RBAC 역할 바인딩



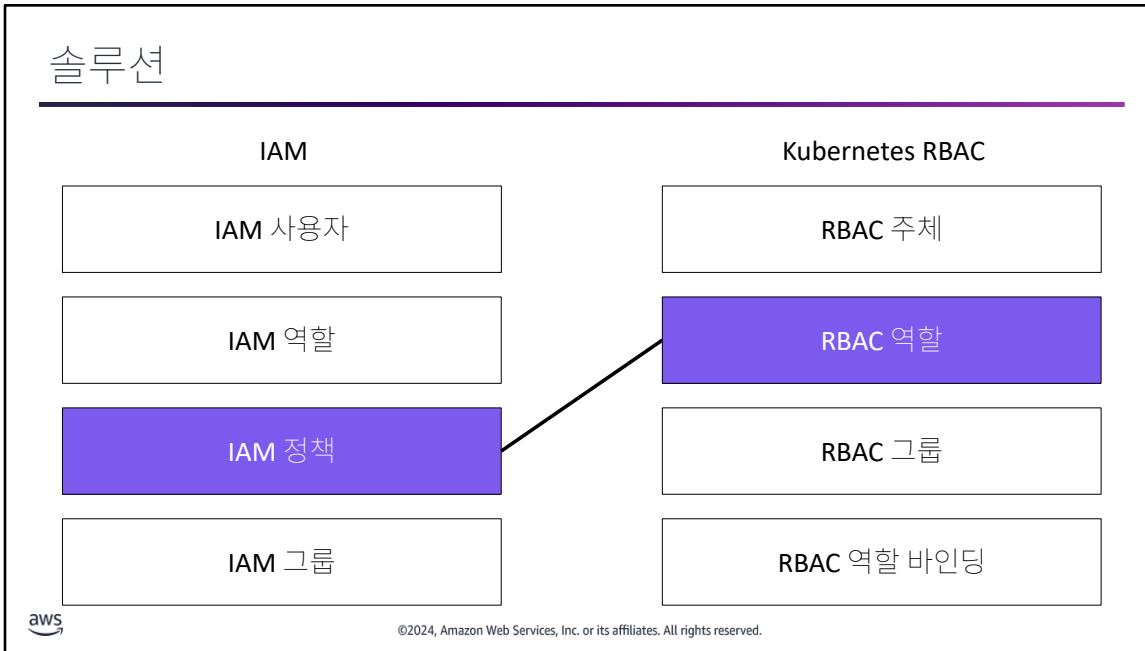
©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

RBAC에서 허용되는 작업 및 리소스 목록은 무엇입니까?



정답은 **IAM 정책**입니다. 질문의 두 번째 부분: **IAM 정책에 상응하는 객체는 Kubernetes RBAC 주체, 역할, 그룹, Role Binding** 중에서 무엇입니까?

솔루션



정답은 RBAC 역할입니다. IAM 정책은 IAM에서 허용되는 작업을 나열하는 반면, RBAC 역할은 Kubernetes RBAC에서 허용되는 작업과 리소스를 나열합니다.



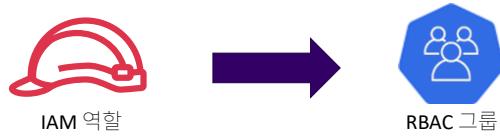
Running Containers on Amazon EKS

IAM 및 RBAC 관리



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

IAM과 RBAC 간 매핑



IAM 역할

RBAC 그룹

©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

역할 같은 AWS 자격 증명을 Kubernetes API 서버를 이용해 인증하면 RBAC는 Kubernetes 명령을 실행을 위해 역할에 필요한 권한 수를 제어합니다. IAM 역할을 RBAC 그룹에 매핑하면 IAM 역할에서 해당 RBAC 그룹에 바인딩된 RBAC 역할에 의해 정의된 권한을 부여합니다.

Summary

Role ARN arn:aws:iam:: 123456789012 :role/eksctl-prod-nodegroup-standard-wo-NodeInstanceRole-LWOBHGZBW0N1

Role description Edit

Instance Profile ARNs arn:aws:iam:: 123456789012 :instance-profile/eks-2eb938f9-cf9f-b128-0fef-981a92190a96

Path /

Creation time 2020-06-01 18:14 EDT

Last activity 2020-07-26 17:50 EDT (Today)

Maximum session duration 1 hour Edit

Permissions Trust relationships Tags (5)

▼ Permissions policies (4 policies applied)

Attach policies

Policy name ▾

- AmazonEKSWorkerNodePolicy
- AmazonEC2ContainerRegistryReadOnly
- CloudWatchAgentServerPolicy
- AmazonEKS_CNI_Policy

```
$ kubectl edit configmap aws-auth -n kube-system
apiVersion: v1
data:
  mapRoles: | 
    - groups:
      - system:bootstrappers
      - system:nodes
    rolearn: arn:aws:iam::123456789012:role/eksctl-prod-nodegroup-standard-wo-NodeInstanceRole-LWOBHGZBW0N1
    username: system:node:{EC2PrivateDNSName}
  kind: ConfigMap
```

©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

~ALT text

~클러스터의 노드에 대해 eksctl에서 생성한 역할의 세부 정보를 보여주는 IAM 콘솔의 스크린샷입니다.

|수강생용 노트

IAM 자격 증명과 RBAC 사용자 또는 그룹 간 매핑은 Kubernetes에서 생성된 **aws-auth**라는 ConfigMap을 통해 수행됩니다. 이 YAML 문서는 IAM 보안 주체(IAM 사용자 또는 역할)의 IAM Amazon Resource Name(ARN)을 RBAC 역할 또는 ClusterRole에 매핑합니다. IAM에서 자격 증명을 인증하면 Kubernetes API 서버는 **kube-system** Namespace에 있는 **aws-auth** ConfigMap을 읽어 사용자와 연결할 RBAC 그룹을 결정합니다. RBAC 그룹은 Kubernetes 리소스 모음에 대해 수행할 수 있는 작업(동사) 집합을 정의한다는 점에서 IAM 역할과 유사합니다.

이 예에서는 클러스터에서 실행 중인 노드에 대해 생성된 IAM 역할과 이 IAM 역할과 연결된 Kubernetes 그룹(**system:bootstrappers** 및 **system:nodes**) 간의 초기 매핑을 보여 줍니다. 이 예에서는 eksctl을 사용하여 클러스터를 만들고 이 IAM 역할을 만들었습니다. 이 역할에는 클러스터의 노드가 AWS 호출을 수행하는 데 필요한 IAM 권한 2개가 모두 있습니다. 또한 이 역할에는 노드가 클러스터에 조인하는데 필요한 RBAC의 권한도 있습니다. 다시 말하면 이 구성은 클러스터의 노드로 작동할 인스턴스에 IAM 및 RBAC 모두에서 올바른 수준의 권한을 할당합니다.

클러스터 만드는 데 `eksctl`을 이용하지 않는다면 이 ConfigMap을 수동으로 만들어야 합니다.
자세한 방법은 **Amazon EKS** 사용 설명서의 ‘클러스터의 사용자 또는 IAM 역할
관리’(<https://docs.aws.amazon.com/eks/latest/userguide/add-user-role.html>)를 참조하십시오.

예: web-admins 그룹의 aws-auth ConfigMap 수정

```
$ kubectl describe configmap aws-auth -n kube-system

apiVersion: v1
data:
  mapRoles: |
    - groups:
        - system:bootstrappers
        - system:nodes
        rolearn: arn:aws:iam::123456789012:role/eksctl-prod-nodegroup-\
          standard-wo-NodeInstanceRole-LWOBHGZBW0N1
        username: system:node:{{EC2PrivateDNSName}}
    - groups:
        - web-admins
        rolearn: arn:aws:iam::123456789012:role/webAdminRole
        username: web-admin
kind: ConfigMap
```



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 강사용 노트

| 시간이 허락한다면 이 슬라이드 다음에 **RBAC** 역할 시연을 진행할 수 있습니다. 다음 슬라이드를 숨김 해제하여 개별로 사용할 수 있습니다. 이 슬라이드는 수강생 가이드에 포함되지 않습니다.

| 데모 노트:

| RBAC 역할 사용자 정의

- | Kubernetes RBAC에서 새로운 역할을 생성합니다.
- | RoleBinding을 생성합니다.
- | RoleBinding을 그룹에 바인딩합니다.
- | aws-auth ConfigMap을 수정합니다.

| 수강생용 노트

다음은 웹 관리자의 권한을 제어하도록 사용자 정의된 aws-auth ConfigMap의 예입니다. 이 예에서 IAM에서 **webAdminRole**을 담당하는 모든 AWS 엔터티는 인증하고, 클러스터 액세스 권한을 얻고, RBAC 그룹 **web-admins**에서 허용하는 Kubernetes API를 사용하여 작업을 수행할 수 있습니다.

지식 확인 1

IAM 자격 증명을 사용하여
Kubernetes 클러스터를
인증하려면 무엇이 필요합니까?

보기	응답
A	Kubernetes의 기본 기능으로 필요한 것이 없습니다.
B	RBAC Service Account를 생성해야 합니다.
C	OpenID Connect 공급자를 구성해야 합니다.
D	IAM 자격 증명을 aws-auth ConfigMap의 Kubernetes 그룹에 매핑해야 합니다.

지식 확인 1: 정답은 D입니다.

IAM 자격 증명을 사용하여

Kubernetes 클러스터를
인증하려면 무엇이 필요합니까?

보기 응답

A

Kubernetes의 기본 기능으로 필요한 것이 없습니다.

B

RBAC Service Account를 생성해야 합니다.

C

OpenID Connect 공급자를 구성해야 합니다.

D

IAM 자격 증명을 aws-auth ConfigMap의 Kubernetes 그룹에
매핑해야 합니다.

정답

 ©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

정답은 D입니다. Kubernetes용 AWS IAM Authenticator는 aws-auth ConfigMap을 사용하여 인증된 IAM 자격 증명을 Kubernetes 주체에 매핑합니다. 인증된 자격 증명은 Kubernetes RBAC로 권한을 부여받습니다.

cluster-admin RBAC 역할 관리

- 자동 생성됨
- **system:masters RBAC** 그룹에 바인딩됨
- 클러스터에 대한 전체 액세스 권한 있음



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

사용자가 클러스터를 생성할 때 **Kubernetes**는 자동으로 **cluster-admin RBAC** 역할을 생성합니다. **cluster-admin RBAC** 역할은 **UNIX/Linux** 시스템의 루트 사용자와 비슷합니다. 이 역할은 아무런 제한 없이 클러스터를 관리할 수 있도록 모든 권한을 부여합니다. **Kubernetes**는 사용자가 클러스터를 생성할 때 자동으로 **system:masters**라는 RBAC 그룹을 생성하고 **cluster-admin RBAC** 역할을 이 RBAC 그룹에 바인딩합니다.

클러스터의 보안을 유지하려면 **system:masters RBAC** 그룹에 대한 액세스를 주의 깊게 관리해야 합니다.

예: 기본 RBAC 그룹

```
$ rbac-lookup -k group
```

GROUP	ROLE
eks:kube-proxy-windows	ClusterRole/system:node-proxier
system:authenticated	ClusterRole/eks:podsecuritypolicy:privileged
system:authenticated	ClusterRole/system:basic-user
system:authenticated	ClusterRole/system:discovery
system:authenticated	ClusterRole/system:public-info-viewer
system:bootstrappers	ClusterRole/eks:node-bootstrapper
system:masters	ClusterRole/cluster-admin
system:node-proxier	ClusterRole/system:node-proxier
system:nodes	ClusterRole/eks:node-bootstrapper
system:unauthenticated	ClusterRole/system:public-info-viewer



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

Kubernetes API 서버는 클러스터를 만들 때 기본 RBAC 객체 집합을 생성합니다. Amazon EKS는 Amazon EKS에서 관리하는 클러스터에 기본 객체를 추가합니다. 이러한 객체 중 대부분에 **system:** 접두사가 포함되어 있으며, 이 접두사는 클러스터 제어 영역에서 직접 관리함을 나타냅니다. **system:** 접두사를 사용하여 RBAC 객체를 수정하면 클러스터가 작동하지 않을 수 있습니다.

이 예에서는 **system:masters** 그룹에 바인딩된 **cluster-admin ClusterRole**을 강조합니다. 이 ClusterRole은 아무런 제한 없이 클러스터를 관리할 수 있도록 모든 권한을 부여합니다. **system:masters** 그룹에 매핑된 모든 IAM 자격 증명에는 이러한 광범위한 권한이 있습니다.

Kubernetes에서 생성한 기본 역할 및 Role Binding에 대한 자세한 내용은 Kubernetes 온라인 설명서의 ‘Default roles and role bindings’(<https://kubernetes.io/docs/reference/access-authn-authz/rbac/#default-roles-and-role-bindings>)를 참조하십시오.

이 예에서는 RBAC 주체에 바인딩된 역할을 검색할 수 있는 kubectl용 플러그인인 **rbac-lookup**을 사용합니다. 이 플러그인에 대한 자세한 내용은 GitHub의 **rbac-lookup** 리포지토리(<https://github.com/FairwindsOps/rbac-lookup>)를 참조하십시오.

이 예에서는 **-k group** 인수를 사용하여 그룹만 표시하도록 출력을 필터링하고 공간 배열과 명확성을 위해 출력을 편집했습니다. 실제 명령 출력에는 나열된 주체의 범위가 포함된 세 번째 열이 포함됩니다. 새 클러스터를 생성할 때 기본적으로 생성되는 그룹은 모두 클러스터 전체를 범위로 합니다.

cluster-admin 역할

```
$ kubectl get clusterroles cluster-admin -o yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  annotations:
    rbac.authorization.kubernetes.io/autoupdate: "true"
  creationTimestamp: "2023-06-01T22:10:57Z"
  labels:
    kubernetes.io/bootstrapping: rbac-defaults
  name: cluster-admin
  resourceVersion: "40"
  selfLink: /apis/rbac.authorization.k8s.io/v1/clusterroles/cluster-admin
  uid: ba460613-66da-4b54-9054-187479cb3a3b
rules:
- apiGroups:
  - '*'
  resources:
  - '*'
  verbs:
  - '*'
- nonResourceURLs:
  - '*'
  verbs:
  - '*'
```



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

사용자가 클러스터를 생성할 때 RBAC는 자동으로 **cluster-admin ClusterRole**을 포함한 ClusterRole을 생성합니다. **cluster-admin ClusterRole**은 모든 리소스에서 어떤 작업이든 수행할 권한이 있는 슈퍼 사용자 역할로 간주됩니다. **cluster-admin ClusterRole**은 사용자가 클러스터를 생성할 때 **system:masters** 그룹에 바인딩된 클러스터입니다. **system:master** 그룹에 매핑되는 모든 IAM 자격 증명은 클러스터 및 모든 네임스페이스의 모든 리소스를 완전히 제어합니다.



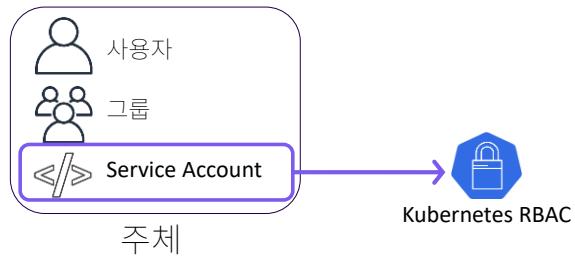
Running Containers on Amazon EKS

RBAC Service Account를 사용한 Pod 권한 관리



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Kubernetes Service Account



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

Kubernetes Service Account는 RBAC 주체 유형 중 하나입니다. RBAC 사용자와 그룹처럼 Service Account도 Role이나 ClusterRole에 바인딩하여 Kubernetes 권한을 부여할 수 있습니다.

Service Account 및 Pod

```
$ kubectl create namespace sa-demo
namespace/sa-demo created

$ kubectl -n sa-demo get serviceaccounts
NAME          SECRETS   AGE
default        0          1m

$ kubectl apply -n sa-demo -f demo-pod.yaml
pod/demo-pod created

$ kubectl -n sa-demo exec demo-pod -- cat \
/run/secrets/kubernetes.io/serviceaccount/token
MIICiTCCAFICCCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAKGA1UEBhMCVVMxCzAJBgNVBAgTA
1dbMRawDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbwF6b24xFDASBgnVBAsTC01BTSBdb25zb2x1MR
IwEAYDVQQDEw1UZXN0Q21sYWMxHzAdBgkqhkiG9w0BCQEWEg5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI
1MjA0NTIxwhcn...
```



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 강사용 노트

| Kubernetes 버전 1.23 및 이하에서는 여기에 표시된 것과 다른 방법을 사용했습니다. 즉, 더 장기 토큰이 포함된 각 Service Account에 대해 Secret이 생성되었습니다. 이에 대한 자세한 내용을 보려면 <https://itnext.io/big-change-in-k8s-1-24-about-serviceaccounts-and-their-secrets-4b909a4af4e0>를 참조하십시오.

| 수강생용 노트

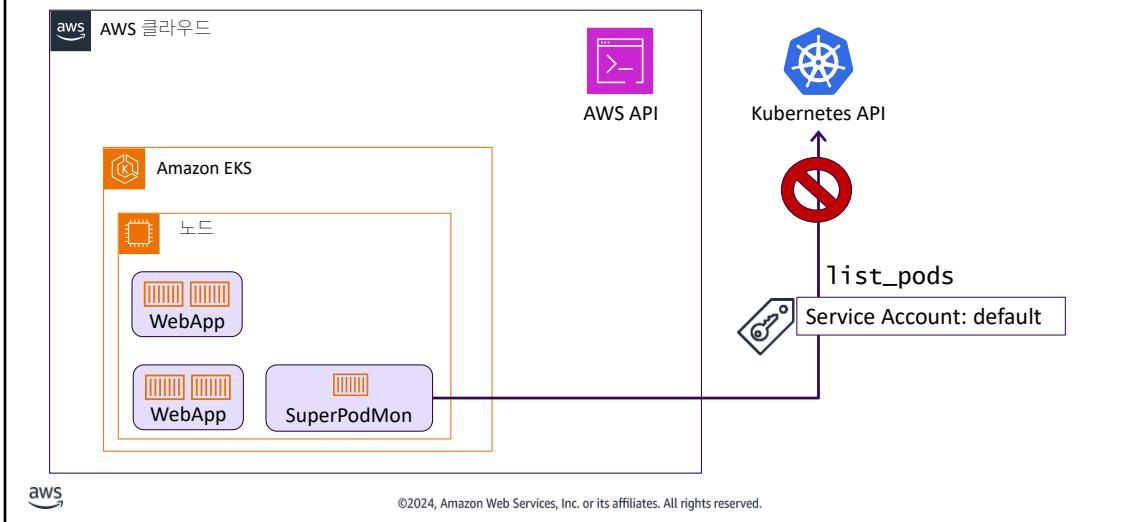
Service Account는 사용자나 그룹과는 다른 방식으로 관리됩니다. RBAC 사용자 및 그룹은 IAM 같은 외부 서비스에 저장되고 인증되는 자격 증명에 매핑됩니다. RBAC Service Account는 RBAC 자체에 저장됩니다. 각 Service Account는 서명된 보유자 토큰을 사용하여 요청을 확인하는 자동으로 활성화된 인증자입니다. 이러한 토큰은 Pod가 시작될 때 ServiceAccount 승인 컨트롤러에 의해 Pod에 할당됩니다.

이 슬라이드에 표시된 것처럼 모든 네임스페이스에는 명시적인 Kubernetes 권한이 없는 "default"라는 Service Account가 있습니다. 기본 Service Account는 사용자 지정 Service Account를 지정하지 않는 모든 Pod와 연결됩니다. 이 예에서는 demo-pod의 매니페스트가 고객 Service Account를 지정하지 않았다고 가정합니다. 결과적으로 Pod는 기본 Service Account에 할당되고 클러스터 권한이 없습니다.

보유자 토큰은 잘 알려진 위치의 Pod에 탑재되며, 이를 통해 클러스터 내 프로세스가 API 서버와 통신할 수 있습니다. 이 예에서는 kubectl exec 명령을 사용하여 demo-pod의 컨테이너로 셸을 열고 토큰이 저장되는 기본 위치를 확인합니다. 이 예의 토큰 길이는 가독성을 위해 잘렸습니다.

Service Account 토큰에 대한 자세한 내용은 Kubernetes 설명서의 ‘Service Account Tokens’(<https://kubernetes.io/docs/reference/access-authn-authz/authentication/#service-account-tokens>)를 참조하십시오.

기본 Service Account 권한



~ALT text

~애플리케이션이 Kubernetes API에서 정보를 요청합니다. 기본 Service Account에는 명령을 실행하는데 필요한 권한이 없습니다. 노트에 자세히 설명되어 있습니다.

|수강생용 노트

이 다이어그램에서는 **SuperPodMon**이라는 환경을 모니터링하기 위해 만든 애플리케이션 예가 나와 있습니다. 이 애플리케이션은 사용자의 환경에 관한 유용한 보고서를 작성하지만, 보고서를 작성하려면 Kubernetes API에 사용자 클러스터 관련 정보를 요청해야 합니다. 기본적으로 모든 Pod에는 Pod에 연결된 기본 Service Account의 토큰이 있습니다. Pod가 명령을 실행하려고 하면 이 토큰이 명령과 함께 Kubernetes API 서버로 전달됩니다. 기본 Service Account에는 이 명령을 실행하는데 필요한 권한이 없습니다. 이 문제를 해결하는 가장 좋은 방법은 새 Service Account를 생성하고 올바른 사용 권한을 계정에 할당하는 것입니다.

새 Service Account 생성 옵션

- 새 Service Account를 얼마나 자주 생성합니까?
- 특정 Service Account를 많이 생성합니까 아니면 일반적인 계정을 몇 개 생성합니까?



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

새 Service Account를 생성하여 Kubernetes 권한을 Pod에 할당합니다. 생성할 올바른 Service Account 수를 결정할 때는 워크로드의 보안 요구 사항과 필요한 관리 수준 사이의 균형을 맞춰야 합니다.

예: Service Account 생성

```
$ kubectl create serviceaccount super-pod-mon
Serviceaccount/super-pod-mon created

$ kubectl describe serviceaccount super-pod-mon
Name:           super-pod-mon
Namespace:      default
Labels:         <none>
Annotations:   <none>
Image pull secrets: <none>
Mountable secrets: <none>
Tokens:        <none>
Events:        <none>
```



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

다음은 **Service Account** 생성의 예입니다. 사용자와 달리 **Service Account**는 네임스페이스가 지정된 리소스입니다. 이 예에서는 **Role**을 **Service Account**에 바인딩하여 부여한 모든 권한이 기본 네임스페이스에만 적용됩니다.

이전 버전의 **Kubernetes**에는 생성 시 장기 토큰이 할당되어 **Secret**으로 저장되었습니다. **Kubernetes** 버전 **1.24** 이상에서는 더 이상 그렇지 않습니다. 대신, 단기 토큰이 해당 **Service Account**와 연결된 **Pod** 내의 컨테이너에 저장됩니다.

예: 역할 생성 후 Service Account에 바인딩

- 새 역할을 생성하여 Service Account에 바인딩합니다.

```
$ kubectl create role pod-scanner --verb=get,list,watch --resource=pods,pods/status  
$ kubectl create rolebinding super-pod-mon-binding --role=pod-scanner \  
--serviceaccount=default:super-pod-mon --namespace=default
```

- PodSpec에 Service Account를 추가합니다.

```
apiVersion: v1  
kind: Pod  
metadata:  
  name: super-pod-mon  
spec:  
  serviceAccountName: super-pod-mon  
  ...
```

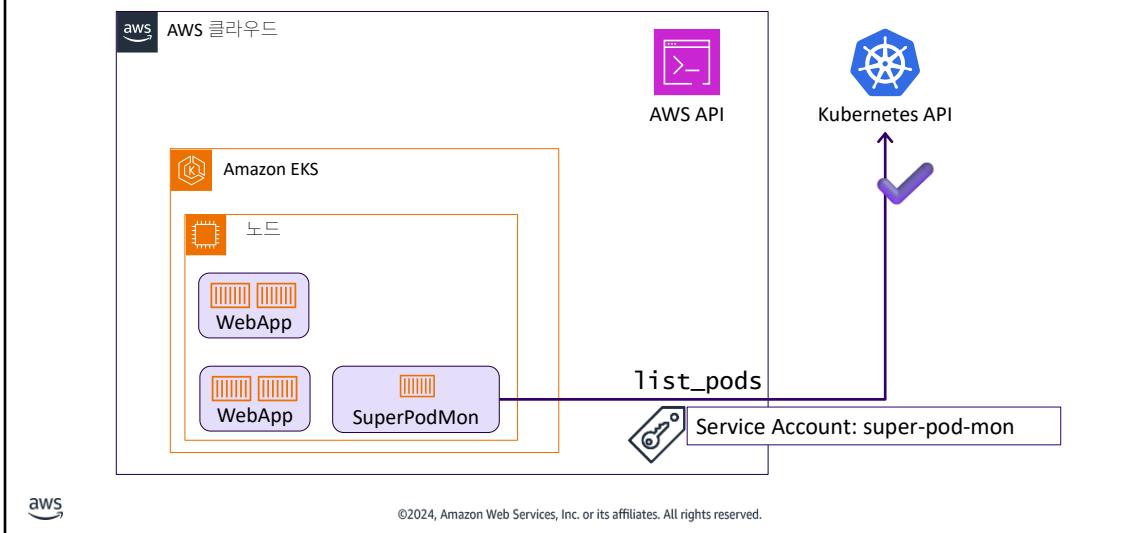


©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

이 예에서는 새 RBAC 역할을 만들어 이전에 만든 Service Account에 바인딩하는 방법을 보여줍니다. **pod-scanner** 역할은 모니터링 애플리케이션에서 Pod를 다루는데 필요한 권한을 부여합니다. 마지막으로 **SuperPodMon** 애플리케이션을 포함하는 Pod의 PodSpec 파일 내 **super-pod-mon** Service Account를 지정해야 합니다. 이러한 Pod는 기본 Service Account가 아닌 **super-pod-mon** Service Account용 토큰을 검색합니다. Pod는 kubernetes Service를 통하여 **Kubernetes Secret**으로 이 토큰에 액세스합니다. Secret은 `/var/run/secrets/kubernetes.io/serviceaccount`에서 Pod에 탑재됩니다.

Service Account 권한 사용자 정의



~ALT text

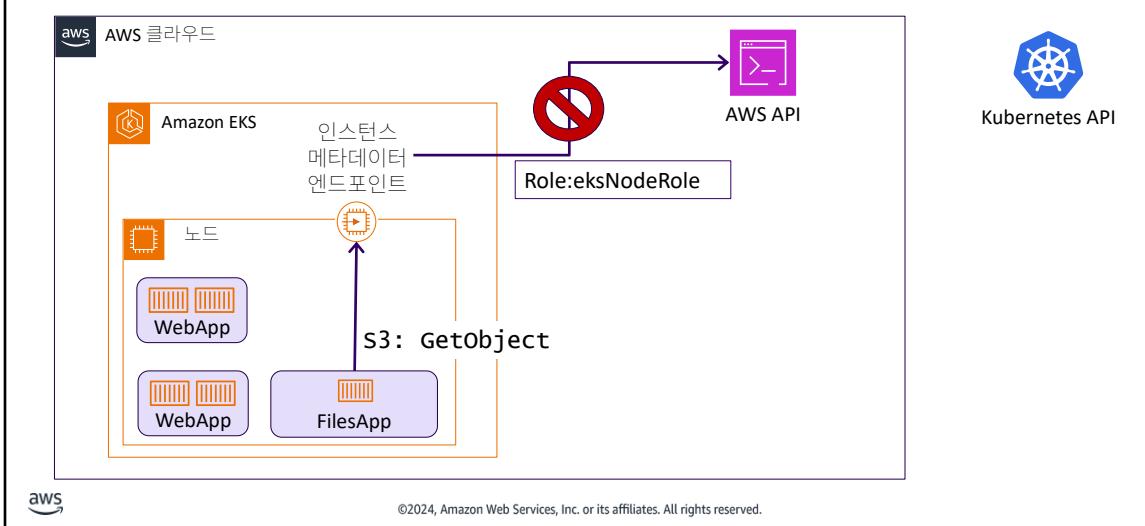
~SuperPodMon의 요청은 super-pod-mon Service Account를 사용하여 승인됩니다.

| 수강생용 노트

올바른 권한을 사용하도록 SuperPodMon용 Service Account를 구성했으니, 이제 모니터링 애플리케이션은 필요한 Kubernetes 명령을 실행할 수 있습니다.

이 예에서는 가상의 애플리케이션을 사용했습니다. Pod로 실행되며 Service Account를 사용하는 실제 모니터링 애플리케이션의 예를 보고 싶다면 GitHub에서 **kube-ops-view** 리포지토리(<https://github.com/hjacobs/kube-ops-view>)를 참조하십시오.

Pod에 어떤 AWS 권한이 있습니까?



~ALT text

Amazon S3에 액세스하려고 하지만 역할에 작업을 허용하는 정책이 없기 때문에 액세스할 수 없는 애플리케이션을 보여주는 아키텍처입니다.

|수강생용 노트

다른 AWS 서비스를 호출해야 하는 Pod에서 애플리케이션을 실행한다면 어떻게 하습니까? 기본적으로 노드에서 실행 중인 Pod에는 인스턴스에 연결된 역할에서 정의하는 권한이 부여됩니다. Amazon EC2 인스턴스는 EC2 인스턴스에서 인스턴스 메타데이터 서비스(IMDS)를 실행합니다. 이 서비스는 IAM 자격 증명을 포함한 인스턴스 관련 정보를 메타데이터 엔드포인트에서 이용할 수 있게 합니다. 이 엔드포인트에는 로컬 주소 169.254.169.254의 인스턴스에서 실행 중인 애플리케이션이 접근할 수 있습니다.

이 예에서 노드에는 AWS IAM의 eksNodeRole 역할이 할당됩니다. 이 역할에는 Amazon Simple Storage Service(Amazon S3)에 대한 액세스를 허용하는 IAM 정책이 없으며, 따라서 AWS API는 Pod의 S3:GetObject 실행 시도를 거부합니다. 이 Pod에 필요한 권한을 어떻게 얻을 수 있습니까?

노드의 역할에 필요한 권한을 추가하는 방법이 있지만 이 방법은 보안과 확장성에 문제가 있습니다. 이 Pod가 실행될 수 있는 클러스터의 모든 노드에 이러한 추가 권한을 추가해야 합니다. 클러스터가 증가하면 모든 노드에 클러스터가 호스트할 만한 모든 Pod에 대한 권한이 필요합니다. 이렇게 되면 관리하기가 점점 더 어려워지며 최소 권한의 원칙도 위반하게 됩니다. 따라서 Kubernetes Service Account를 사용하여 Pod에 AWS 권한을 직접 할당하는 것이 좋습니다.

Service Account의 IAM 역할(IRSA)

- IAM 역할을 RBAC Service Account와 간접적으로 연결
- RBAC에서 만든 토큰을 디지털로 서명하는 OpenID Connect (OIDC) 공급자 사용
- IAM의 권한 부여 기능 처리



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

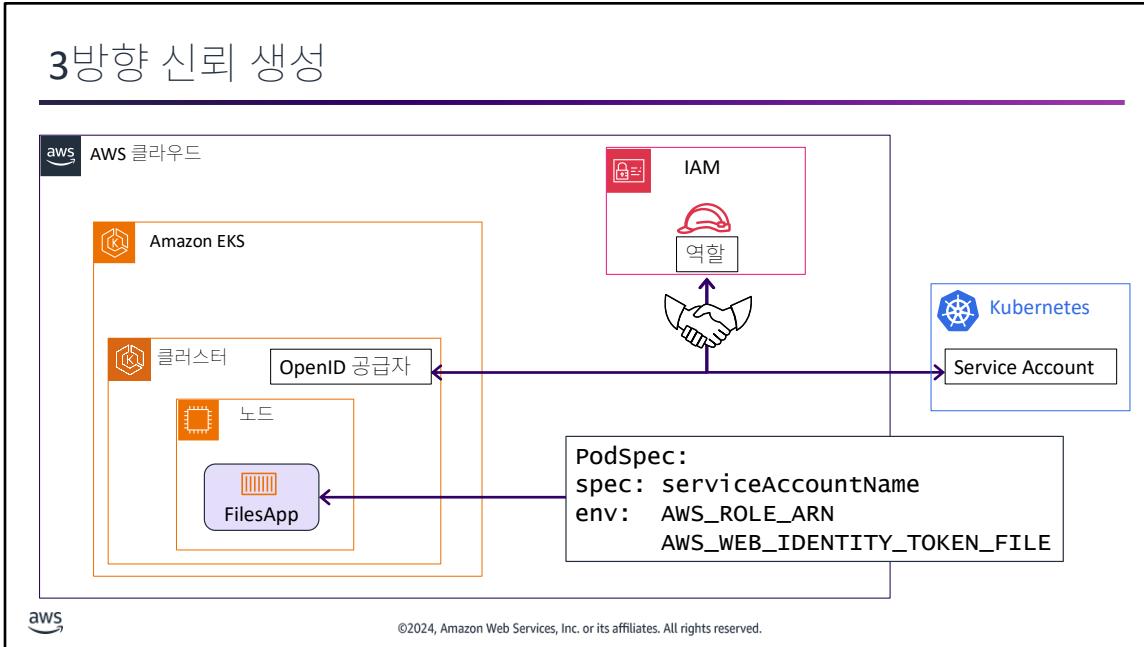
| 수강생용 노트

AWS 명령을 실행할 권한을 Pod에 부여하는 가장 쉬운 방법은 Pod에 IAM 역할을 할당하는 것입니다. 하지만 현재는 Pod에 IAM 역할을 할당할 수 없습니다. 문제는 인증입니다. 기본적으로 IAM은 Kubernetes Service Account와 연결된 토큰을 유효한 자격 증명으로 허용하지 않습니다.

이를 해결하기 위해서는 OpenID Connect(OIDC) 공급자를 사용하여 토큰을 디지털 서명하는 Service Account의 IAM 역할(IRSA)을 사용해야 합니다. RBAC는 Service Account의 토큰을 생성하기 때문에 IAM은 이를 페더레이션 사용자로 수락합니다. 토큰을 사용하면 Kubernetes Service Account에 IAM 역할을 할당할 수 있습니다.

IRSA를 사용하여 Kubernetes Service에 IAM 역할을 할당하면 일반적인 방법으로 권한 부여를 관리할 수 있습니다(IAM 역할에 IAM 정책 할당).

3방향 신뢰 생성



~ALT text

~아키텍처: Kubernetes Service Account와 연결된 OpenID 공급자를 보여주는 아키텍처, 노트에 자세히 설명되어 있습니다.

~Pod: Kubernetes Service Account의 이름이 포함된 Pod의 PodSpec을 보여주는 애니메이션입니다.

|수강생용 노트

IRSA를 사용하면 Pod가 AWS 명령을 실행하도록 허용하는 권한을 가진 IAM 역할을 Kubernetes Service Account가 담당할 수 있습니다. 사용자는 신뢰 관계가 있는 IAM 역할을 생성합니다. 이 관계는 Amazon EKS 클러스터에서 호스팅하는 OIDC Connect(OIDC) 공급자 및 Kubernetes Service Account로 범위가 지정됩니다. 신뢰 범위를 단일 Service Account가 아닌 Namespace의 모든 Service Account로 지정할 수 있습니다. IAM 역할에 연결된 IAM 정책은 Pod에 부여하는 권한을 연결된 Kubernetes Service Account를 사용하여 정의합니다.

AWS IAM Authenticator를 사용할 때처럼 IRSA는 토큰 기반 인증을 사용합니다. IAM은 OIDC를 사용하는 페더레이션 사용자를 지원합니다. Kubernetes는 Service Account에 대한 OIDC JSON Web Token(JWT)을 생성합니다. Amazon EKS는 이러한 JSON 웹 토큰에 대한 서명 키를 포함하는 각 클러스터에 퍼블릭 OIDC 검색 엔드포인트를 호스팅합니다. 이렇게 하면 IAM은 Kubernetes가 발행한 OIDC 토큰을 검증하고 수락할 수 있습니다. IAM은 여전히 인증을 담당하지만, 이 경우 인증되는 자격 증명(Service Account)은 Kubernetes에서 관리하는 페더레이션 사용자입니다. 또한 IAM은 Service Account에 AWS 서비스와 상호 작용할 수 있는 임시 역할 자격 증명을 부여하는 방식으로 권한을 부여합니다.

IRSA를 사용할 각 Pod에는 PodSpec에서 사용할 Kubernetes Service Account의 이름과 다음 두 가지 환경 변수가 있어야 합니다.

1. Service Account를 통해 담당할 역할의 Amazon Resource Name
2. 토큰 파일 경로(/var/run/secrets/eks.amazonaws.com/serviceaccount/token)

클러스터의 Amazon EKS Pod Identity Webhook(<https://github.com/aws/amazon-eks-pod-identity-webhook>)은 IRS를 사용하는 것으로 주석이 달린 Service Account와 연결된 Pod를 모니터링합니다. Webhook은 여기에 환경 변수를 적용합니다. 클러스터는 환경 변수 및 토큰 파일 탑재를 구성하기 위해 변종 Webhook을 사용하지 않아도 됩니다. 이러한 환경 변수를 수동으로 추가하도록 Pod를 구성할 수 있습니다.

자세한 내용은 Amazon EKS 사용 설명서의 ‘Service Account에 대한 IAM 역할’(<https://docs.aws.amazon.com/eks/latest/userguide/iam-roles-for-service-accounts.html>)을 참조하십시오.

Service Account의 IAM 역할(IRSA) 구성

- 클러스터의 IAM OIDC 공급자를 생성합니다.

```
$ eksctl utils associate-iam-oidc-provider --cluster dev-cluster --approve --region us-west-2
```

- IAM 역할 및 이에 해당하는 Kubernetes Service Account를 생성합니다.

```
$ eksctl create iamserviceaccount --name aws-s3-read --namespace default --cluster prod --attach-policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess --approve --region us-west-2
[i] eksctl version 0.22.0-rc.0
[i] using region us-west-2
[i] 1 iamserviceaccount (default/aws-s3-read) was included (based on the include/exclude rules)
[!] serviceaccounts that exists in Kubernetes will be excluded, use --override-existing-serviceaccounts to override
[i] 1 task: { 2 sequential sub-tasks: { create IAM role for serviceaccount "default/aws-s3-read", create serviceaccount "default/aws-s3-read" } }
[i] building iamserviceaccount stack "eksctl-prod-addon-iamserviceaccount-default-aws-s3-read"
[i] deploying stack "eksctl-prod-addon-iamserviceaccount-default-aws-s3-read"
[i] created serviceaccount "default/aws-s3-read"
```



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

IAM은 OpenID Connect(OIDC)를 사용하는 페더레이션 사용자를 지원합니다. 이 기능을 사용하면 지원되는 자격 증명 공급자를 이용해 AWS API 호출을 인증하고 유효한 OIDC JSON 웹 토큰을 수신할 수 있습니다. 엔터티는 이 토큰을 AWS Security Token Service(AWS STS) AssumeRolewithWebIdentity API 작업에 전달하여 임시 IAM 역할 자격 증명을 받을 수 있습니다. 이 자격 증명은 Amazon S3 또는 Amazon DynamoDB와 같은 모든 AWS 서비스와 상호 작용하는 데 사용됩니다.

Kubernetes Service Account는 Pod가 자동 탑재 토큰을 사용하여 Kubernetes API로 인증할 수 있게 합니다. Kubernetes 버전 1.12 이후부터는

ProjectedServiceAccountToken이라는 기능을 지원합니다. 이 기능은 Service Account 자격 증명을 포함하고 구성 가능한 대상을 지원하는 OIDC JSON 웹 토큰입니다.

Amazon EKS는 **ProjectedServiceAccountToken** JSON 웹 토큰의 서명 키를 포함하는 각 클러스터에 퍼블릭 OIDC 검색 엔드포인트를 호스팅합니다. 이렇게 하면 IAM은 Kubernetes에서 발행한 OIDC 토큰을 검증하고 수락할 수 있습니다.

IRSA 요약 보기

Role ARN: arn:aws:iam::1234567890123:role/eksctl-prod-addon-iamserviceaccount-default-Role1-A5874PD60FWW

Role description: Edit

Instance Profile ARNs: /

Path: /

Creation time: 2020-07-27 18:58 EDT

Last activity: Not accessed in the tracking period

Maximum session duration: 1 hour Edit

Permissions | Trust relationships | Tags (4) | Access Advisor | Revoke sessions

You can view the trusted entities that can assume the role and the access conditions for the role. Show policy document

Edit trust relationship

Trusted entities

The following trusted entities can assume this role.

Trusted entities

arn:aws:iam::1234567890123:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/31BBC1669FF14DF65EC04E04688898

Conditions

The following conditions define how and when trusted entities can assume the role.

Condition	Key	Value
StringEquals	oidc.eks.us-west-2.amazonaws.com/id/31BBC1669FF14DF65EC04E04688898;aud	sts.amazonaws.com
StringEquals	oidc.eks.us-west-2.amazonaws.com/id/31BBC1669FF14DF65EC04E04688898;sub	system:serviceaccount:default:aws-s3-read

~ALT text

~IAM 콘솔에 역할의 세부 정보가 표시되어 있습니다. 신뢰 관계 탭이 선택되어 신뢰할 수 있는 엔티티 및 조건이 강조 표시되어 있습니다.

|수강생용 노트

이 예에서는 IAM에서 구성된 항목을 확인할 수 있습니다. **AmazonS3ReadOnlyAccess** 정책이 할당된 역할이 생성되었습니다. Amazon EKS 클러스터에서 호스팅하는 OIDC 공급자는 신뢰할 수 있는 엔터티로 구성됩니다. 다음 두 가지 조건이 구성되었습니다.

1. Service Account는 AWS STS AssumeRoleWithWebIdentity API 작업(Kubernetes API 서버에서 생성한 OIDC JSON 웹 토큰 사용)을 사용해야 합니다.
2. 지정된 Service Account(기본값 : aws-s3-read)만 이 역할을 담당할 수 있습니다.

예: IRSA용 Service Account

```
$ kubectl get serviceaccount/aws-s3-read -o yaml

apiVersion: v1
kind: ServiceAccount
metadata:
  annotations:
    eks.amazonaws.com/role-arn: arn:aws:iam::123456789012:role/eksctl-prod-addon-
    iamServiceAccount-default-role1-A5874PD60Fww
    creationTimestamp: "2023-07-27T22:58:59Z"
  name: aws-s3-read
  namespace: default
  resourceVersion: "8361407"
  selfLink: /api/v1/namespaces/default/serviceaccounts/aws-s3-read
  uid: 23ea075b-9b8e-456d-9416-6e862901b801
  secrets:
  - name: aws-s3-read-token-99zq2
```

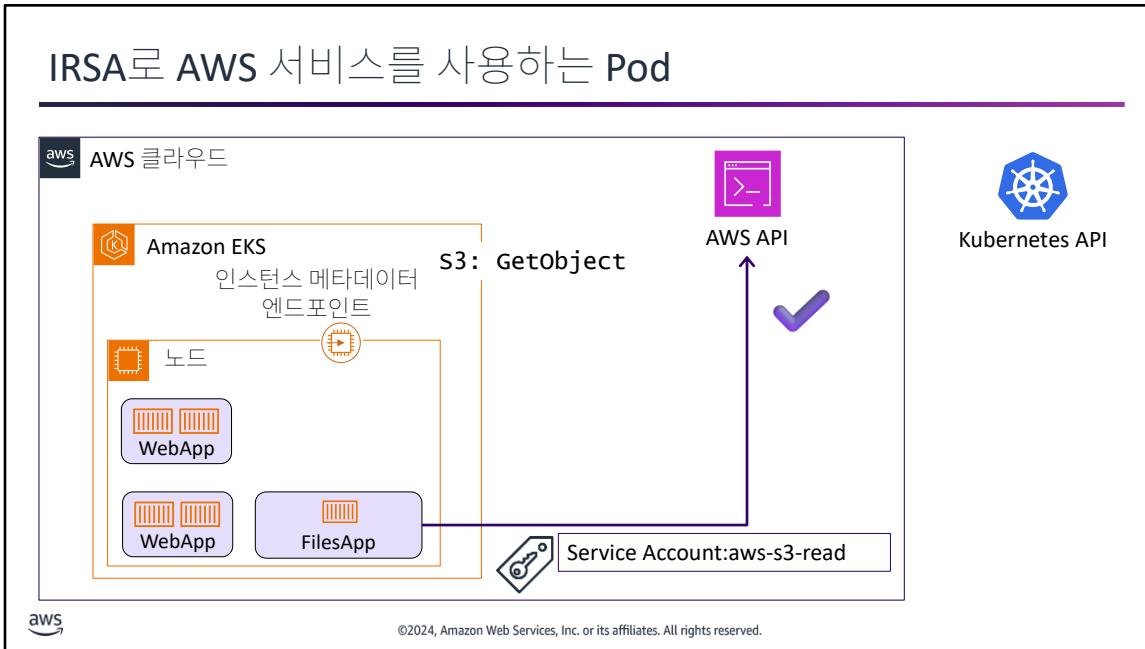


©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

이 예에서는 **eksctl**에서 생성한 **Kubernetes Service Account**를 확인할 수 있습니다.
이 **Service Account**를 사용하는 **Pod**에서 담당할 **IAM** 역할의 **Amazon Resource Name**을 사용하여 **PodSpec**에 주석을 추가해야 합니다.

IRSA로 AWS 서비스를 사용하는 Pod



~ALT text

~FilesApp Pod는 IMDS 역할 대신 IRSA Service Account를 사용하여 S3에 성공적으로 연결됩니다.

|수강생용 노트

IRSA가 성공적으로 구성되면 Pod는 Kubernetes Service Account를 사용하여 AWS 서비스와 상호 작용하는 데 필요한 권한을 가진 IAM 역할을 담당할 수 있습니다.

Pod의 IRSA를 구현하면 Pod의 컨테이너에는 Service Account 및 노드의 IAM 역할에 할당된 모든 권한이 포함됩니다. 클러스터 전역의 모든 Pod에 대해 IRSA를 구성하는 경우, Pod 내 컨테이너가 노드의 IAM 역할에 할당된 권한을 사용하지 못하게 해야 할 수도 있습니다.

그러나 노드 IAM 역할에는 Pod가 작동하는 데 필요한 특정 키 권한이 있을지도 모른다는 사실을 명심해야 합니다. Service Account IAM 역할의 범위를 적절하게 지정하여 Pod에 필요한 모든 권한을 부여해야 합니다. 예를 들어 노드 IAM 역할에 Amazon Elastic Container Registry(Amazon ECR)에서 컨테이너 이미지를 가져오는 권한을 할당해야 합니다. 이러한 권한이 할당되지 않은 Pod는 Amazon ECR에서 컨테이너 이미지를 가져올 수 없습니다.

Amazon EC2 인스턴스 프로파일 자격 증명 액세스 제한에 대한 자세한 내용은 **Amazon EKS** 사용 설명서의 ‘Amazon EC2 인스턴스 프로파일 자격 증명에 대한 액세스 제한’(<https://docs.aws.amazon.com/eks/latest/userguide/restrict-ec2-credential-access.html>)을 참조하십시오.

활동



권한 제어

이 활동에서는 다음을 수행합니다.

- 필요한 최소 권한을 할당하기 위해 구성해야 하는 객체를 파악합니다.

©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

활동: 제시된 각 시나리오에서 필요한 최소 권한을 할당하기 위해 구성해야 하는 객체를 식별합니다.

권한 제어 질문 1

관리자는 **dev** 네임스페이스의 Pod를 관리해야 합니다.
(2개 선택)

보기	응답
A	인증을 위한 IAM 내 자격 증명(사용자, 그룹, 역할)
B	올바른 권한이 있는 자격 증명에 연결된 IAM 내 정책
C	올바른 권한이 있는 RBAC 내 그룹에 매핑
D	올바른 권한이 있는 그룹 구성원인 RBAC 내 Service Account
E	IAM 역할, RBAC Service Account, OIDC 공급자 간 신뢰
F	RBAC system:masters 그룹에 매핑

권한 제어 질문 1: 정답은 A와 C입니다.

관리자는 **dev** 네임스페이스의 Pod를 관리해야 합니다.
(2개 선택)

보기	응답
A 정답	인증을 위한 IAM 내 자격 증명(사용자, 그룹, 역할)
B	올바른 권한이 있는 자격 증명에 연결된 IAM 내 정책
C 정답	올바른 권한이 있는 RBAC 내 그룹에 매핑
D	올바른 권한이 있는 그룹 구성원인 RBAC 내 Service Account
E	IAM 역할, RBAC Service Account, OIDC 공급자 간 신뢰
F	RBAC system:masters 그룹에 매핑

 ©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

|수강생용 노트
A와 C는 필수입니다.

권한 제어 질문 2

Pod에서 실행 중인 애플리케이션은 다른 Pod의 속성을 봐야 하는 경우

보기	응답
A	인증을 위한 AWS IAM 내 자격 증명(사용자, 그룹, 역할)
B	올바른 권한이 있는 자격 증명에 연결된 IAM 내 정책
C	올바른 권한이 있는 RBAC 내 그룹에 매핑
D	올바른 권한이 있는 그룹 구성원인 RBAC 내 Service Account
E	IAM 역할, RBAC Service Account, OIDC 공급자 간 신뢰
F	RBAC system:masters 그룹에 매핑

권한 제어 질문 2: 정답은 D입니다.

Pod에서 실행 중인
애플리케이션은 다른 Pod의
속성을 봐야 하는 경우

보기	응답
A	인증을 위한 AWS IAM 내 자격 증명(사용자, 그룹, 역할)
B	올바른 권한이 있는 자격 증명에 연결된 IAM 내 정책
C	올바른 권한이 있는 RBAC 내 그룹에 매핑
D	올바른 권한이 있는 그룹 구성원인 RBAC 내 Service Account
E	IAM 역할, RBAC Service Account, OIDC 공급자 간 신뢰
F	RBAC system:masters 그룹에 매핑

 ©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트
D는 필수입니다.

권한 제어 질문 3

관리자가 Amazon EKS
클러스터에서 노드를 생성하고
관리해야 하는 경우 (3개 선택)

보기	응답
A	인증을 위한 AWS IAM 내 자격 증명(사용자, 그룹, 역할)
B	올바른 권한이 있는 자격 증명에 연결된 IAM 내 정책
C	올바른 권한이 있는 RBAC 내 그룹에 매핑
D	올바른 권한이 있는 그룹 구성원인 RBAC 내 Service Account
E	IAM 역할, RBAC Service Account, OIDC 공급자 간 신뢰
F	RBAC system:masters 그룹에 매핑

권한 제어 질문 3: 정답은 A, B, C입니다.

관리자가 Amazon EKS

클러스터에서 노드를 생성하고
관리해야 하는 경우 (3개 선택)

보기 응답

A

정답

인증을 위한 AWS IAM 내 자격 증명(사용자, 그룹, 역할)

B

정답

올바른 권한이 있는 자격 증명에 연결된 IAM 내 정책

C

정답

올바른 권한이 있는 RBAC 내 그룹에 매핑

D

정답

올바른 권한이 있는 그룹 구성원인 RBAC 내 Service Account

E

정답

IAM 역할, RBAC Service Account, OIDC 공급자 간 신뢰

F

정답

RBAC system:masters 그룹에 매핑



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

|수강생용 노트
A, B, C는 필수입니다.

권한 제어 질문 4

Pod에서 실행하는 애플리케이션이 다른 **Pod**의 데이터를 수집하고, 데이터를 분석하고, 결과를 **Amazon S3**에 저장해야 하는 경우
(4개 선택)

보기	응답
A	인증을 위한 AWS IAM 내 자격 증명(사용자, 그룹, 역할)
B	올바른 권한이 있는 자격 증명에 연결된 IAM 내 정책
C	올바른 권한이 있는 RBAC 내 그룹에 매핑
D	올바른 권한이 있는 그룹 구성원인 RBAC 내 Service Account
E	IAM 역할, RBAC Service Account, OIDC 공급자 간 신뢰
F	RBAC system:masters 그룹에 매핑

권한 제어 질문 4: 정답은 A, B, D, E입니다.

Pod에서 실행하는 애플리케이션이 다른 Pod의 데이터를 수집하고, 데이터를 분석하고, 결과를 Amazon S3에 저장해야 하는 경우 (4개 선택)

보기	응답
A 정답	인증을 위한 AWS IAM 내 자격 증명(사용자, 그룹, 역할)
B 정답	올바른 권한이 있는 자격 증명에 연결된 IAM 내 정책
C	올바른 권한이 있는 RBAC 내 그룹에 매핑
D 정답	올바른 권한이 있는 그룹 구성원인 RBAC 내 Service Account
E 정답	IAM 역할, RBAC Service Account, OIDC 공급자 간 신뢰
F	RBAC system:masters 그룹에 매핑

 ©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

|수강생용 노트
A, B, D, E는 필수입니다.

모듈 요약



이 모듈에서 학습한 내용:

- 클러스터에 대한 액세스를 보호하려면 IAM과 **Kubernetes RBAC**가 모두 필요합니다.
- AWS API와 Kubernetes API에 대한 액세스는 **Service Account**를 사용하여 관리합니다.

©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 강사용 노트

| 수강생용 노트



Running Containers on Amazon EKS

실습 6: Capstone 실습



©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 수강생용 노트

이 **120분짜리 capstone** 실습에서는 이 과정에서 배운 모든 내용을 활용해야 합니다.

실습 6



aws

목표:

- 새 버전의 애플리케이션을 배포합니다. 버전 1은 1-Pod 앱입니다. 버전 2는 전용 스토리지가 있는 다중 서비스입니다.

과제:

- 배포 준비
- 앱 영구 스토리지 구성 – 없음 또는 인스턴스 스토어에서 EFS로 전환
- 애플리케이션 배포

©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Running Containers on Amazon EKS

감사합니다.



수정 사항이나 피드백 또는 기타 질문이 있으십니까?

<https://support.aws.amazon.com/#/contacts/aws-training>에서 문의해 주십시오.
모든 상표는 해당 소유자의 자산입니다.

©2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

| 강사용 노트

| 수강생용 노트