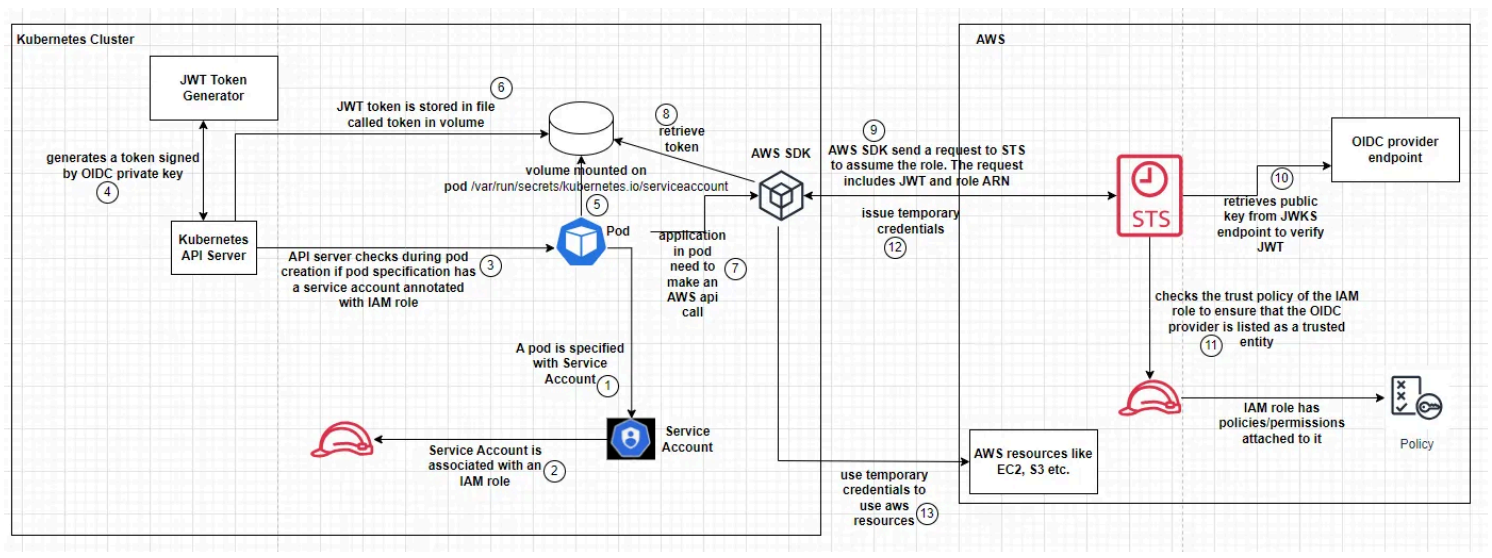
Implementing and Understanding IAM Roles for Service Accounts in AWS EKS

Anil Goyal · Follow
6 min read · Mar 4, 2024

2 1

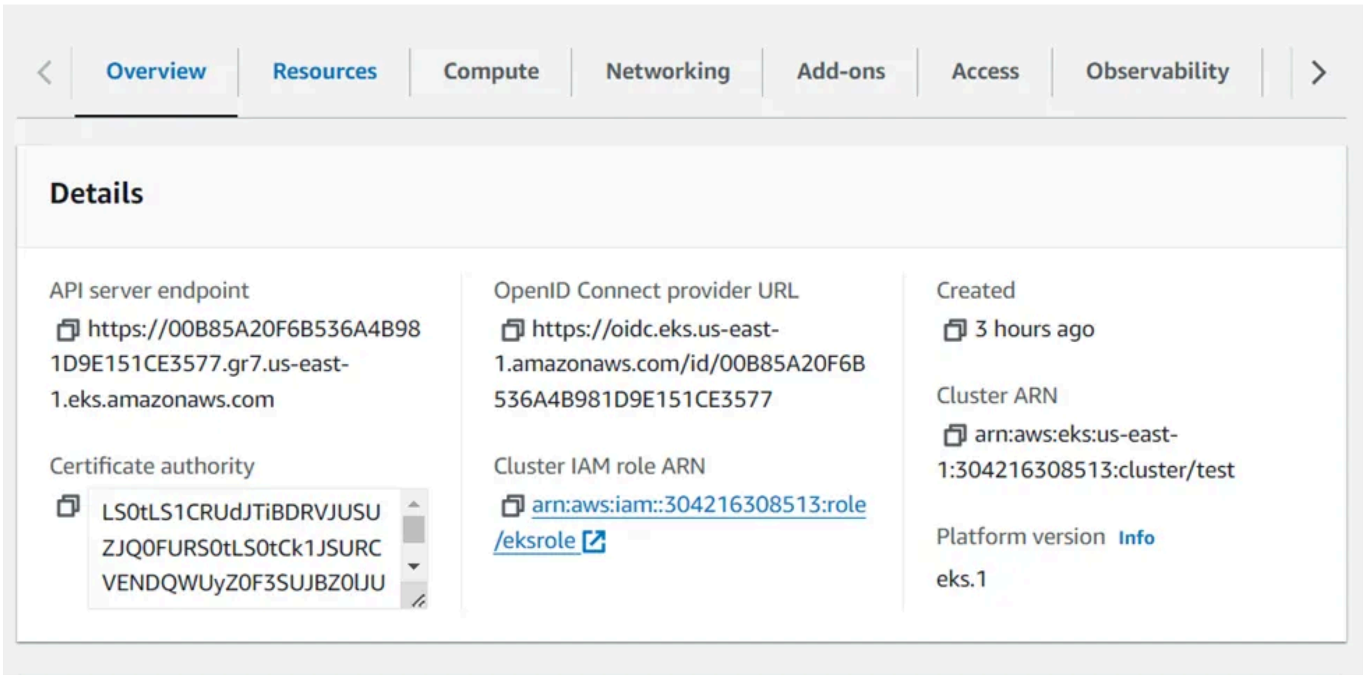
When it comes to managing access control within AWS's Elastic Kubernetes Service (EKS), IAM Roles for Service Accounts (IRSA) plays a crucial role. In this post, we will not only understand what IRSA is and how it operates but also walk through practical examples to demonstrate its implementation.


Kubernetes Cluster
JWT Token Generator
generates a token signed by OIDC private key 4
Kubernetes API Server
6 JWT token is stored in file called token in volume
volume mounted on pod /var/run/secrets/kubernetes.io/serviceaccount 5
8 retrieve token
API server checks during pod creation if pod specification has a service account annotated with IAM role 3
Pod
application in pod need to make an AWS api call 7
AWS SDK
A pod is specified with Service Account 1
Service Account
Service Account is associated with an IAM role 2
AWS
9 AWS SDK send a request to STS to assume the role. The request includes JWT and role ARN
issue temporary credentials 12
STS
10 retrieves public key from JWKS endpoint to verify JWT
OIDC provider endpoint
checks the trust policy of the IAM role to ensure that the OIDC provider is listed as a trusted entity 11
IAM role has policies/permissions attached to it
Policy
AWS resources like EC2, S3 etc.
use temporary credentials to use aws resources 13

Create an Amazon EKS Cluster

You can create an Amazon EKS cluster using the AWS Management Console, AWS CLI, or AWS SDKs. See this link for detail https://docs.aws.amazon.com/eks/latest/userguide/create-cluster.html

After creation of cluster, it has an OpenID Connect (OIDC) provider URL associated with it as shown below:



Details

API server endpoint
https://00B85A20F6B536A4B981D9E151CE3577.gr7.us-east-1.eks.amazonaws.com

Certificate authority
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURCVENDQWUyZ0F3SUJBZ0lJU

OpenID Connect provider URL
https://oidc.eks.us-east-1.amazonaws.com/id/00B85A20F6B536A4B981D9E151CE3577

Cluster IAM role ARN
arn:aws:iam::304216308513:role/eksrole

Created
3 hours ago

Cluster ARN
arn:aws:eks:us-east-1:304216308513:cluster/test

Platform version Info
eks.1

Now To use AWS Identity and Access Management (IAM) roles for service accounts, an IAM OIDC provider must exist for your cluster's OIDC issuer URL.

After the EKS cluster is created, you can find the OIDC provider URL in the cluster's details. The URL is in the format *oidc.eks.<region>.amazonaws.com/id/<OIDC_ID>*, where <region> is the AWS region where the cluster is located and <OIDC_ID> is a unique identifier for the OIDC provider. Once you have the OIDC provider URL, you can create an identity provider in AWS IAM that represents the same OIDC provider. Here's how you can do it in the AWS Management Console.

Identity Provider in AWS IAM

1. Open the AWS Management Console and navigate to the IAM service.

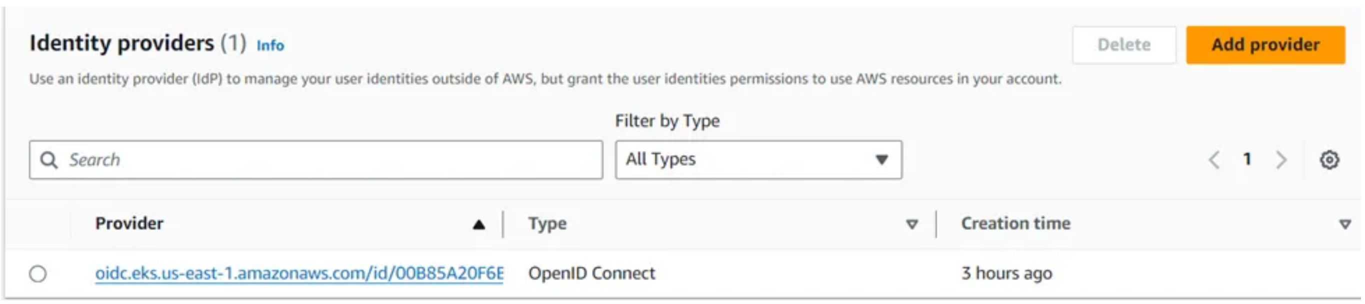
2. In the IAM dashboard, click on "Identity providers" in the left navigation

4. For the "Provider Type", select "OpenID Connect".

5. For the "Provider URL", enter the OIDC provider URL of your EKS cluster.

6. For the "Audience", enter "sts.amazonaws.com".

7. Click "Next Step", review the details, and then click "Create".



Identity providers (1) Info

Use an identity provider (IdP) to manage your user identities outside of AWS, but grant the user identities permissions to use AWS resources in your account.

Filter by Type: All Types

Provider	Type	Creation time
oidc.eks.us-east-1.amazonaws.com/id/00B85A20F6E	OpenID Connect	3 hours ago

You can find the steps here as well https://docs.aws.amazon.com/eks/latest/userguide/enable-iam-roles-for-service-accounts.html

Now, the identity provider in IAM and the OIDC provider in EKS are essentially the same entity. They both represent the same OIDC issuer, and they use the same public and private keys to sign and verify JWTs. The role of the OIDC provider in IAM is to allow AWS STS to trust JWTs that are issued by the OIDC provider in EKS.

Create an IAM Role for a Kubernetes Service Account

Create an IAM role that your Kubernetes service account can assume. You can do this in the AWS Management Console:

1. Go to the IAM service.

2. Click on "Roles" in the left navigation pane.

3. Click on "Create role".

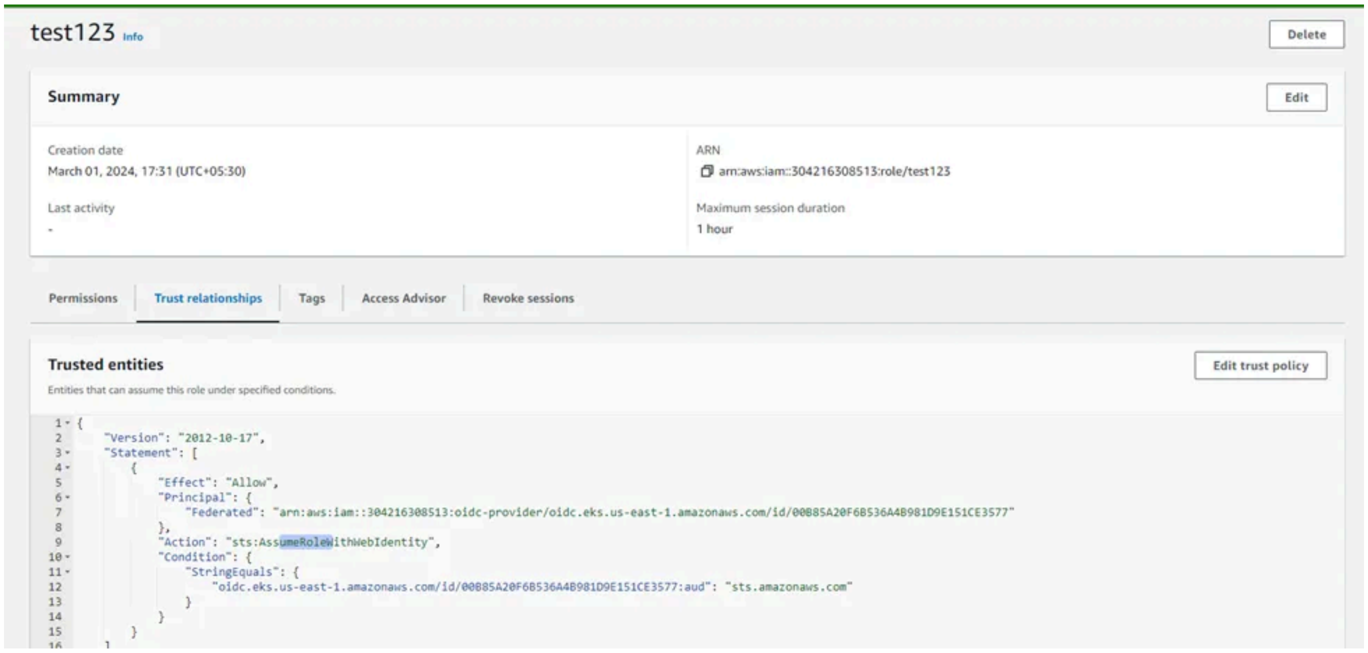
4. Select "Web identity" as the type of trusted entity.

5. In the "Identity provider" dropdown, select the identity provider that corresponds to your EKS cluster.

6. In the "Audience" field, enter "sts.amazonaws.com".

7. Attach the necessary policies to the role and then give your role a name and a description.

8. Click “Create role”.


test123 Info

Delete

Summary

Edit

Creation date
March 01, 2024, 17:31 (UTC+05:30)

Last activity
-

ARN
arn:aws:iam::304216308513:role/test123

Maximum session duration
1 hour

Trusted entities

Entities that can assume this role under specified conditions.

Edit trust policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::304216308513:oidc-provider/oidc.eks.us-east-1.amazonaws.com/id/00B85A20F6B536A4B981D9E151CE3577"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.us-east-1.amazonaws.com/id/00B85A20F6B536A4B981D9E151CE3577:aud": "sts.amazonaws.com"
                }
            }
        }
    ]
```

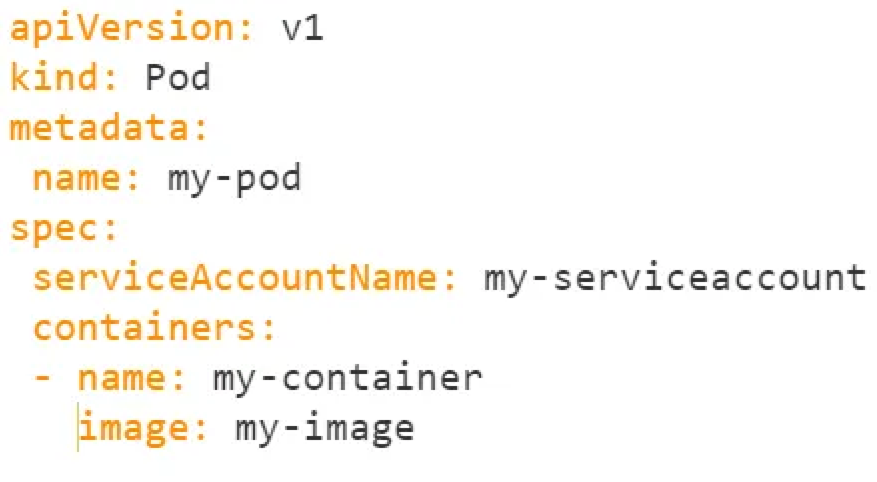
Associate the IAM Role with a Kubernetes Service Account

You can associate the IAM role with a Kubernetes service account by adding an annotation to the service account in your Kubernetes manifest file:

```
apiVersion: v1
kind: ServiceAccount
metadata:
 name: my-serviceaccount
 annotations:
  eks.amazonaws.com/role-arn: 'arn:aws:iam::123456789012:role/my-role'
```

Create a Pod that Uses the Service Account

Create a pod that uses the service account:



```yaml
apiVersion: v1
kind: Pod
metadata:
 name: my-pod
spec:
 serviceAccountName: my-serviceaccount
 containers:
 - name: my-container
   image: my-image
```

Use the AWS SDK in Your Application

In your application, use the AWS SDK to make calls to AWS services. The SDK will automatically use the credentials provided by the DefaultCredentialsProvider, which includes the WebIdentityTokenFileCredentialsProvider that uses the token file provided by the Kubernetes service account. Here's an example in Java:

```java
import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.services.s3.S3Client;

public class Main {
    public static void main(String[] args) {
        DefaultCredentialsProvider credentialsProvider = DefaultCredentialsProvi

        S3Client s3 = S3Client.builder()
                .credentialsProvider(credentialsProvider)
                .build();

        // Use the S3 client
    }
}
```

When a pod is launched in a Kubernetes cluster with a service account that is associated with an IAM role, the following steps occur:

1. **Token Generation and Injection:** When the Kubernetes API server creates a pod, it checks pod specification to see if any service account is specified or not. If service account is specified, it then check if service account is associated with an IAM role or not. If it is, the API server generates a JSON Web Token (JWT) that represents the identity of the service account. This JWT includes:

· **Issuer:** This is OIDC provider url.

· **Subject:** It is unique identifier for the service account.

· **Audience:** Intended recipient of the token i.e., "sts.amazonaws.com"

· **Expiration Time:** expiration time of token.

· **Issued At:** when token was issued.

· **Kubernetes Namespace:** namespace of service account.

· **Service Account Name:** name of service account

The JWT is then signed by the Kubernetes cluster's OIDC provider's private key.

2. **Token Mounting:** The Kubernetes API server injects the JWT into the pod. It does this by creating a special type of volume called a ServiceAccount volume. This volume is automatically mounted into the pod at the path

/var/run/secrets/kubernetes.io/serviceaccount/. The JWT is stored in a file in this volume called token.

3. **Token Retrieval by the AWS SDK:** When your application starts in the pod, it uses the AWS SDK to make calls to AWS services. The AWS SDK includes a credentials provider called WebIdentityTokenFileCredentialsProvider. This credentials provider is designed to work with Kubernetes service accounts. It reads the JWT from the token file in the ServiceAccount volume.

4. **Token Verification and Role Assumption by AWS STS:** When your application makes a call to an AWS service, the AWS SDK sends a request to AWS STS to assume the IAM role that is associated with the service account. This request includes the JWT. AWS STS uses the issuer url in the JWT to match the incoming JWT to the correct OIDC identity provider in IAM. After that STS retrieves the provider public key from the provider JWKS (JSON Web Key Set) endpoint. It then uses these public keys to verify the signature of the JWT. If the JWT is valid, AWS STS check the trust policy of the IAM role to verify whether the OIDC provider that issue the JWT is a trusted entity for that IAM role or not. If both checks pass, it assumes the IAM role and returns temporary security credentials that are associated with that IAM role to the AWS SDK. These credentials (include an access key ID, a secret access key and a session token) are used to authenticate the request to the AWS service. The AWS service use the access key ID to look up the IAM role and its permissions i.e. whether the role has necessary permissions to perform the requested operation or not and based on that process or denies the request.

```
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleWithWebIdentityReques

public class Main {
    public static void main(String[] args) {
        StsClient stsClient = StsClient.create();

        AssumeRoleWithWebIdentityRequest request = AssumeRoleWithWebIdentityRequ
                .roleArn("arn:aws:iam::123456789012:role/my-role")  // Replace w
                .roleSessionName("my-role-session")
                .webIdentityToken("my-jwt")  // Replace with your JWT
                .build();

        stsClient.assumeRoleWithWebIdentity(request);
    }
}
```

In conclusion, IAM Roles for Service Accounts (IRSA) is a feature in AWS EKS that allows Kubernetes service accounts to be associated with IAM roles. This provides a secure and efficient way to give your applications running on EKS the permissions they need to call other AWS services.

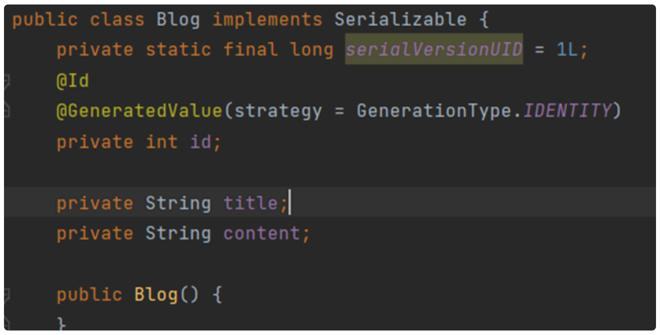
Kubernetes Serviceaccount Iam Roles Aws Sts Jwt Token Aws Sdk

Written by Anil Goyal

23 Followers

Follow

More from Anil Goyal



```
public class Blog implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    private String title;
    private String content;

    public Blog() {
    }
```

Anil Goyal

Mastering Caching in Spring Boot with MySQL and Redis

Many a time, we all come to a phase when our application does not perform well as it is...

5 min read · Jan 29, 2024

4

CodeWhisperer-HelloWorld.py
Untitled-1
CodeWhisperer-HelloWorld.py > my_function

```
def my_function(param1, param2):
    return param1 + param2
```

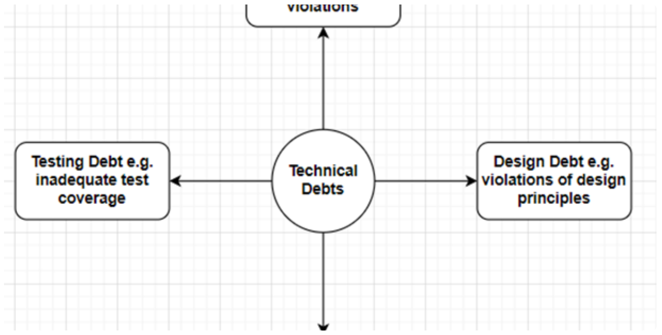
Anil Goyal

Github Copilot vs Amazon CodeWhisperer

Both Github Copilot and Amazon Code Whisperer are AI coding assistant tool i.e., an...

5 min read · Jan 18, 2024

16


violations
Testing Debt e.g. inadequate test coverage
Technical Debts
Design Debt e.g. violations of design principles

Anil Goyal

Refactoring for Design Smells

Up to 64% of Software defects can be tracked back to errors in software design in enterpri...

9 min read · Aug 3, 2023

5

Anil Goyal

Mastering the Sliding Window Algorithm with Practical Example...

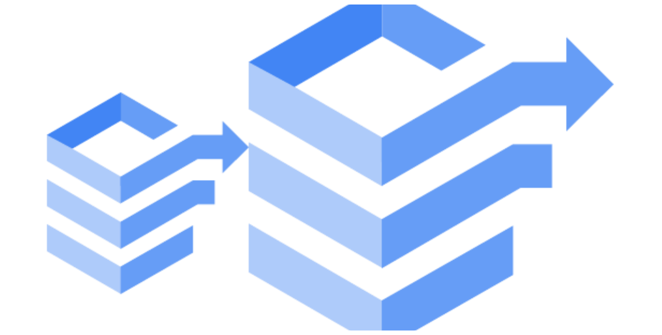
Sliding Window Algorithm

11 min read · Apr 23, 2024

6

See all from Anil Goyal

Recommended from Medium



Saloni Singh

A Step-by-Step Guide to successful AWS Data Migration Project: A...

Today, everyone is looking for a hands-on experience over Cloud Migration Projects,...

31 min read · Apr 28, 2024

20

Andreas Leicher in kotaicode

AWS IAM Roles for Kubernetes Pods in EKS

Amazon Elastic Kubernetes Service (EKS) simplifies the deployment and management...

5 min read · Jan 19, 2024

16

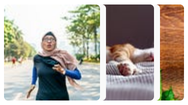
Lists

Staff Picks

634 stories · 948 saves

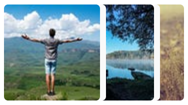
Stories to Help You Level-Up at Work

19 stories · 600 saves



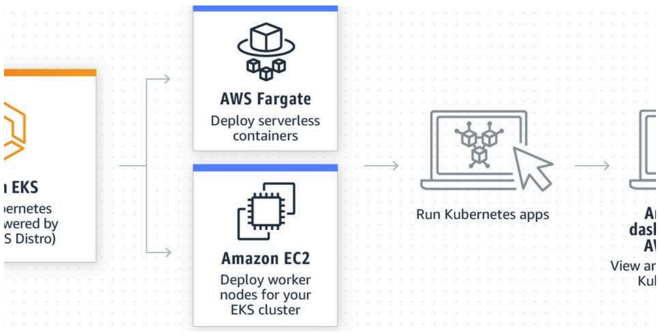
Self-Improvement 101

20 stories · 1736 saves



Productivity 101

20 stories · 1611 saves


AWS Fargate
Deploy serverless containers
Amazon EC2
Deploy worker nodes for your EKS cluster
Run Kubernetes apps
EKS

Foued Jbali

How to create an EKS Cluster in AWS ?

Amazon Elastic Kubernetes Service (Amazon EKS) is a fully-managed service offered by…

4 min read · Dec 14, 2023

5

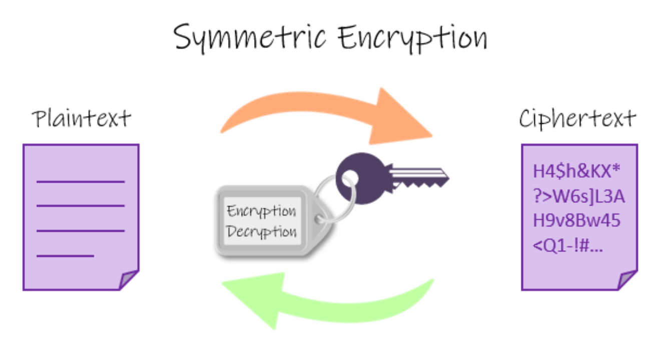
Muhammed Suhail in AWS in Plain English

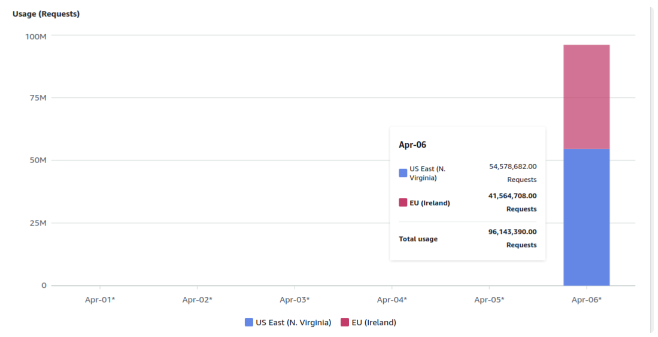
Deploying Application on Amazon EKS

Kubernetes is an open-source container orchestration platform designed to automat…

13 min read · Dec 9, 2023

18 2





Abdie Mohamed

AWS Cloud Security Lab: Using KMS and OpenSSL for Encryption

Hi there! If you're an organization or a practitioner looking to migrate their…

9 min read · Nov 24, 2023

26

Maciej Pocwierz

How an empty S3 bucket can make your AWS bill explode

Imagine you create an empty, private AWS S3 bucket in a region of your preference. What…

4 min read · Apr 29, 2024

8.9K 121

See more recommendations