Amazon EKS

GW 直前！まだまにあうコンテナバケーションwith Amazon EKS

RBAC DeepDive

SAML Authentication / IAM Roles for Service Accounts

松田 和樹

スタートアップ ソリューションアーキテクト

アマゾン ウェブ サービス ジャパン株式会社

#EKSMatsuri

kind: 自己紹介

metadata:

name: 松田 和樹 （まつだ かずき） mats16k

spec:

org: アマゾン ウェブ サービス ジャパン株式会社

role: スタートアップ ソリューションアーキテクト

like:

- AWS Fargate
- AWS Lambda
- Elasticsearch

セッション対象者とゴール

#EKSMatsuri

想定聴講者

- Kubernetes (Amazon EKS) を利用している
- 「俺は 雰囲気で RBAC を理解している」
- 「なんなら、IAM Role, STS も雰囲気だ」

ゴール

- AWS IAM や SAML 認証を絡めた際の RBAC の動きについて理解する
- 関連する OSS の活用やトラブルシュートが出来るようになる

aws

アジェンダ

- Amazon EKS における IAM と RBAC
- SAML 認証と RBAC
- IAM Roles for Service Accounts
- まとめ

#EKSMatsuri

Amazon EKS における IAM と RBAC

aws

Amazon EKS における IAM と RBAC の関係性

#EKSMatsuri

認証 : IAM

- 「誰であるか」の実証
- AWS の認証情報を利用

認可 : RBAC

- 「特定の権限を持つこと」の実証
- 特定の Namespace に Deployment を展開していいとか

※ EKS は CreateCluster API を叩いた IAM User を、Administrator として RBAC に登録する

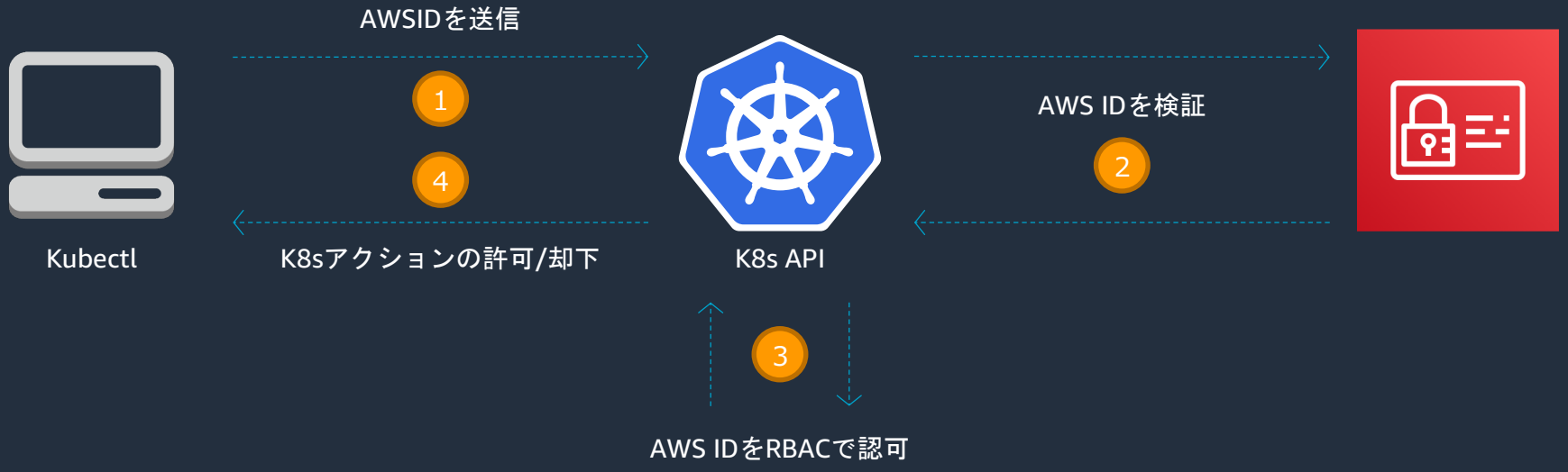
aws

Amazon EKS における IAM と RBAC の関係性

#EKSMatsuri

EKSでは IAMとKubernetes RBAC (Role Based Access Control) が連携

- IAMで許可を与えただけでは Kubernetes API サーバーへのコマンド実行は不可
- Kubernetes 内の aws-auth ConfigMap にIAMユーザーのARNなど設定



クラスター認証の管理　https://docs.aws.amazon.com/ja_jp/eks/latest/userguide/managing-auth.html
クラスターのユーザーまたは IAM ロールの管理　https://docs.aws.amazon.com/ja_jp/eks/latest/userguide/add-user-role.html

aws

Amazon EKS における IAM と RBAC の関係性

#EKSMatsuri

サーバサイドは Amazon EKS が設定してくれるので対応不要
クライアント側の設定は下記コマンドで OK

```
$ aws eks update-kubeconfig --name devCluster
```

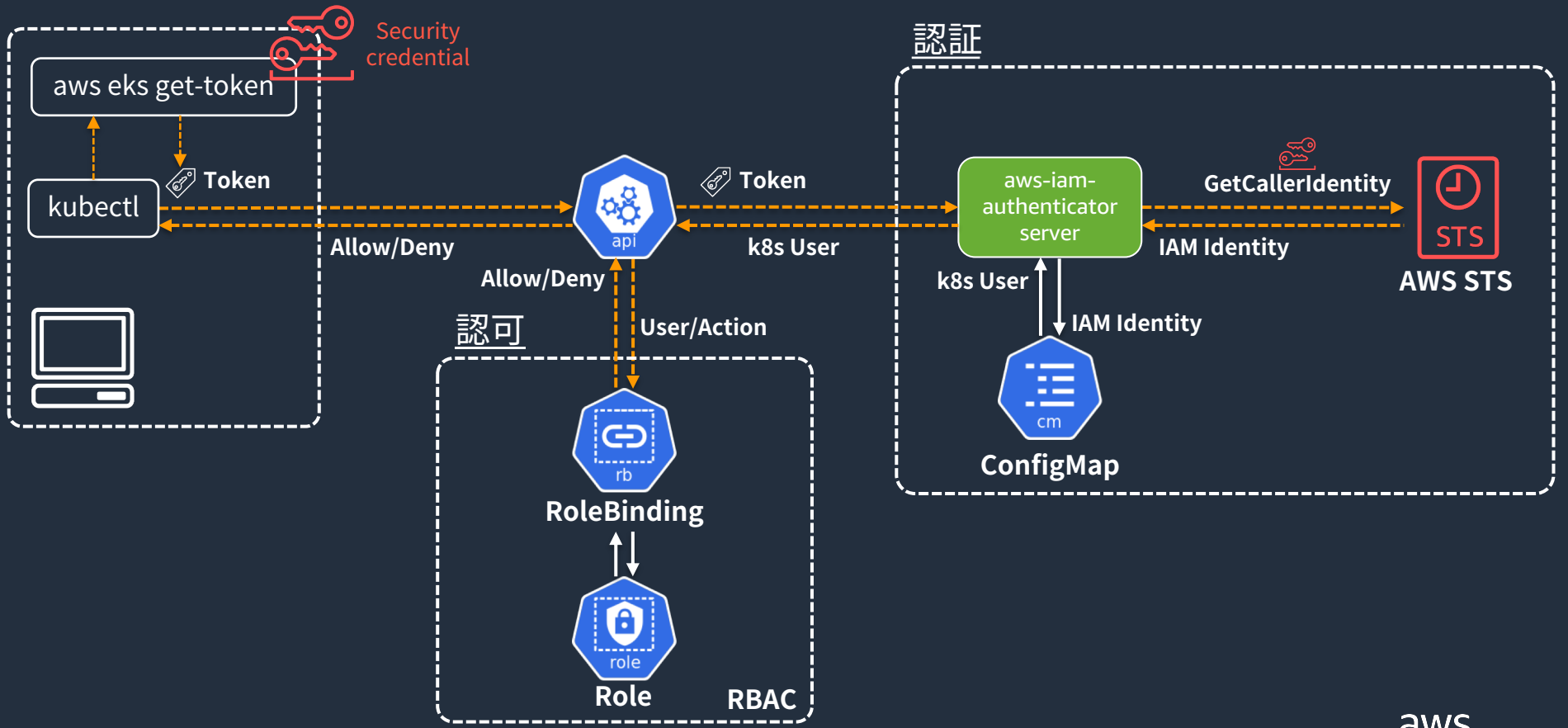
```
- name: arn:aws:eks:us-east-1:123456789012:cluster/devCluster
  user:
    exec:
      apiVersion: client.authentication.k8s.io/v1alpha1
      args:
        - --region
        - us-east-1
        - eks
        - get-token
        - --cluster-name
        - devCluster
      command: aws
```

```
$ aws eks get-token --cluster-name devCluster
```


aws

Amazon EKS における IAM と RBAC の関係性

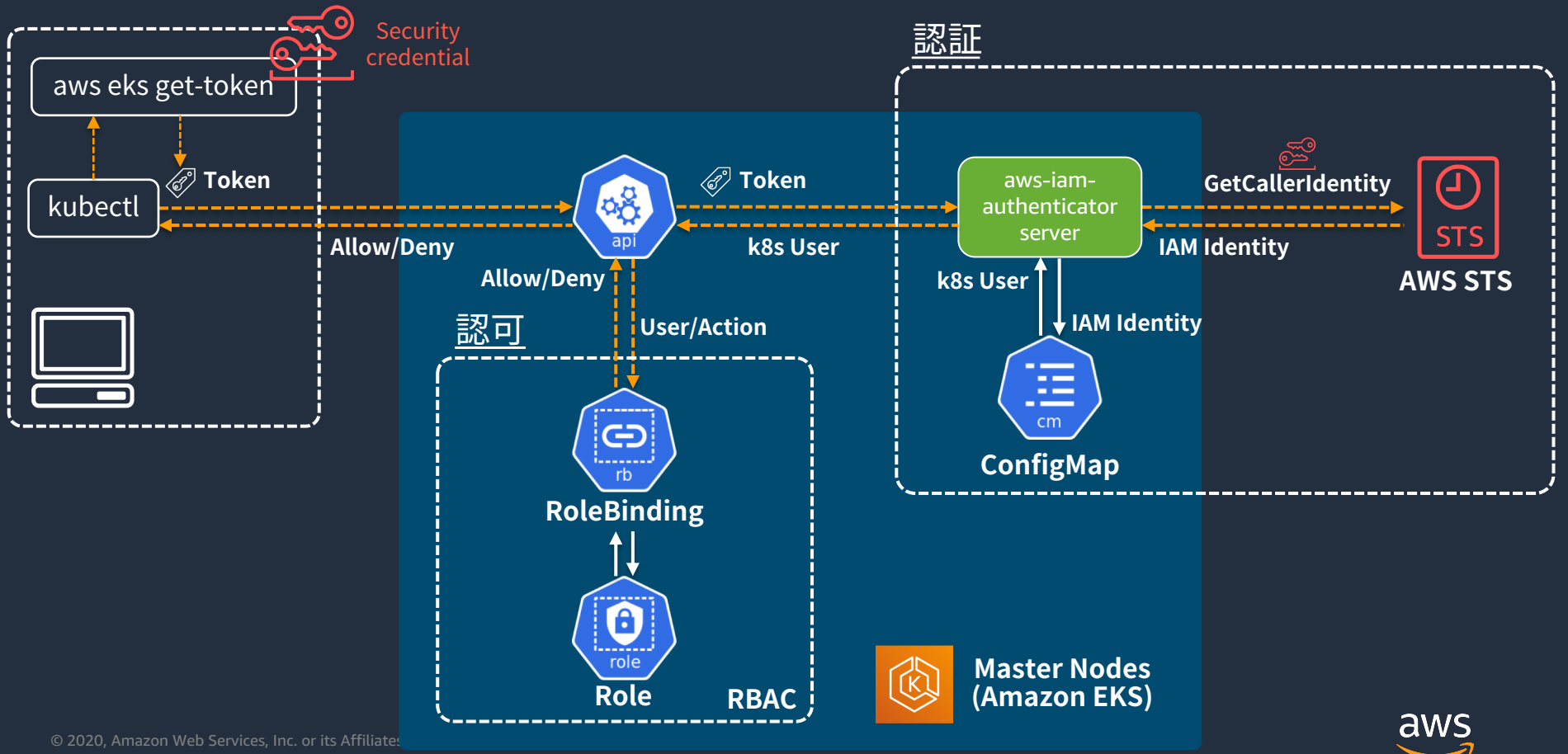
#EKSMatsuri



aws

Amazon EKS における IAM と RBAC の関係性

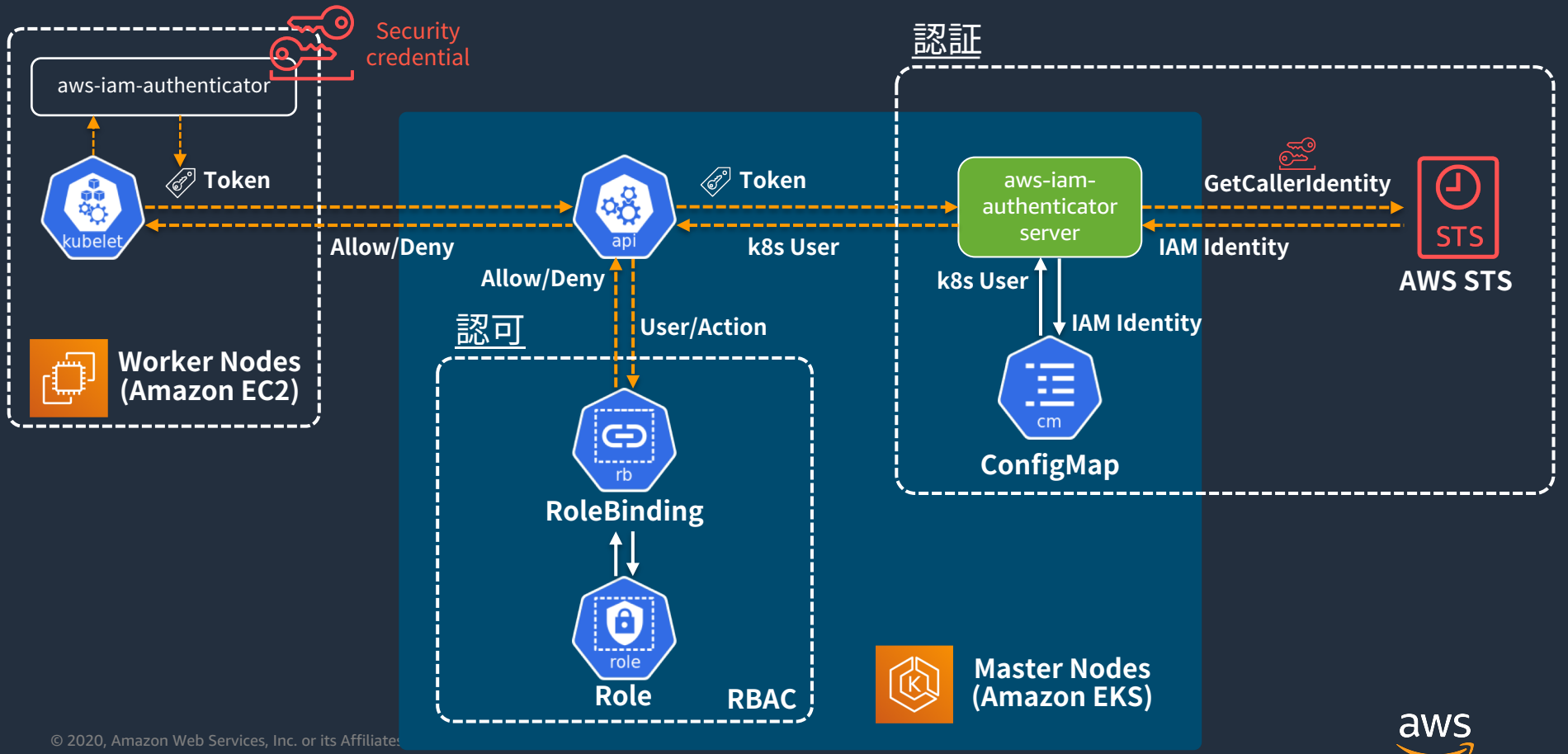
#EKSMatsuri



aws

Amazon EKS における IAM と RBAC の関係性

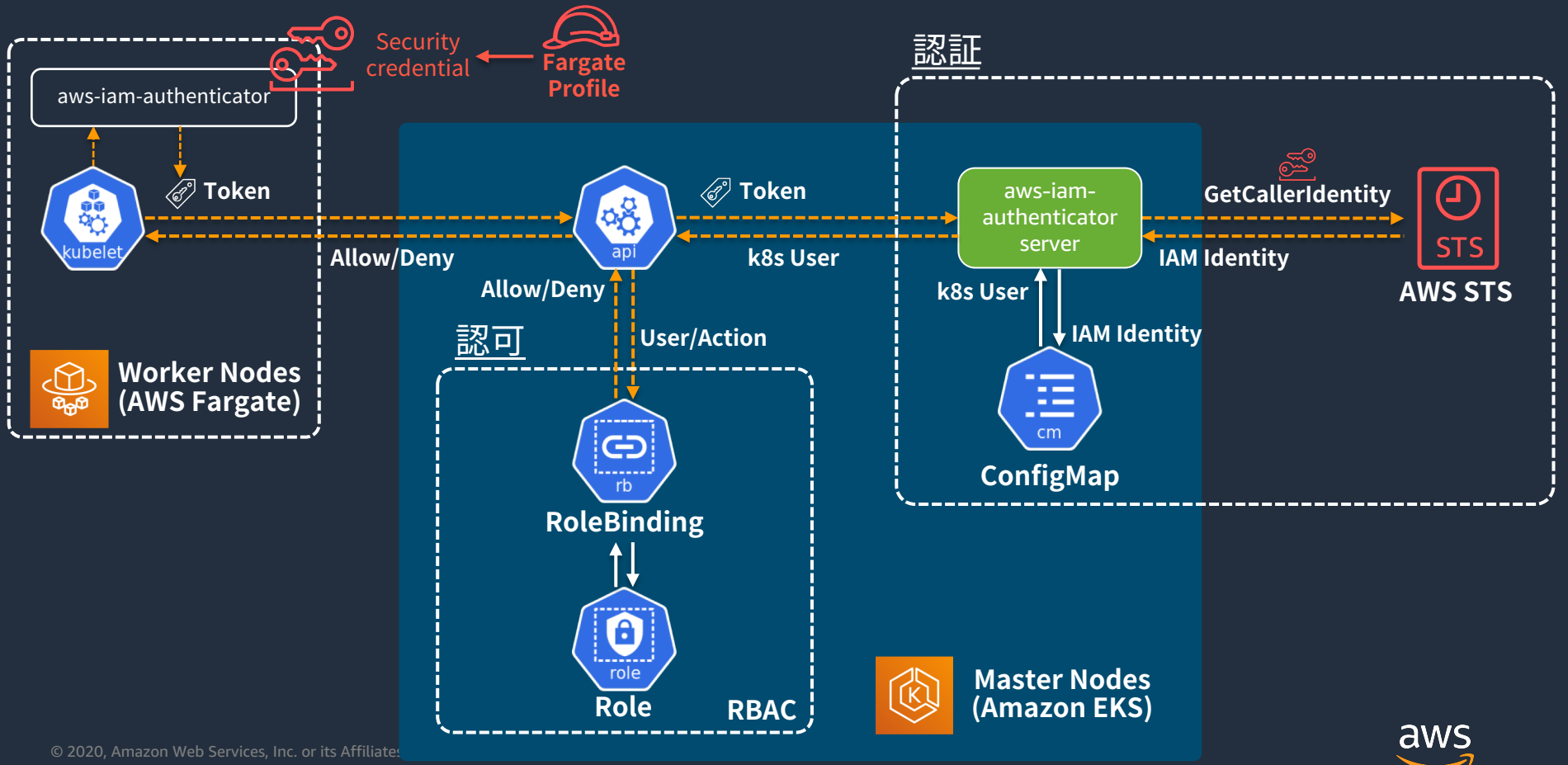
#EKSMatsuri



aws

Amazon EKS における IAM と RBAC の関係性

#EKSMatsuri



aws

RBAC への IAM User/Role の登録


```
$ aws iam create-user --user-name k8s-user
$ aws iam create-access-key --user-name k8s-user
```

IAM User を作成

AccessKey を取得

```
$ kubectl apply -n kube-system -f aws-auth.yaml
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapUsers: |
    - userarn: arn:aws:iam::123456789012:user/k8s-user
      username: k8s_user
      groups:
        - system:masters
```

ConfigMap に mapUsers を追加

userarn に ARN を記載する点に注意

username は k8s 上の User

RBAC への IAM User/Role の登録

#EKSMatsuri

IAM Role を利用する WorkerNode の場合

```
$ kubectl apply –n kube-system -f aws-auth.yaml
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles:
    - rolearn: arn:aws:iam::123456789012:role/EksWorkerNodeRole
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
```

ConfigMap に mapRoles を追加

rolearn を記述する点に注意

username に static な値を入れると、識別が出来なくなる

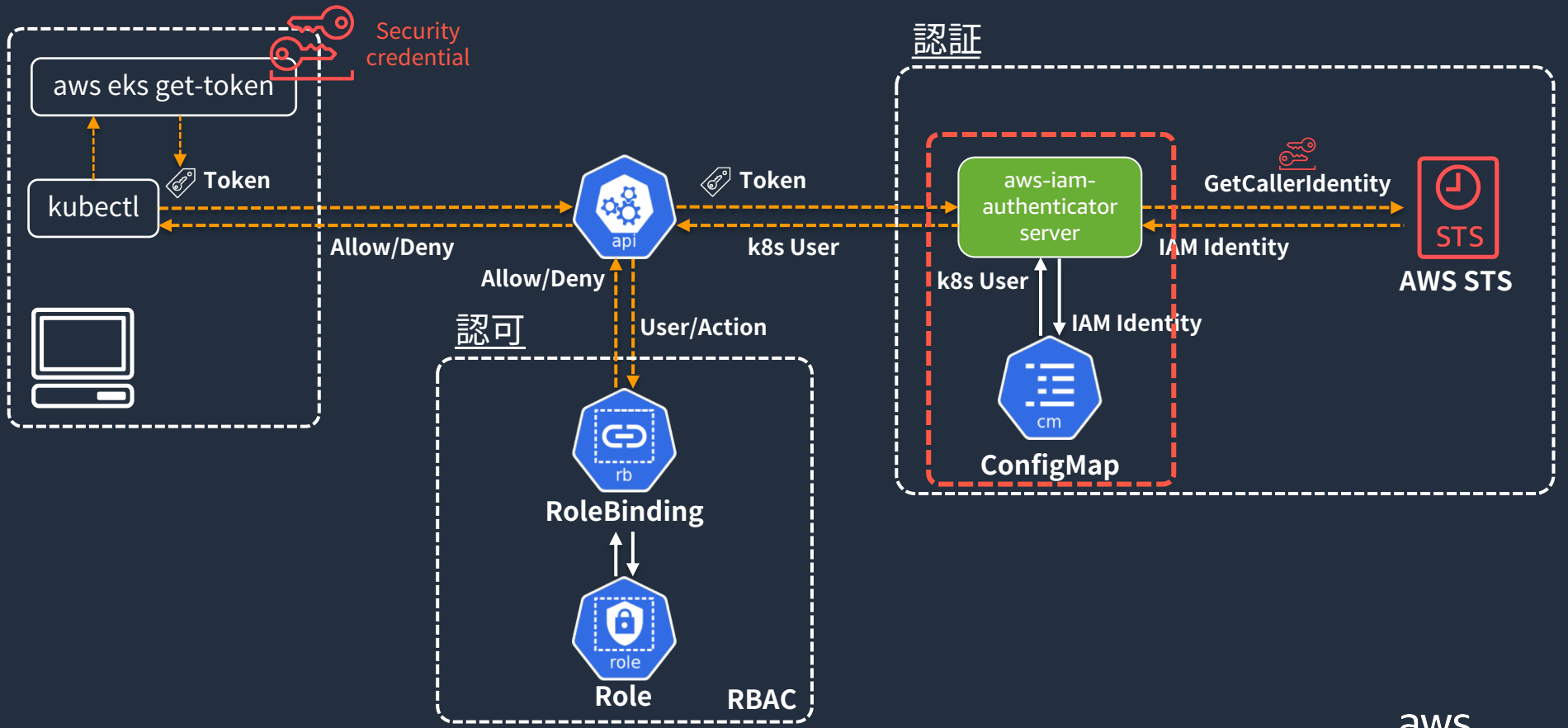
利用可能な変数は

- {{AccountID}}
- {{EC2PrivateDNSName}}
- {{SessionName}}

aws

authenticator のログは CloudWatch で確認可能

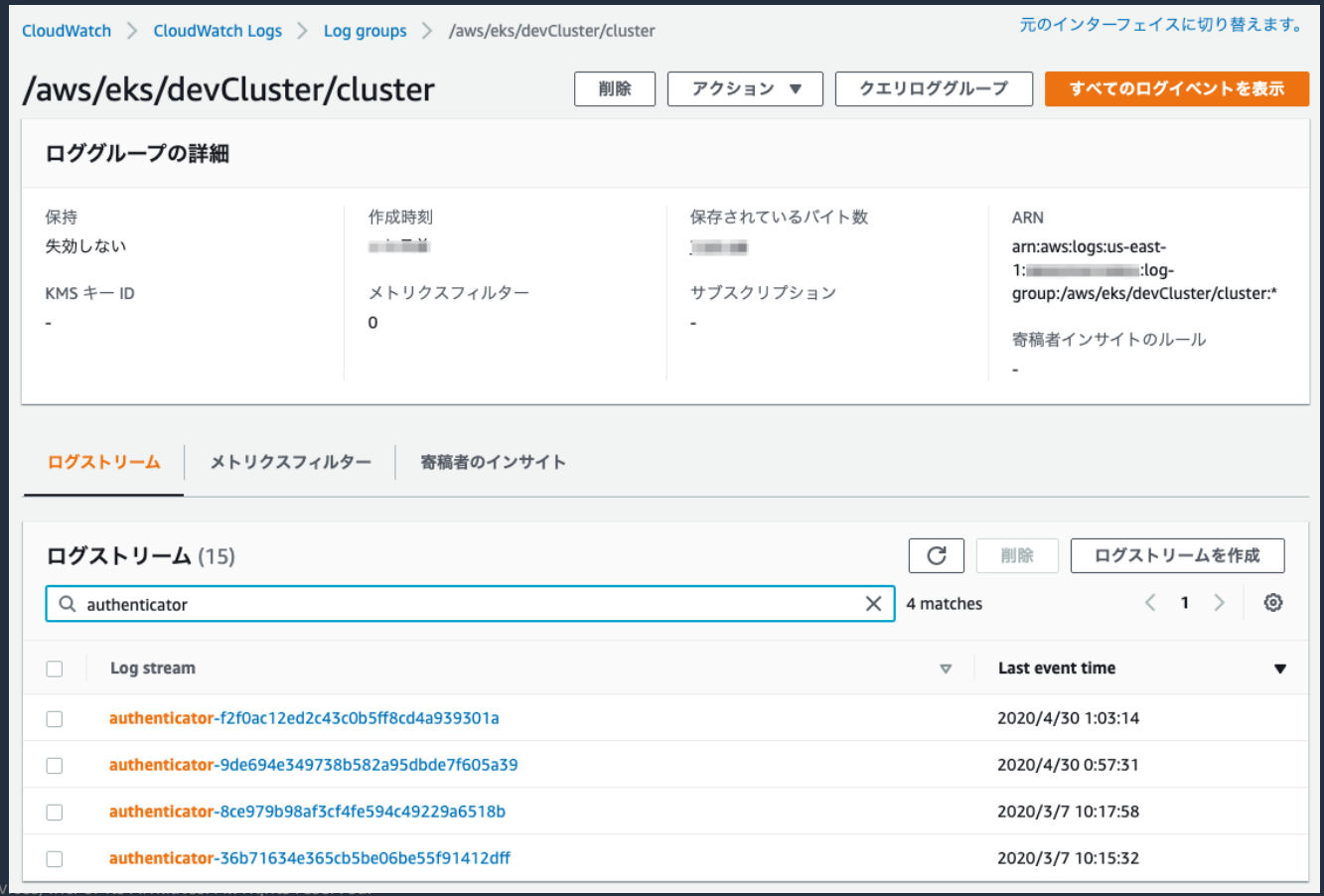
#EKSMatsuri



aws

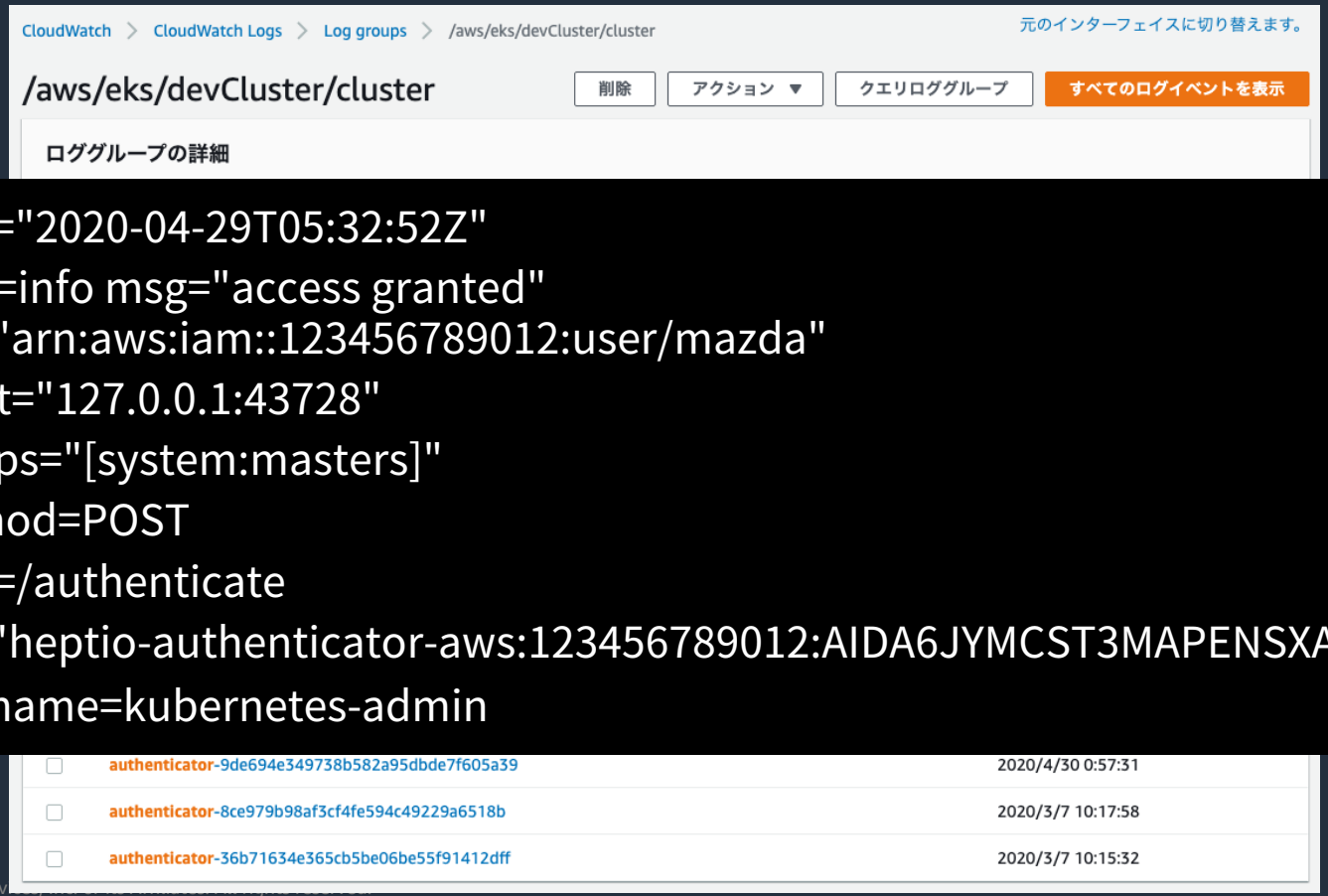
authenticator のログは CloudWatch で確認可能

#EKSMatsuri



authenticator のログは CloudWatch で確認可能

#EKSMatsuri



aws

authenticator が STS の API を叩いているのも CloudTrail で確認可能

	イベント時間	ユーザー名	イベント名	発信元 IP アドレス
▼	2020-04-29, 09:41:12 PM	mazda-cli	GetCallerIdentity	3.209.144.150

AWS アクセスキー	AKIA6JYMCST3ANRRVBW4	イベント時間	2020-04-29, 09:41:12 PM
AWS リージョン	us-east-1	読み取り専用	true
エラーコード		リクエスト ID	44b7902c-bbd5-4c03-b0b4-8e558780cd06
イベント ID	c4f1394f-6695-4aed-8473-5be7cee0adee	発信元 IP アドレス	3.209.144.150
イベント名	GetCallerIdentity	ユーザー名	mazda-cli
イベントソース	sts.amazonaws.com		

参照リソース (0)

イベントの表示

※ 余談ですが userAgent とかも見れます "userAgent": "Go-http-client/1.1"

#EKSMatsuri

SAML 認証で AWS を利用している場合の Amazon EKS

aws

SAML 認証での AWS の利用

AWS への 認証 に SAML 認証を利用可能

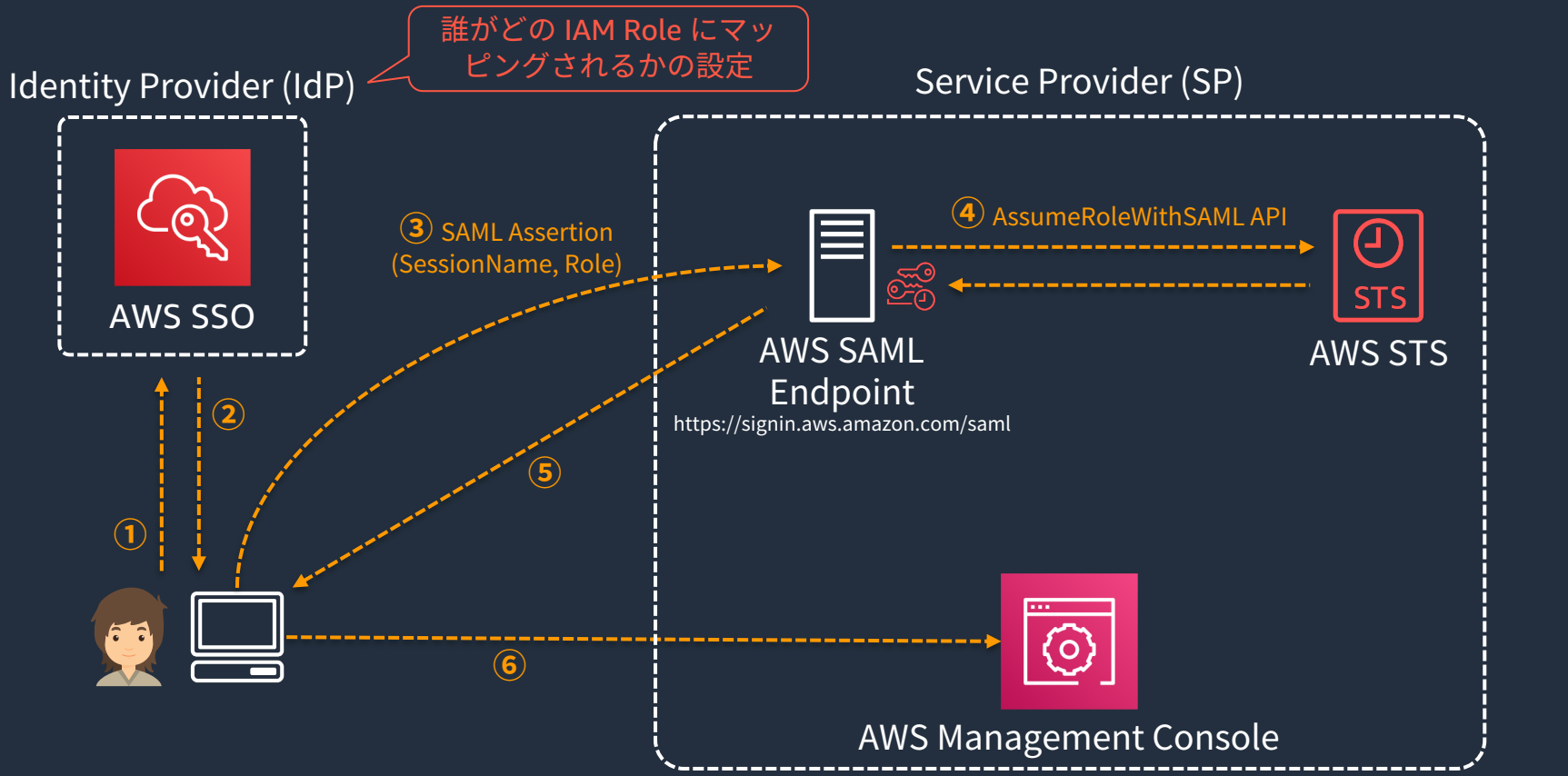
- AWS Single Sign-On
- G Suite
- Azure AD など

メリット

- IAM での権限管理は今まで通り
- ID 基盤の集約、アカウント管理の簡素化
    - IAM User を管理しなくて良い！！！

SAML 認証での AWS の利用

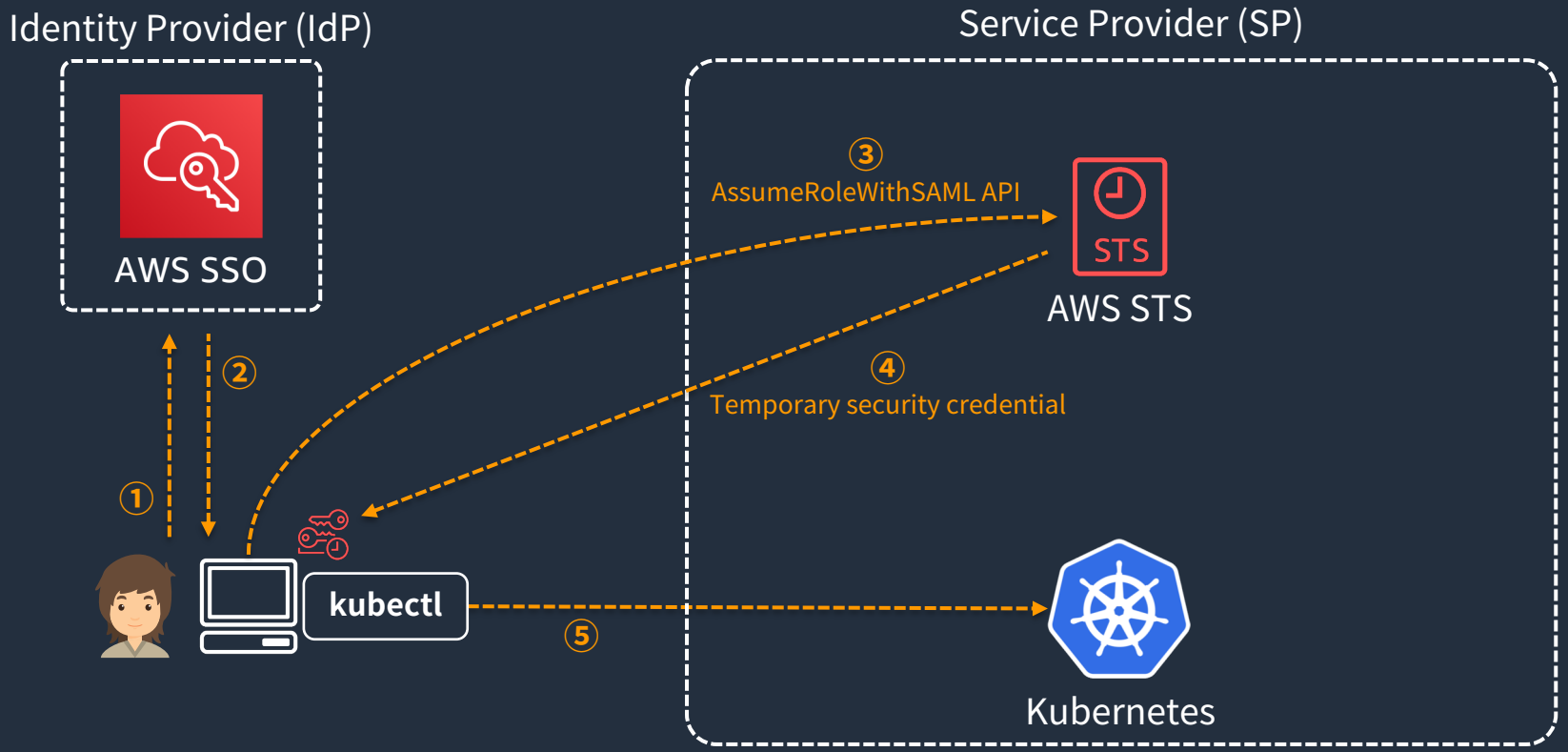
#EKSMatsuri



aws

SAML 認証での Kubernetes (Amazon EKS) の利用

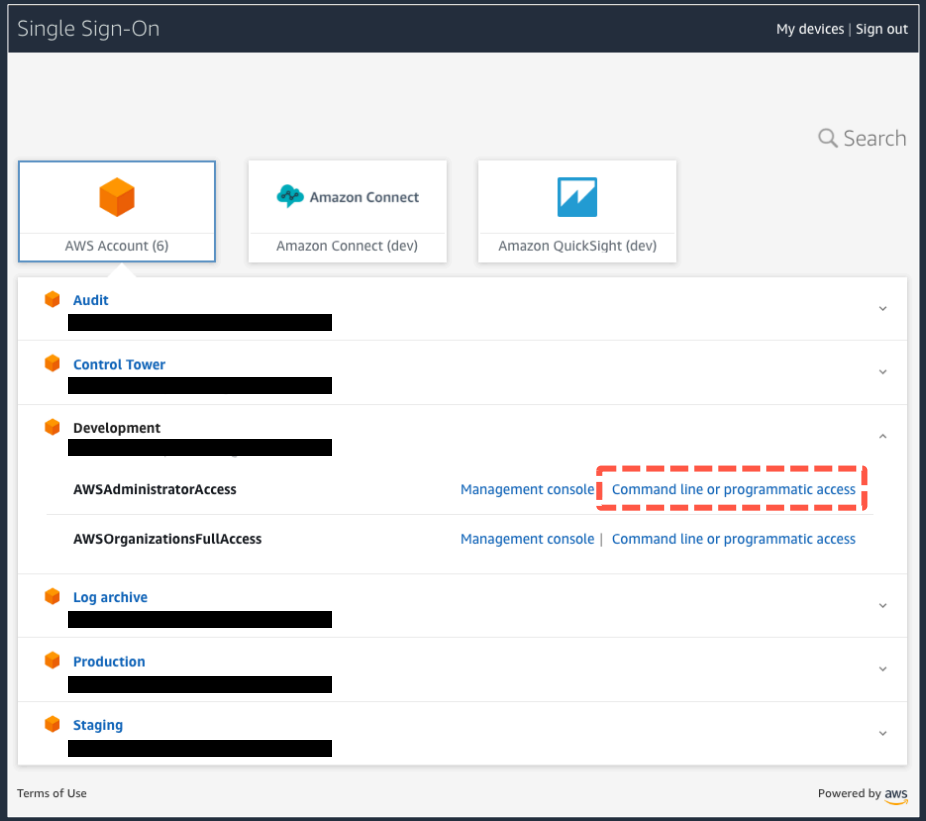
#EKSMatsuri

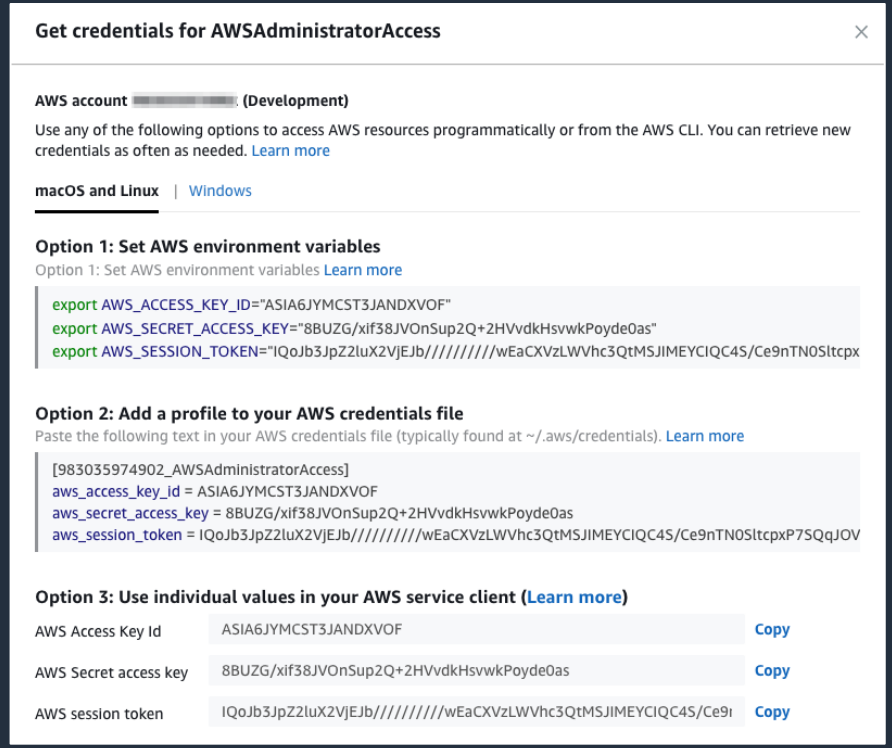


aws

SAML 認証での Kubernetes (Amazon EKS) の利用

#EKSMatsuri

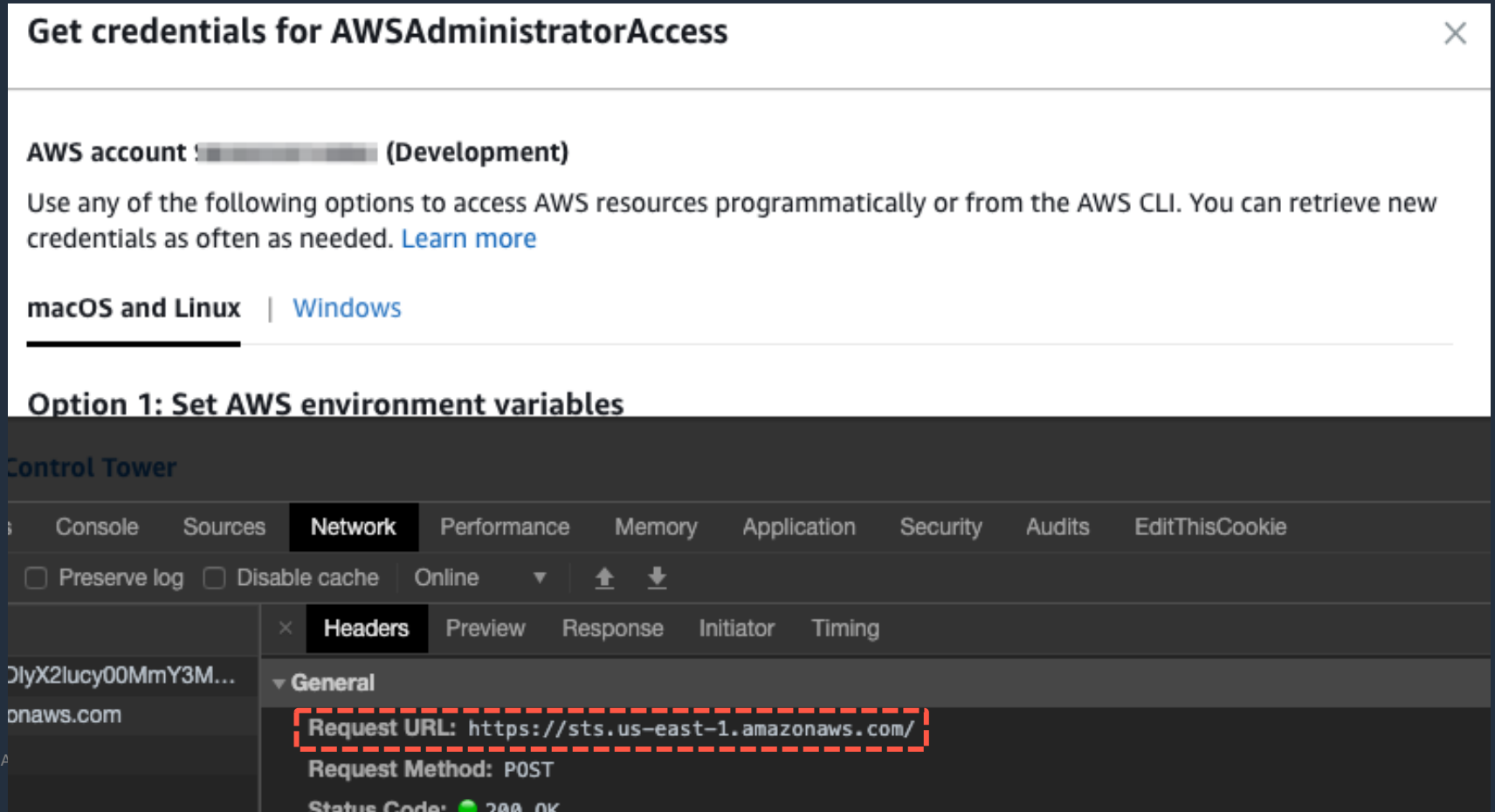




※ CLI やプログラムからも実行可能

SAML 認証での Kubernetes (Amazon EKS) の利用

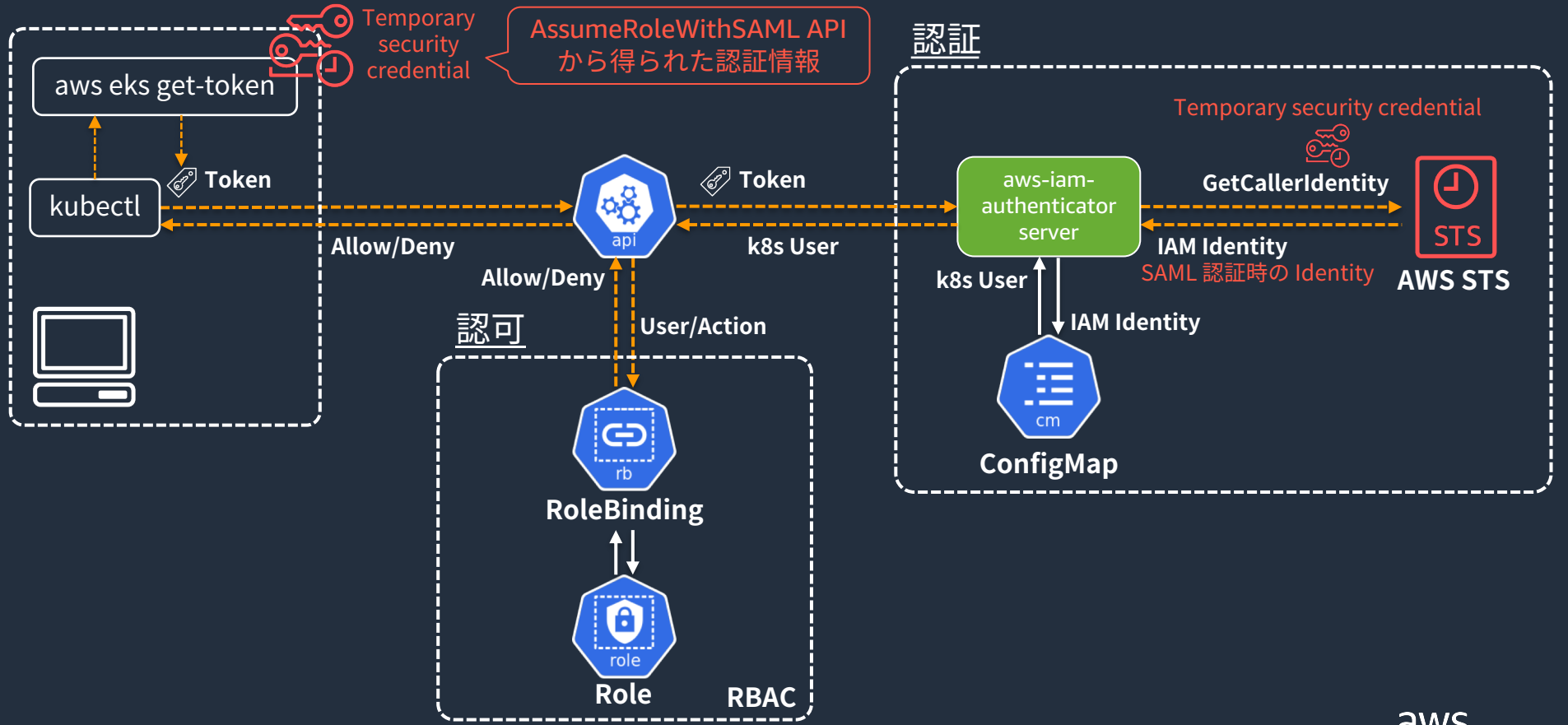
#EKSMatsuri



Kubernetes 上の User は何になるのか？

SAML 認証での Kubernetes (Amazon EKS) の利用

#EKSMatsuri



aws

SAML 認証での Kubernetes (Amazon EKS) の利用

IAM Role を利用する場合は、mapRoles を利用する
IdP 側の識別子（多くの場合はメールアドレス）が
SessionName として参照可能なので、username として利用する
（ユーザー識別が目的の場合、AccountID や EC2PrivateDNSName は適していない）

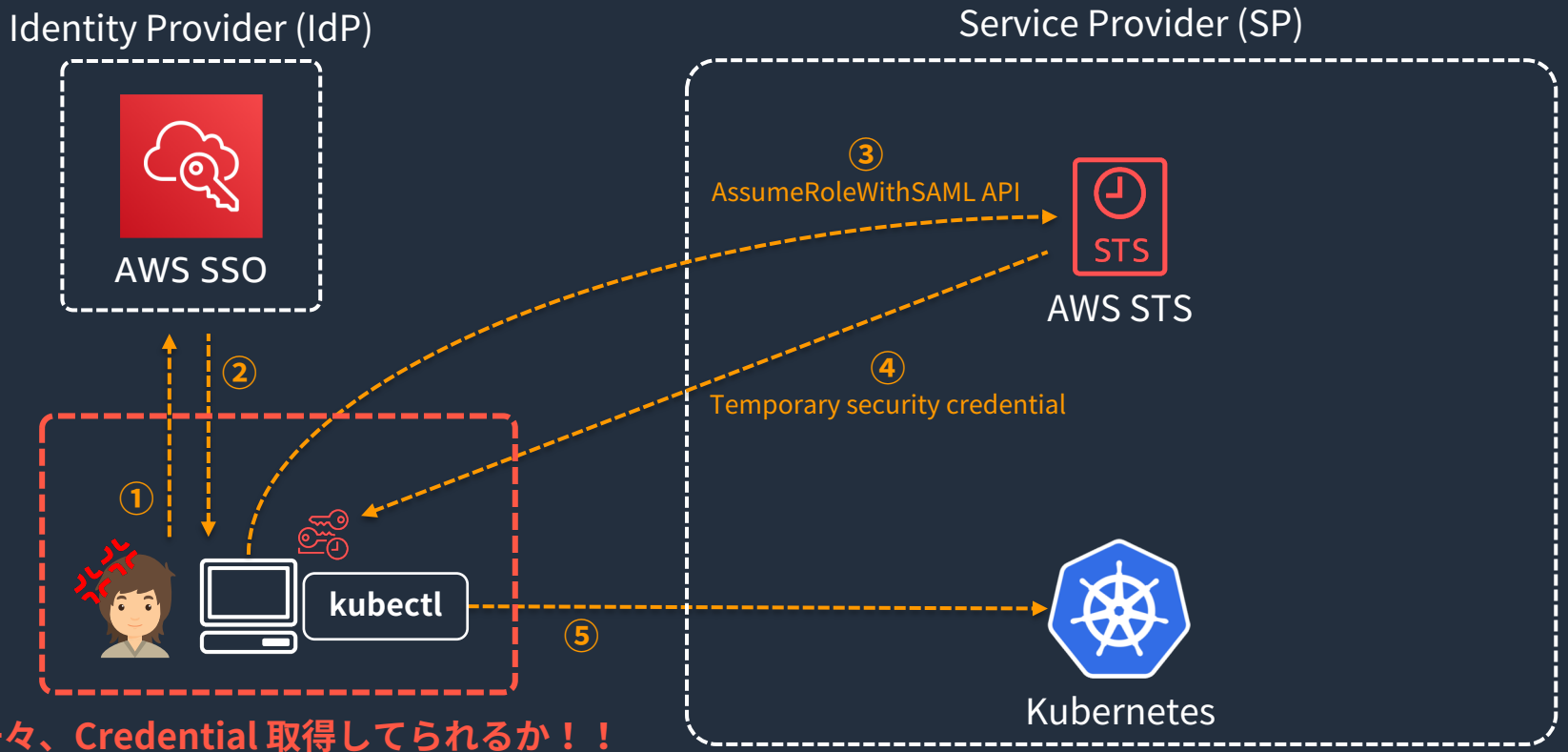
```
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles:
    - rolearn: arn:aws:iam::123456789012:role/AWSReservedSSO_EKS_Admins
      username: adminuser:{{SessionName}}
      groups:
        - system:masters
```

ここまでは普通のお話

仕組みは分かったけど、正直めんどくさい・・・？

SAML 認証での Kubernetes (Amazon EKS) の利用

#EKSMatsuri



一々、Credential 取得してられるか！！

aws

Amazon Web Services ブログ

AWS CLI v2 が一般利用可能となりました

by AWS Japan Staff | on 12 FEB 2020 | in AWS CLI | Permalink | Share

AWS CLI バージョン 2 (v2) の v2.0.0 GA リリースを発表できることを嬉しく思います。

AWS CLI v2 は AWS CLI v1 をベースに構築され、コミュニティのフィードバックに基づいた多くの機能と拡張機能が含まれています。

新機能

AWS CLI v2 には、改良されたインストーラ、AWS シングルサインオン（SSO）などの新しい設定オプション、さまざまなインタラクティブ機能など、いくつかの新機能があります。

新しいインストールメカニズム

AWS CLI v2 には、Windows、Linux、および macOS 用のビルド済みのバイナリが用意されています。 AWS CLI を使用するために Python をインストールする必要はありません。 互換性のあるPythonバージョン、仮想環境、または競合するPythonパッケージについて心配する必要はありません。 WindowsではMSIインストーラを提供し、macOSでは.pkgインストーラを提供します。 AWS CLI v2 のインストールの詳細については、インストール手順を参照してください。

AWS CLIの設定メカニズム

AWS CLI v2 では、認証情報を設定するためのいくつかの新しいメカニズムが導入されています。 AWS コンソールで生成された .csv ファイルから認証情報をインポートする新しい `aws configure import` コマンドが追加されました。

AWS CLI v2

#EKSMatsuri

AWS CLI v2 は AWS SSO をサポート

ブラウザが自動で起動
AWS SSO 未ログインの場合は
ここでログイン

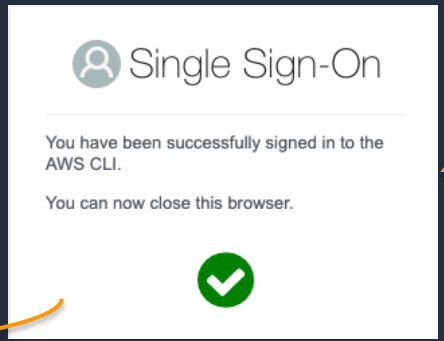
```
$ aws sso login (--profile sso)

Attempting to automatically open the SSO authorization page in your default browser.
If the browser does not open or you wish to use a different device to authorize this request,
open the following URL:

https://device.sso.us-east-1.amazonaws.com/

Then enter the code:
                                （いろいろ出ますが、基本無視してOKです）
RNKX-JWWP


Successully logged into Start URL: https://hoge.awsapps.com/start
```



aws

kubectl with AWS CLI v2


```
$ aws sso login (--profile sso)

$ aws eks update-kubeconfig --name devCluster (--profile sso)

$ kubectl get pods
```

※ cli v2 を aws2 の様な名前で利用している場合、
aws2 eks update-kubeconfig はそのことを認識しないため、
手動で ~/.kube/config を修正する必要があります。

kubectl with AWS CLI v2
CloudWatch も見てみる

#EKSMatsuri

	イベント時間	ユーザー名	イベント名	発信元 IP アドレス
▼	2020-04-30, 01:08:34 AM	mazda@amazon.com	GetCallerIdentity	3.209.144.150

AWS アクセスキー ASIA6JYMCST3MMWM66FH

AWS リージョン us-east-1

エラーコード

イベント ID efd50ebe-14c5-44af-9b79-e3ed73c752a6

イベント名 GetCallerIdentity

イベントソース sts.amazonaws.com

イベント時間 2020-04-30, 01:08:34 AM

読み取り専用 true

リクエスト ID e50e763c-4b4b-49e9-8d83-f03efd55a0f0

発信元 IP アドレス 3.209.144.150

ユーザー名 mazda@amazon.com

参照リソース (0)

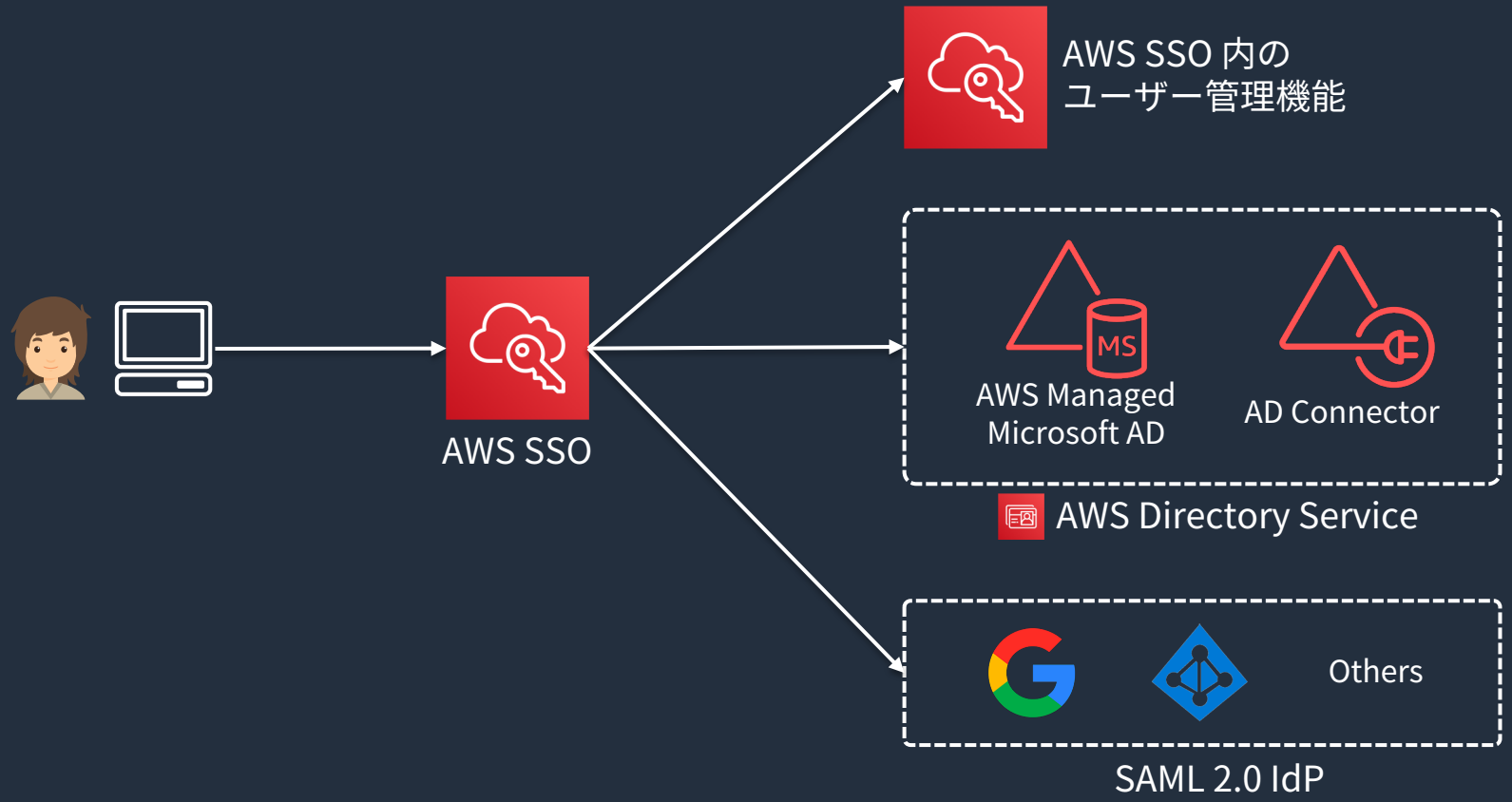
イベントの表示

aws

ID 基盤が G Suite なんですけど・・・

AWS Single Sign-On の外部 IdP サポート

#EKSMatsuri



AWS Single Sign-On の外部 IdP サポート

kubectl with SAML Authentication まとめ

#EKSMatsuri

- aws-iam-authenticator は SAML 認証でも問題なく動く
- AWS CLI v2 なら、ターミナル操作時の SAML 認証が容易 (AWS SSO)
- AWS SSO は外部の IdP を扱える

任意の IdP でアカウント管理をおこないつつ、
セキュアに Kubernetes を利用することが出来る

aws

IAM Roles for Service Accounts

IAM Roles for Service Accounts

#EKSMatsuri

Kubernetes 上の Service Account に IAM Role を割り当てる機能

- Node に IAM Role を割り当てると、全ての Pod に権限がついてしまう

aws

IAM Roles for Service Accounts

#EKSMatsuri

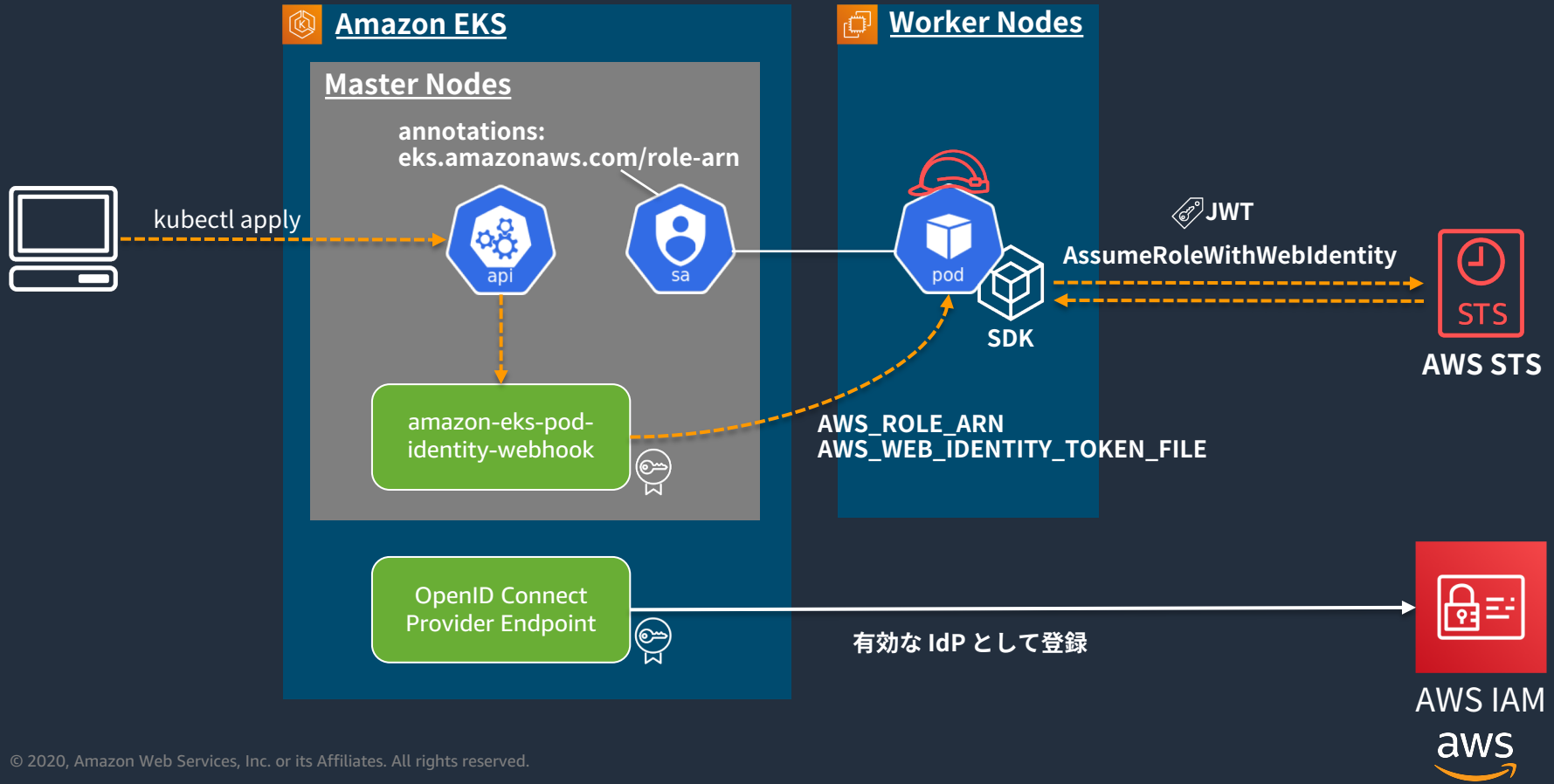
Kubernetes 上の Service Account に IAM Role を割り当てる機能

- Node に IAM Role を割り当てると、全ての Pod に権限がついてしまう
- Pod に紐付ける Service Account 毎に IAM Role を紐付ける

aws

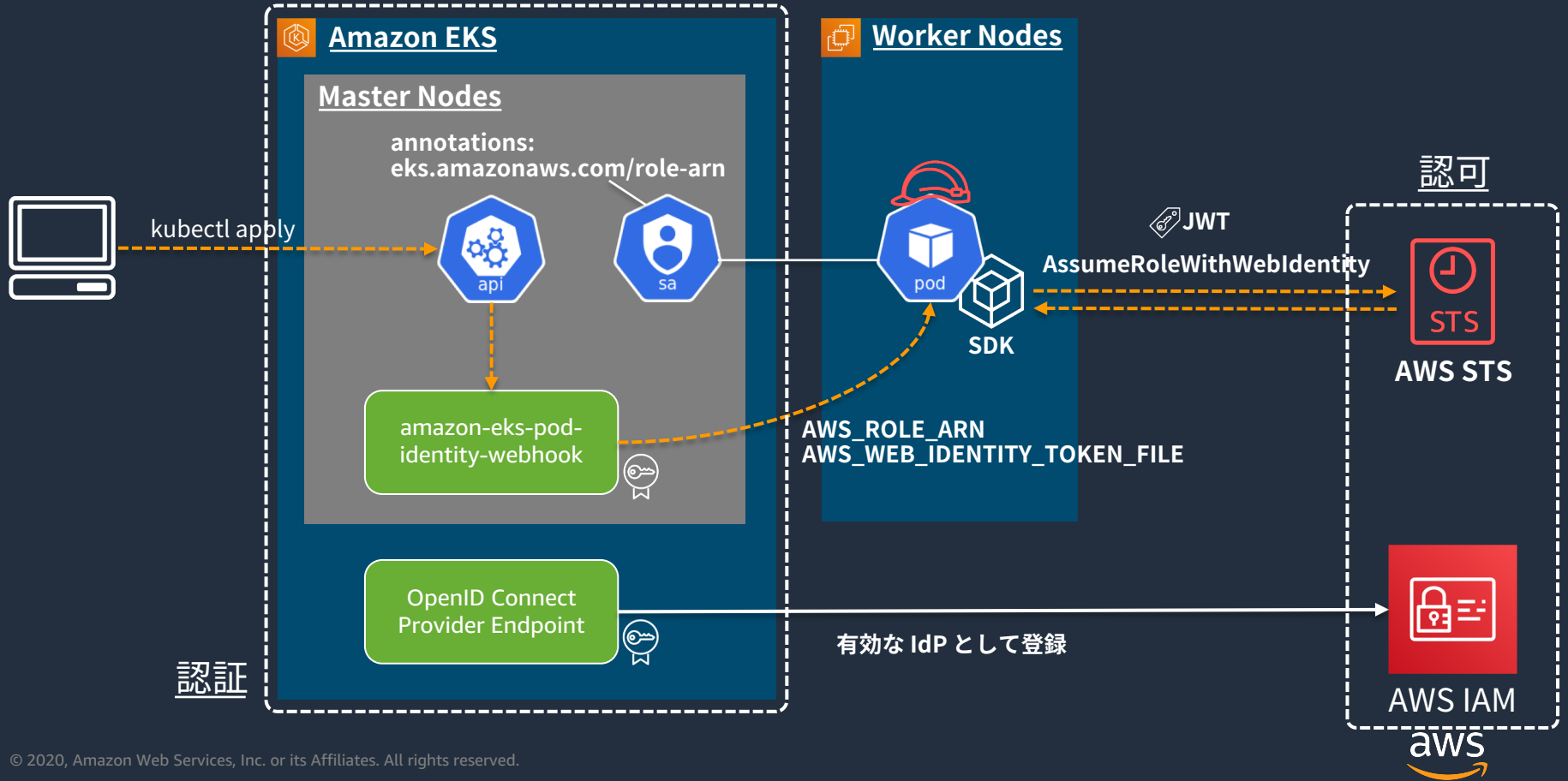
IAM Roles for Service Accounts の仕組み

#EKSMatsuri



IAM Roles for Service Accounts の仕組み

#EKSMatsuri



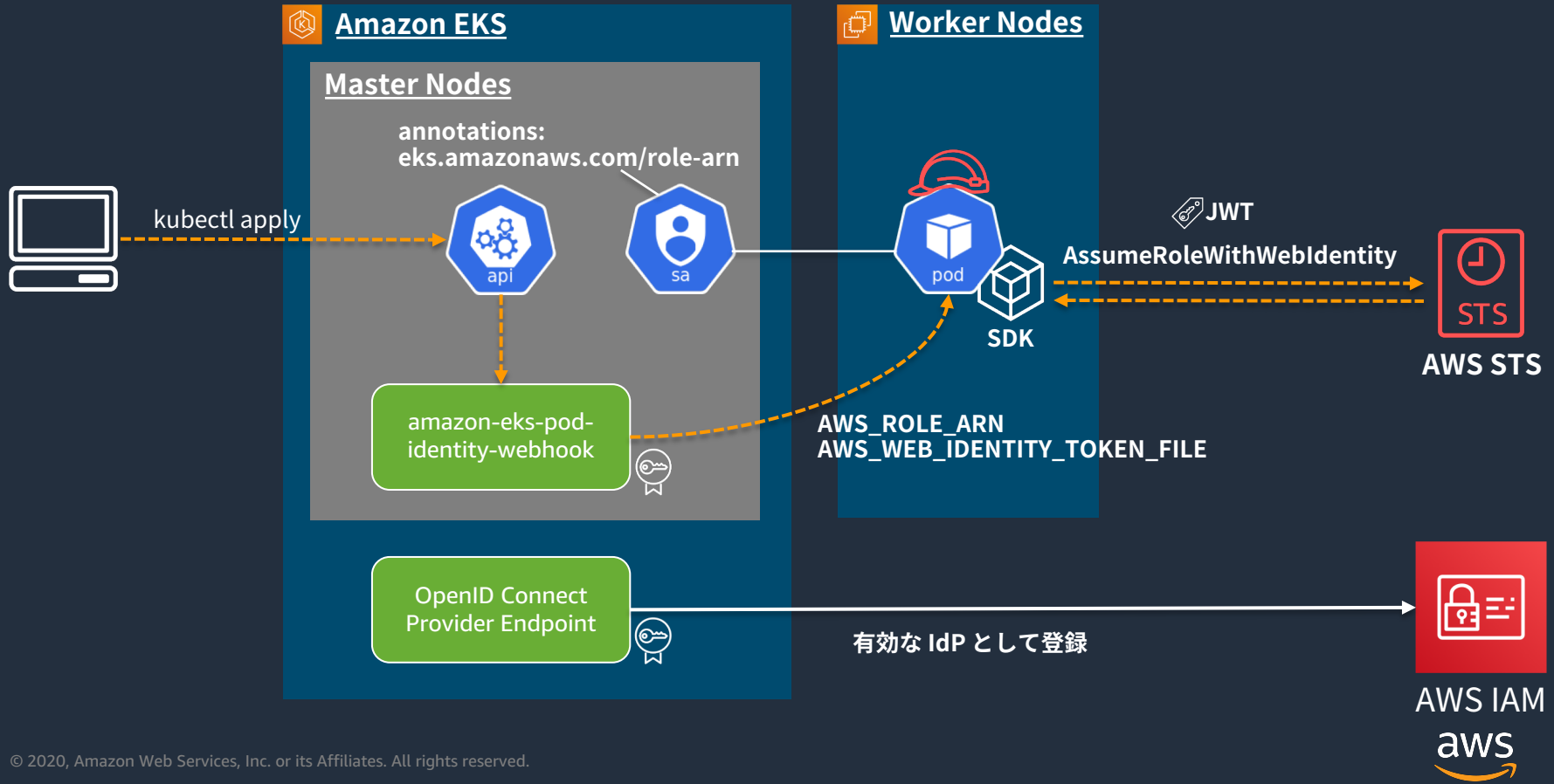
IAM Roles for Service Accounts の仕組み

AssumeRoleWithWebIdentity に対応したバージョンの AWS SDK が必要

- Java 1.11.623
- Java2 2.7.36
- Go 1.23.13
- Python 1.9.220
- Node 2.521.0
- Ruby 2.11.345
- PHP 3.110.7
- .NET 3.3.580.0

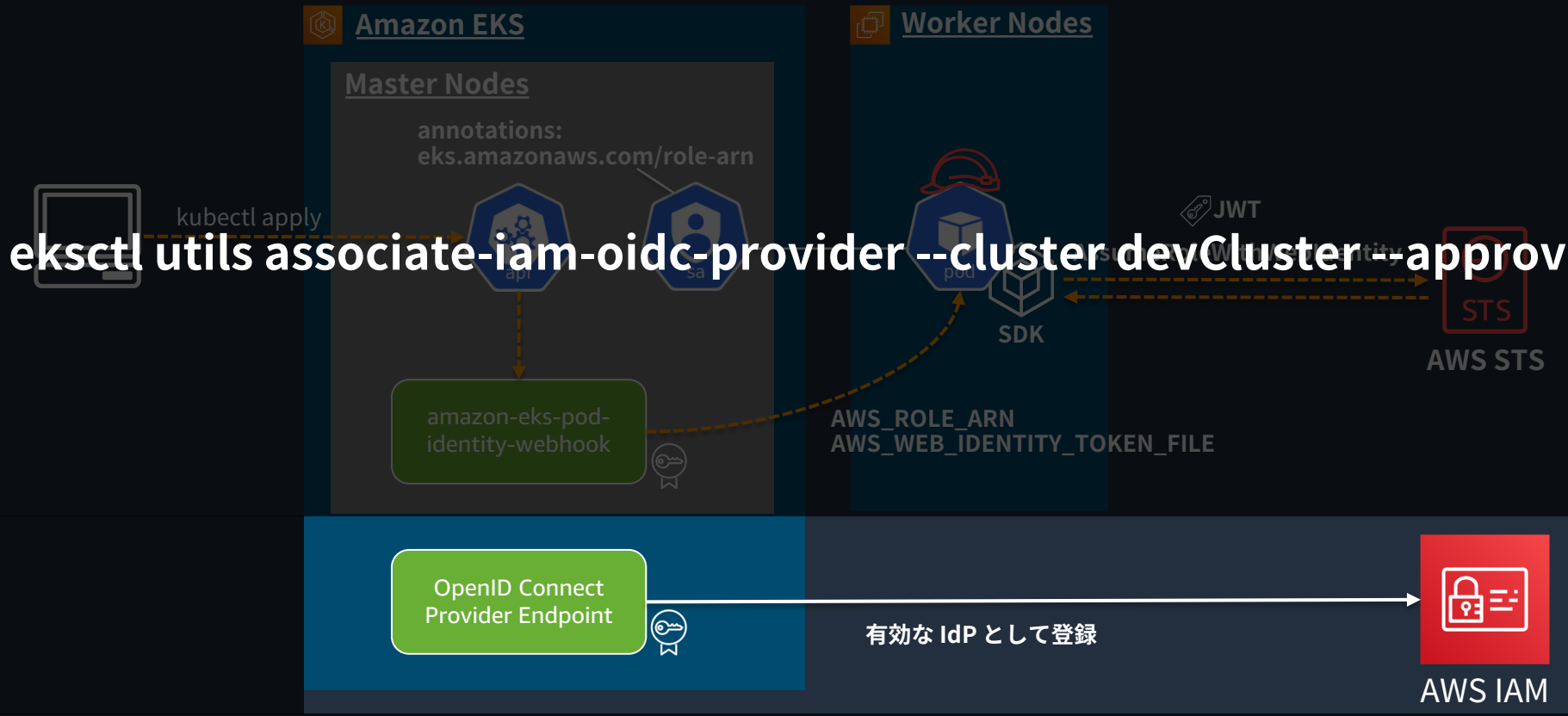
IAM Roles for Service Accounts の設定方法 (eksctl)

#EKSMatsuri



IAM Roles for Service Accounts の設定方法 (eksctl)

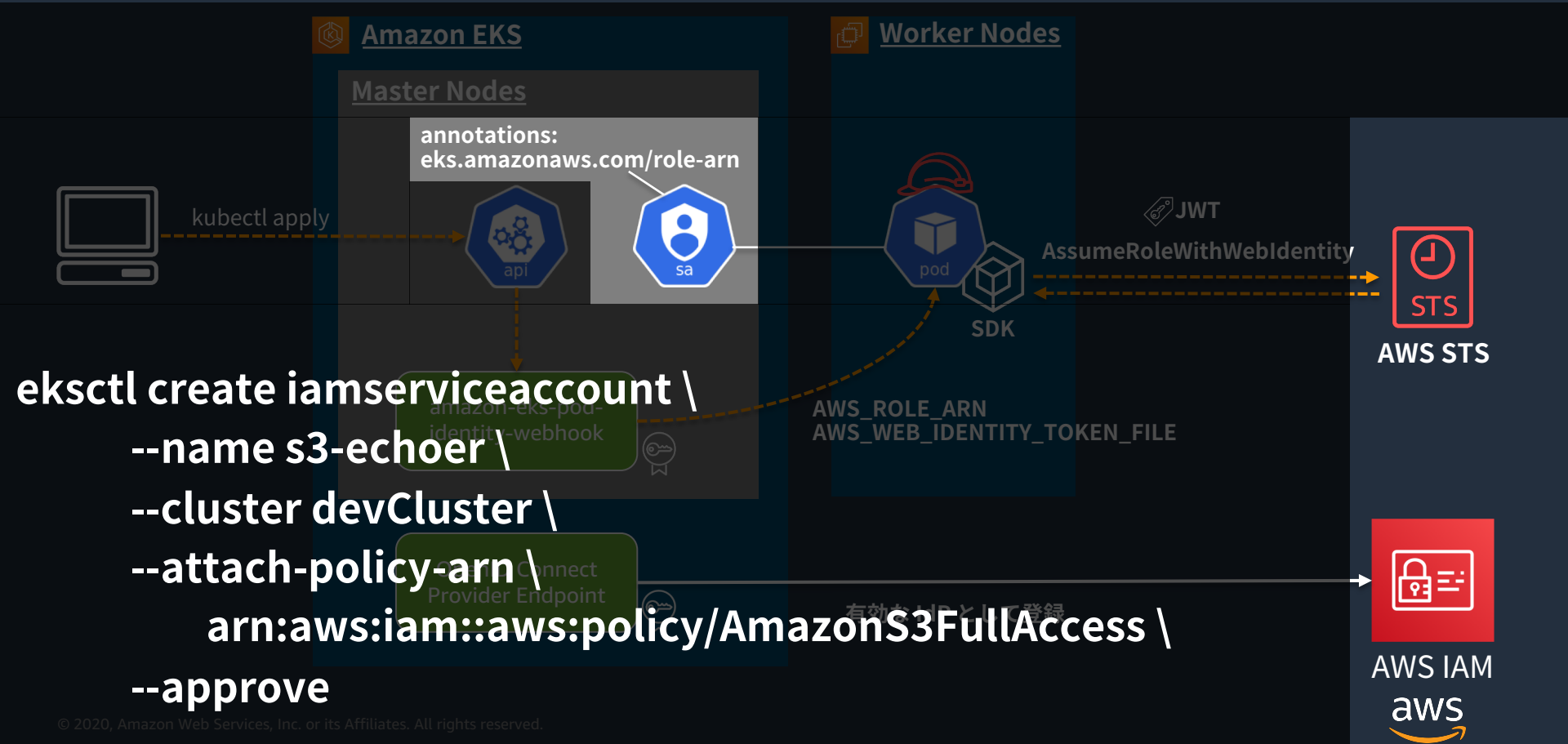
#EKSMatsuri



eksctl utils associate-iam-oidc-provider --cluster devCluster --approve

IAM Roles for Service Accounts の設定方法(eksctl)

#EKSMatsuri




```
eksctl create iamserviceaccount \
        --name s3-echoer \
        --cluster devCluster \
        --attach-policy-arn \
             arn:aws:iam::aws:policy/AmazonS3FullAccess \
        --approve
```

Json Web Token の中身

/var/run/secrets/eks.amazonaws.com/serviceaccount/token

eyJhbGciOiJSUzI1NiIsImtpZCI6ImI2ZDAxYTk2MmUyNWEwMmExMzlhMzhlMzUyZDkxZGI0OGEwMjI0YmQifQ.eyJhdWQiOlsic3RzLmFtYXpvbmF3cy5jb20iXSwiZXhwIjoxNTg4MjUyMzM2LCJpYXQiOjE1ODgxNjU5MzYsImlzcyI6Imh0dHBzOi8vb2lkYy5la3MudXMtZWFzdC0xLmFtYXpvbmF3cy5jb20vaWQvQjU2RTQzNUNDOTA3NzA0RDg0NDU4NEE2QTZCRUUzMTEiLCJrdWJlcm5ldGVzLmlvIjp7Im5hbWVzcGFjZSI6ImRlZmF1bHQiLCJwb2QiOnsibmFtZSI6Im5naW54LTZkZGRkZmM0N2QtZ2x3MmsiLCJ1aWQiOiIwYmYwNjViNy04YTFiLTExZWEtOGZmZC0wMmQ5ZGFkMjE0NTkifSwic2VydmljZWFjY291bnQiOnsibmFtZSI6InMzLWVjaG9lciIsInVpZCI6ImNiMmQ1MjNjLThhMTYtMTFlYS04MTg3LTEyNGQ4MGQyNTBhNSJ9fSwibmJmIjoxNTg4MTY1OTM2LCJzdWIiOiJzeXN0ZW06c2VydmljZWFjY291bnQ6ZGVmYXVsdDpzMy1lY2hvZXIifQ.UF1lGZIq9Ittno2OilnC6oHT6HULFlw-OYLah4eLjmTI3iLdWE5R7O_806OSsNgGVtRFJ082jXbKKUSv3xH-LFW3yC_a9aS-uzt8WxH7K_NvQcOFWY2g2mZ8X6wCrPcaFS-dtbrSkSwptlrSyNhc_SCObSo1FPZd5E2w2BDQHatC9_ROVObGKBM7pVIzivCoP76PUZsiI1EdYAXiX2EwsCBUlM_37qgjL6RGqHyQ4vk0wXOKa3G7JtQlSTF22xfIOCMrEJ_4pTbxS4h3MY2LM_w4bwYKp9A-vn_PzJnSId8vc8-dQfnv2-blRQ9Dpy4L9aVo8KmNVMqfmq5ukR7_OQ

Json Web Token の中身

/var/run/secrets/eks.amazonaws.com/serviceaccount/token

eyJhbGciOiJSUzI1NiIsImtpZCI6ImI2ZDAxYTk2MmUyNWEwMmExMzlhMzhlMzUyZDkxZGI0OGEwMjI0YmQifQ.eyJhdWQiOlsic3RzLmFtYXpvbmF3cy5jb20iXSwiZXhwIjoxNTg4MjUyMzM2LCJpYXQiOjE1ODgxNjU5MzYsImlzcyI6Imh0dHBzOi8vb2lkYy5la3MudXMtZWFzdC0xLmFtYXpvbmF3cy5jb20vaWQvQjU2RTQzNUNDOTA3NzA0RDg0NDU4NEE2QTZCRUUzMTEiLCJrdWJlcm5ldGVzLmlvIjp7Im5hbWVzcGFjZSI6ImRlZmF1bHQiLCJwb2QiOnsibmFtZSI6Im5naW54LTZkZGRkZmM0N2QtZ2x3MmsiLCJ1aWQiOiIwYmYwNjViNy04YTFiLTExZWEtOGZmZC0wMmQ5ZGFkMjE0NTkifSwic2VydmljZWFjY291bnQiOnsibmFtZSI6InMzLWVjaG9lciIsInVpZCI6ImNiMmQ1MjNjLThhMTYtMTFlYS04MTg3LTEyNGQ4MGQyNTBhNSJ9fSwibmJmIjoxNTg4MTY1OTM2LCJzdWIiOiJzeXN0ZW06c2VydmljZWFjY291bnQ6ZGVmYXVsdDpzMy1lY2hvZXIifQ.UF1lGZIq9Ittno2OilnC6oHT6HULFlw-OYLah4eLjmTI3iLdWE5R7O_806OSsNgGVtRFJ082jXbKKUSv3xH-LFW3yC_a9aS-uzt8WxH7K_NvQcOFWY2g2mZ8X6wCrPcaFS-dtbrSkSwptlrSyNhc_SCObSo1FPZd5E2w2BDQHatC9_ROVObGKBM7pVIzivCoP76PUZsiI1EdYAXiX2EwsCBUlM_37qgjL6RGqHyQ4vk0wXOKa3G7JtQlSTF22xfIOCMrEJ_4pTbxS4h3MY2LM_w4bwYKp9A-vn_PzJnSId8vc8-dQfnv2-blRQ9Dpy4L9aVo8KmNVMqfmq5ukR7_OQ

ヘッダー

ペイロード

署名

Json Web Token の中身 (base64 decord)


```
{
  "alg":"RS256",
  "kid":"b6d01a962e25a02a139a38e352d91db48a0224bd”
}
{
  "aud":["sts.amazonaws.com"],
  "exp":1588252336,
  "iat":1588165936,
  "iss":"https://oidc.eks.us-east-1.amazonaws.com/id/XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
  "kubernetes.io":{
    "namespace":"default",
    "pod":{
      "name":"nginx-6ddddfc47d-glw2k",
      "uid":"0bf065b7-8a1b-11ea-8ffd-02d9dad21459”
    },
    "serviceaccount":{
      "name":"s3-echoer",
      "uid":"cb2d523c-8a16-11ea-8187-124d80d250a5”
    }
  },
  "nbf":1588165936,
  "sub":"system:serviceaccount:default:s3-echoer”
}
```

ヘッダー

ペイロード

CloudTrail も見てみる

#EKSMatsuri

	イベント時間	ユーザー名	イベント名	発信元 IP アドレス
▼	2020-04-29, 10:01:29 PM	system:serviceaccount:d…	AssumeRoleWithWebIdentity	54.86.228.21

AWS アクセスキー

AWS リージョン us-east-1

エラーコード

イベント ID 7c9a1627-39ed-410a-9f95-082812903fb6

イベント名 AssumeRoleWithWebIdentity

イベントソース sts.amazonaws.com

イベント時間 2020-04-29, 10:01:29 PM

読み取り専用 true

リクエスト ID 9cf4c019-620e-4af4-ab9f-f16f35e4ab5c

発信元 IP アドレス 54.86.228.21

ユーザー名 system:serviceaccount:default:s3-echoer

aws

IAM Roles for Service Accounts まとめ

#EKSMatsuri

- Kubernetes 側が IdP として認証を行い、IAM が認可を担う

- Amazon EKS であれば、Master Nodes 上で動かす必要のある amazon-eks-pod-identity-webhook や、OIDC Provider Endpoint の構築、運用が不要

- 対応している AWS SDK のバージョンを利用する必要がある
    - Pod 起動時に、AWS CLI で Credential を取得するなど、迂回方法が無いわけではない

aws

まとめ

- SAML, OIDC と組み合わせることで、Kubernetes はより便利によりセキュアに利用することが出来る

- Amazon EKS を利用することで、Master Nodes 上で動かす必要のあるモジュールの管理運用も AWS にオフロードすることが出来る
    - aws-iam-authenticator
    - amazon-eks-pod-identity-webhook

- CloudWatch Logs に出力される Master Nodes のログや、CloudTrail を確認することで、内部挙動を確認できる

Thank You !

Kazuki Matsuda  **mats16k**

Startup Solutions Architect
Amazon Web Services Japan