



このコンテンツは公開から3年以上経過しており内容が古い可能性があります
最新情報については[サービス別資料](#)もしくはサービスのドキュメントをご確認ください

[AWS Black Belt Online Seminar] Amazon Elastic Container Service for Kubernetes (Amazon EKS)

サービスカットシリーズ

アマゾンウェブサービスジャパン株式会社
ソリューションアーキテクト 浅野 佑貴
2019/4/10

AWS 公式 Webinar

<https://amzn.to/JPWebinar>



過去資料

<https://amzn.to/JPArchive>



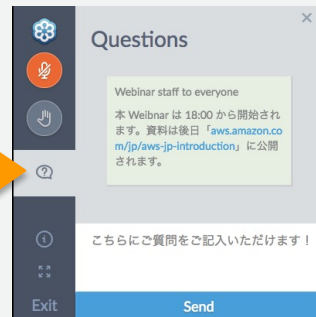
AWS Black Belt Online Seminar とは

「サービス別」「ソリューション別」「業種別」のそれぞれのテーマに分かれて、アマゾンウェブサービス ジャパン株式会社が主催するオンラインセミナーシリーズです。

質問を投げることができます！

- 書き込んだ質問は、主催者にしか見えません
- 今後のロードマップに関するご質問はお答えできませんのでご了承下さい

- ① 吹き出しをクリック
- ② 質問を入力
- ③ Sendをクリック



Twitter ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では2019年4月10日時点のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(<http://aws.amazon.com>)にてご確認ください。
- 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます。
- 価格は税抜表記となっております。日本居住者のお客様が東京リージョンを使用する場合、別途消費税をご請求させていただきます。
- AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at <http://aws.amazon.com/agreement/>. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

Agenda

- なぜコンテナなのか
- Kubernetes概要
- Amazon EKS概要

Agenda

- なぜコンテナなのか
- Kubernetes概要
- Amazon EKS概要

なぜコンテナなのか？



パッケージング

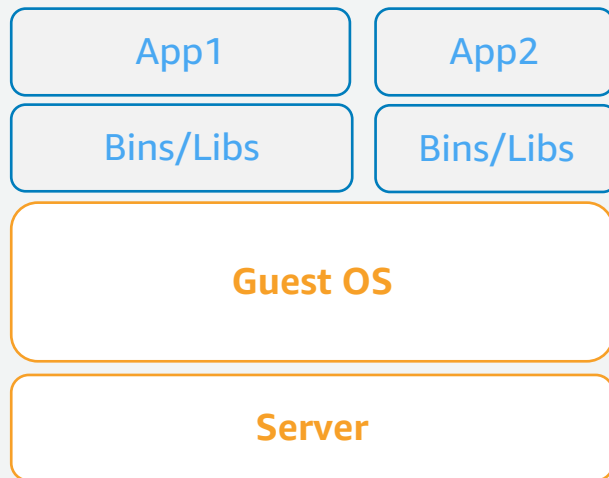


配布

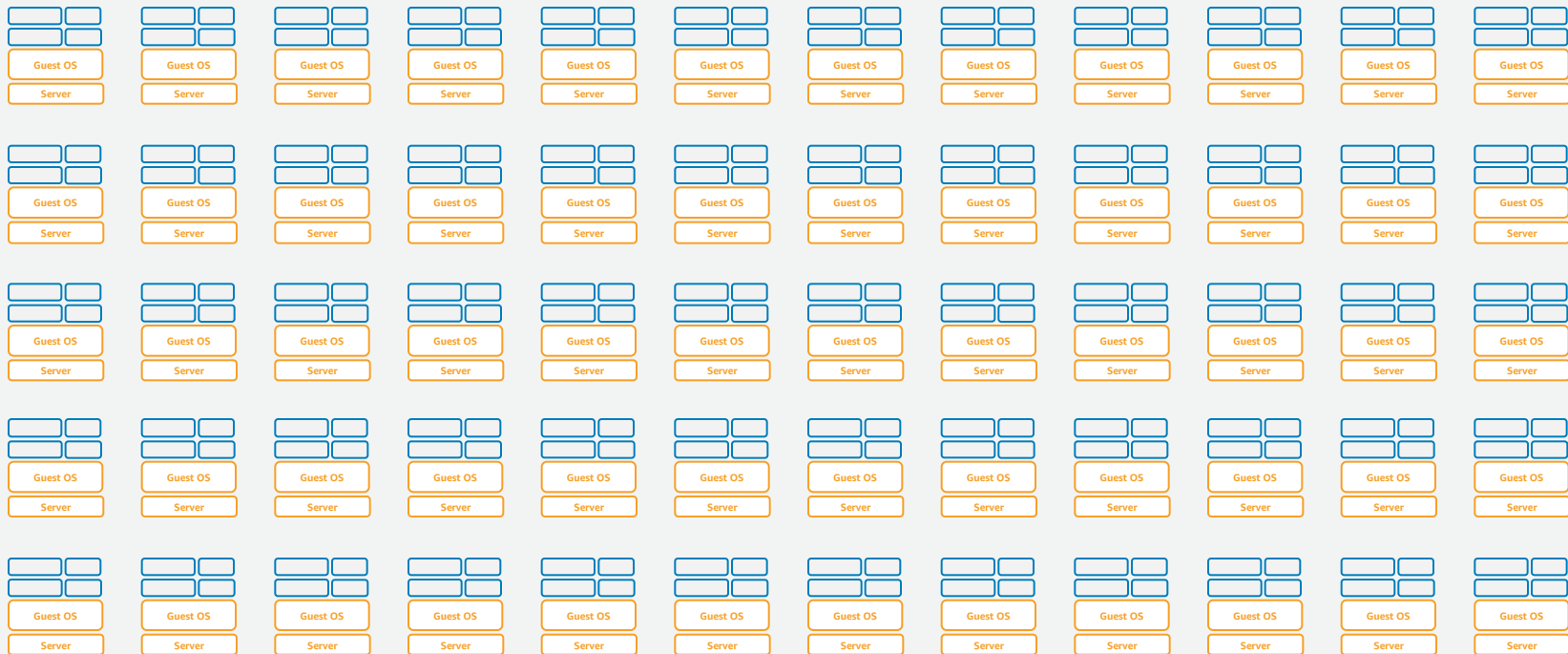


イミュータブル
インフラストラクチャ

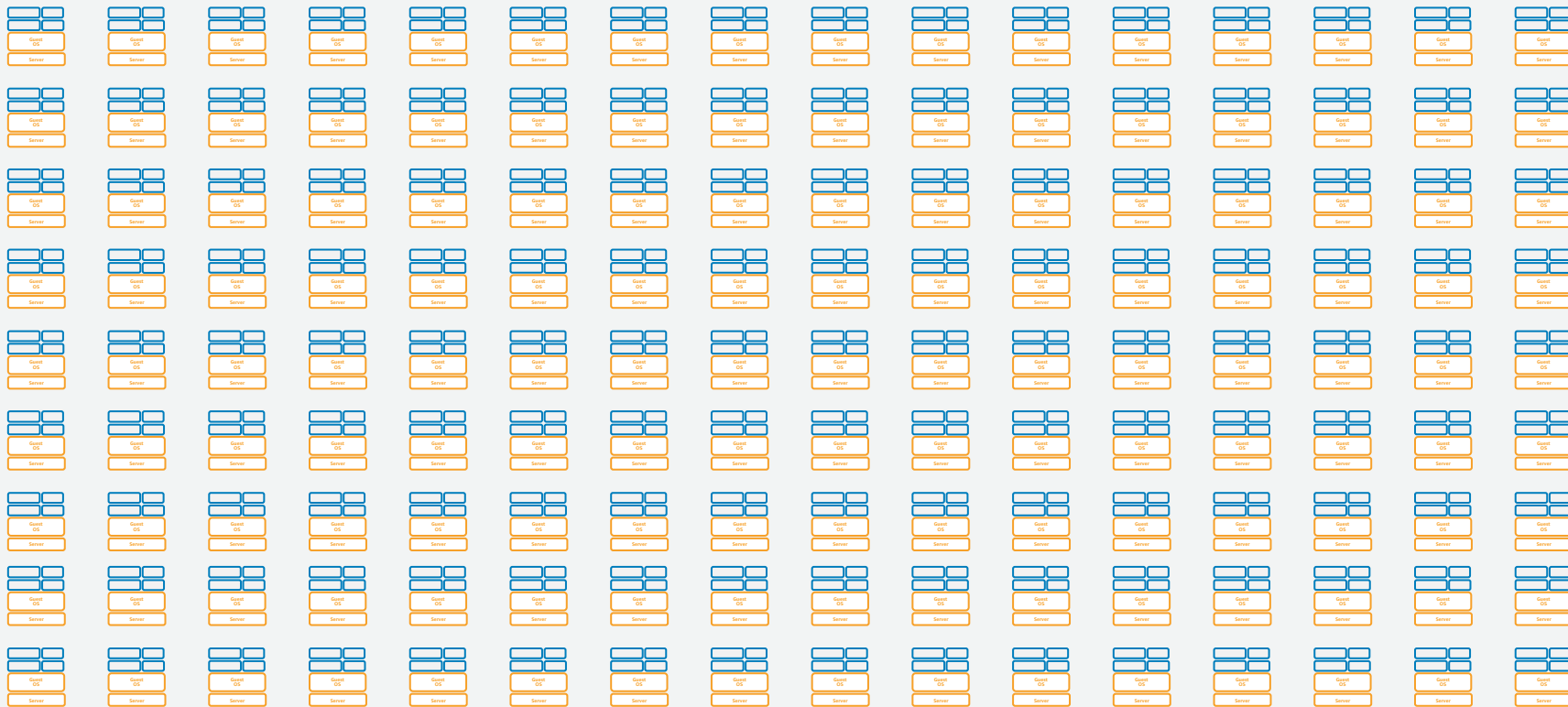
1台のサーバでDockerコンテナを使うのは簡単



サーバが増えると？



さらに増える？



AWSのコンテナ関連サービス

レジストリ

コンテナイメージのリポジトリ



Amazon EC2 Container Registry

コントロールプレーン

デプロイ, スケジューリング, スケーリング,
コンテナアプリケーションの管理



Amazon Elastic Container Service



Amazon Elastic Container Service for
Kubernetes

データプレーン

コンテナの実行環境



Amazon EC2



AWS Fargate

Agenda

- なぜコンテナなのか
- **Kubernetes概要**
- Amazon EKS概要

Kubernetes(K8s)とは

- 複数のホスト間でコンテナ化されたアプリケーションを管理するオープンソースシステム
- デプロイ、メンテナンス、スケーリングといった基本的な機能を提供している
- Cloud Native Computing Foundation (CNCF) によって管理、推進されている



Kubernetes(K8s)で実現できること

- ホスト管理、スケジューリング
- コンテナの死活監視
- オートリカバリ
- サービスディスカバリ、ロードバランシング
- シークレットやアプリケーション設定の管理
- バッチ実行
- エコシステムとの連携



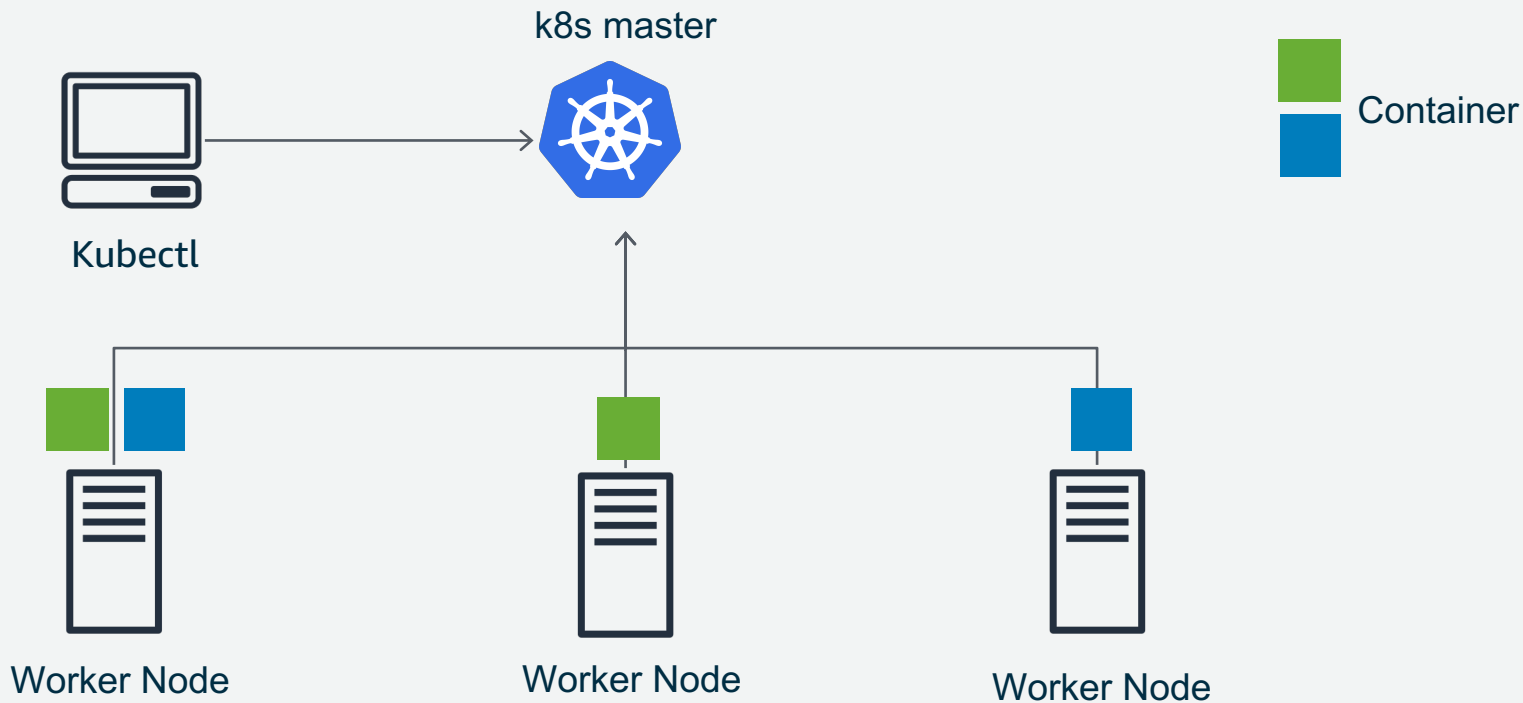
KubernetesとAWS



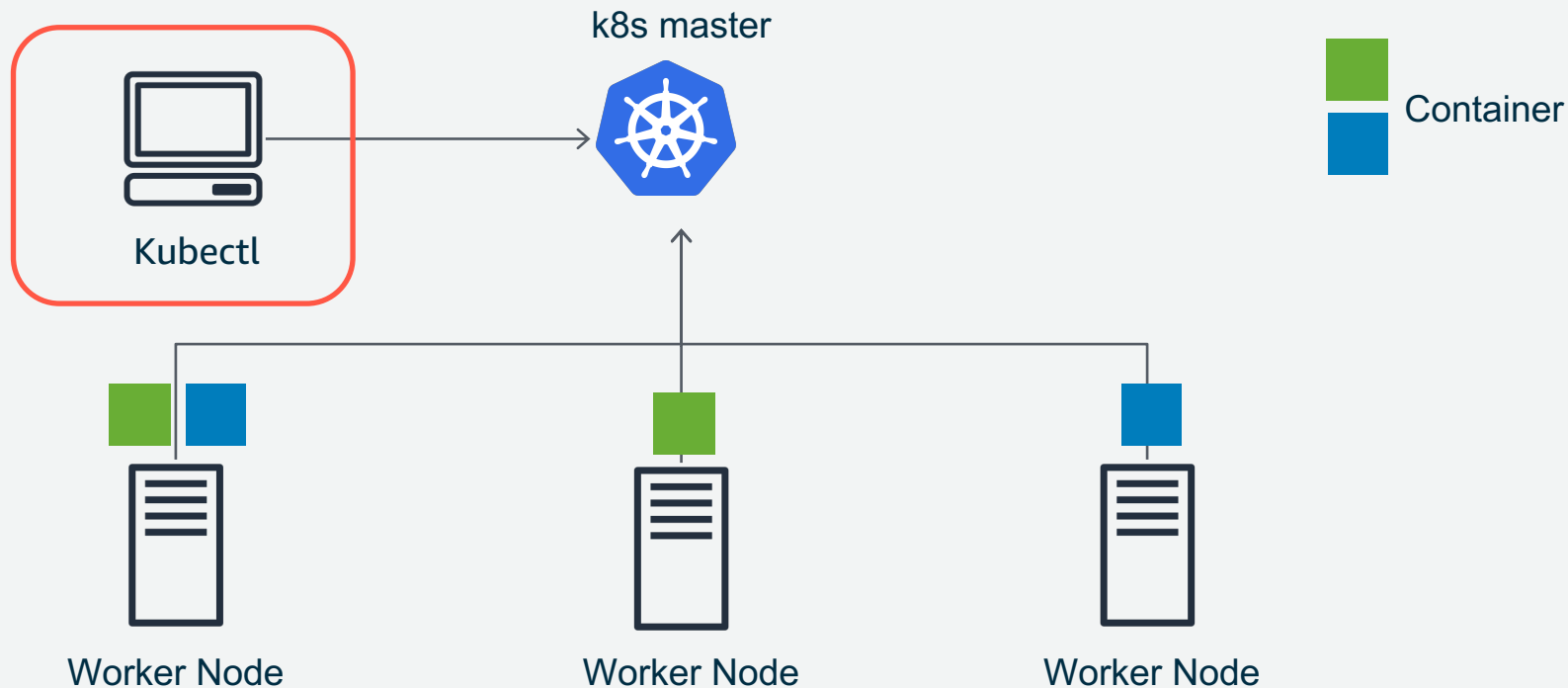
51%

of Kubernetes workloads
run on AWS today
— Cloud Native Computing
Foundation

Kubernetes(K8s) のアーキテクチャ



Kubernetes(K8s) のアーキテクチャ



kubectl

- k8s APIを実行する為のコマンドラインインターフェイス
- 簡単に使いやすい直感的な使い慣れたコマンド群
 - get, create, describe, delete, apply etc

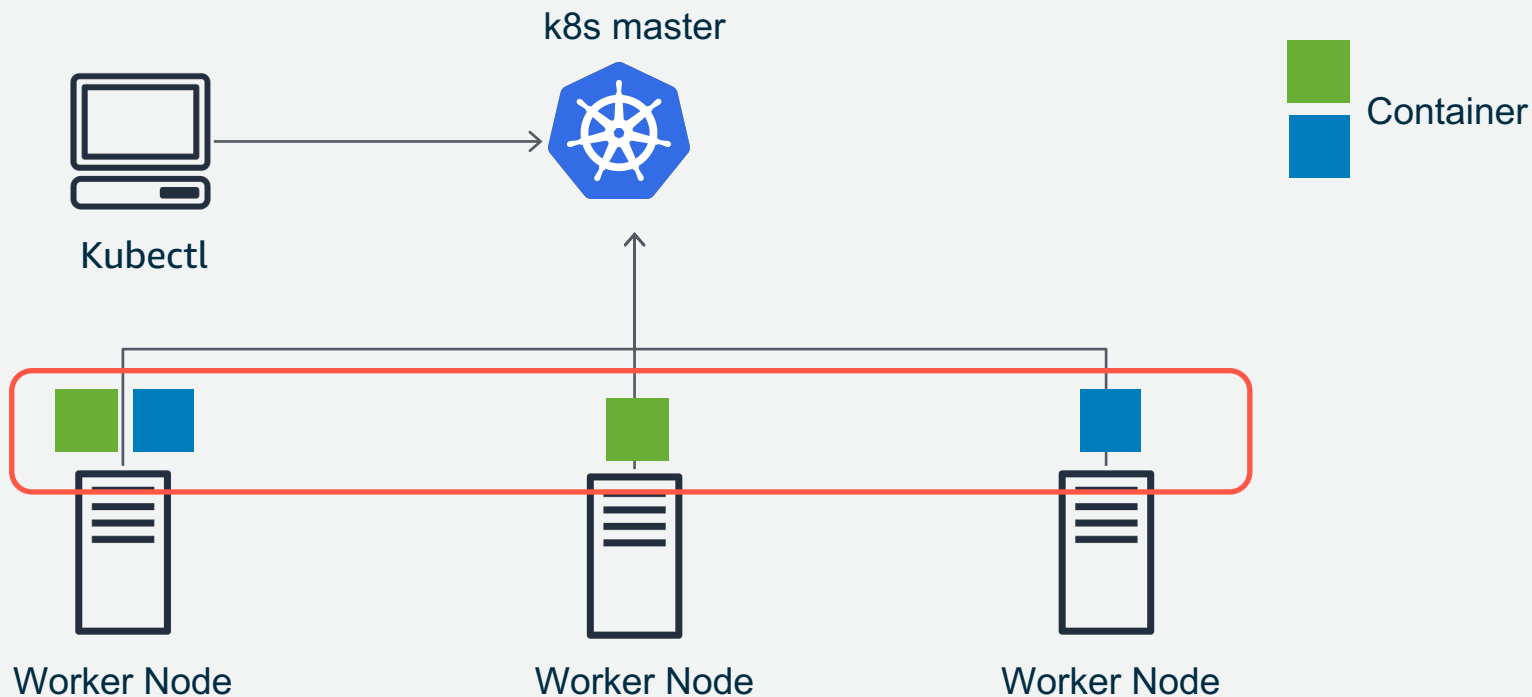
```
$ kubectl get nodes
NAME                                STATUS              AGE
ip-10-100-100-50.us-west-2.compute.internal  Ready,SchedulingDisabled  17m
ip-10-100-100-83.us-west-2.compute.internal  Ready                    16m
ip-10-100-101-39.us-west-2.compute.internal  Ready                    16m
ip-10-100-101-50.us-west-2.compute.internal  Ready,SchedulingDisabled  17m
ip-10-100-102-160.us-west-2.compute.internal  Ready                    16m
ip-10-100-102-50.us-west-2.compute.internal  Ready,SchedulingDisabled  18m
$
```



Kubectl



Kubernetes(K8s) のアーキテクチャ概要



k8sのオブジェクト例-Workloadsリソース

コンテナの実行に関連するリソース

- Pod
- ReplicaSet
- Deployment
- DaemonSet
- Job
- CronJob
- StatefulSet



Pods

- 作成、スケジュール、管理できる最小のデプロイ可能な単位
 - シンプルなユースケースであれば、1Podに1つのコンテナとなる
 - Sidecarの様に1 Podに複数のコンテナを含む事も勿論可能。
- 同じPod内のコンテナはVolumeやNetworkを共有する
 - Pod内のコンテナはlocalhostで通信可能

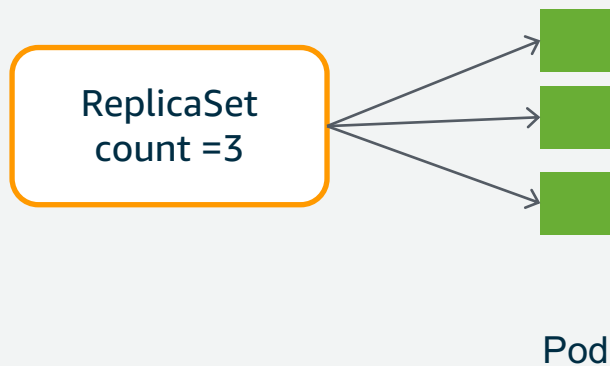
```
$ cat pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: pod.firstrun
spec:
  containers:
    - name: web
      image: nginx:latest
```

```
$ kubectl create -f pod.yaml
pod/pod.firstrun created

$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
pod.firstrun  1/1     Running   0           27s
```

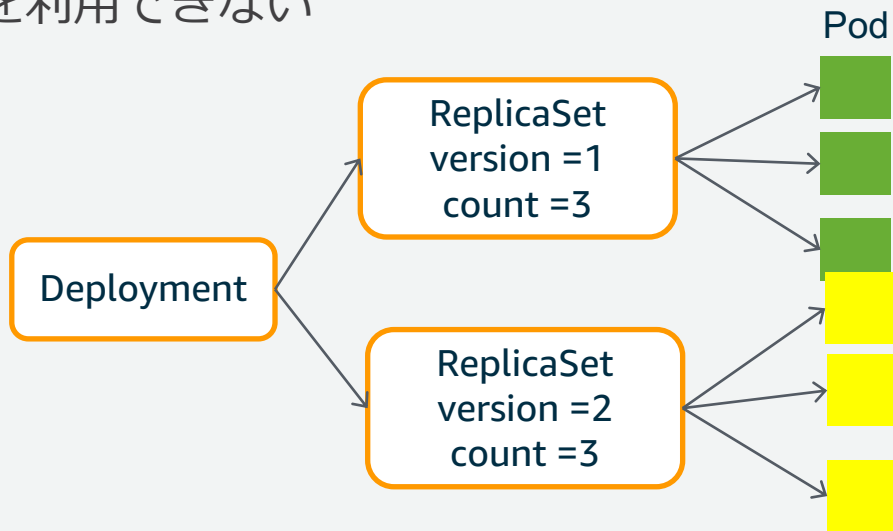
ReplicaSet

- Podのレプリカを作成し、Podを指定された数に維持するリソース
 - ロングランニングなPod向け
 - Worker Nodeの障害時には、指定されたPodの数を維持する為に別NodeにPodを作成する



Deployment

- 複数のReplicaSetを管理し、Podのローリングアップデートやロールバックを実現する為のリソース
- ReplicaSetでPodを利用した場合は、Podの更新時にローリングアップデートなどの機能を利用できない



Deploymentを利用したPodの作成例

```
$cat deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: deployment.firstrun
  labels:
    app: web
spec:
  replicas: 1
  selector:
    matchLabels:
      app: web
  template:
    metadata:
      labels:
        app: web
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
```

```
$ kubectl apply -f deployment.yaml
deployment.apps/deployment.firstrun created
```

```
$ kubectl get deployment
NAME                DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
deployment.firstrun  1        1        1           1          19s
```

```
$ kubectl get pod
NAME                READY  STATUS   RESTARTS  AGE
deployment.firstrun-9b6c8d86d-gvwg4  1/1    Running  0         2m
pod.firstrun                1/1    Running  0         17m
```

```
$ kubectl get replicaset
NAME                DESIRED  CURRENT  READY  AGE
deployment.firstrun-9b6c8d86d  1        1        1      6m
```

Deploymentの更新例

```
$cat deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: deployment.firstrun
  labels:
    app: web
spec:
  replicas: 1
  selector:
    matchLabels:
      app: web
  template:
    metadata:
      labels:
        app: web
    spec:
      containers:
        - name: httpd
          image: httpd:latest
          ports:
            - containerPort: 80
```

```
$ kubectl apply -f deployment.yaml
deployment.apps/deployment.firstrun configured
```

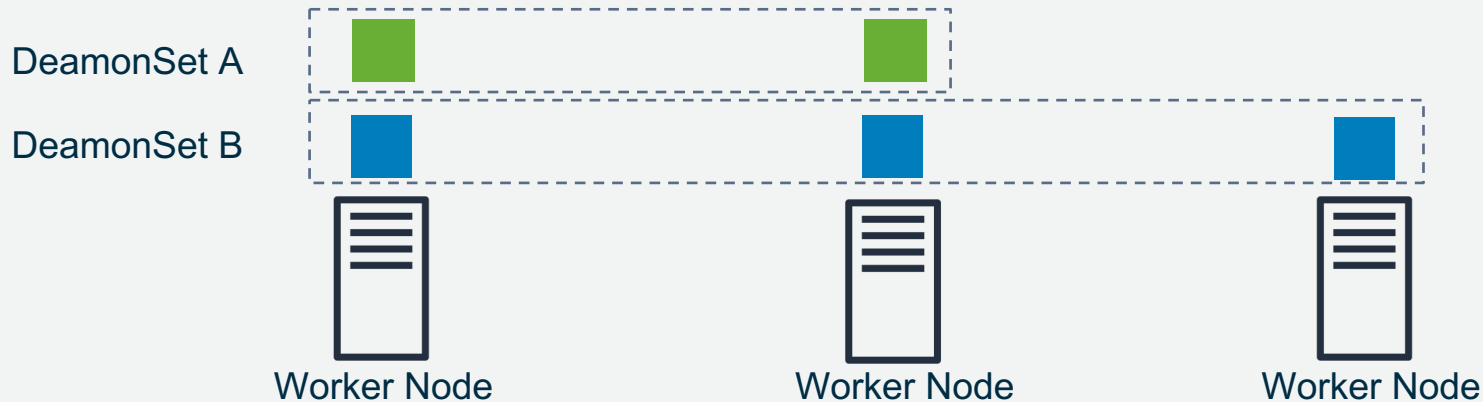
```
$ kubectl get deployment
NAME                DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
deployment.firstrun  1         1         1            1           19s
```

```
$ kubectl get pod
NAME                READY  STATUS   RESTARTS  AGE
deployment.firstrun-6cb5cf69d8-4fdhh  1/1    Running   0          6s
deployment.firstrun-9b6c8d86d-765zh  0/1    Terminating  0          2m
pod.firstrun                1/1    Running   0          16m
```

```
$ kubectl get replicaset
NAME                DESIRED  CURRENT  READY  AGE
deployment.firstrun-6cb5cf69d8  1         1         1      15s
deployment.firstrun-9b6c8d86d  0         0         0       2m
```

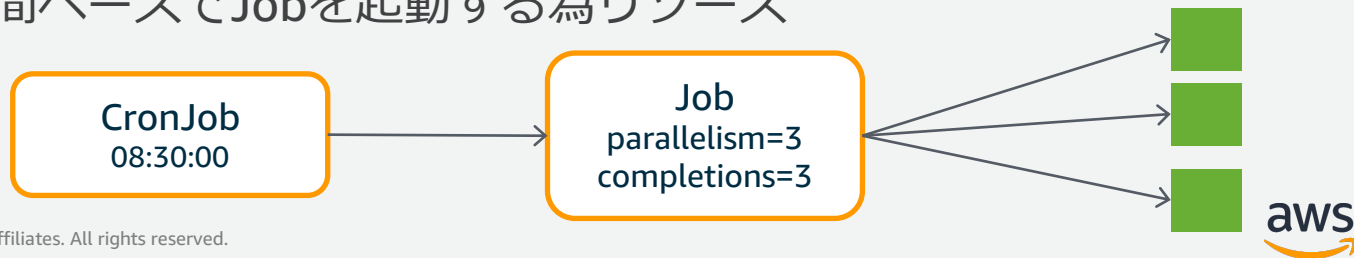

DaemonSet

- 選択したノードの組に対して、各ノードにPodを1つ動かす為のリソース
 - 1台のWorker nodeに複数つつPodを配置する事はできない
- ユースケース例
 - 各Worker nodeで動かすログ収集用のdaemon(fluentdなど)
 - 各Worker nodeで動かすモニタリング用のdaemon



Job

- 1つ以上のPodを作成し、指定された数のPodが正常に完了する様にコントロールするリソース
 - 指定した数のPodが正常に完了するとJob自体も完了する
 - Podが失敗/削除された場合、新しいPodを作成する
- JobとReplicaSetの違い
 - ReplicaSet: ロングランニングなPod向け
 - Job: 終了する事が予想されるPod向け
- CronJob : 時間ベースでJobを起動する為リソース



k8sのリソース例-Service/Ingressリソース

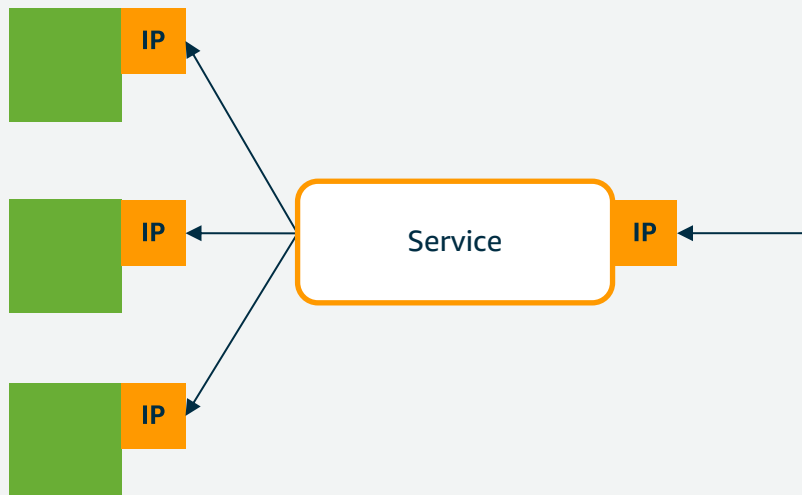
コンテナを公開する様なエンドポイントを提供するリソース

- Service
- Ingress



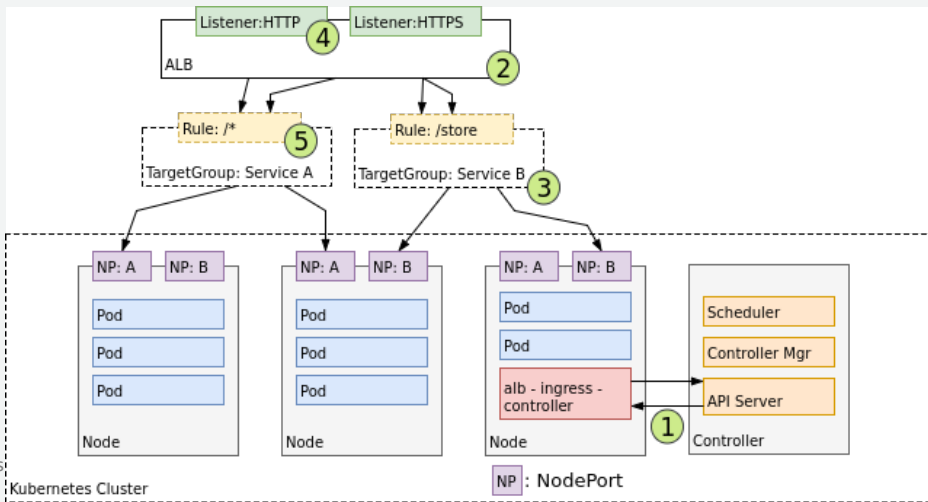
Service

- Podの論理セットとアクセスする為のポリシーを定義するリソース
- コンテナに対してトラフィックを流す事ができる(L4ロードバランシング)
- ServiceにIPアドレスを割り当てる事もできる

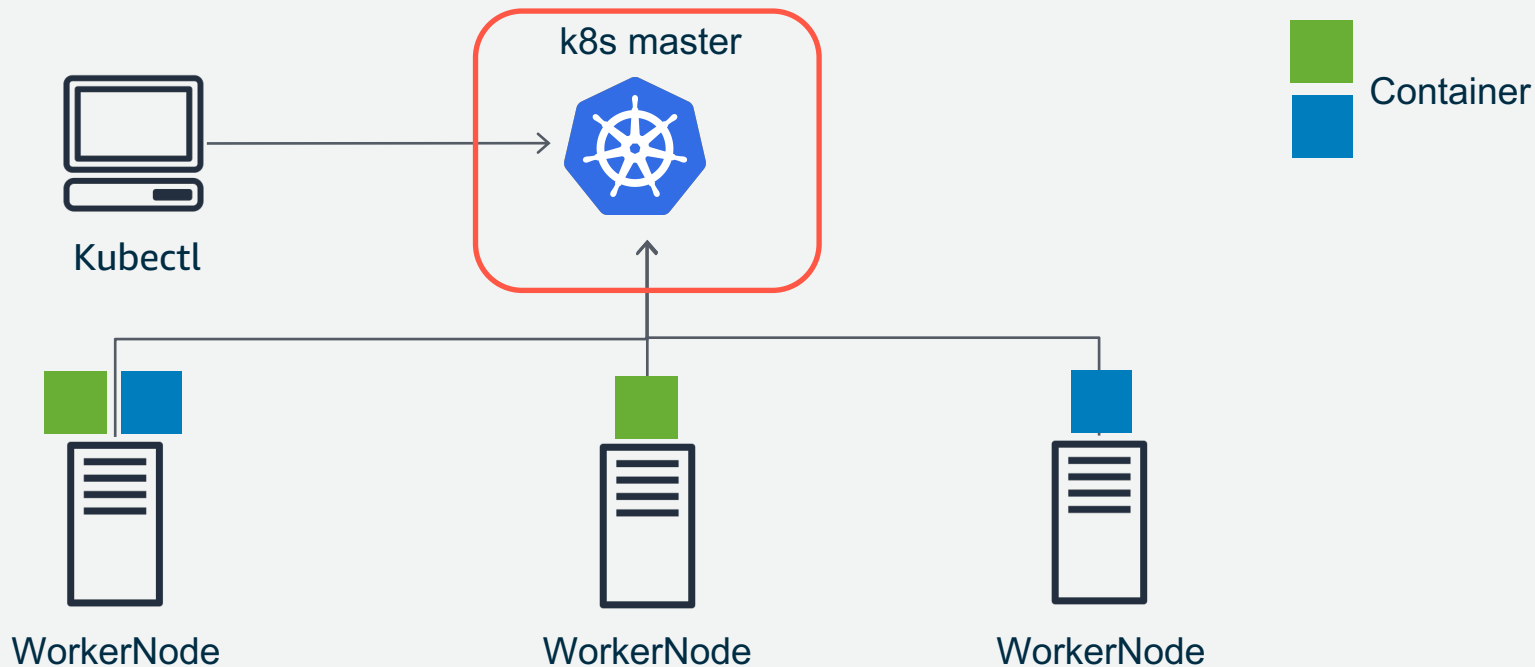


Ingress

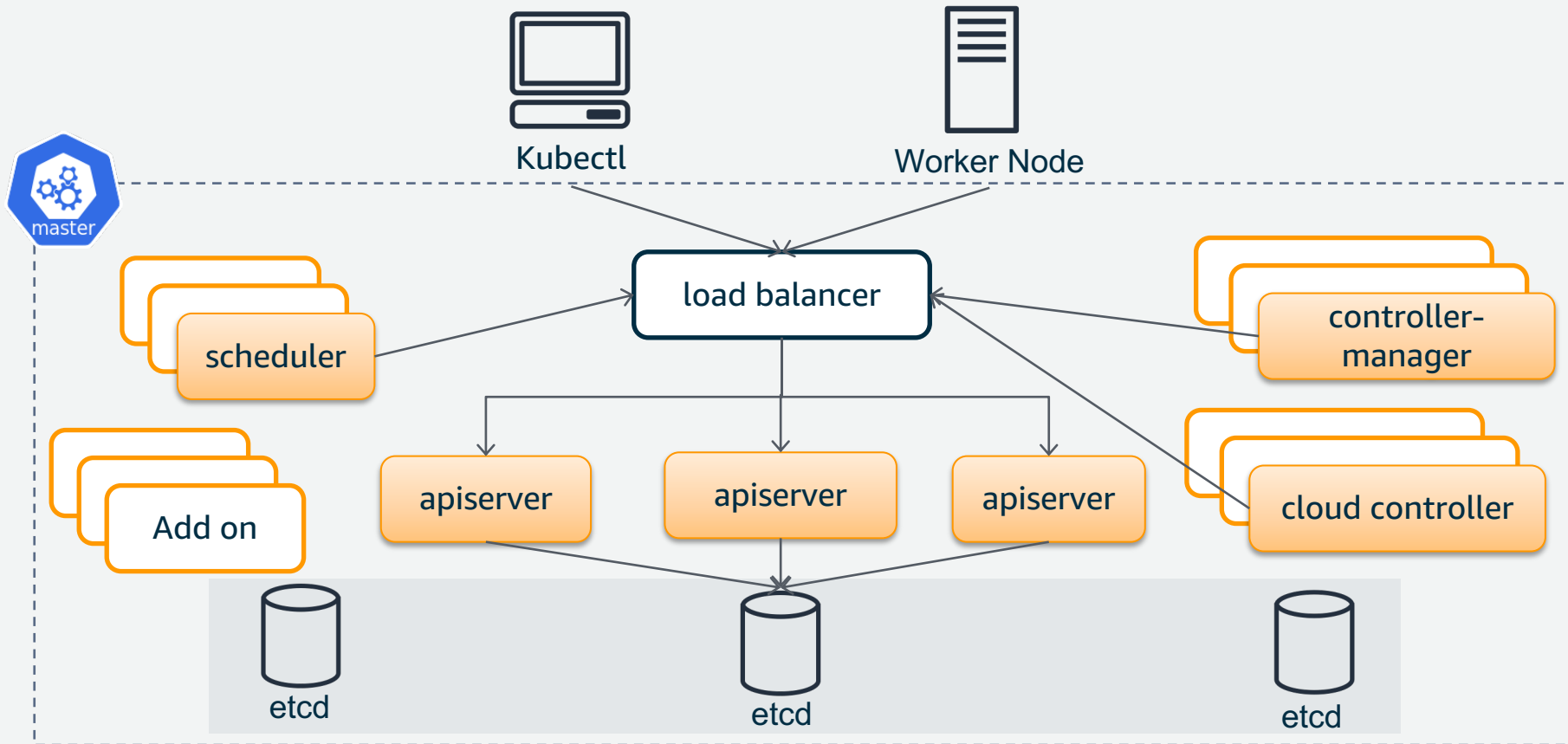
- コンテナに対してトラフィックを流す事ができる(L7ロードバランシング)
- AWS ALB Ingress コントローラー: AWSがサポート
 - Ingress リソースとして Application Load Balancer (ALB) を公開できる
 - ホスト名またはパスによるコンテンツベースルーティングをサポートしたL7の負荷分散が可能



Kubernetes(K8s) アーキテクチャ-Control plane



Kubernetes(K8s) アーキテクチャ-Control plane





“私の代わりにKubernetesを実行して”



“AWSとネイティブに統合”



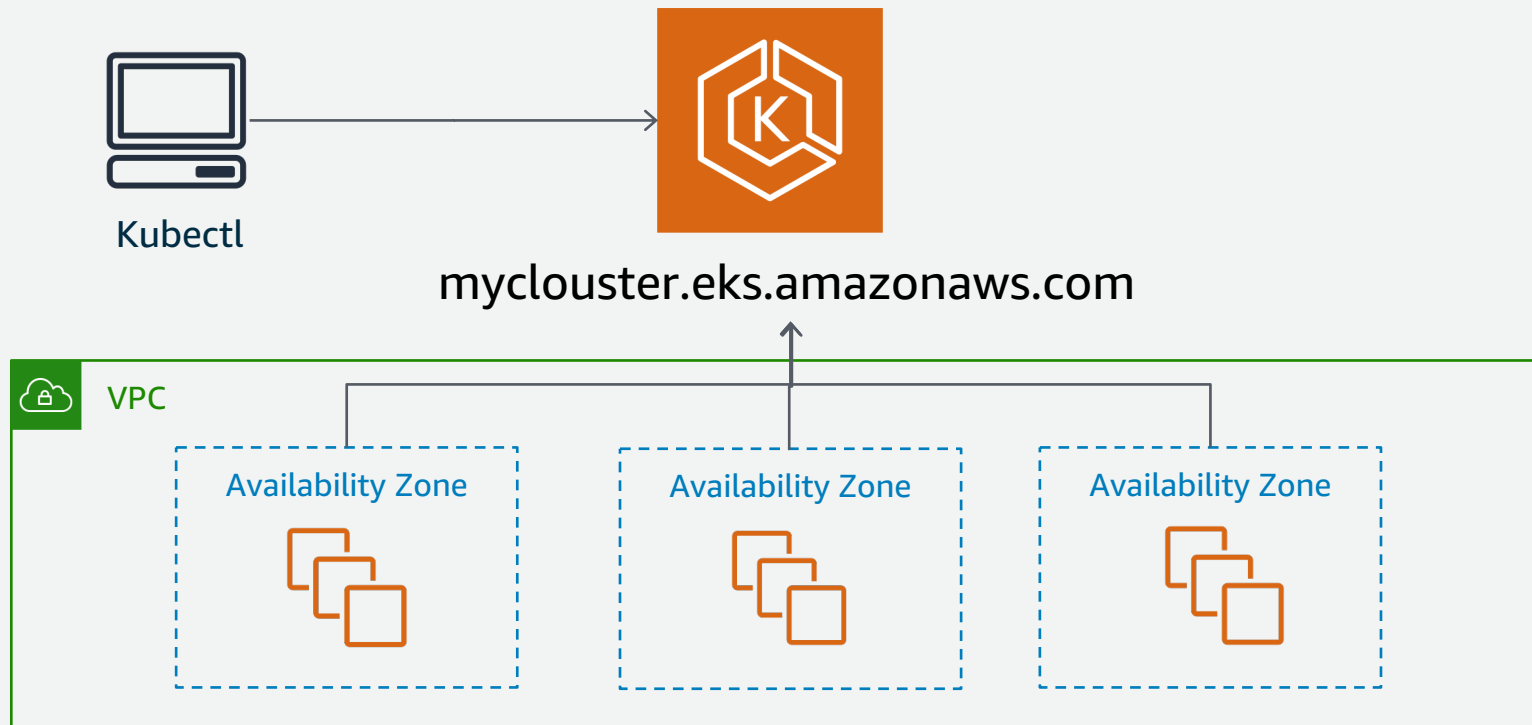
"オープンソースとしてのKubernetes 体験"

東京リージョンを含む13リージョンで利用可能



ELASTIC CONTAINER SERVICE FOR KUBERNETES
(EKS)

K8s Control planeをマネージドサービスとして提供



Agenda

- なぜコンテナなのか
- Kubernetes概要
- Amazon EKS概要

Amazon Elastic Container Service for Kubernetes (Amazon EKS)

- 運用難易度の高いKubernetes Controle planeをマネージドサービスで提供
- KubernetesエコシステムのOSSやツールがそのまま動かせる
- AWSサービスとの連携
- EKSサービスチームとAWS OSSチームによるKubernetesコミュニティへの貢献



2018年の主要アップデート

update

April: Kubernetes準拠に適合

June: Amazon EKS GA!

June: HIPAA準拠

July: GitHubでAMIビルドスクリプトを公開

August: EKS最適化AMIとノード展開用のCloud Formationテンプレートを更新

August: GPUインスタンスのサポート

August: プラットフォームバージョン2のリリース

August: カスタムメトリックのHPAをサポート

September: アイルランド、ダブリンリージョンで利用可能に

September: update-kubeconfigコマンドの追加

October: Dynamic Admission Controllers (Istio)のサポート

November: オハイオリージョンで利用可能に

December: 東京リージョンで利用可能に

Amazon EKSのSLA /コンプライアンス対応状況

update

SLA:99.9 %

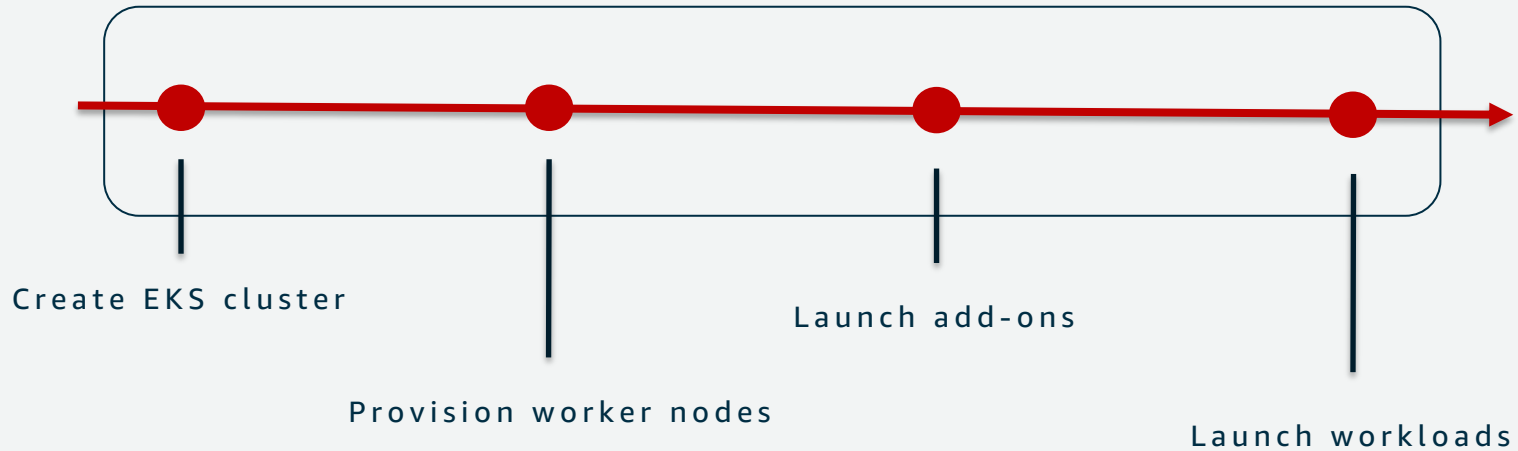
利用できる全リージョンが対応

- 準拠しているコンプライアンスプログラム
- HIPAA-eligible
- ISO 9001, 27001, 27017, 27018
- PCI DSS Level 1

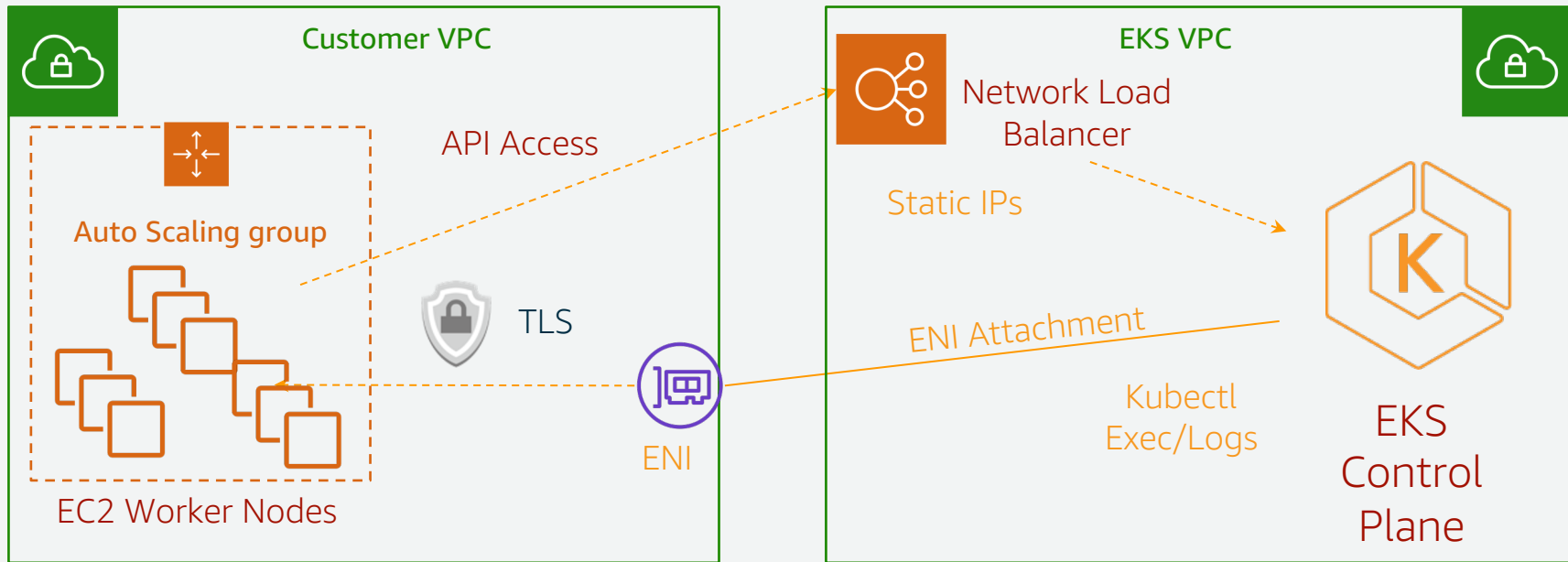
<https://aws.amazon.com/jp/eks/sla/>

<https://aws.amazon.com/jp/compliance/services-in-scope/>

EKSの開始方法

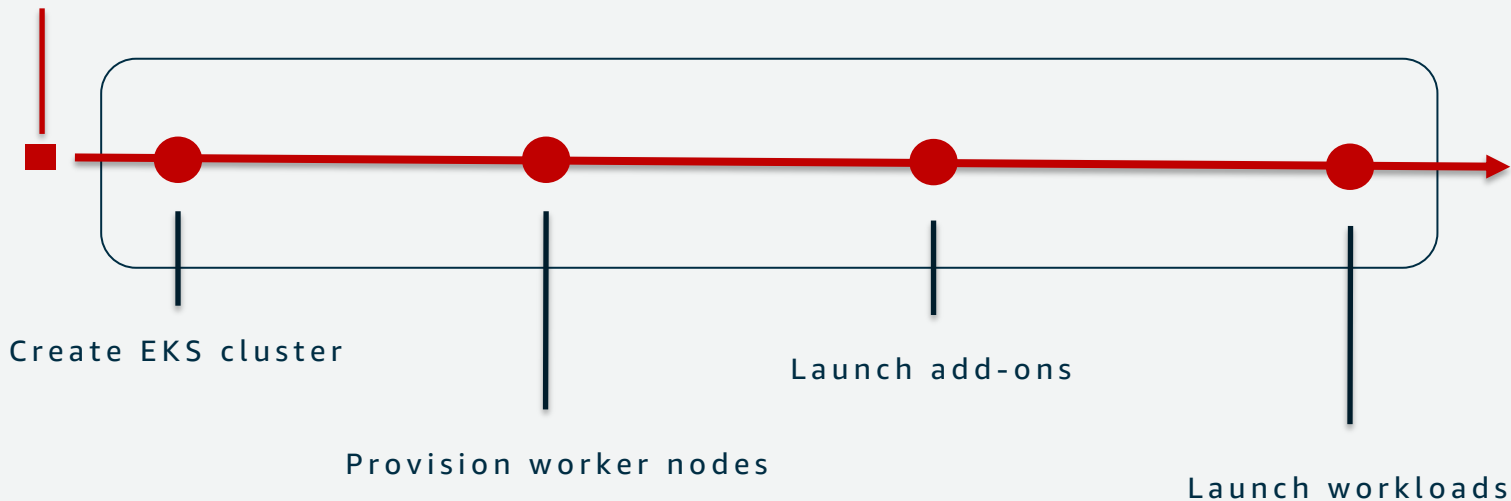


EKSアーキテクチャ



EKSの開始方法

preparation



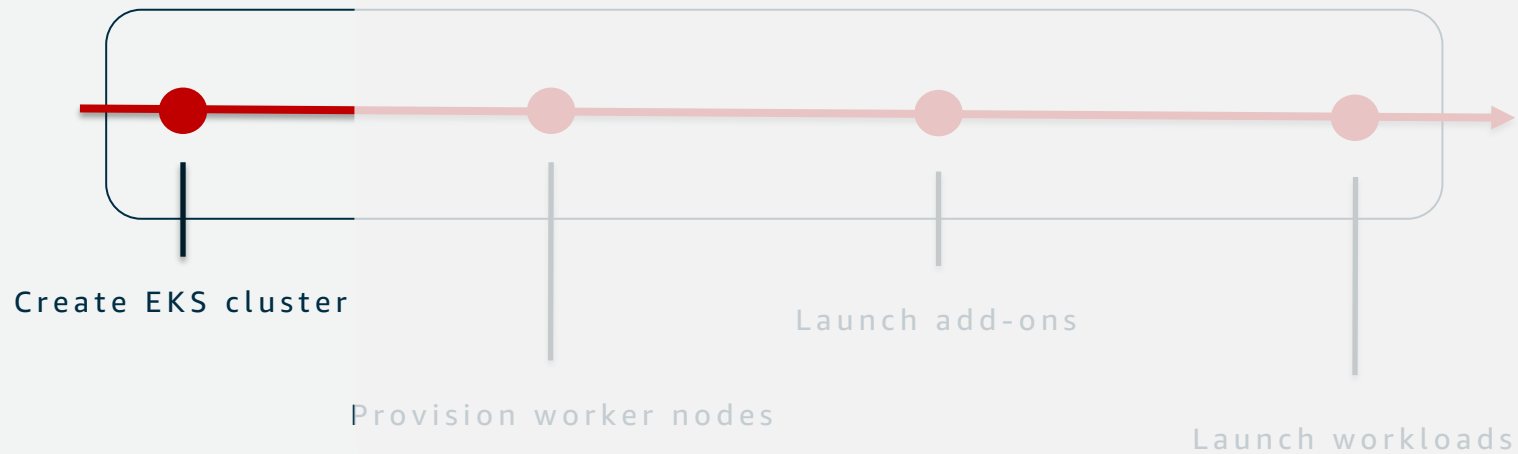
preparation

- クライアント環境のセットアップ
 - kubectlのインストール
 - aws-iam-authenticatorのインストール

- AWS環境のセットアップ
 - EKSサービスロールを作成する
 - (任意)EKS用のVPCを作成する
 - EKSコントロールプレーンの通信用に利用するSecurityGroupが必要

https://docs.aws.amazon.com/ja_jp/eks/latest/userguide/getting-started.html#eks-prereqs

EKSの開始方法

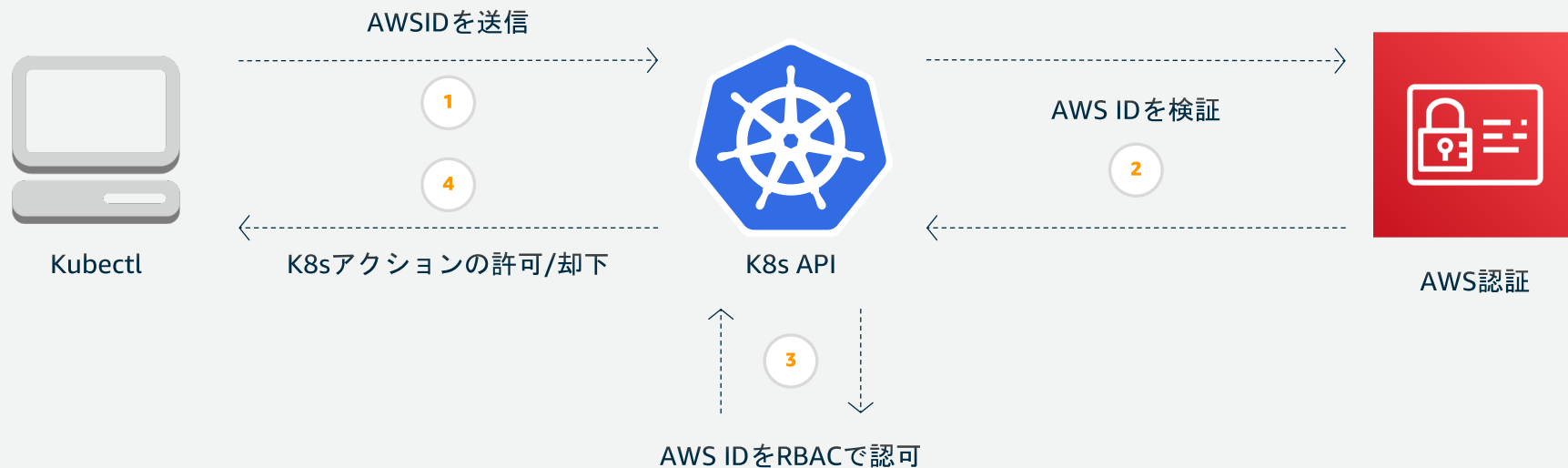


Create EKS cluster

```
aws eks create-cluster --name eks-first-run --role-arn arn:aws:iam::account-id:role/eksServiceRole --resources-vpc-config subnetIds=subnet-public-az1-id,subnet-private-az1-id,subnet-private-az2-id,securityGroupIds=security-group-name
```

IAM roles for Kubectl

github.com/kubernetes-sigs/aws-iam-authenticator



K8s API サーバーエンドポイントへのアクセス

update

- Work Node/kubectl と EKSにより提供されているK8s APIサーバ間のトラフィックをVPC内に制限することが可能に
 - Endpoint Public Accessを無効にすることでインターネット経由のアクセスを無効化
- Private Accessを有効化する場合
 - VPCで以下を有効にする
 - enableDnsHostnames
 - enableDnsSupport

Update API server endpoint access

一般設定
クラスター名 eks-firstrun
Networking
API server endpoint access Configure access to the Kubernetes API server endpoint.
Private access Configure access to the API server endpoint from within your VPC. <input type="radio"/> Disabled
Public access Configure access to the API server endpoint from outside of your VPC. <input checked="" type="radio"/> Enabled

キャンセル

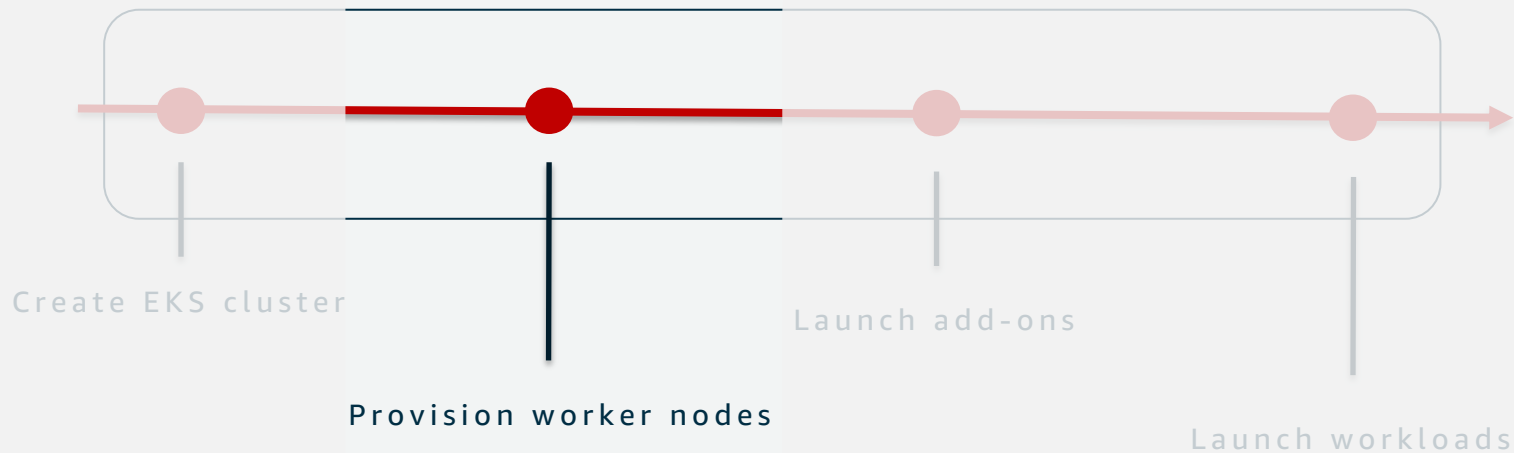
UPDATE

kubeconfig の作成

```
$ aws eks --region region update-kubeconfig --name cluster_name
```

```
apiVersion: v1
clusters:
- cluster:
  server: <endpoint-url>
  certificate-authority-data: <base64-
  encoded-ca-cert>
  name: kubernetes
contexts:
- context:
  cluster: kubernetes
  user: aws
  name: aws
current-context: aws
kind: Config
preferences: {}
users:
- name: aws
  user:
    exec:
      apiVersion:
      client.authentication.k8s.io/v1alpha1
      command: aws-iam-authenticator
      args:
      - "token"
      - "-i"
      - "<cluster-name>"
```

EKSの開始方法



Provision worker nodes

Worker Nodeのデプロイ用AWS CloudFormationテンプレートを提供

パラメータ

EKS Cluster

ClusterName The cluster name provided when the cluster was created. If it is incorrect, nodes will not be able to join the cluster.

ClusterControlPlaneSecurityGroup The security group of the cluster control plane.

Worker Node Configuration

NodeGroupName Unique identifier for the Node Group.

NodeAutoScalingGroupMinSize Minimum size of Node Group ASG.

NodeAutoScalingGroupMaxSize Maximum size of Node Group ASG.

NodeInstanceType EC2 instance type for the node instances

NodeImageId AMI id for the node instances.

NodeVolumeSize Node volume size

KeyName The EC2 Key Pair to allow SSH access to the instances

BootstrapArguments Arguments to pass to the bootstrap script. See files/bootstrap.sh in <https://github.com/awslabs/amazon-eks-ami>

Worker Network Configuration

VpcId The VPC of the worker instances

Subnets
 The subnets where workers can be created.

- Stack name
- ClusterName
- ClusterControlPlaneSecurityGroup
- NodeGroupName
- NodeAutoScalingGroupMinSize
- NodeAutoScalingGroupMaxSize
- NodeInstanceType
- NodeImageId
- KeyName
- VpcId
- Subnets

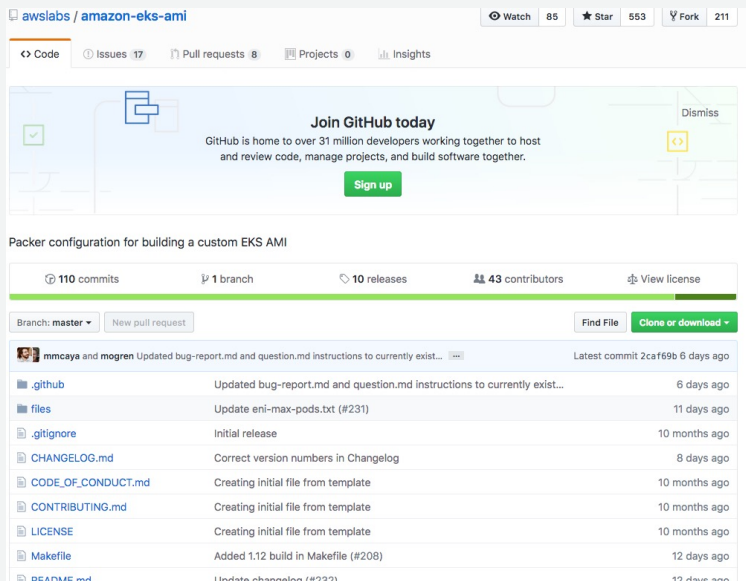
Amazon EKS 最適化 AMI

- Amazon Linux 2
- 設定なしでEKSと動作する事が可能
 - Docker,kublet,AWS IAMオーセンティケータが含まれる
 - コントロールプレーンを自動的に検知して接続を許可するbootstrap script
- GPUに対応したAMIも提供
 - P2およびP3インスタンスのみをサポート
 - NVIDIA end user license agreement (EULA)への同意が必要

<https://docs.aws.amazon.com/eks/latest/userguide/eks-optimized-ami.html>

カスタマイズAMIも利用可能

- Amazon EKS としてサービスチームが用意しているAMIと同じ構成のAMIを作成できるPackerのスク립トをGithubで公開



The screenshot shows the GitHub repository page for 'aws-labs / amazon-eks-ami'. At the top, it displays 'aws-labs / amazon-eks-ami' with 85 watches, 553 stars, and 211 forks. Below this, there are navigation links for Code, Issues (17), Pull requests (8), Projects (0), and Insights. A prominent banner encourages users to 'Join GitHub today' with a 'Sign up' button. The main content area is titled 'Packer configuration for building a custom EKS AMI' and shows repository statistics: 110 commits, 1 branch, 10 releases, and 43 contributors. The current branch is 'master'. A list of recent commits is visible, including updates to bug reports, question reports, and initial releases of files like .github, files, .gitignore, CHANGELOG.md, CODE_OF_CONDUCT.md, CONTRIBUTING.md, LICENSE, Makefile, and README.md.

Worker NodeをClusterに参加させる

```
$ curl -O https://amazon-eks.s3-us-west-2.amazonaws.com/cloudformation/2018-11-07/aws-auth-cm.yaml
$ kubectl apply -f aws-auth-cm.yaml
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
    - rolearn: <ARN of instance role>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
```

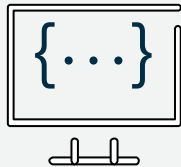


Container Networking Interface (CNI)

<https://github.com/aws/amazon-vpc-cni-k8s>



CNIプラグインによる
ネイティブVPC
ネットワーキング



複数のPodはVPC内に存在
するようにPod内に同じ
VPCアドレスを持つ

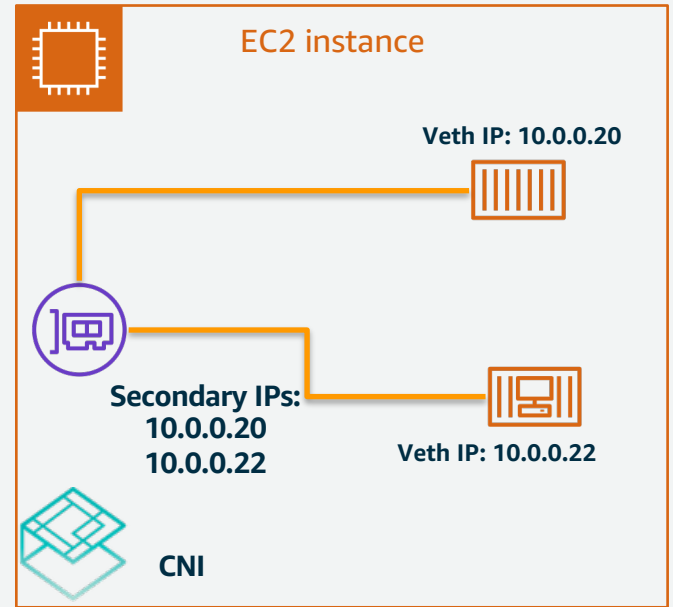
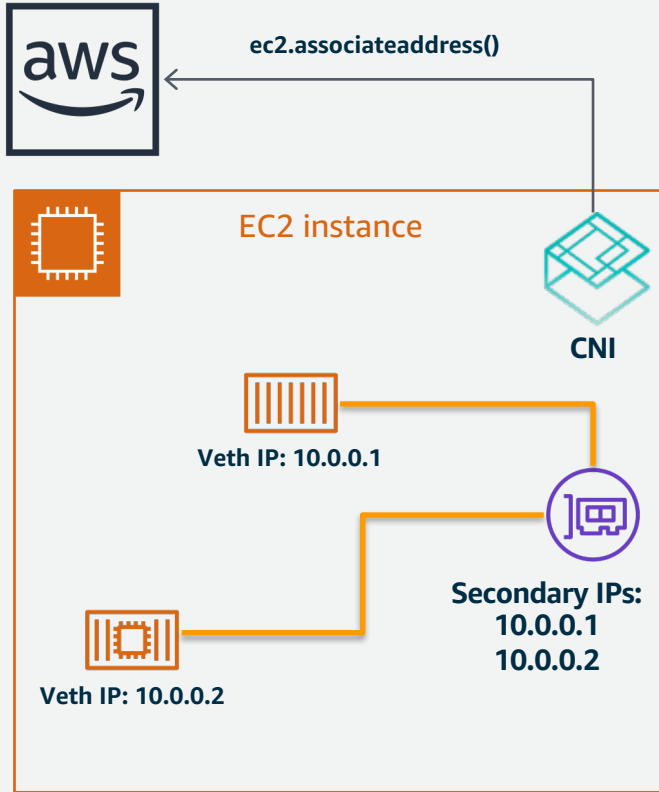


シンプルでセキュアな
ネットワーク

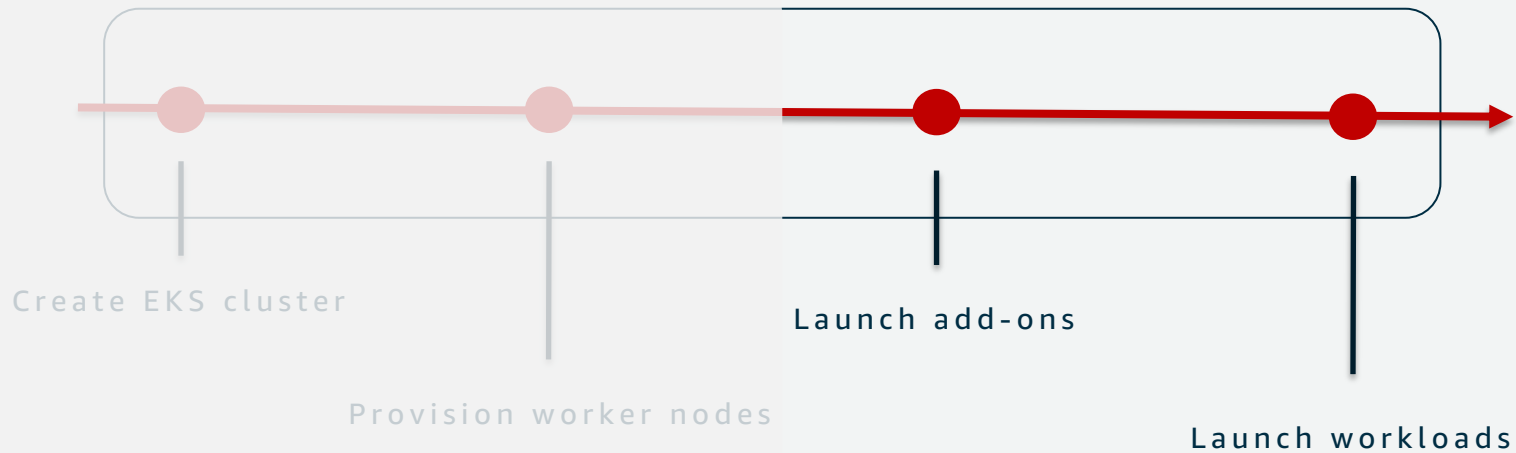


GitHub上で
公開されている
オープンソース

VPC CNI plugin



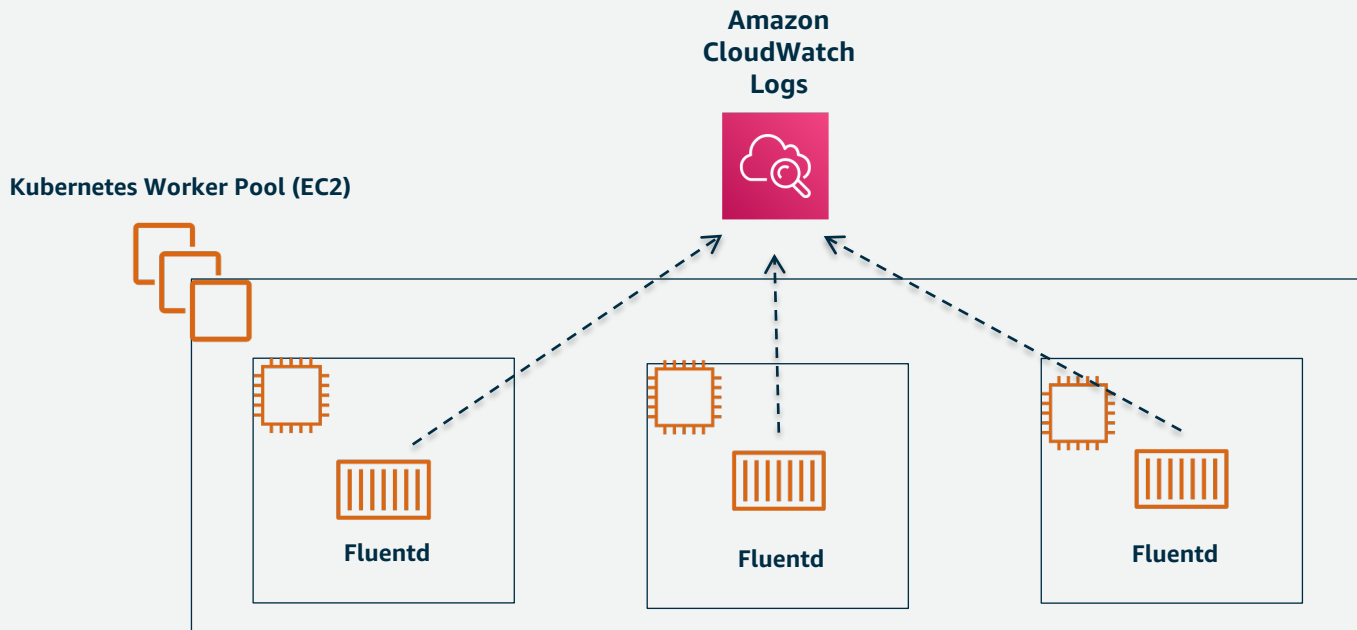
EKSの開始方法



オートスケーリング

- Worker Nodeの場合 : Cluster Autoscaler
 - 詳細はGitHubのドキュメントを参照
 - <https://github.com/kubernetes/autoscaler/tree/master/cluster-autoscaler/cloudprovider/aws>
- Podの場合 : Horizontal Pod Autoscaler(HPA)
 - targetAverageUtilizationにCPU利用率が近づく様にスケーリング
 - プラットフォームバージョン 1.10 eks.2、1.11以降から対応
 - Kubernetes Metrics Server バージョン 0.3.0 以上が必要

Fluentdを用いた Cloudwatch Logsへのログ集約



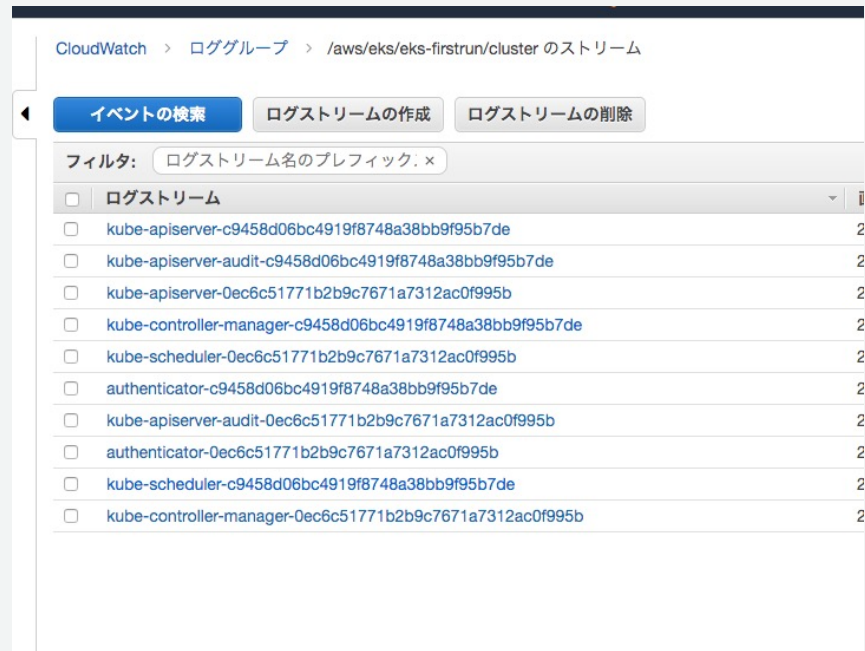
Fluentd Daemonset

Ensures a pod with a Fluentd container on each node in the worker pool with the host's `/var/lib/docker/containers` mounted so that it can package and ship container logs to CWLogs.

CloudWatch LogsへControl planeログを配信可能に

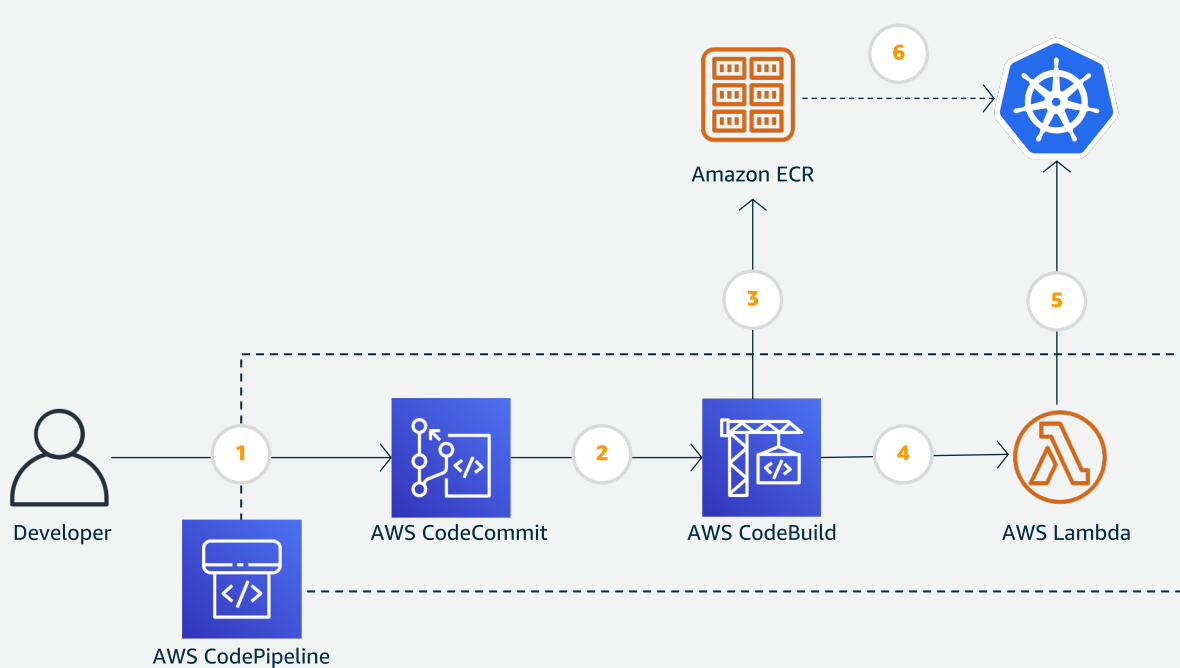
update

- EKSクラスターへの監査やアクティビティを監視が可能に
- 次のControl planeのログが送信可能
 - audit
 - API server
 - authenticator
 - controller-manager
 - scheduler



<https://aws.amazon.com/about-aws/whats-new/2019/04/amazon-eks-now-delivers-kubernetes-control-plane-logs-to-amazon/>

EKS Deployment Pipeline



- 1 Developerがメインブランチに変更をpush
- 2 新しいバージョンが作成されるとパイプラインの実行をトリガー
- 3 build id でタグ付けされたイメージをECRにpush
- 4 アプリケーションデプロイのためにLambdaをトリガー
- 5 Golang SDK を利用してDeploymentを更新
- 6 新しいコンテナイメージを取得し、ローリングアップデート

Preview/alpha projects

update

- Public Preview of Windows Container Support

<https://aws.amazon.com/about-aws/whats-new/2019/03/amazon-eks-opens-public-preview-of-windows-container-support/>

- CSI Drivers for Amazon EFS and Amazon FSx for Lustre

<https://aws.amazon.com/about-aws/whats-new/2019/04/aws-introduces-csi-drivers-for-amazon-efs-and-amazon-fsx-for-lus/>

コンテナ関連サービスのロードマップの一部を公開

The screenshot displays the GitHub repository 'aws / containers-roadmap' with a Kanban board. The board is organized into five columns: Researching (13 items), We're Working On It (29 items), Coming Soon (9 items), Developer Preview (1 item), and Just Shipped (39 items). Each card represents a feature request or update, including details like the service name, issue number, and the person who opened it. For example, in the 'Researching' column, there are cards for 'EKS [request]: allow to configure ipv6 kube-proxy mode', 'ECS CodeDeploy canary deployments', and 'EKS [request]: Security Groups per Pod'. The 'Just Shipped' column includes 'EKS Control Plane Logs', 'Amazon EFS CSI Driver for Kubernetes', and 'Kubernetes Version 1.11.8'.

<https://github.com/aws/containers-roadmap>

4月のBlack Belt Online Seminar 配信予定

<https://amzn.to/JPWebinar>

4/2 (火) 12:00-13:00 Let's dive deep into AWS Lambda Part1

4/9 (火) 12:00-13:00 Let's dive deep into AWS Lambda Part2

4/10 (水) 18:00-19:00 Amazon Elastic Container Service for Kubernetes
(Amazon EKS)

4/17 (水) 18:00-19:00 Amazon VPC Advanced

4/23 (火) 12:00-13:00 ヘルスケア・ライフサイエンス業界におけるAWS活用

4/24 (水) 18:00-19:00 Amazon Aurora MySQL Compatible Edition latest update



AWS の日本語資料の場所「AWS 資料」で検索

AWS クラウドサービス活用資料集トップ

アマゾン ウェブ サービス (AWS) は安全なクラウドサービスプラットフォームで、ビジネスのスケールと成長をサポートする処理能力、データベースストレージ、およびその他多種多様な機能を提供します。お客様は必要なサービスを選択し、必要な分だけご利用いただけます。それらを活用するために役立つ日本語資料、動画コンテンツを多数ご提供しております。(本サイトは主に、AWS Webinar で使用した資料およびオンデマンドセミナー情報を掲載していません。)

AWS Webinar お申込 »

AWS 初心者向け »

サービス別資料 »

<https://amzn.to/JPArchive>

[申込受付中！] AWS Innovate オンラインカンファレンス



INNOVATE
ONLINE CONFERENCE

2019年4月8日(月) ~ 5月7日(火) 開催

今すぐ無料参加申込み >>

<https://amzn.to/AWSInnovateJP>

AWS Well-Architected 個別技術相談会

毎週“W-A個別技術相談会”を実施中

- AWSのソリューションアーキテクト(SA)に
対策などを相談することも可能

- 申込みはイベント告知サイトから

(<https://aws.amazon.com/jp/about-aws/events/>)

AWS イベント で[検索]



AWS Well-Architected



aws

ご視聴ありがとうございました

AWS 公式 Webinar

<https://amzn.to/JPWebinar>



過去資料

<https://amzn.to/JPArchive>

