
AWS
re:Invent

NET204

Looking at Amazon EKS through a networking lens

Ikenna Izugbokwe

Principal Solutions Architect
AWS

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

- Kubernetes cluster networking
- Amazon Elastic Kubernetes Service (Amazon EKS) overview
- Amazon Virtual Private Cloud (Amazon VPC) and pod networking
- Elastic Load Balancing (ELB) integration



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Kubernetes cluster networking



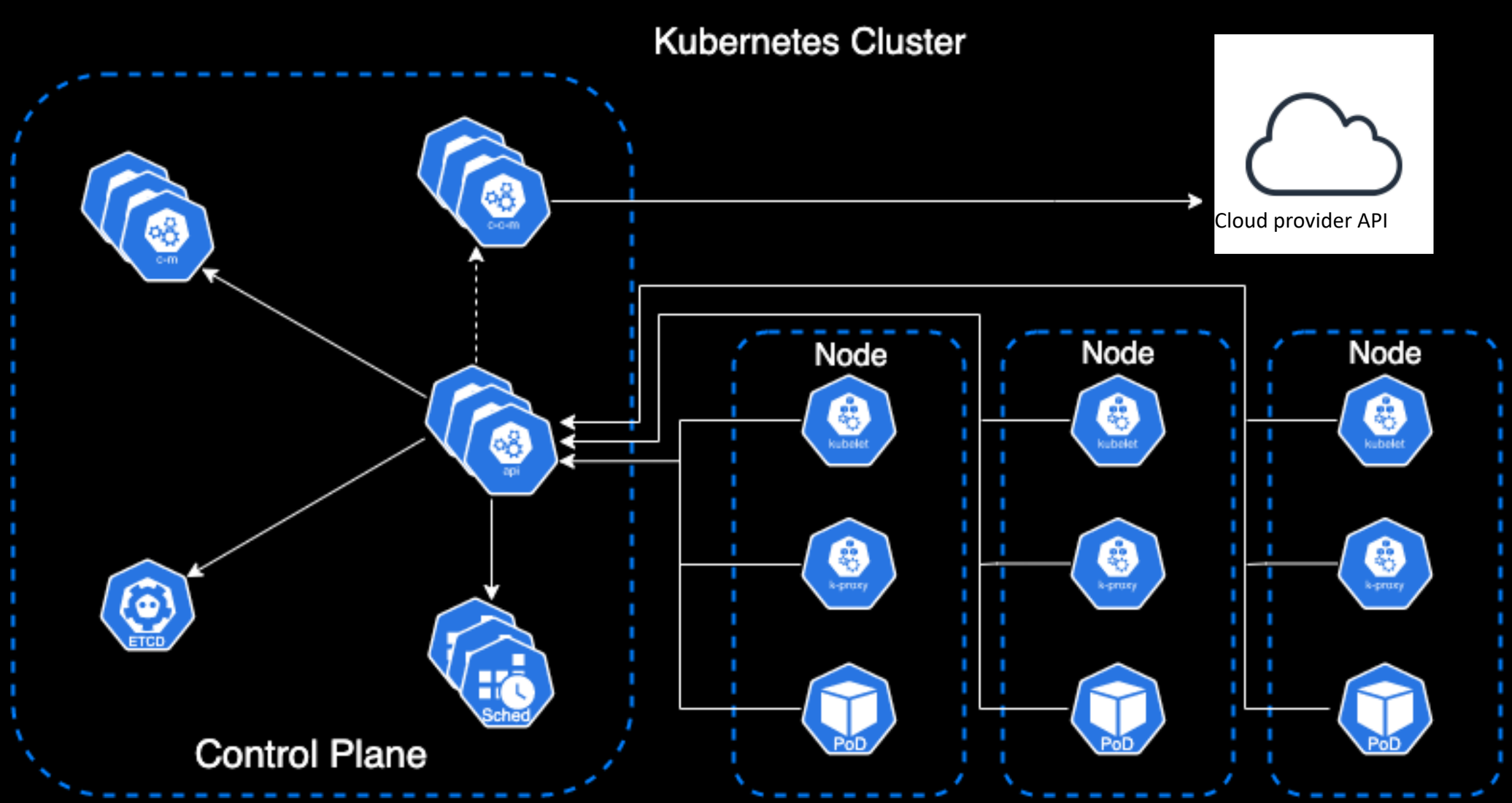
AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Kubernetes networking model focus

- Pod-to-pod communications
- Pod-to-service communications
- External-to-service communications

Kubernetes architecture



API server

Cloud Controller Manager (optional)

Controller Manager

kubelet

kube-proxy

Scheduler

kubernetes.io/docs/concepts/overview/components

EKS cluster overview



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Deployment options

Management

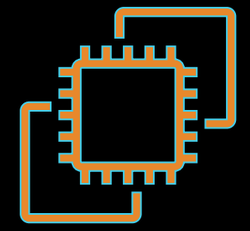


Amazon ECS



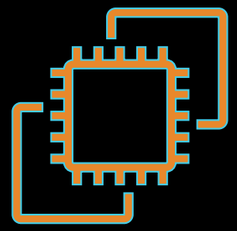
Amazon EKS

Kubernetes +



Kubernetes + Amazon EC2

Hosting



Amazon Elastic Compute Cloud (Amazon EC2)



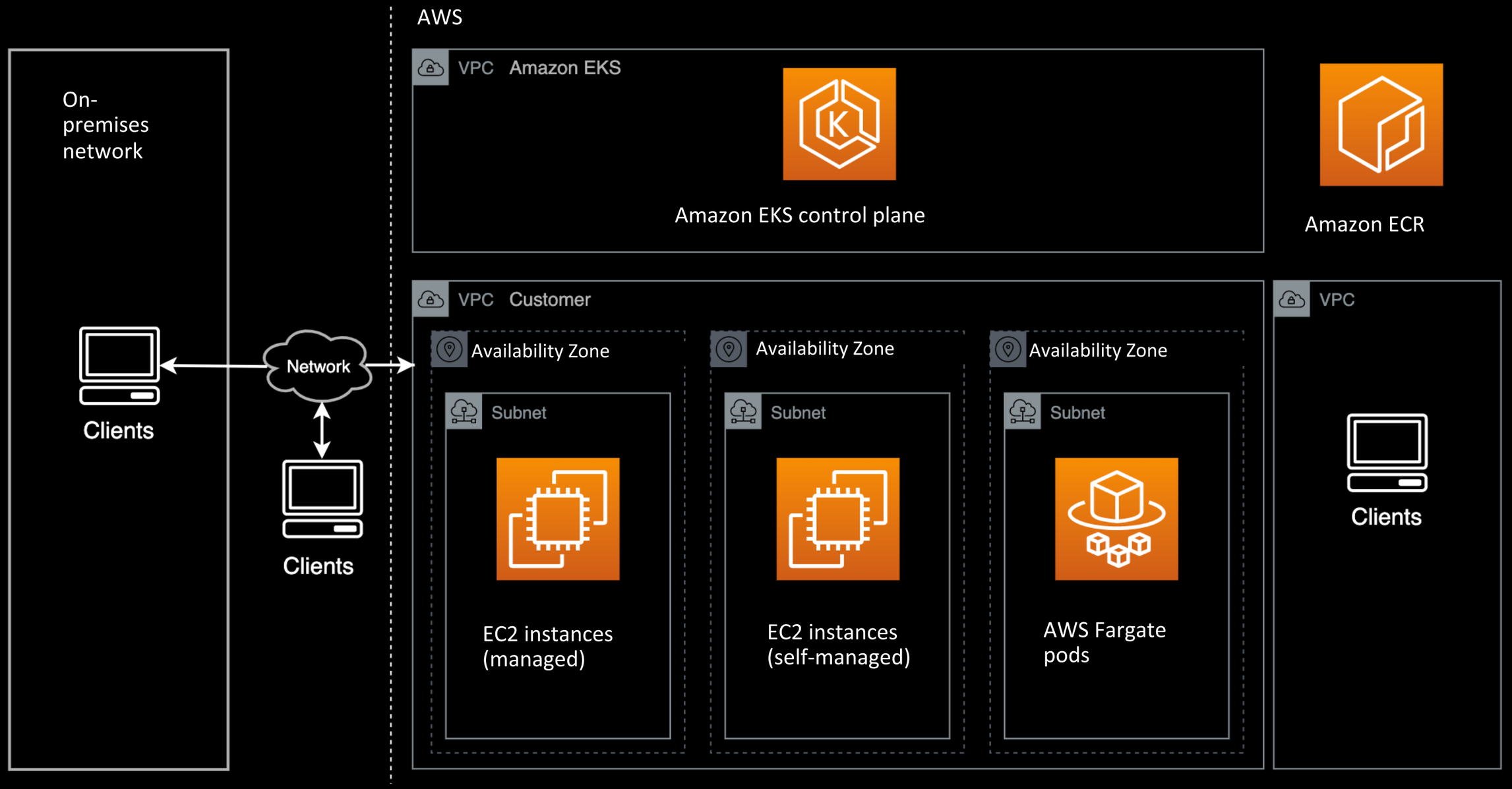
AWS Fargate

Registry

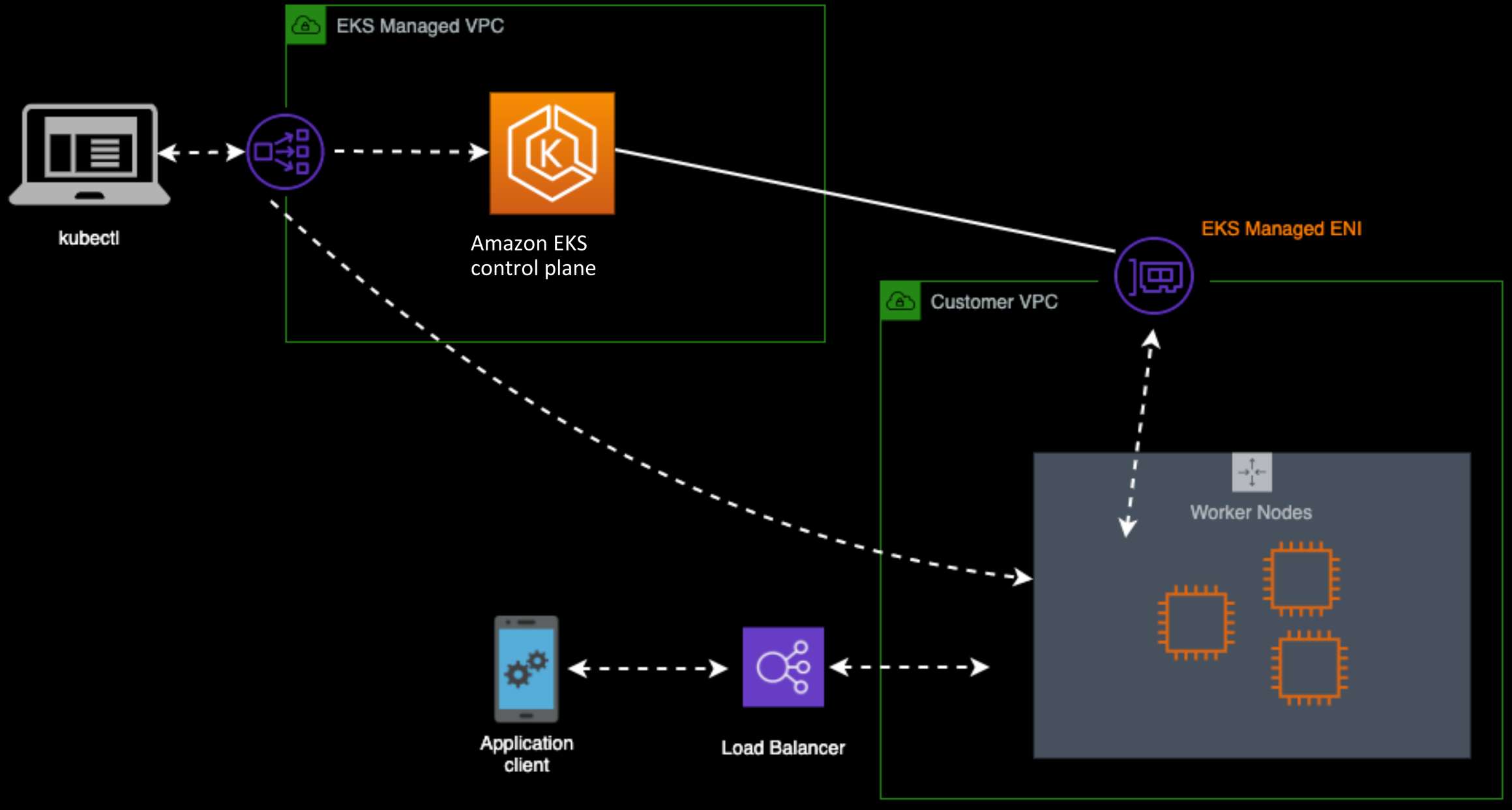


Amazon Elastic Container Registry (ECR)

Amazon EKS cluster components



Amazon EKS cluster components



VPC networking considerations



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Key considerations

- Amazon EC2 instance type
- Amazon EKS Cluster CIDR





- Amazon EKS control plane requires subnets in two AZs
- VPC DNS hostname and DNS resolution support

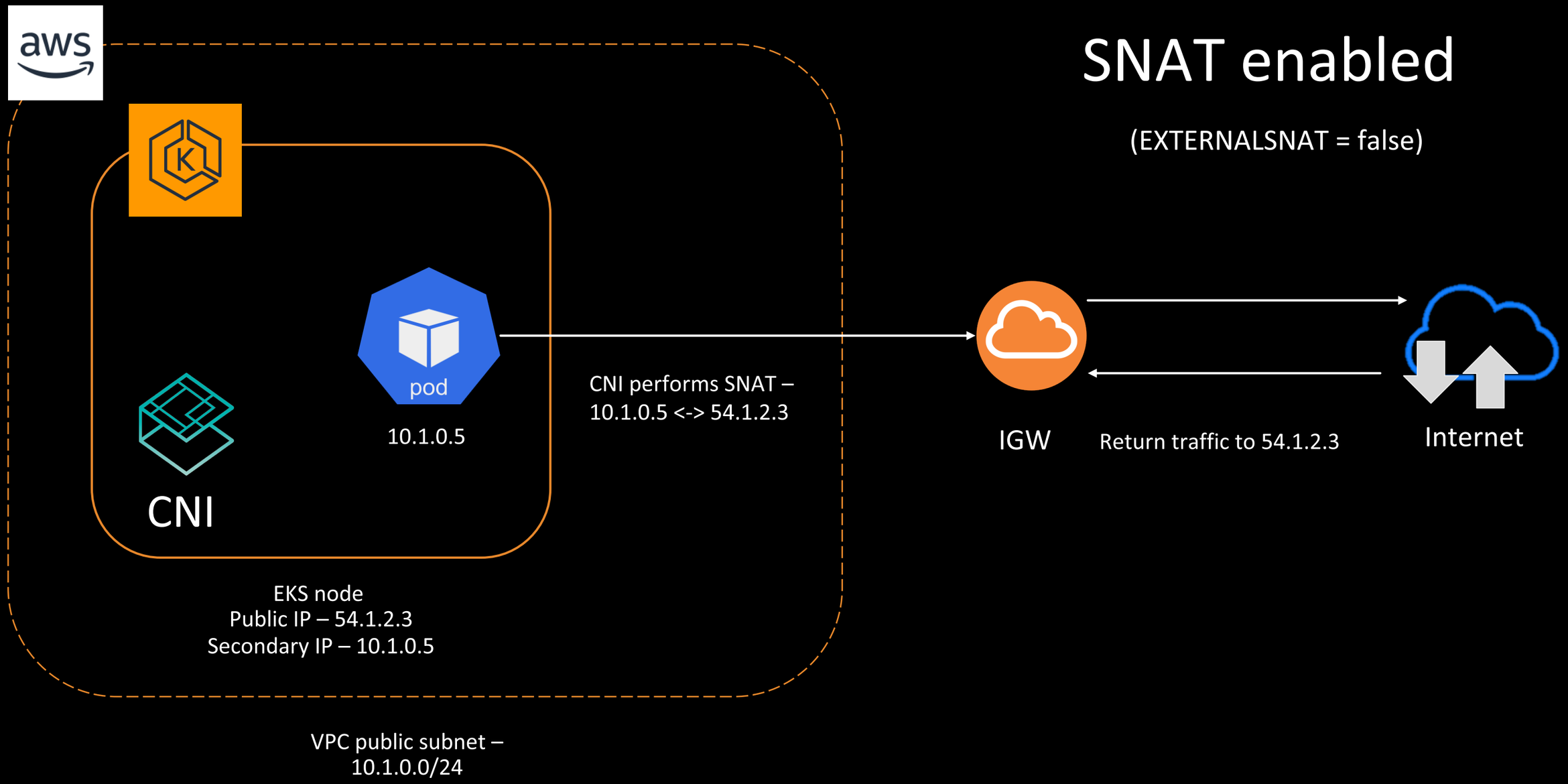
Key considerations

- Subnet tagging

- Private subnets for worker nodes and internal LBs
  - *key: kubernetes.io/cluster/<cluster-name> value: shared*
  - *key: kubernetes.io/role/internal-elb value: 1*

- Public subnets for LBs (IP addresses assignment)
  - *key: kubernetes.io/cluster/<cluster-name> value: shared*
  - *key: kubernetes.io/role/elb value:1*

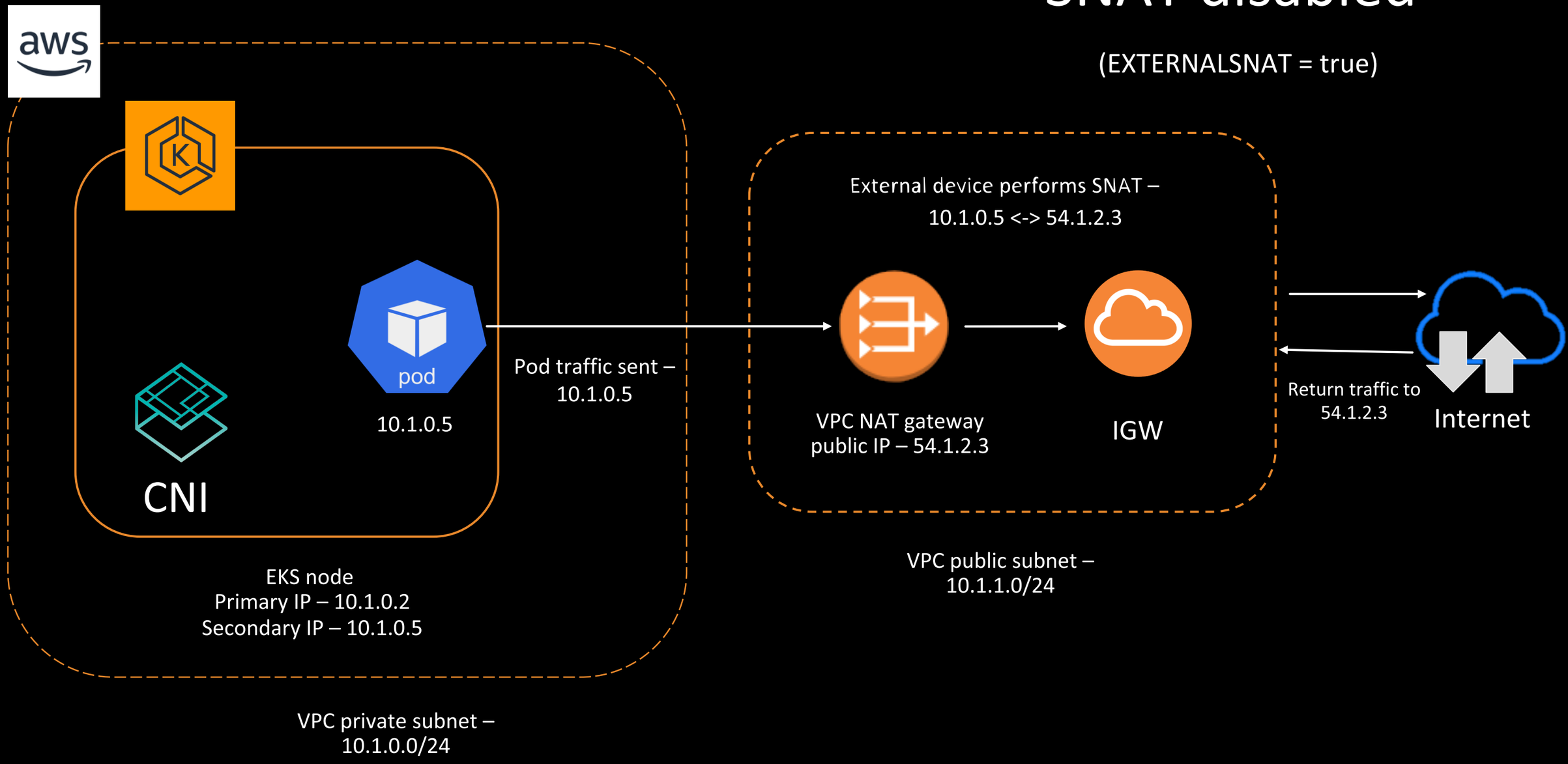
SNAT consideration



SNAT consideration

SNAT disabled

(EXTERNALSNAT = true)



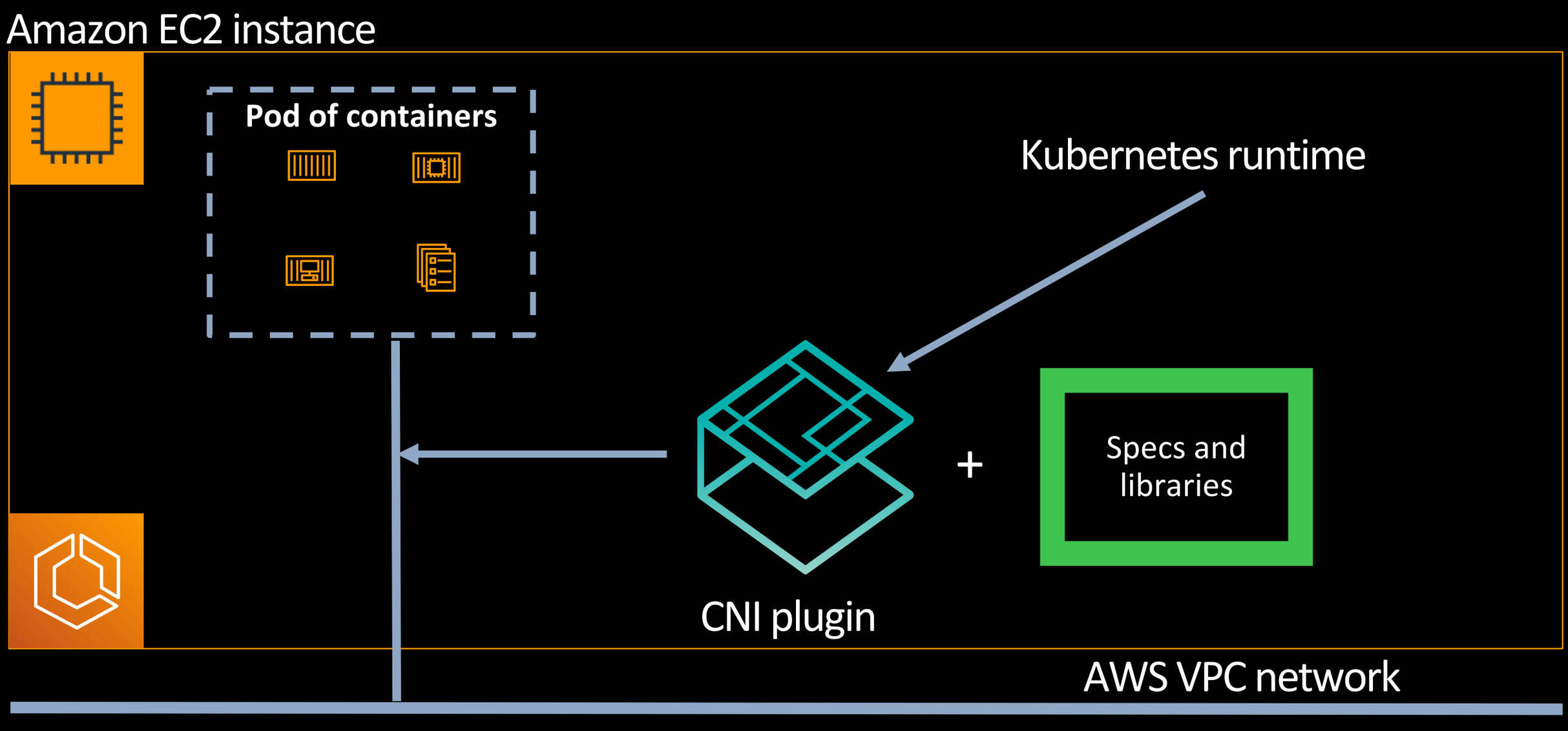
Amazon VPC CNI: Pod networking



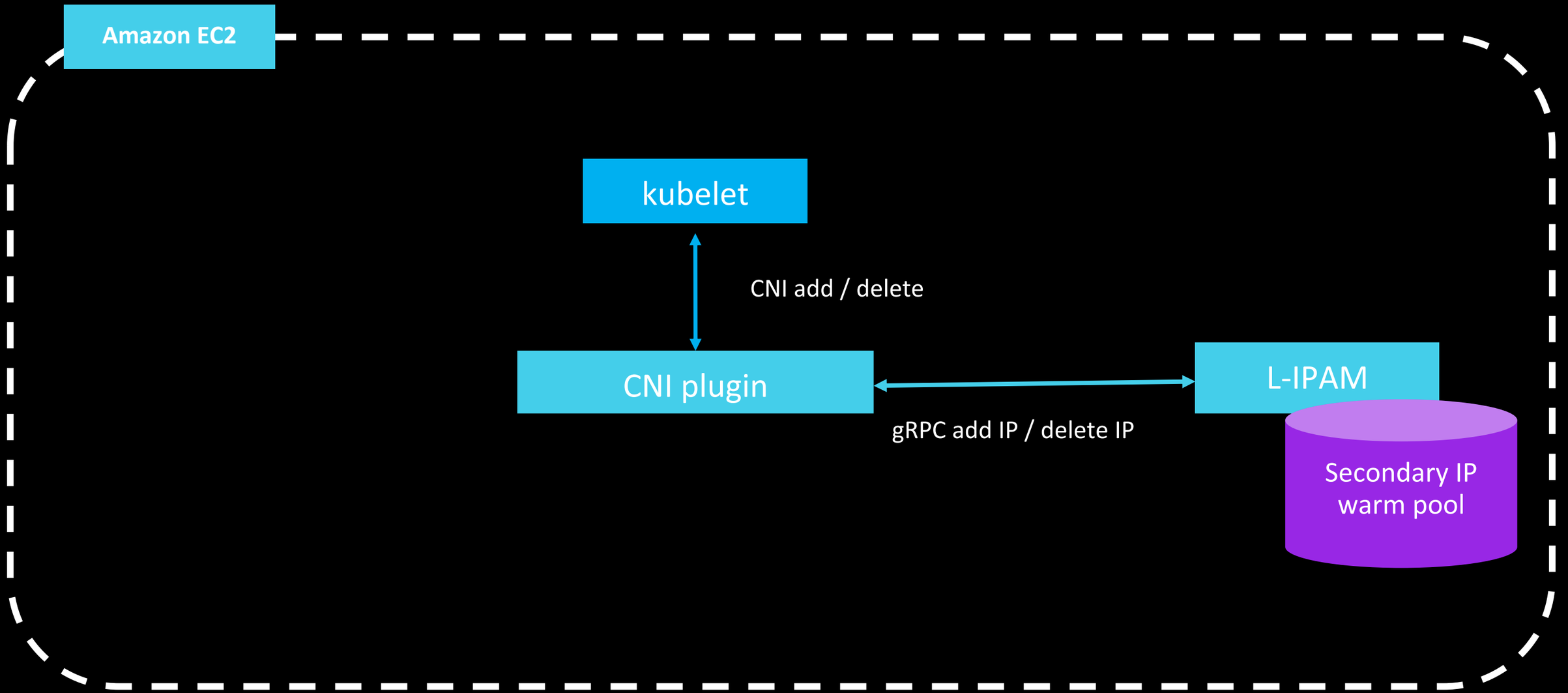
AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

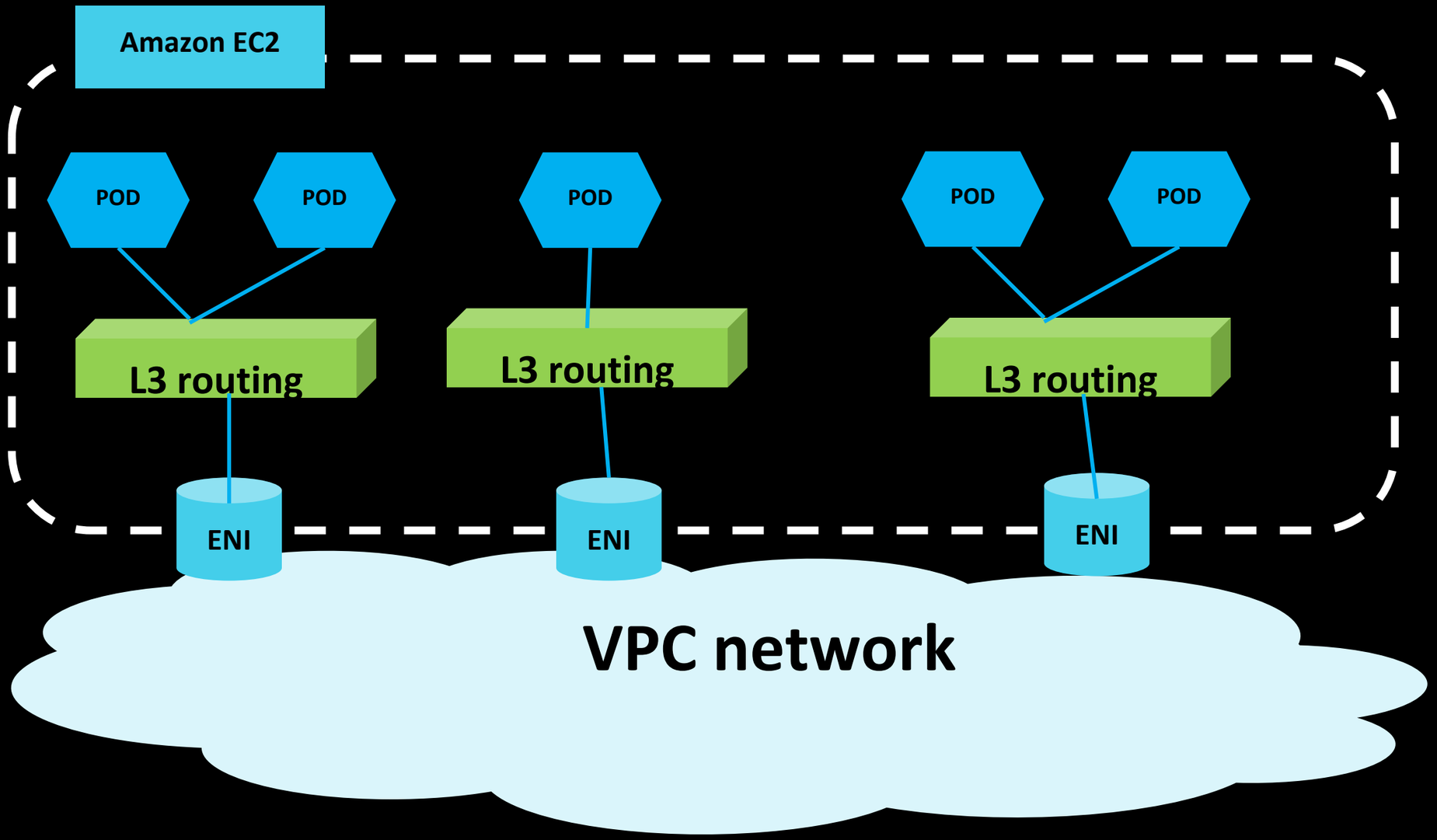
CNI project overview



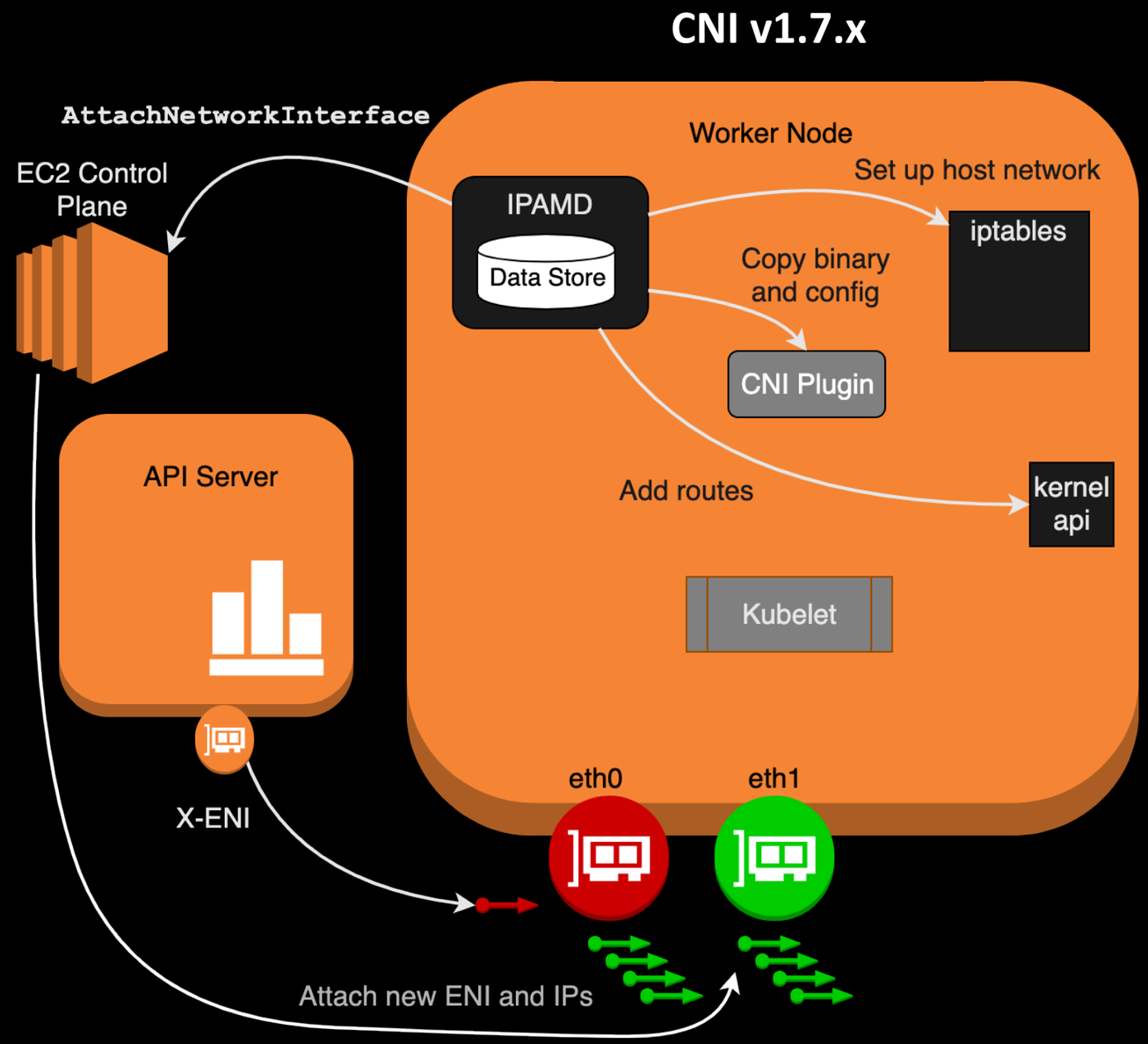
Amazon VPC CNI (amazon-vpc-cni-k8s) control plane components



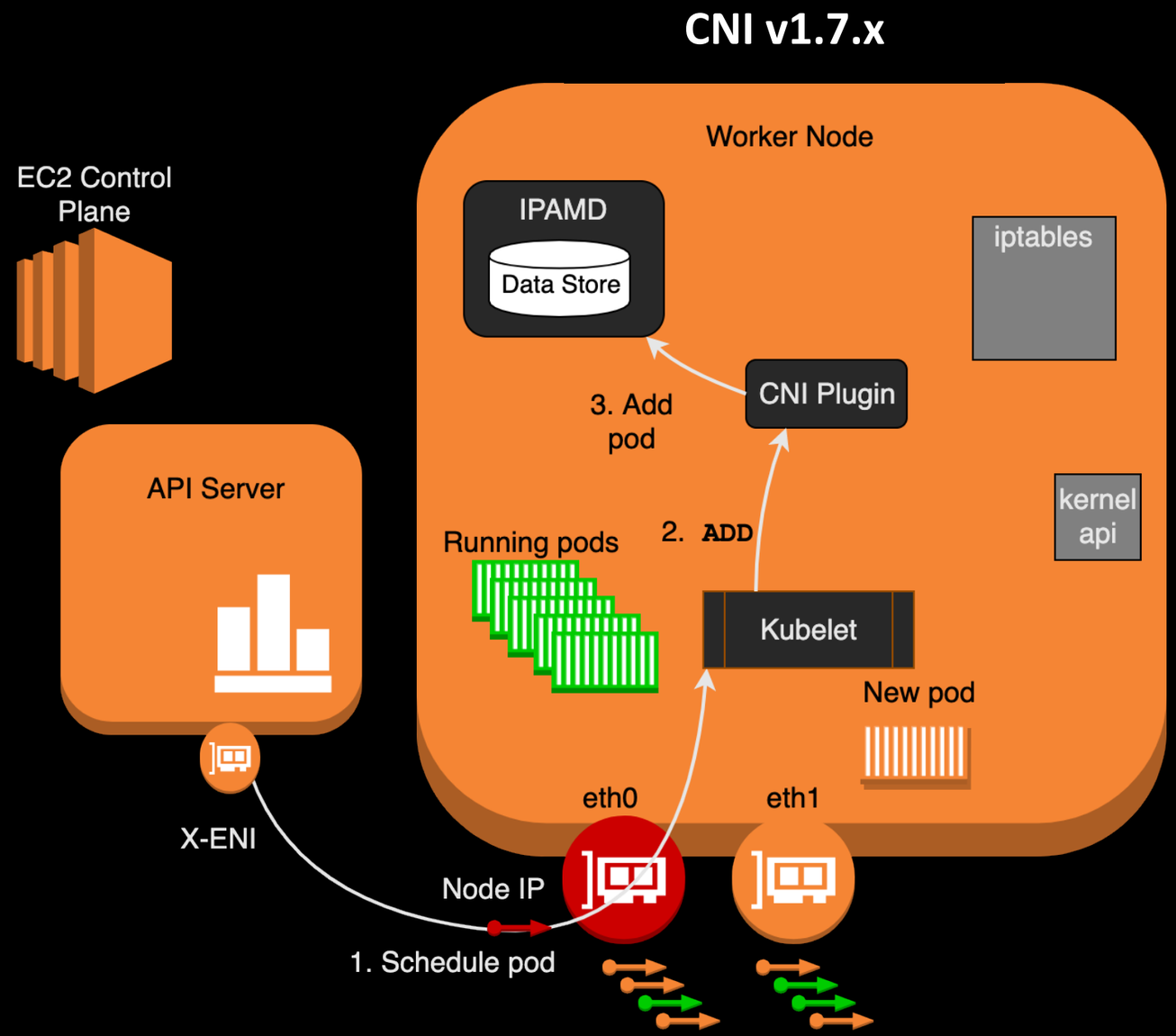
Amazon VPC CNI (amazon-vpc-cni-k8s) data plane



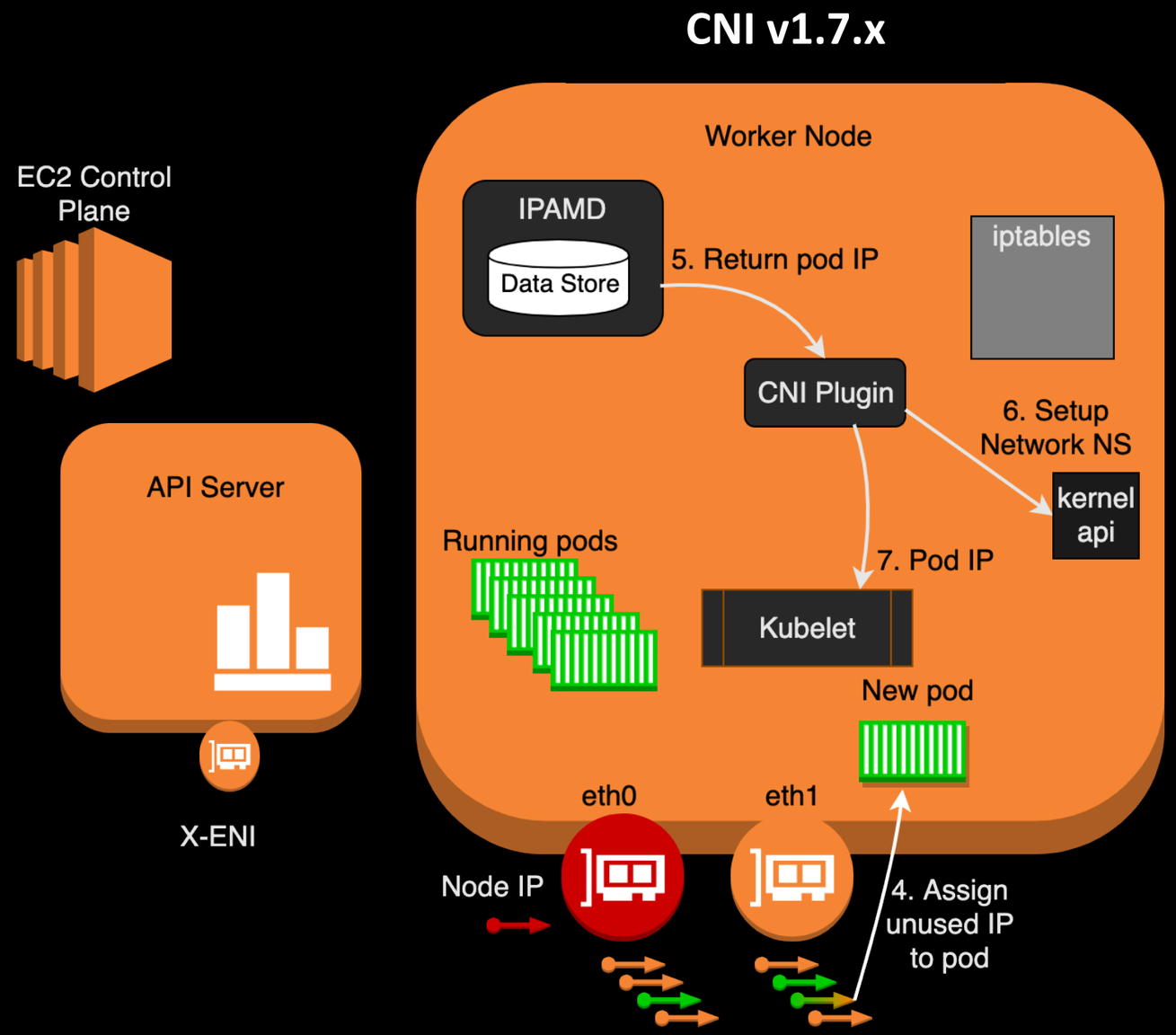
amazon-vpc-cni-k8s: New node starting



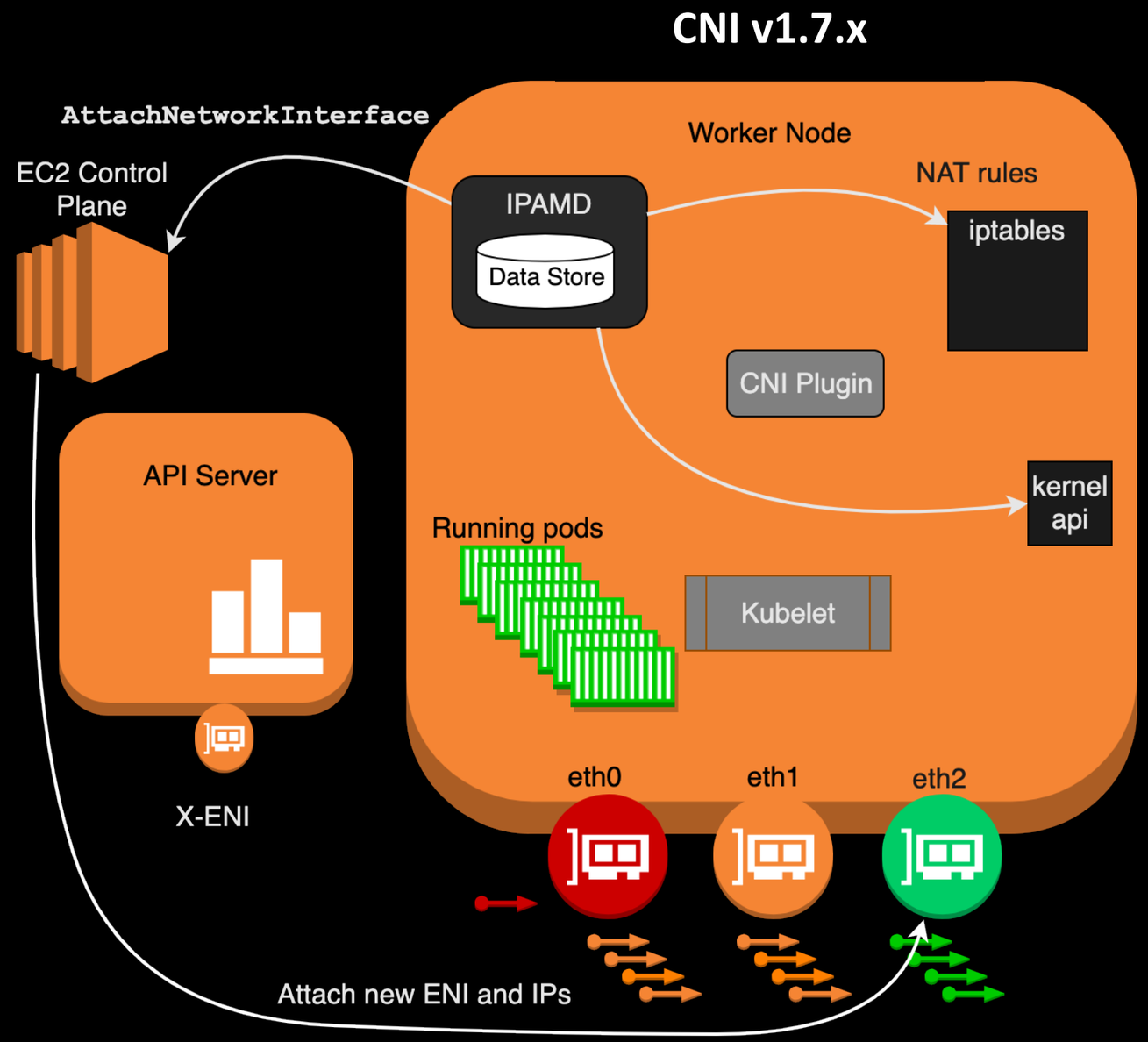
amazon-vpc-cni-k8s: Pod scheduled



amazon-vpc-cni-k8s: Pod scheduled



amazon-vpc-cni-k8s: More IPs added



Max pods per instance

- (Number of network interfaces for the instance type × (number of IP addresses per network interface - 1)) + 2

- For maximum number of pods supported by each instance type, see eni-max-pods.txt file:

  github.com/awslabs/amazon-eks-ami/blob/master/files/eni-max-pods.txt



Additional CNI features

- Custom networking
- Security groups for pods
- Metrics helper
- Higher pod density (roadmap item)

Key CNI configuration variables for scaling

	Default	Purpose
WARM_IP_TARGET	None	Leverage for dynamic scaling and to optimize IP allocation with MINIMUM_IP_TARGET; overrides WARM_ENI_TARGET
WARM_ENI_TARGET	1	Increase to preallocate more IPs for clusters with a lot of pod churn (also related to MAX_ENI)
MINIMUM_IP_TARGET	None	Leverage for prescaling and to set floor for allocated IPs
MAX_ENI	None	Leverage for dynamic scaling
AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG	False	Leverage to expand IP address pool

Alternate compatible CNI plugins

Calico

Cilium

Weave Net

Antrea

AWS ELB Kubernetes integration: Service networking



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Kubernetes load balancing integration methods

- Integration Method 1
  - K8s ServiceType = LoadBalancer
  - K8s environment and annotation define LB implementation
  - Handled by the Kubernetes Controller Manager (in-tree)
  - Deploys AWS CLB and NLB in instance mode

- Integration Method 2 (NEW)
  - AWS Load Balancer Controller
  - Formerly AWS ALB Ingress Controller
  - Deploys ALB in 2 modes for ingress resources
  - Deploys NLB in IP mode for service resources

ServiceType = Loadbalancer CLB deployment

```
apiVersion: v1
kind: Service
metadata:
        name: nginx
        namespace: default
        labels:
                app: nginx
        annotations:

spec:
        externalTrafficPolicy: Local
        ports:
        - name: http
          port: 80
          protocol: TCP
          targetPort: 80
        selector:
          app: nginx

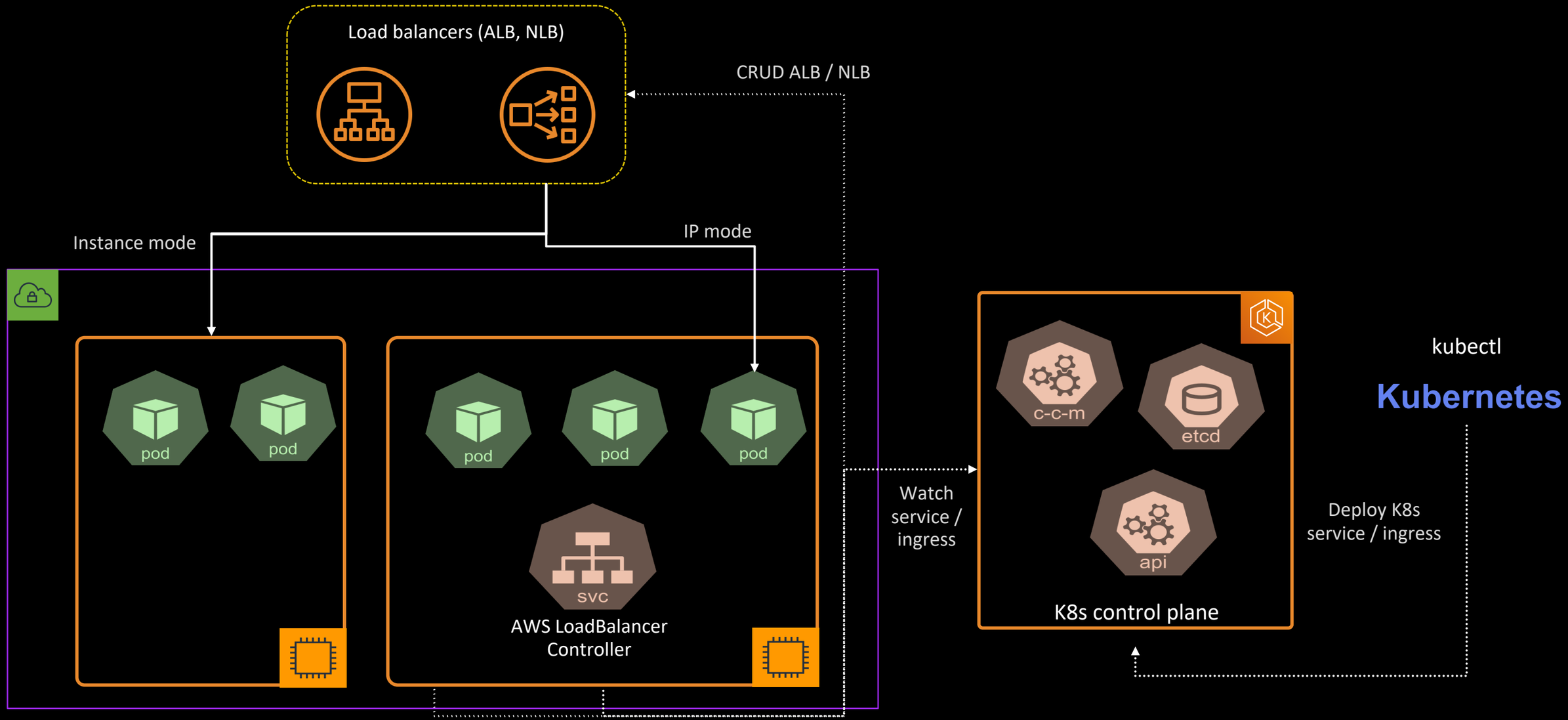
type: LoadBalancer
```

ServiceType = Loadbalancer NLB (instance mode) deployment

```
apiVersion: v1
kind: Service
metadata:
        name: nginx
        namespace: default
        labels:
                app: nginx
        annotations:
                service.beta.kubernetes.io/aws-load-balancer-type: "nlb"
spec:
        externalTrafficPolicy: Local
        ports:
        - name: http
          port: 80
          protocol: TCP
          targetPort: 80
        selector:
          app: nginx

type: LoadBalancer
```

AWS LoadBalancer Controller architecture (NEW)



ServiceType = Loadbalancer NLB (IP mode) deployment

```
apiVersion: v1
kind: Service
metadata:
        name: nginx
        namespace: default
        labels:
                app: nginx
        annotations:
                # route traffic directly to pod IPs
                service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"
spec:
        externalTrafficPolicy: Local
        ports:
        - name: http
          port: 80
          protocol: TCP
          targetPort: 80
        selector:
          app: nginx

type: LoadBalancer
```

ALB deployment (instance mode)

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: nginx
  namespace: nginx
  annotations:
    kubernetes.io/ingress.class: “alb”
    alb.ingress.kubernetes.io/target-type: instance
spec:
  rules:
    - http:
        paths:
          - path: /*
            backend:
              serviceName: nginx
              servicePort: 8080
```

ALB deployment (IP mode)

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: nginx
  namespace: nginx
  annotations:
    kubernetes.io/ingress.class: “alb”
    alb.ingress.kubernetes.io/target-type: ip
spec:
  rules:
    - http:
        paths:
          - path: /*
            backend:
              serviceName: nginx
              servicePort: 8080
```

Resources



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

References

- github.com/aws/amazon-vpc-cni-k8s
- aws.amazon.com/ec2/instance-types
- kubernetes.io/docs/concepts/cluster-administration/networking
- kubernetes.io/docs/concepts/services-networking/service

Thank you!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.