AWS re:Invent

NET403-R

# Deep dive: Container networking at scale on Amazon EKS & Amazon EC2

**Ikenna Izugbokwe**
Sr. Solutions Architect
Amazon Web Services

**Paavan Mistry**
Specialist SA, Security
Amazon Web Services

AWS re:Invent



aws

# Agenda

- Introduction / Workshop environment overview
- Section 1: Kubernetes networking review
- Section 2: Amazon VPC networking
- Section 3: Amazon EKS networking
- Section 4: Kubernetes on EC2 networking (kops)

# Workshop overview

# Workshop environment setup

Workshop website https://www.awsk8snetworkshops.com

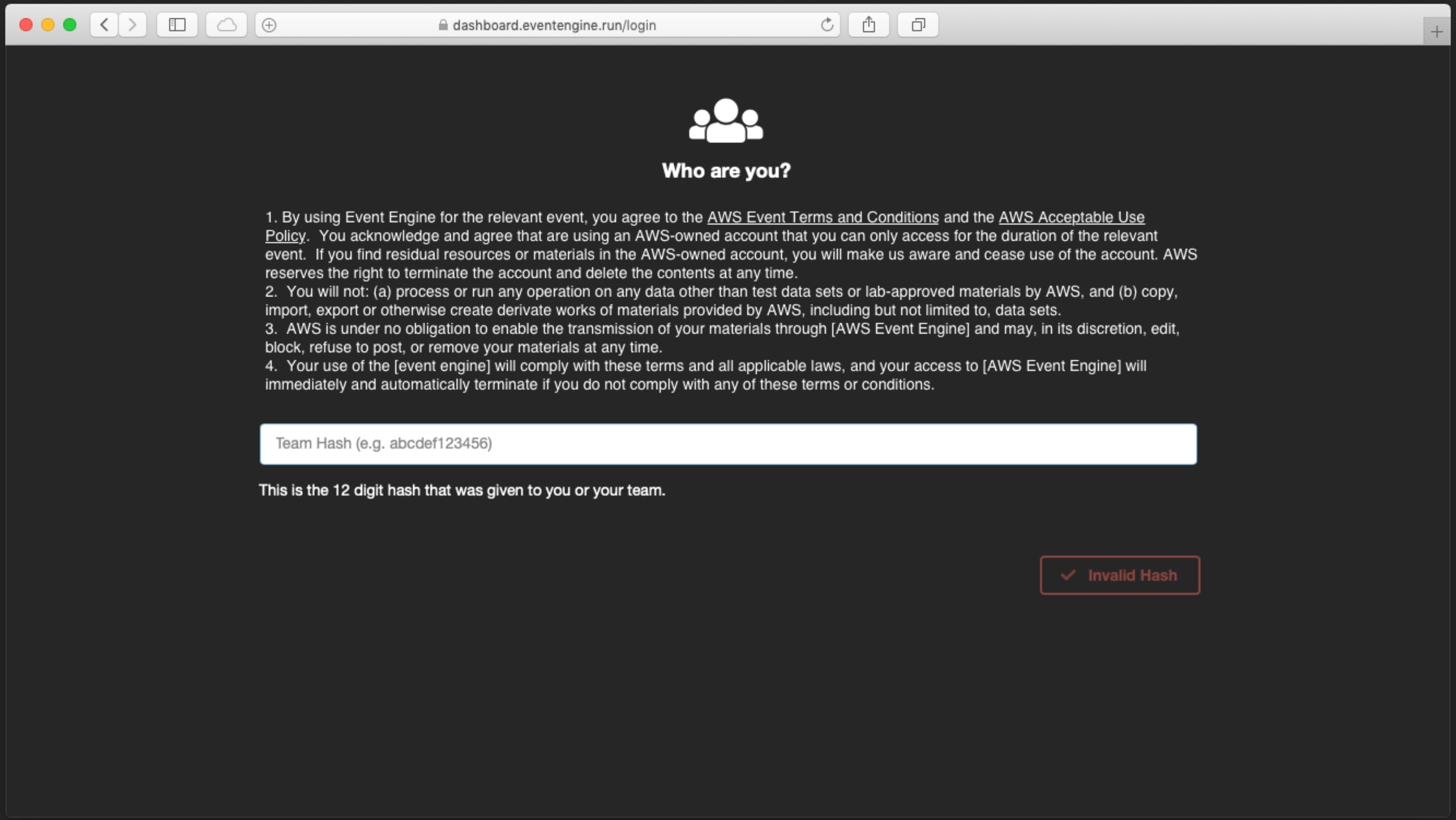
AWS Event Engine https://dashboard.eventengine.run/

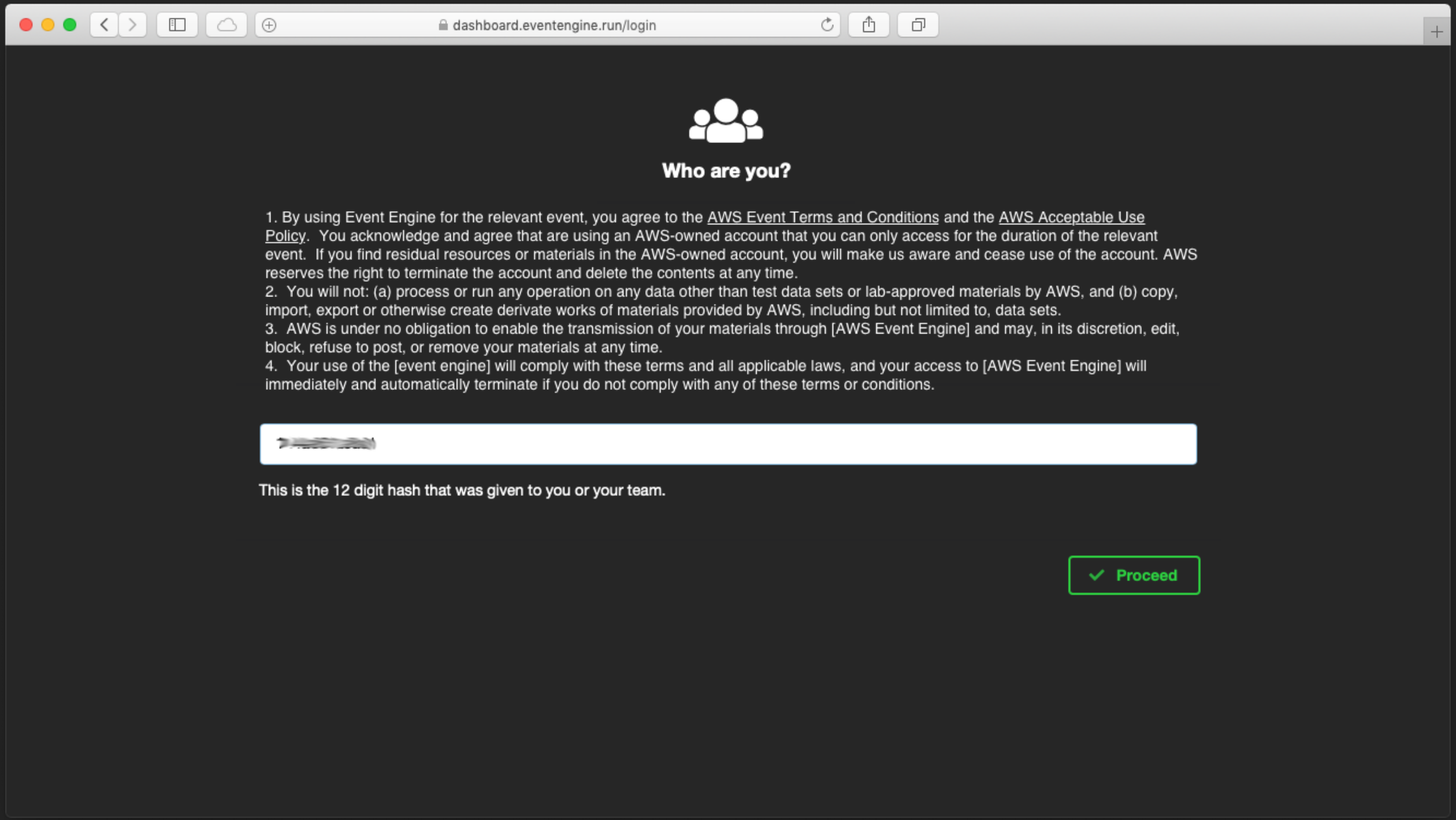
AWS CloudFormation template creates two AWS Cloud9 EC2 environments

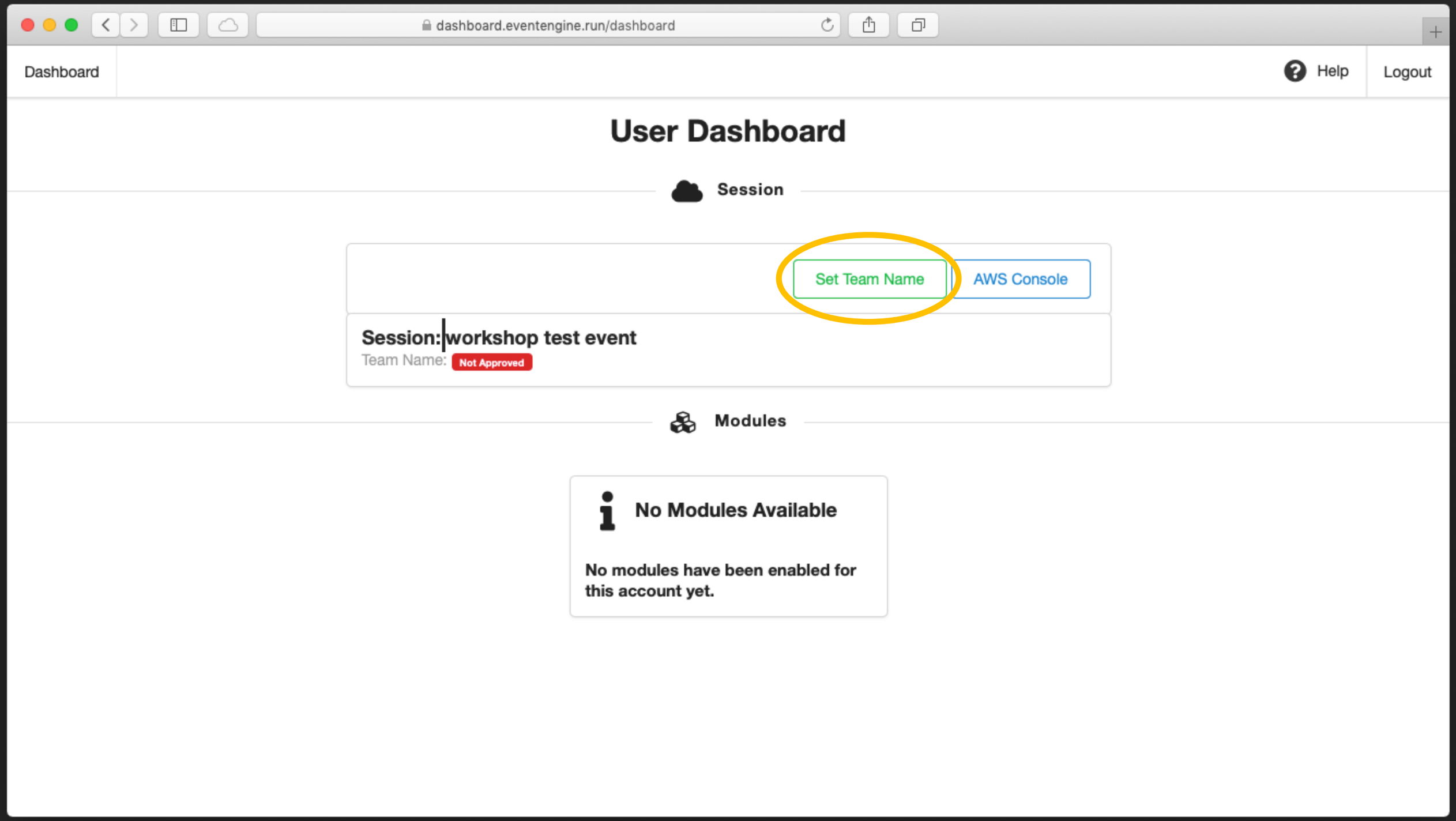
Amazon EKS AWS Cloud9 environment

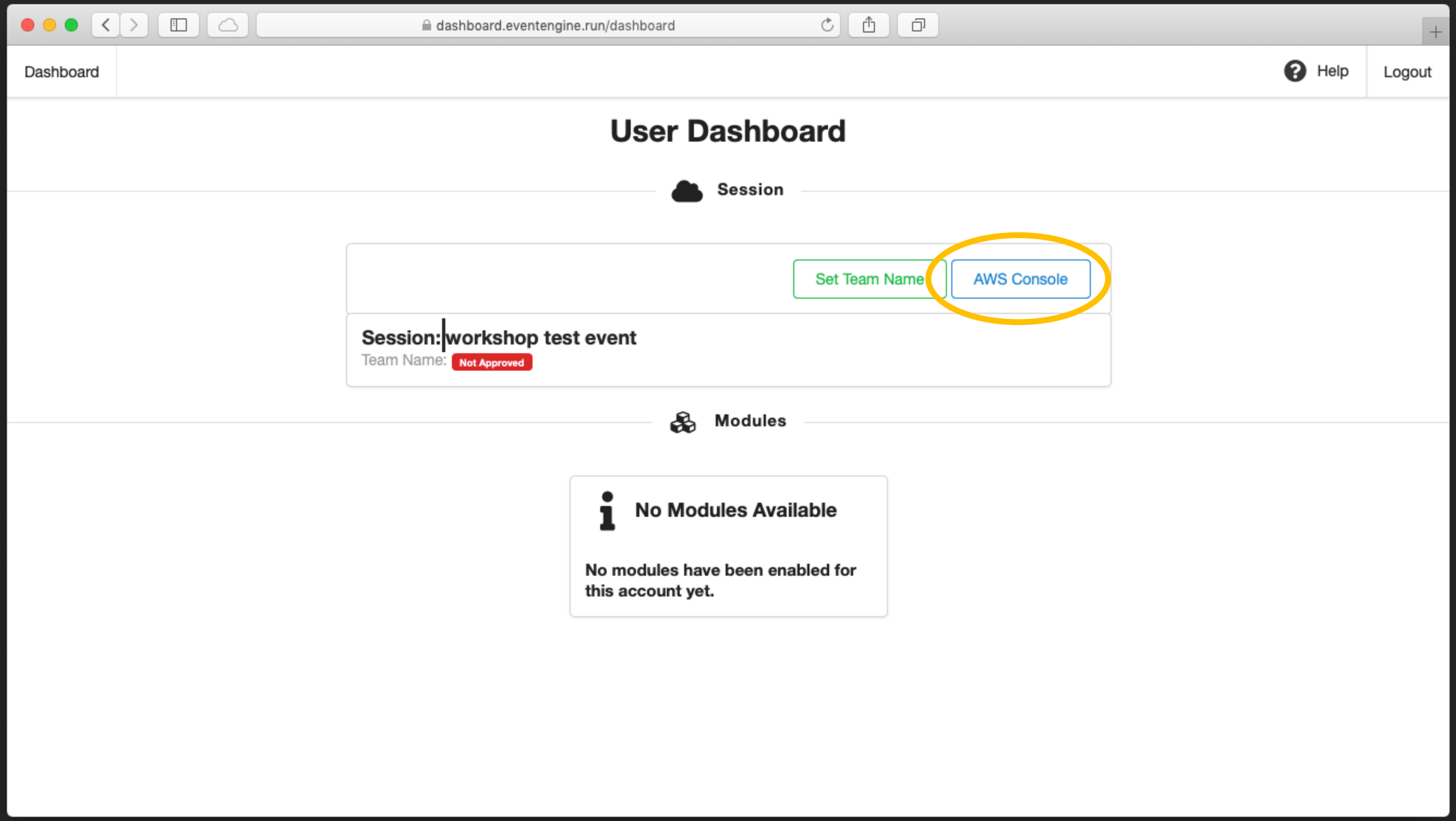
Kops AWS Cloud9 environment

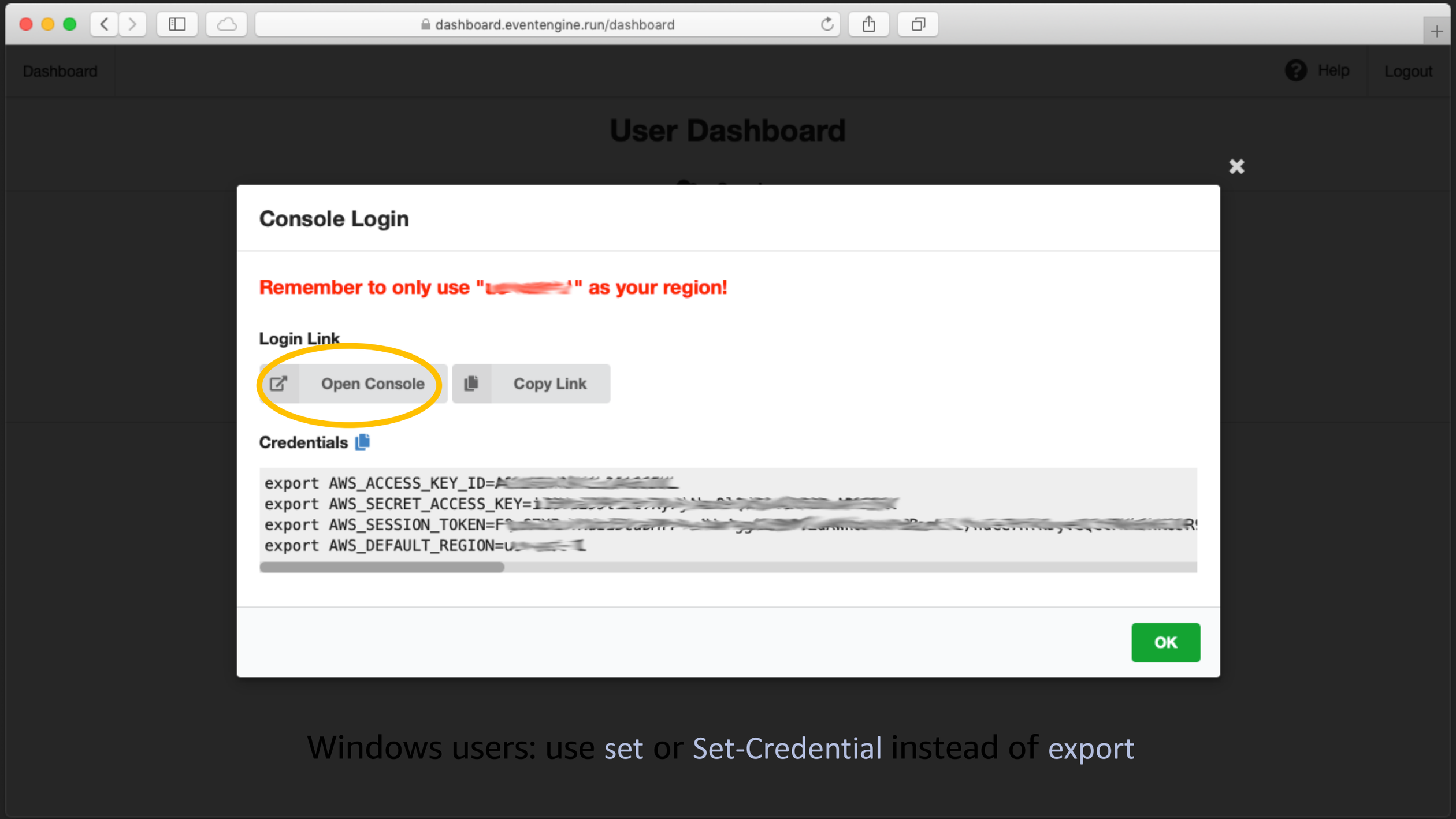
https://dashboard.eventengine.run


dashboard.eventengine.run/login
Who are you?
1. By using Event Engine for the relevant event, you agree to the AWS Event Terms and Conditions and the AWS Acceptable Use Policy. You acknowledge and agree that are using an AWS-owned account that you can only access for the duration of the relevant event. If you find residual resources or materials in the AWS-owned account, you will make us aware and cease use of the account. AWS reserves the right to terminate the account and delete the contents at any time.
2. You will not: (a) process or run any operation on any data other than test data sets or lab-approved materials by AWS, and (b) copy, import, export or otherwise create derivate works of materials provided by AWS, including but not limited to, data sets.
3. AWS is under no obligation to enable the transmission of your materials through [AWS Event Engine] and may, in its discretion, edit, block, refuse to post, or remove your materials at any time.
4. Your use of the [event engine] will comply with these terms and all applicable laws, and your access to [AWS Event Engine] will immediately and automatically terminate if you do not comply with any of these terms or conditions.
Team Hash (e.g. abcdef123456)
This is the 12 digit hash that was given to you or your team.
Invalid Hash


dashboard.eventengine.run/login
Who are you?
1. By using Event Engine for the relevant event, you agree to the AWS Event Terms and Conditions and the AWS Acceptable Use Policy. You acknowledge and agree that are using an AWS-owned account that you can only access for the duration of the relevant event. If you find residual resources or materials in the AWS-owned account, you will make us aware and cease use of the account. AWS reserves the right to terminate the account and delete the contents at any time.
2. You will not: (a) process or run any operation on any data other than test data sets or lab-approved materials by AWS, and (b) copy, import, export or otherwise create derivate works of materials provided by AWS, including but not limited to, data sets.
3. AWS is under no obligation to enable the transmission of your materials through [AWS Event Engine] and may, in its discretion, edit, block, refuse to post, or remove your materials at any time.
4. Your use of the [event engine] will comply with these terms and all applicable laws, and your access to [AWS Event Engine] will immediately and automatically terminate if you do not comply with any of these terms or conditions.
This is the 12 digit hash that was given to you or your team.
Proceed


dashboard.eventengine.run/dashboard
Dashboard
Help
Logout
User Dashboard
Session
Set Team Name
AWS Console
Session:workshop test event
Team Name:
Not Approved
Modules
No Modules Available
No modules have been enabled for this account yet.


dashboard.eventengine.run/dashboard
Dashboard
Help
Logout
User Dashboard
Session
Set Team Name
AWS Console
Session:workshop test event
Team Name:
Not Approved
Modules
No Modules Available
No modules have been enabled for this account yet.




Windows users: use set or Set-Credential instead of export

# Section 1

# AWS container services overview

# Deployment options

Amazon ECS

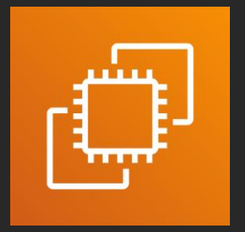
Amazon EKS

AWS Fargate

Kubernetes

+

Amazon ECR



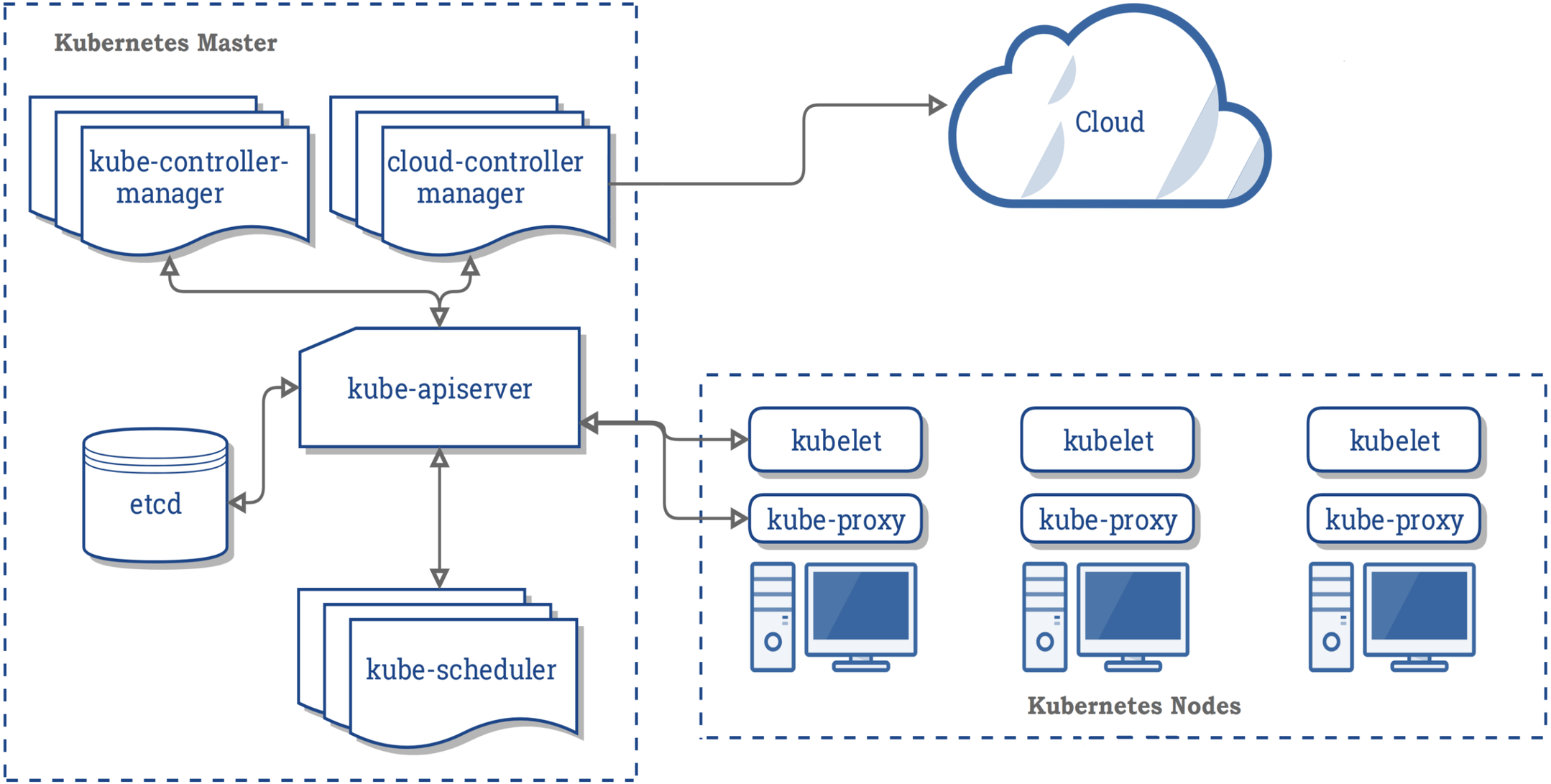
Amazon EC2

# Kubernetes concepts and architecture

# Key Kubernetes concepts

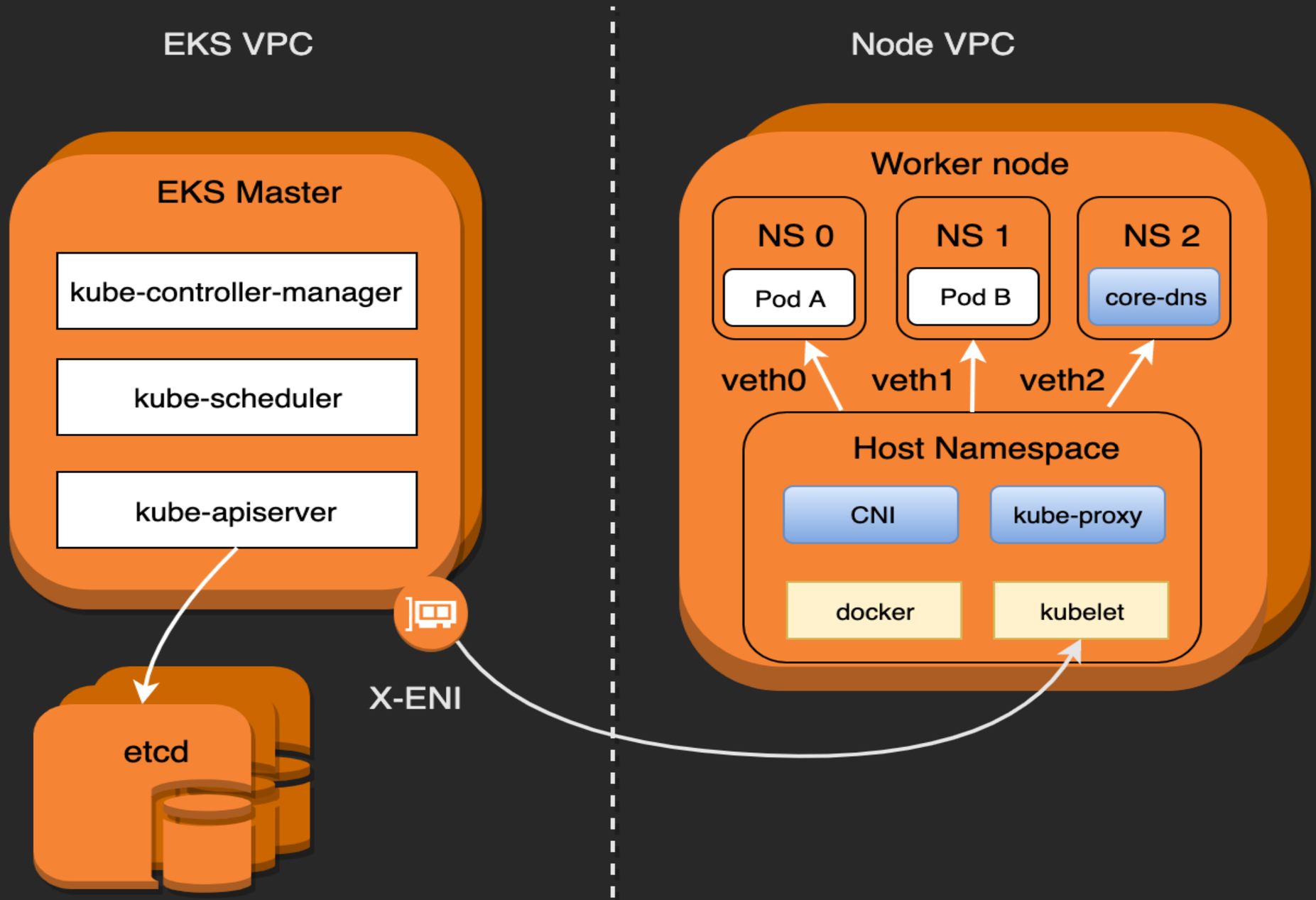
- Kubernetes control plane
  - Provided by master node objects/components
- Kubernetes data plane
  - Provided by worker nodes objects/components
- Kubernetes master node
  - Kube-apiserver
  - Kube-controller-manager
  - Kube-scheduler
  - etcd
- Kubernetes worker nodes
  - Kubelet
  - Kube-proxy
  - Container runtime
  - Pods

# Kubernetes architecture




https://kubernetes.io/docs/concepts/overview/components/

# EKS Kubernetes network architecture



# Kubernetes container networking

# Four networking problems

- Container-to-container communications
- Pod-to-pod communications
- Pod-to-service communications
- External-to-internal communications

# What is a pod?

- Smallest and simplest computing unit
- Group of one or more containers
- Co-located and co-scheduled
- Share a network stack and storage
- Containers within a Pod share an IP address

# From Kubernetes's perspective

- Pods can communicate with other Pods
- Every pod gets its own IP address
- Mapping container ports to node(host) port not required

# Section 2

# Kubernetes networking implementation

AWS re:Invent



aws

# Kubernetes networking implementations: CNIs

- Kubenet
- Calico
- Multus
- Cilium
- Cni-ipvlan-vpc-k8s
- Amazon-vpc-cni-k8s

# Amazon EC2, VPC & hybrid networking considerations for Amazon EKS

AWS re:Invent



aws

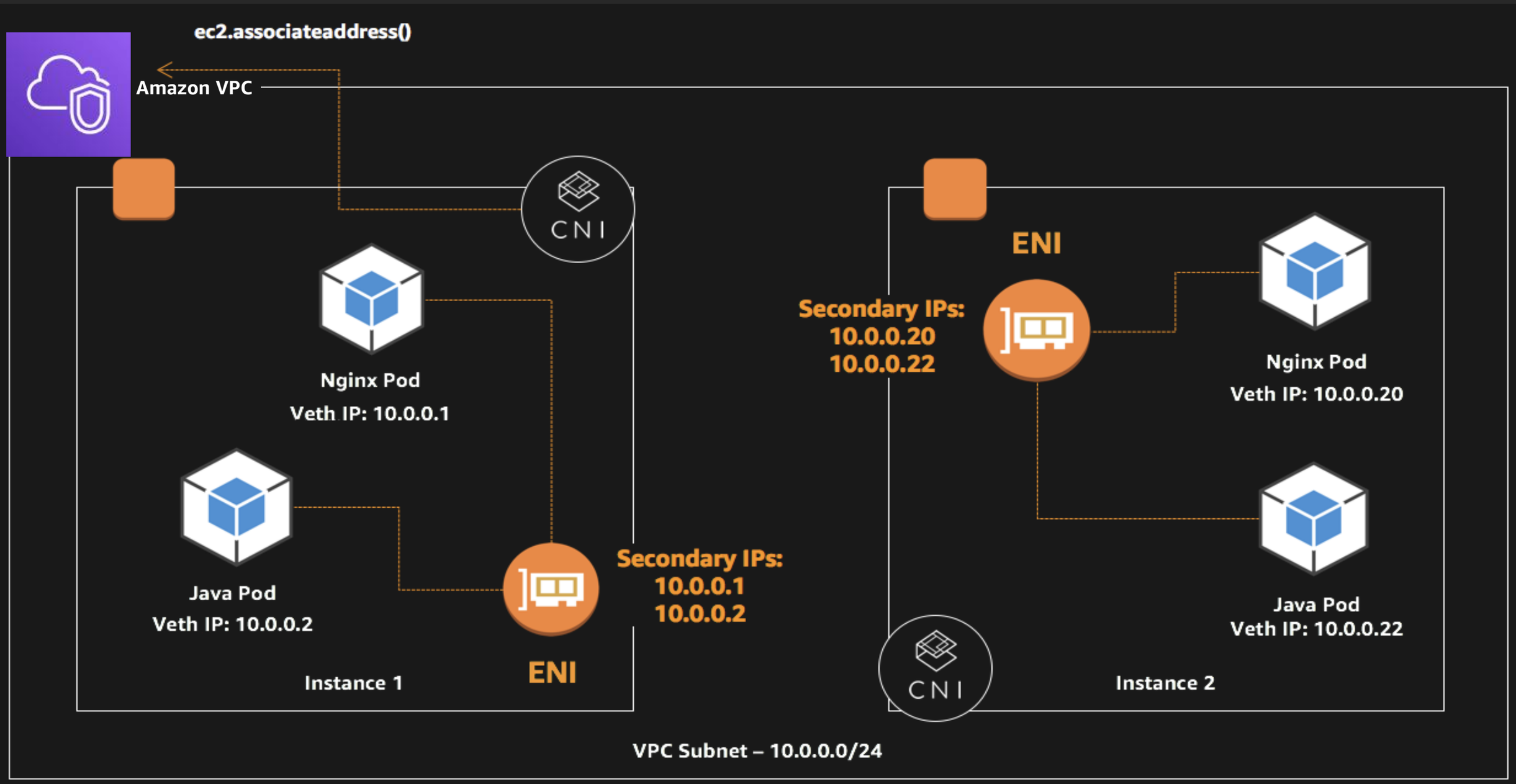
# Amazon EC2 and VPC considerations: Amazon EKS

- Amazon EC2 instance type
- Amazon EKS requires subnets in at least two AZs
- Use a separate VPC for each Amazon EKS cluster
- VPC DNS hostname and DNS resolution support

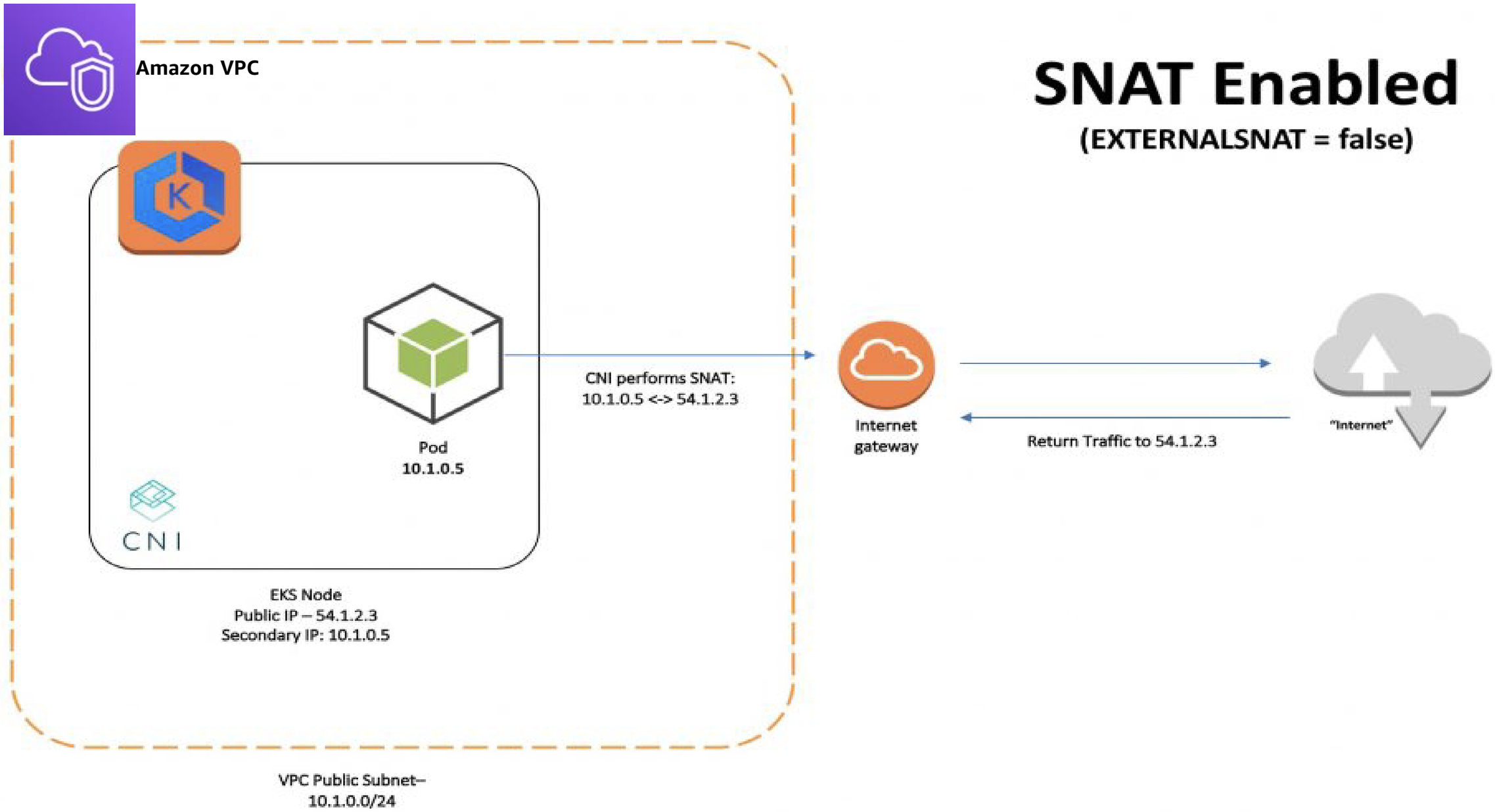
# Amazon EC2 and VPC considerations (cont'd)

- Private subnets for worker nodes recommended
- Public subnets for load balancers
- Cluster upgrades require 2-3 IP's per initial cluster subnet
- Docker runs in the 172.17.0.0/16 CIDR range in Amazon EKS clusters
- Disable SNAT external VPC, VPN of AWS Direct Connect access

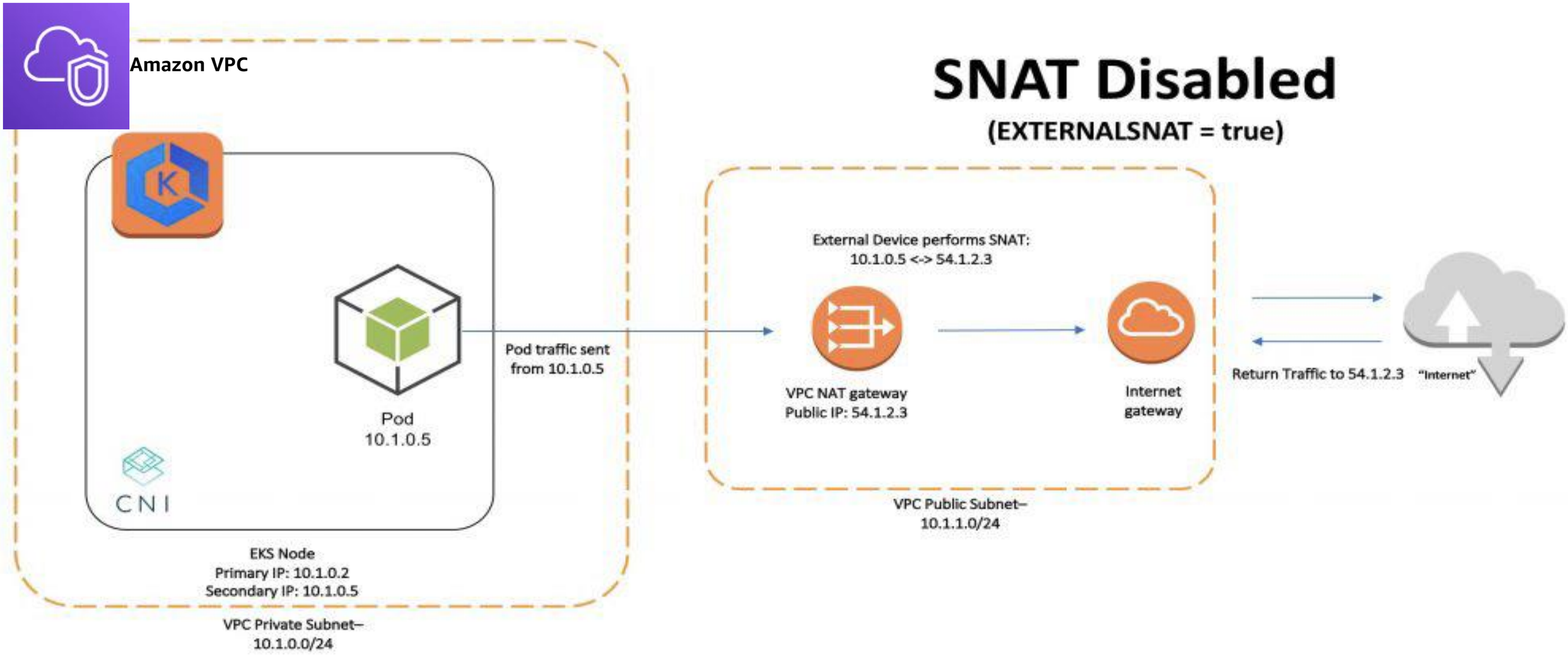
# Pod IP wiring within VPC



# SNAT consideration



# SNAT consideration cont'd



# Section 3

# Amazon EKS: CNI details

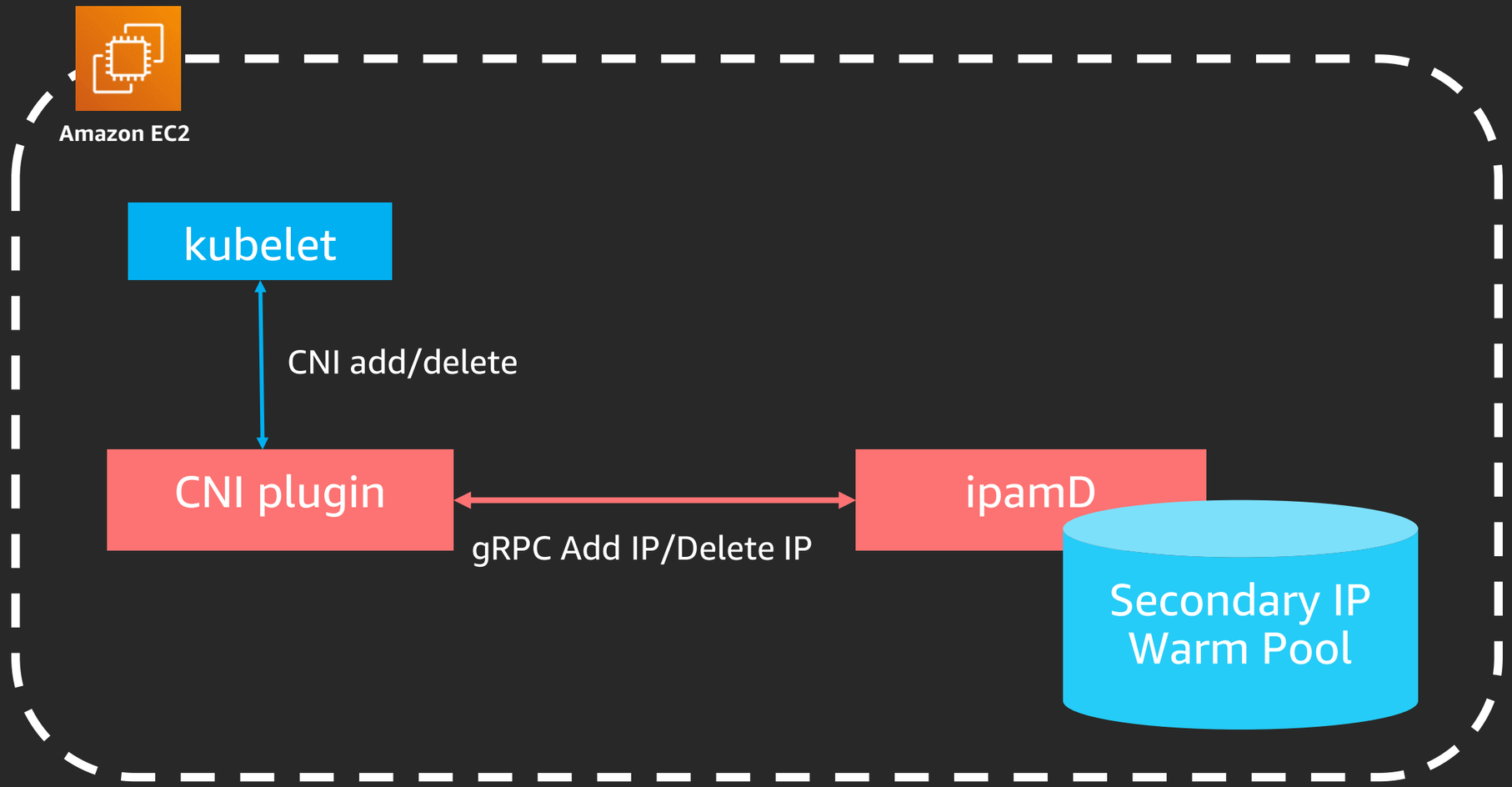
# CNI overview


- Set up network namespace
- Assign an IP to a pod
- Clean up when a pod goes away
- Tear down network namespace

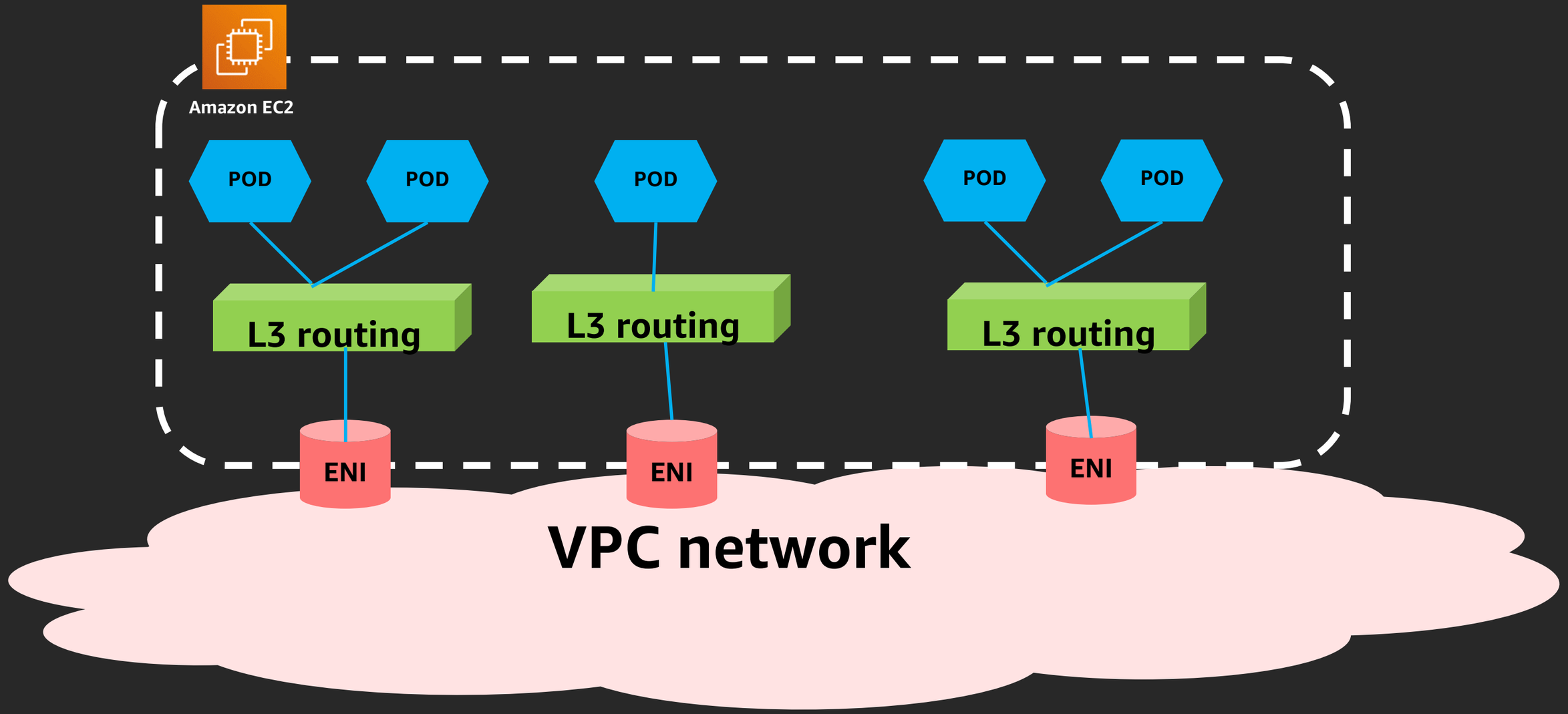
# CNI networking details: Control plane

- Kubelet invokes CNI Add or Delete commands for pods
- CNI requests secondary IPs from ipamD and sets up networking stack for pod
- For fast pod startup time, the IP address manager database (ipamD) creates a secondary IP warm pool with 1 more ENI and its IP address

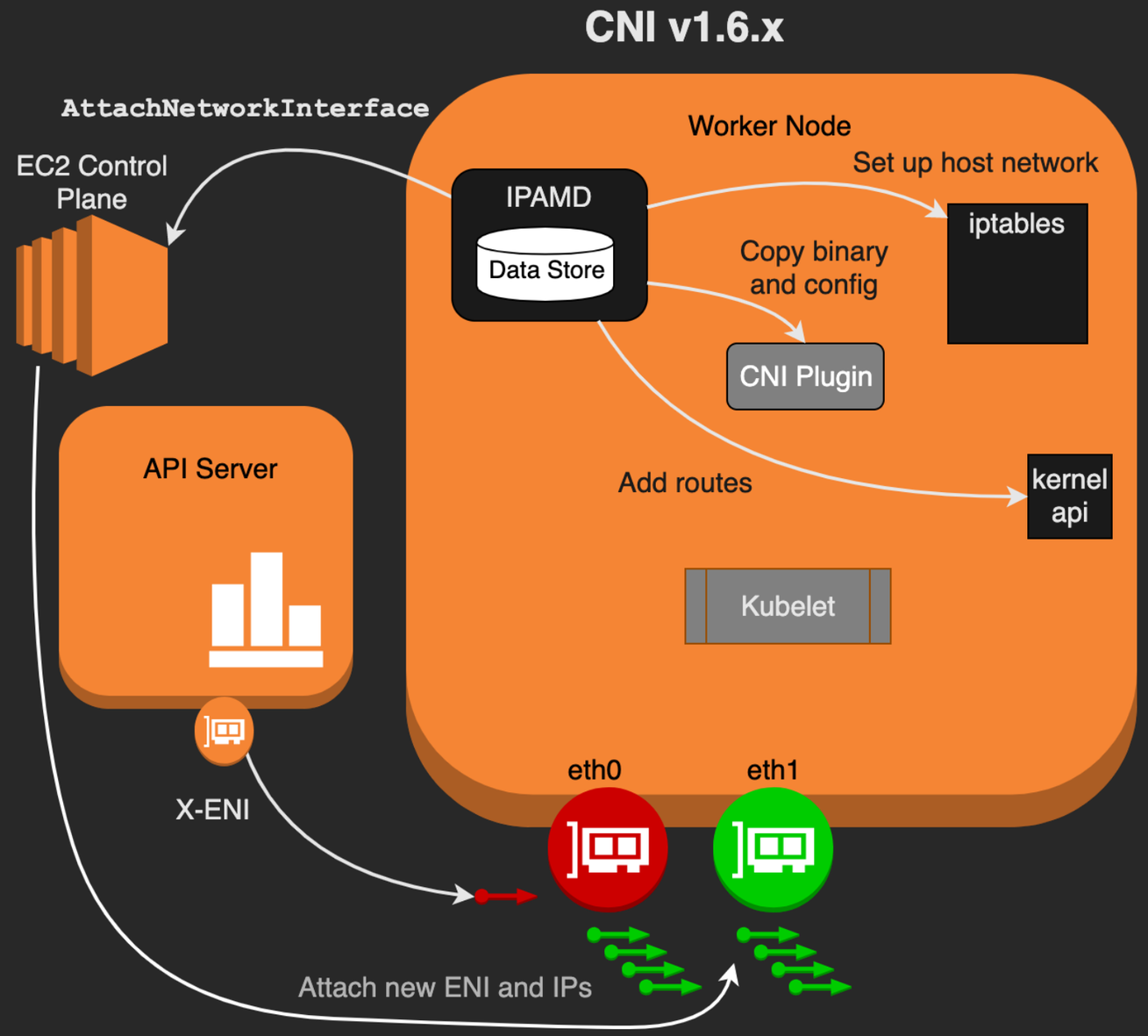
# CNI networking details: Control plane



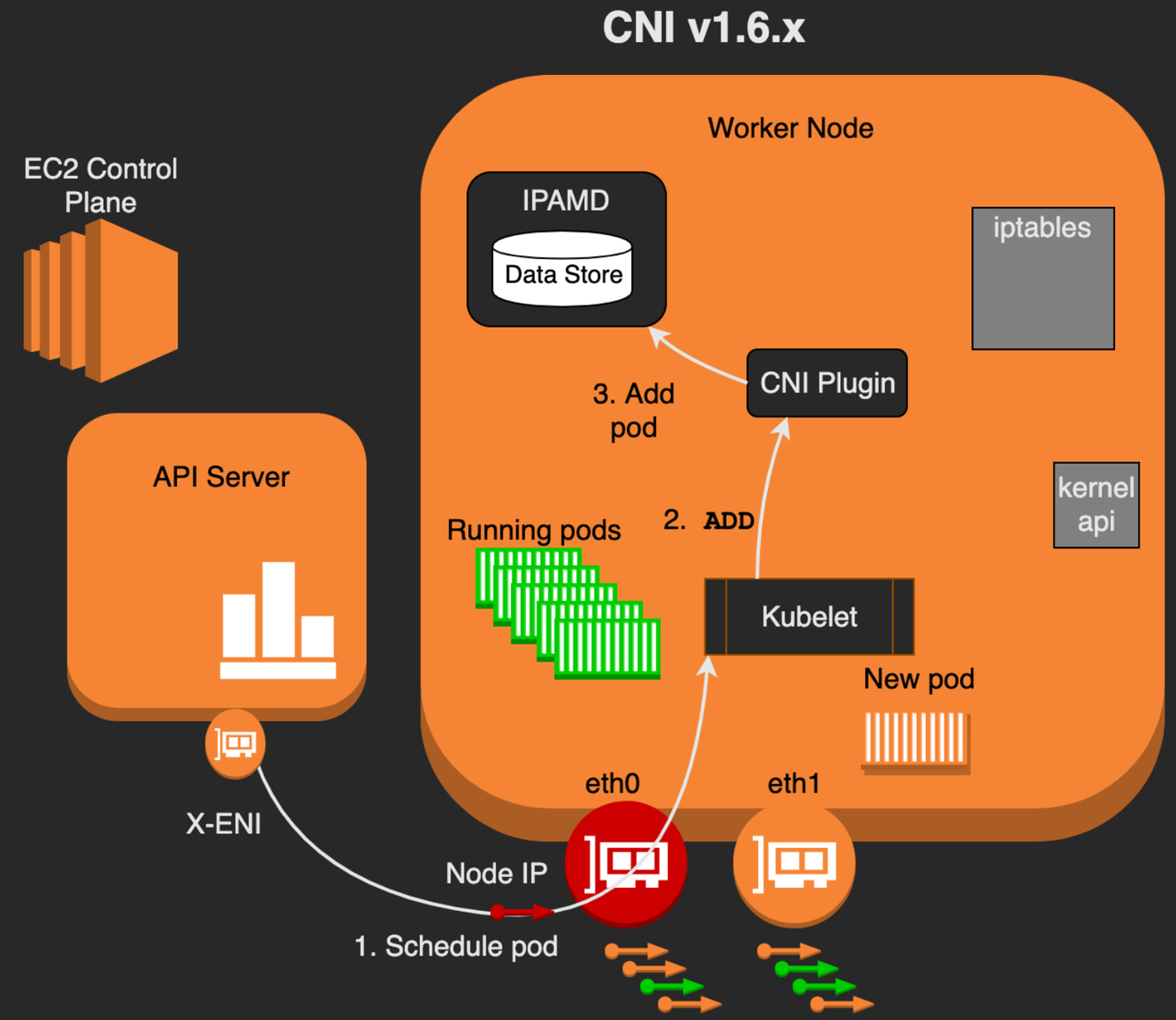
# CNI networking details: Data plane



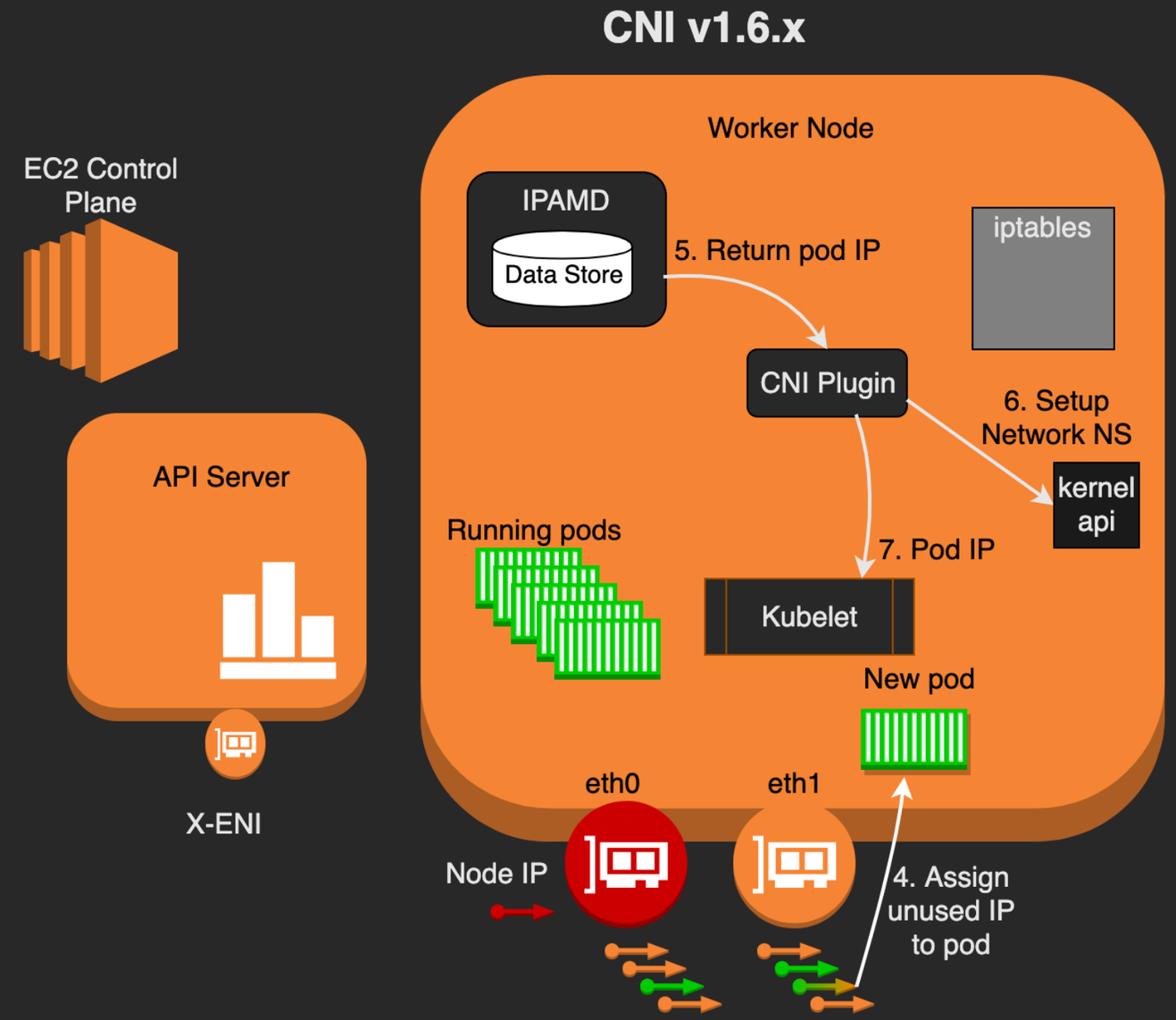
# amazon-vpc-cni-k8s: New node starting



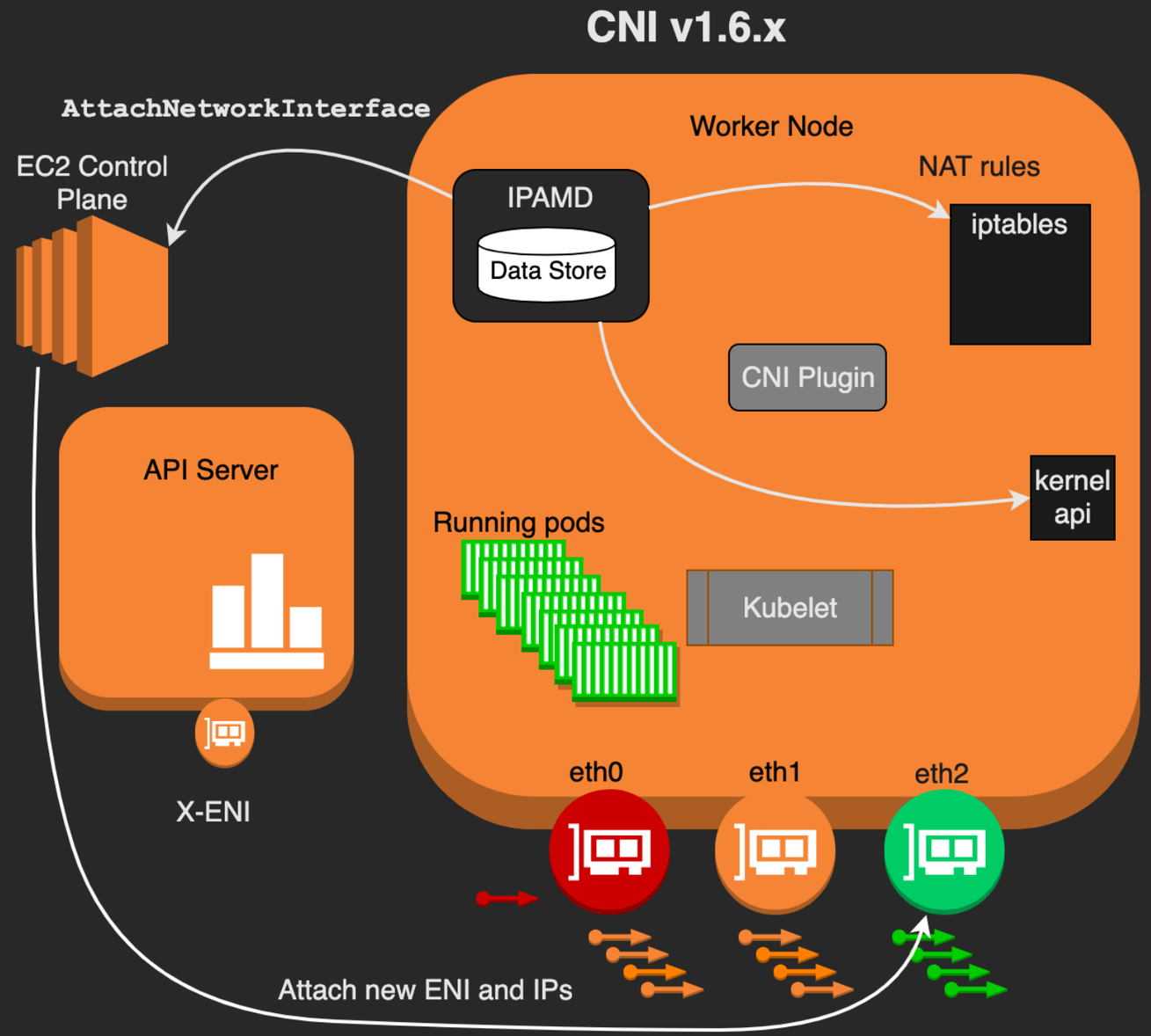
# amazon-vpc-cni-k8s: Pod scheduled



# amazon-vpc-cni-k8s: Pod scheduled



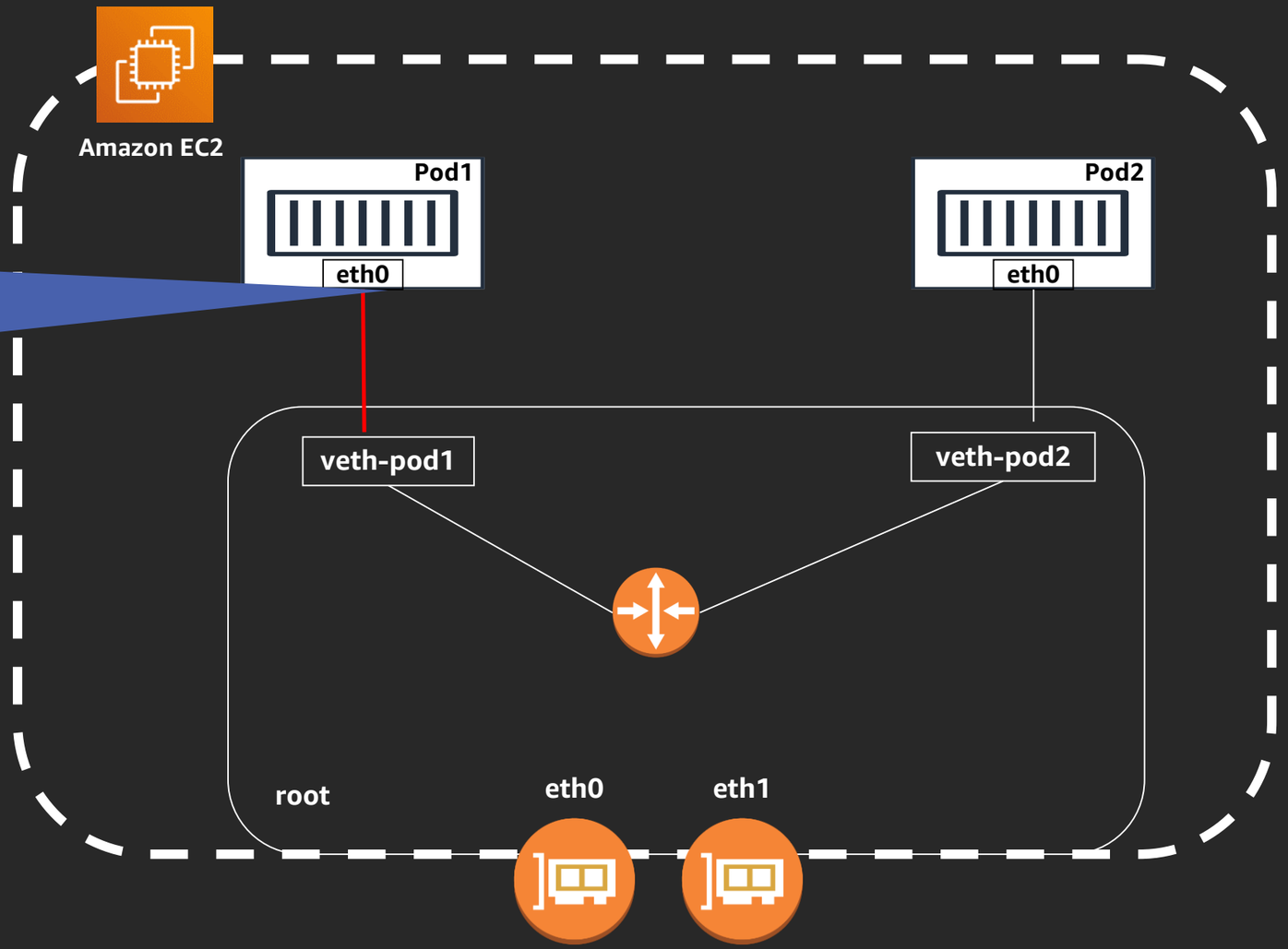
# amazon-vpc-cni-k8s: More IPs added



# Life of a packet: pod1-to-pod2, inside node

SRC-MAC: Pod1's eth0 MAC
SRC-IP: Pod1

DST-MAC: veth-pod1 MAC
DST-IP: Pod2

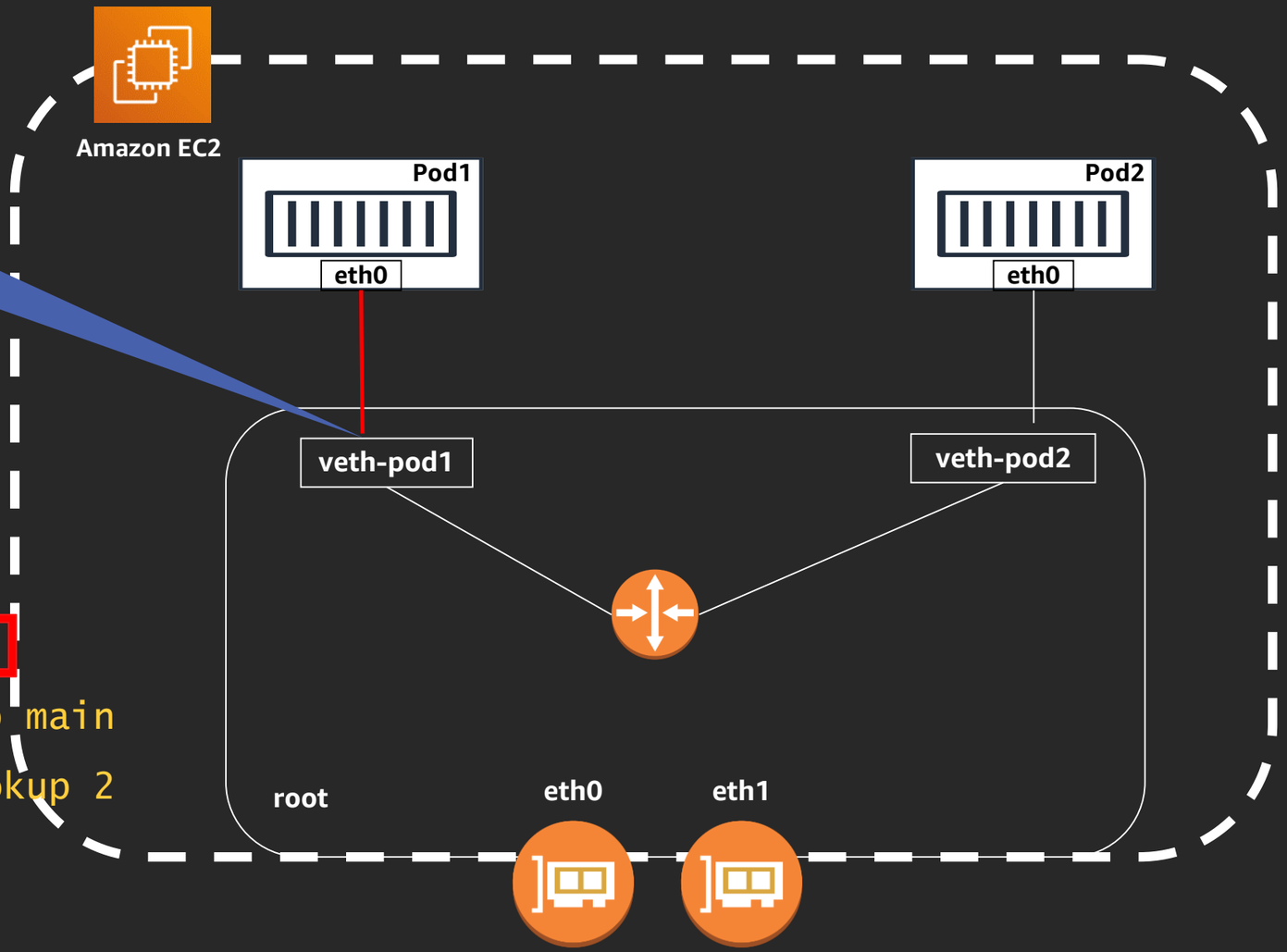


# Life of a packet: pod1-to-pod2, inside node

SRC-IP: Pod1

DST-IP: Pod2

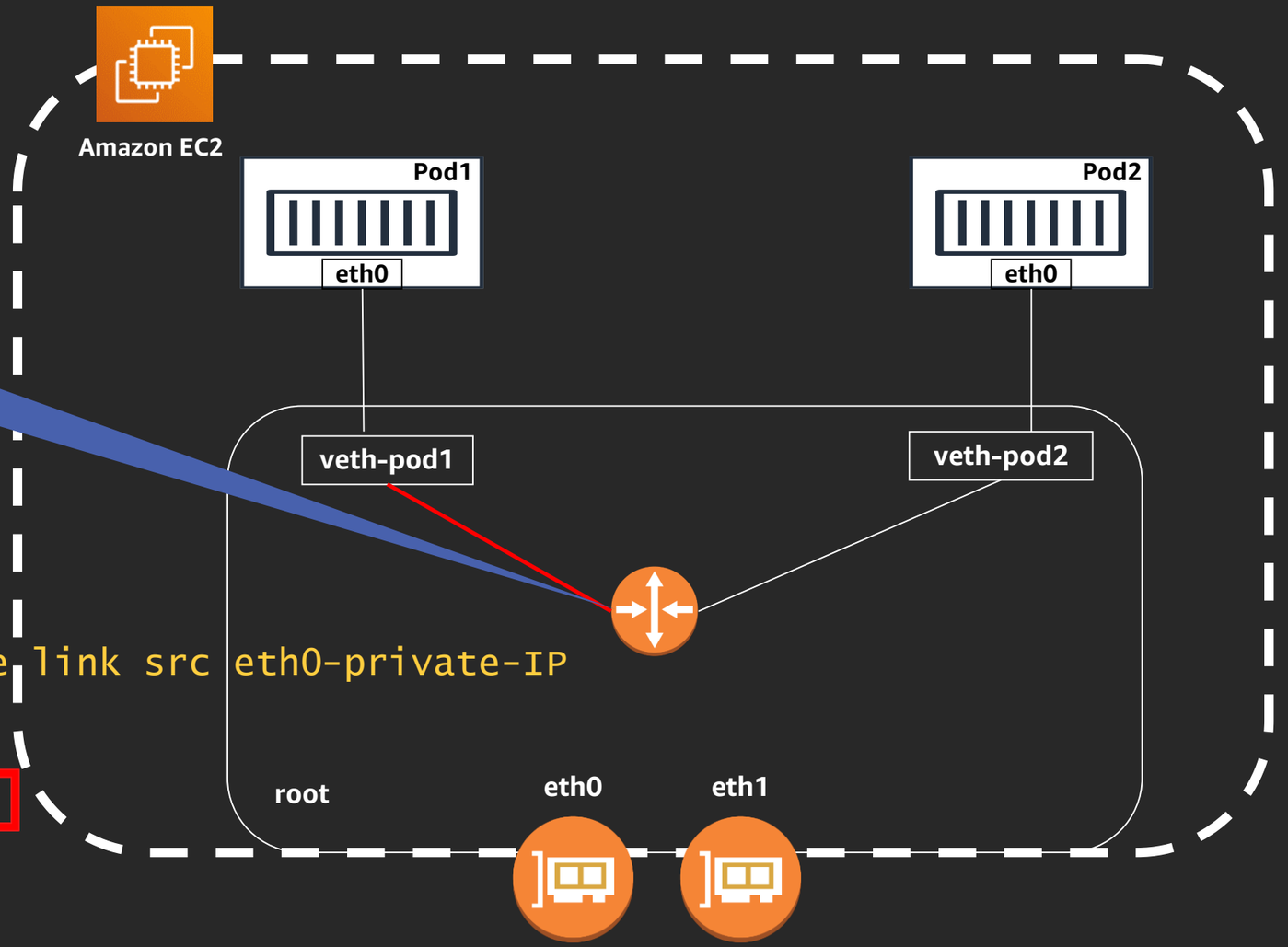
```
> ip rule
0:      from all lookup local
512:    from all to Pod1-IP lookup main
512:    from all to Pod2-IP lookup main
1024:   from all fwmark 0x80/0x80 lookup main
1536:   from Pod2-IP to 10.10.0.0/16 lookup 2
32766:  from all lookup main
32767:  from all lookup default
```



# Life of a packet: pod1-to-pod2, inside node

SRC-IP: Pod1

DST-IP: Pod2



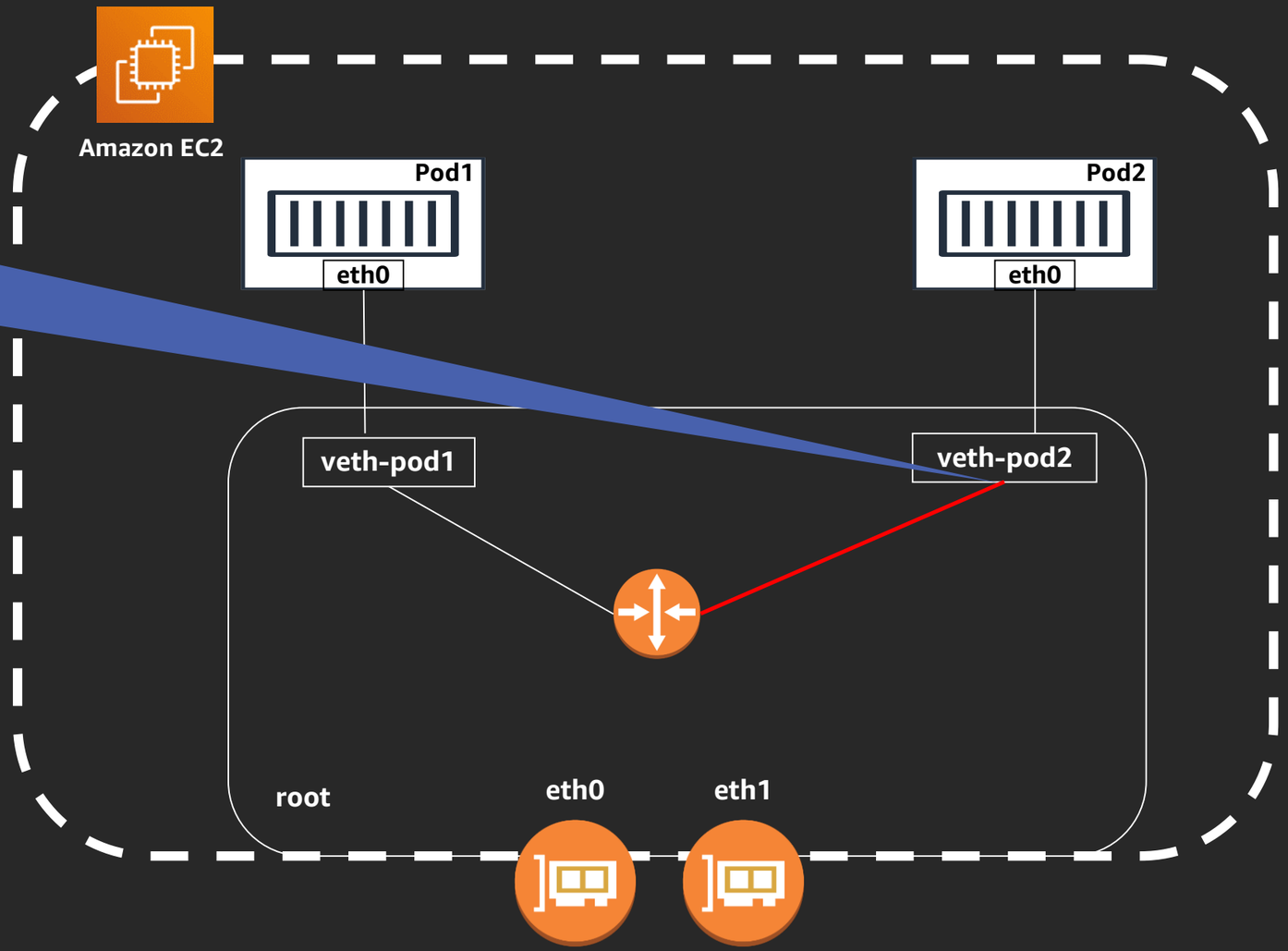

```
➢ ip route show table main

default via 10.10.0.1 dev eth0
10.10.0.0/19 dev eth0 proto kernel scope link src eth0-private-IP
Pod1-IP dev veth-pod1 scope link
Pod2-IP dev veth-pod2 scope link
169.254.169.254 dev eth0
```

# Life of a packet: pod1-to-pod2, inside node

SRC-IP: Pod1

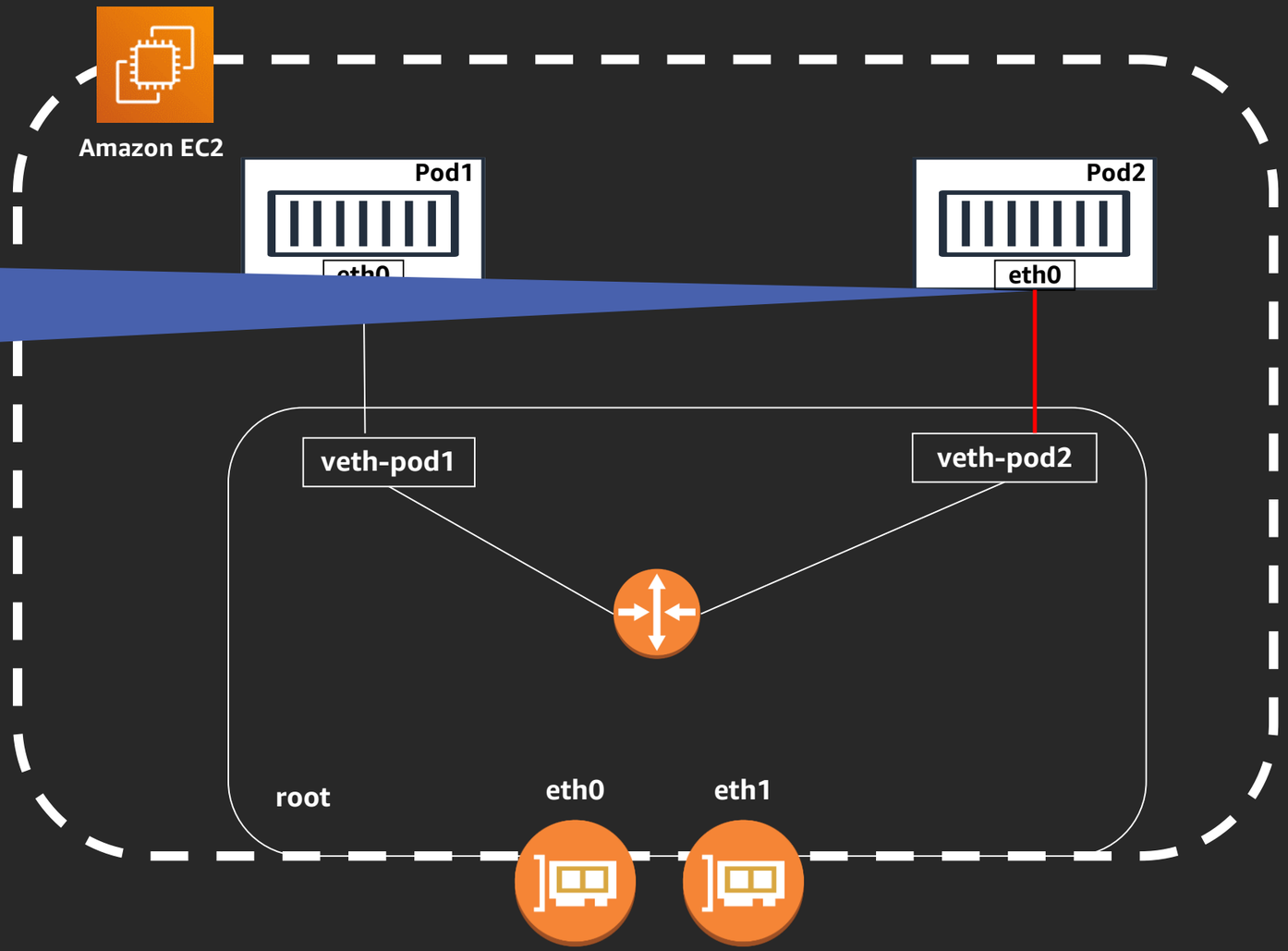
DST-MAC: veth-pod2
DST-IP: Pod2



# Life of a packet: pod1-to-pod2, inside node

SRC-MAC: veth-pod2 MAC
SRC-IP: Pod1

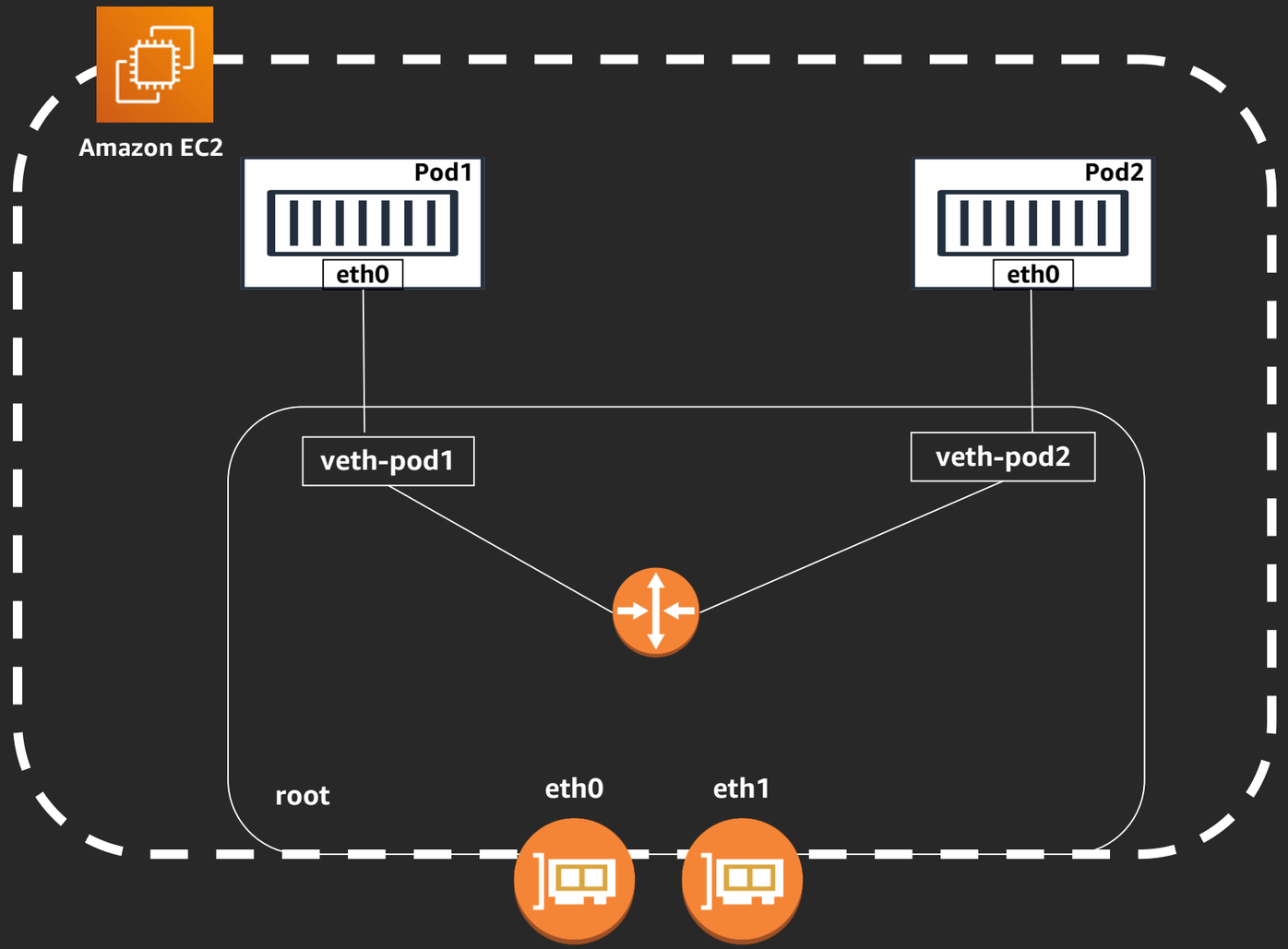
DST-MAC: Pod2 eth0 MAC
DST-IP: Pod2



# Life of a packet: pod1-to-pod2, inside node

Done!

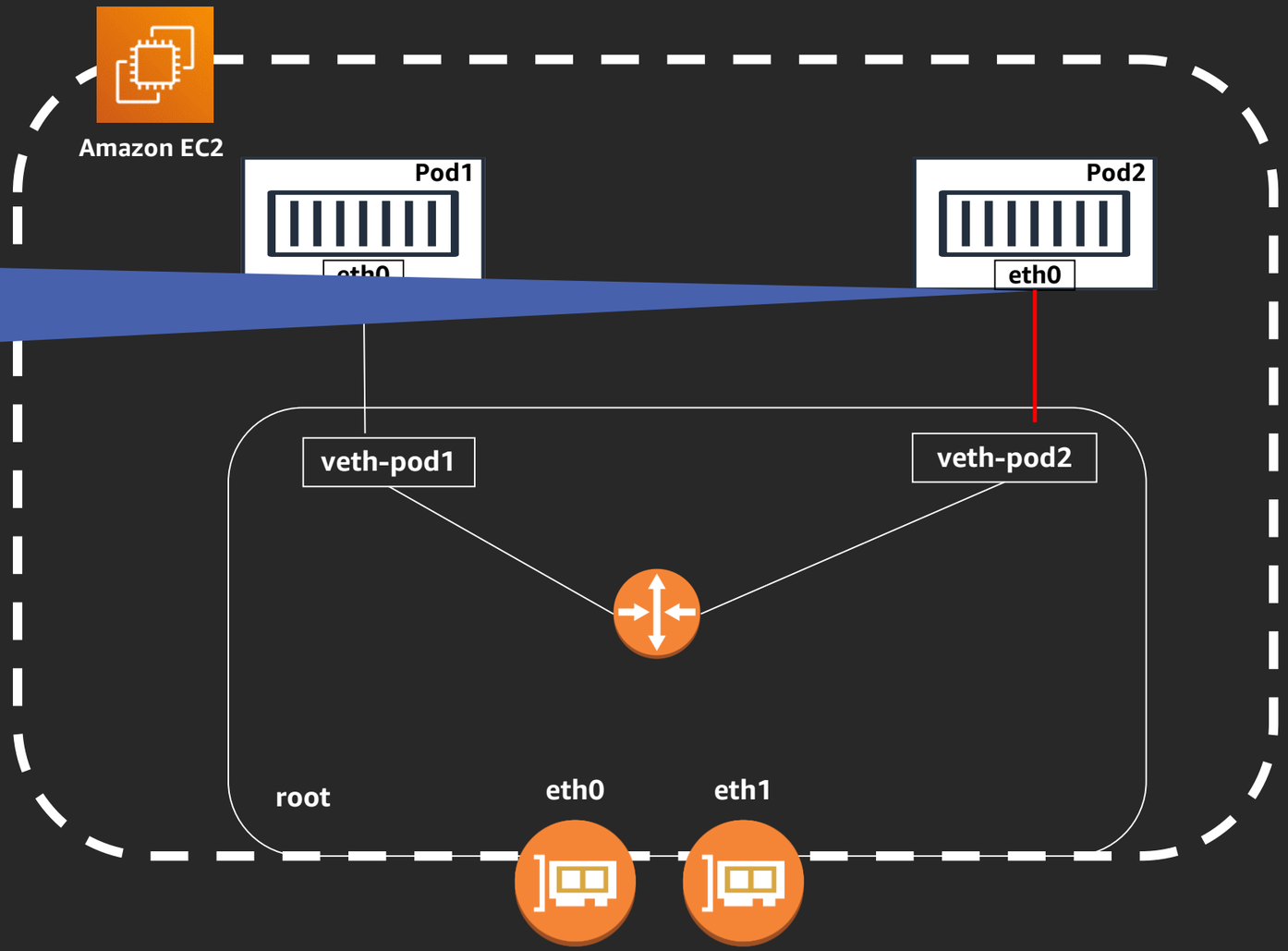
Now, pod 2 will send a package to pod 3 on another node



# Life of a packet: pod2-to-pod3, across nodes

SRC-MAC: Pod2's eth0 MAC
SRC-IP: Pod2

DST-MAC: veth-pod2 MAC
DST-IP: Pod3

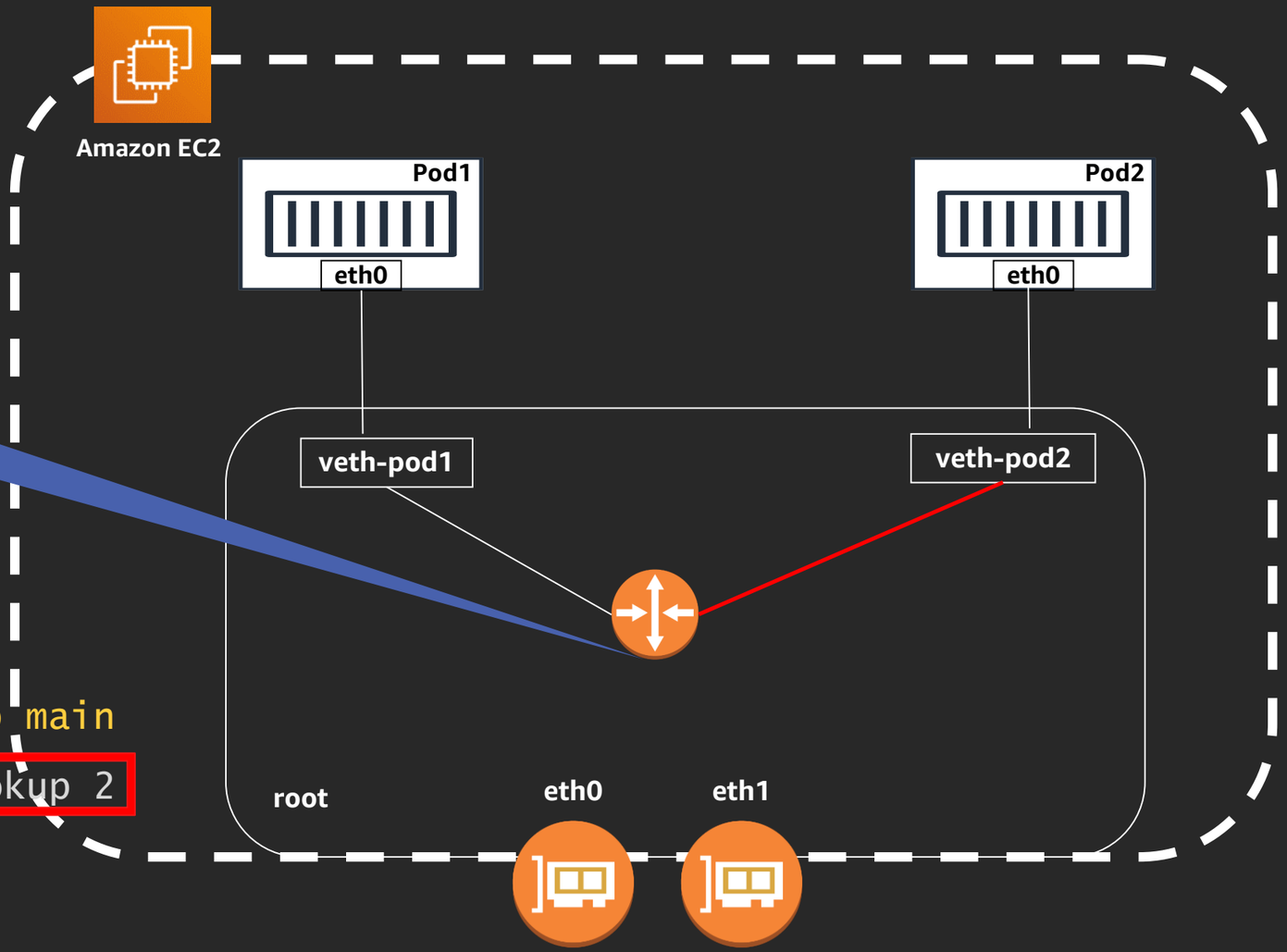


# Life of a packet: pod2-to-pod3, across nodes

SRC-IP: Pod2

DST-IP: Pod3

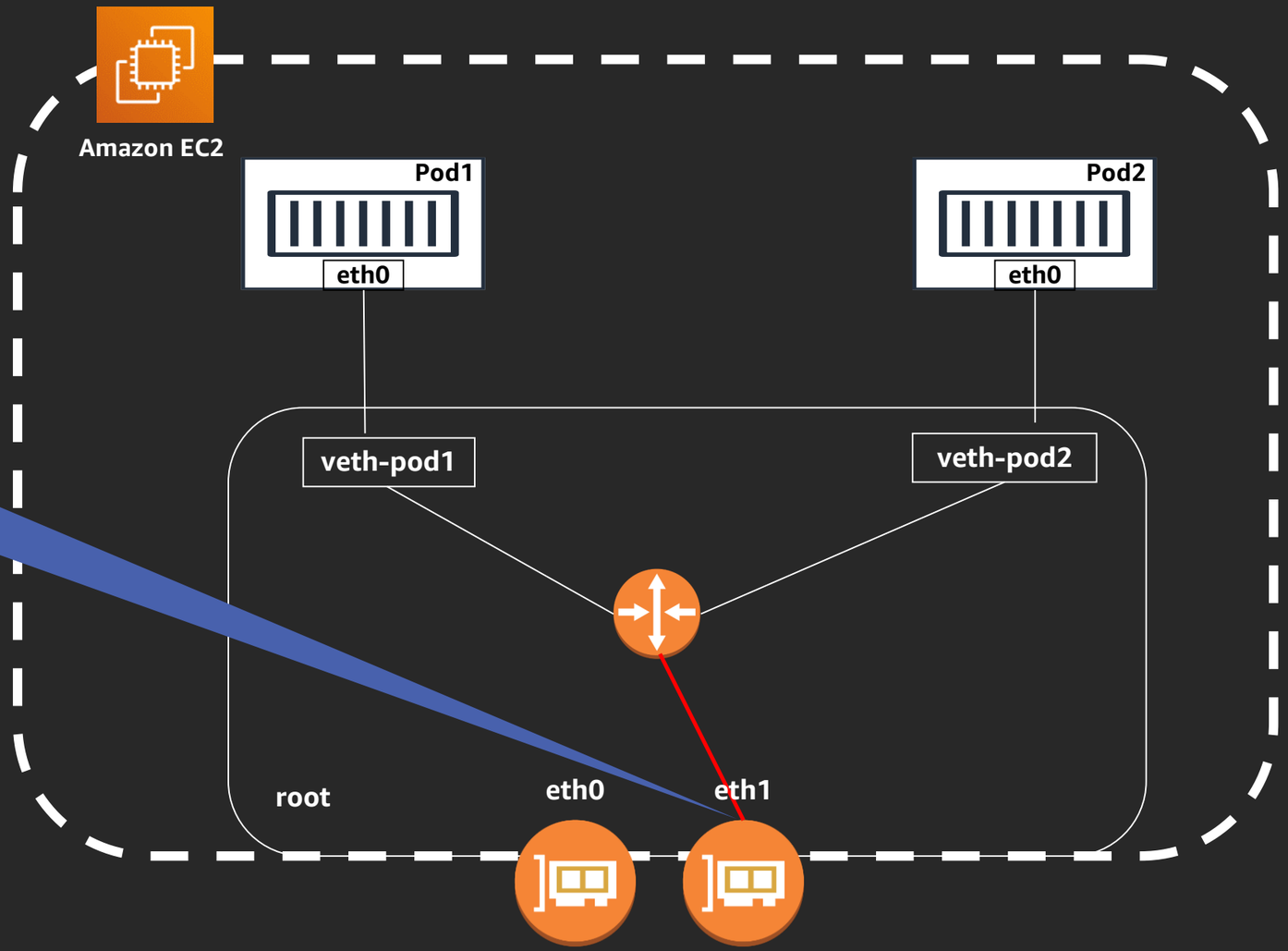
```
> ip rule
0:      from all lookup local
512:    from all to Pod1-IP lookup main
512:    from all to Pod2-IP lookup main
1024:   from all fwmark 0x80/0x80 lookup main
1536:   from Pod2-IP to 10.10.0.0/16 lookup 2
32766:  from all lookup main
32767:  from all lookup default
```



# Life of a packet: pod2-to-pod3, across nodes

SRC-MAC: eth1 MAC
SRC-IP: Pod2

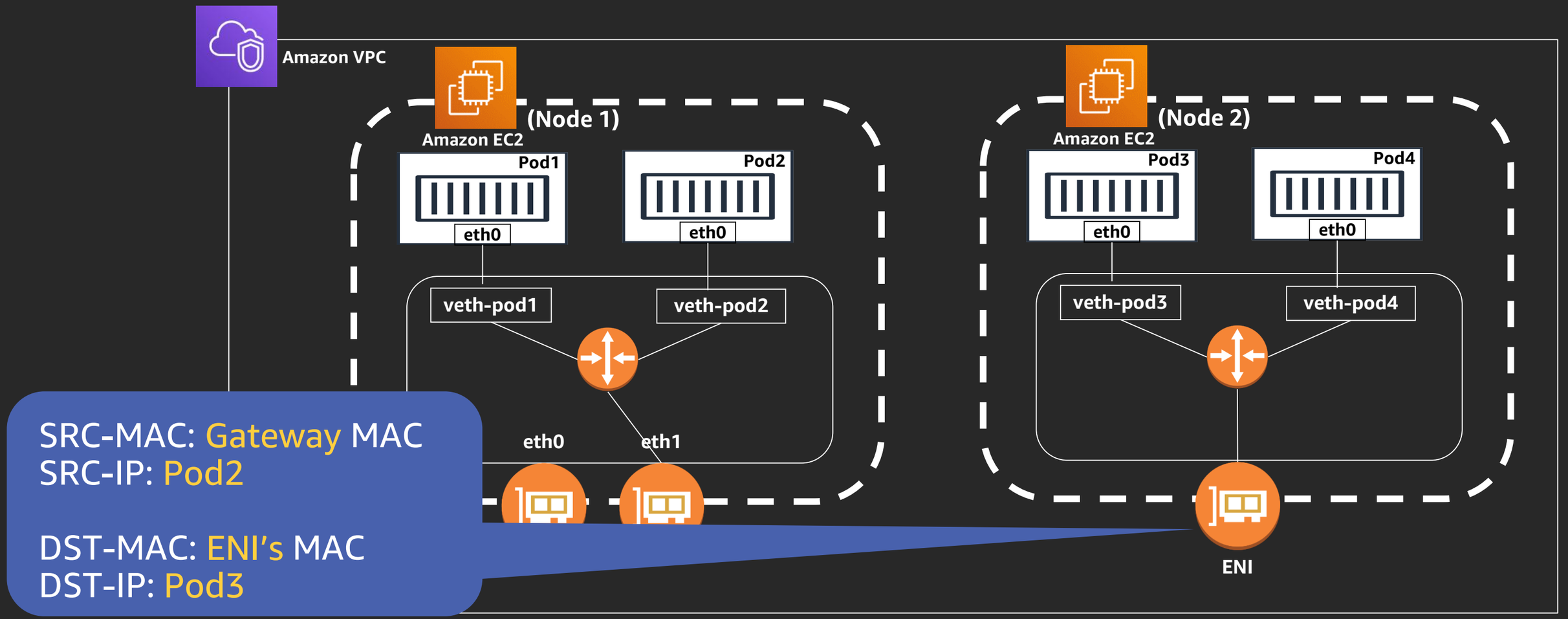
DST-MAC: Gateway MAC
DST-IP: Pod3




➢ `ip route show table 2`

```
default via VPC-router-IP dev eth1
VPC-router-IP dev eth1 scope link
```

# Life of a packet: pod2-to-pod3, across nodes



# HPC, ML, Big Data workload optimizations

- AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG
- WARM_ENI_TARGET
- WARM_IP_TARGET

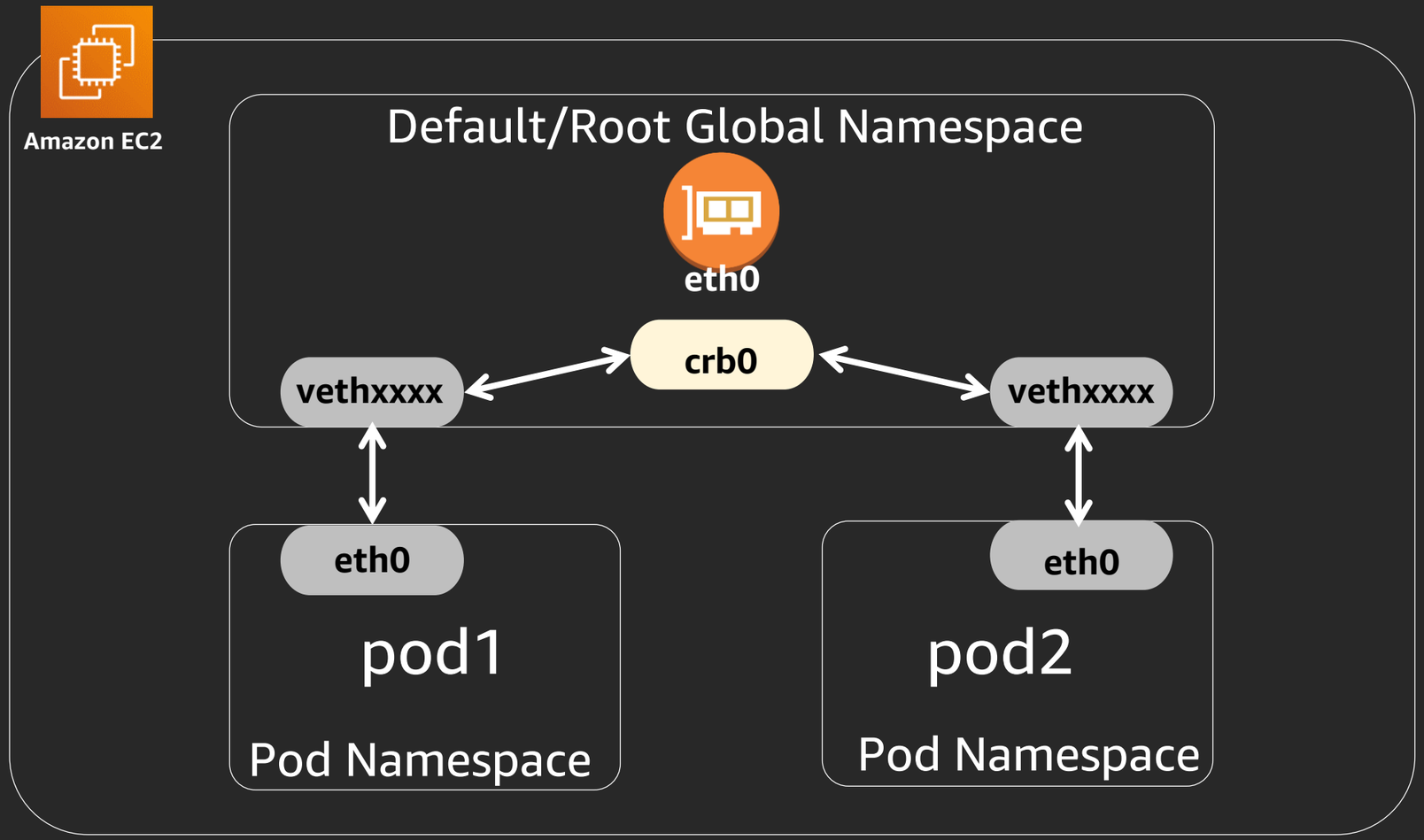
# amazon-vpc-cni-k8s: Configuration

| | Default | Purpose |
|---|---|---|
| WARM_IP_TARGET | 0 | For small subnets, reduce the IP usage; for small clusters with low pod churn |
| WARM_ENI_TARGET | 1 | Increase to pre-allocate more IPs for clusters with a lot of pod churn (also related to `MAX_ENI`) |
| AWS_VPC_K8S_CNI_ EXTERNALSNAT | false | When you have an external NAT gateway for the VPC |
| AWS_VPC_K8S_CNI_ EXCLUDE_SNAT_CIDRS | “” | When you have peered VPCs |
| AWS_VPC_K8S_CNI_ LOG_FILE | “” | Common to set to **`stdout`**. (Adjustable `_LOGLEVEL`) |

# Section 4

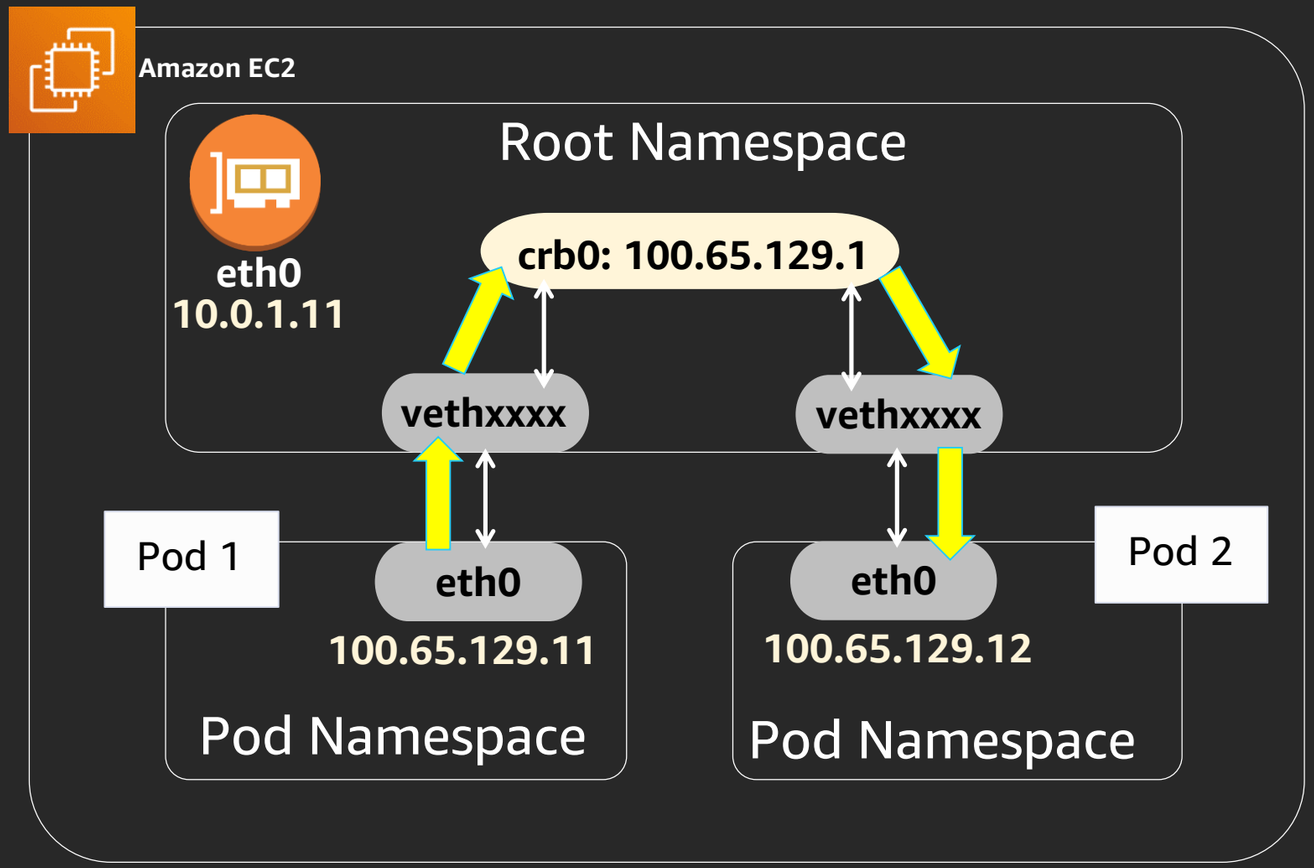
# Kubernetes on EC2–kops

# Networking explained



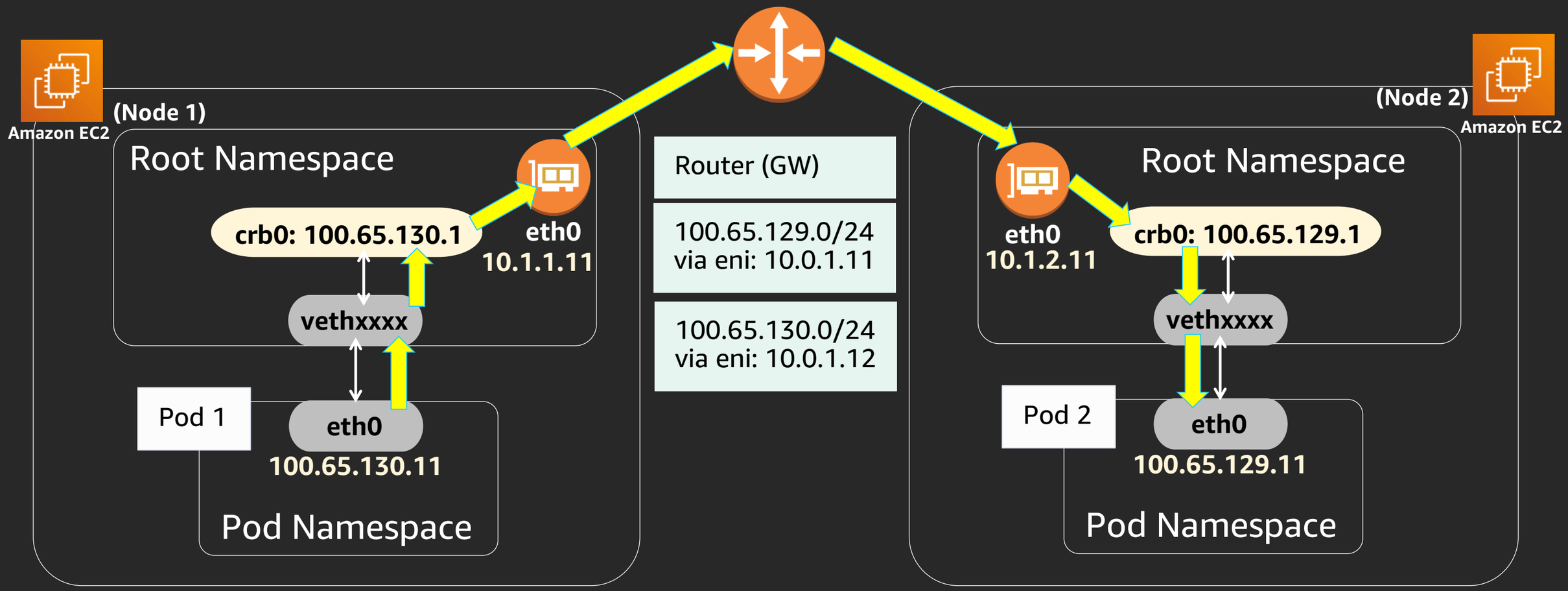
# Pod-to-pod communications (kops)

- Pods on the same instance:

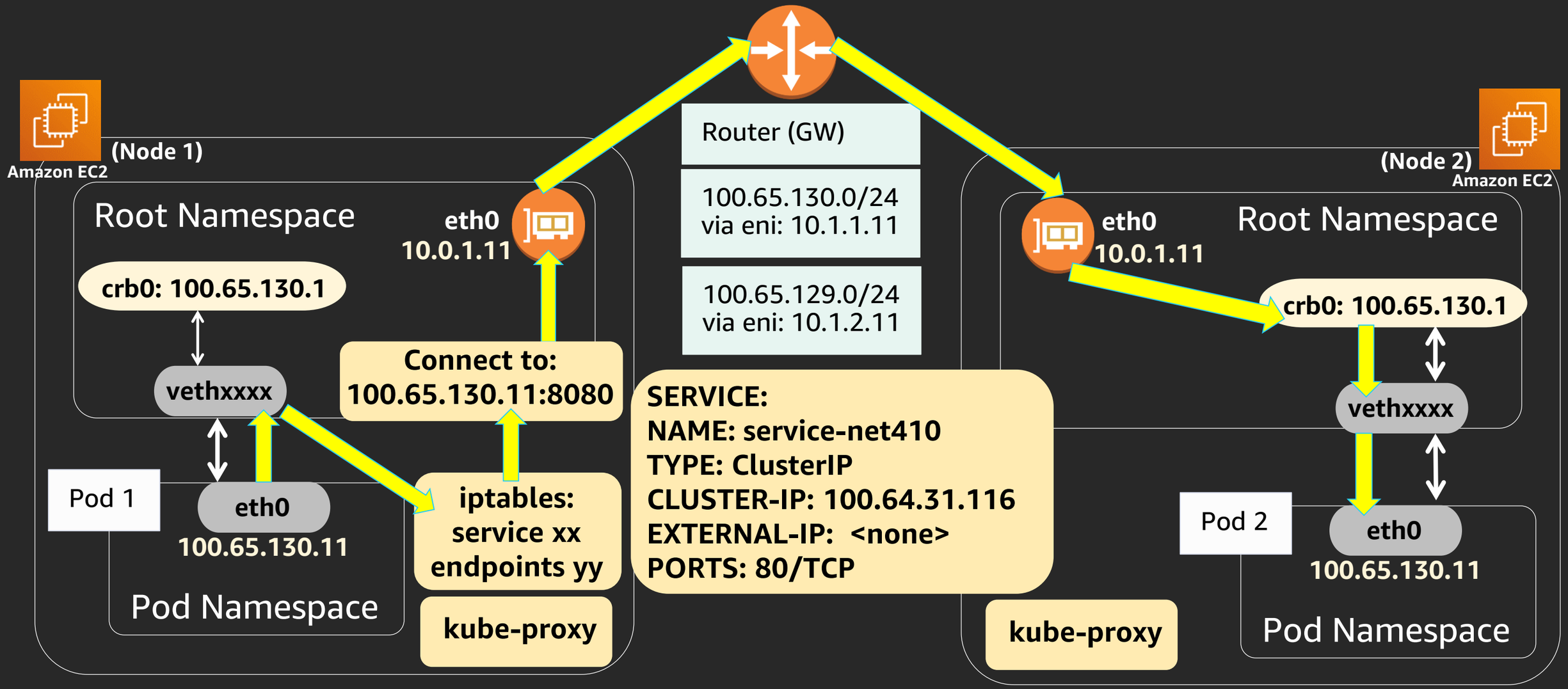


# Pod-to-pod communications (kops)

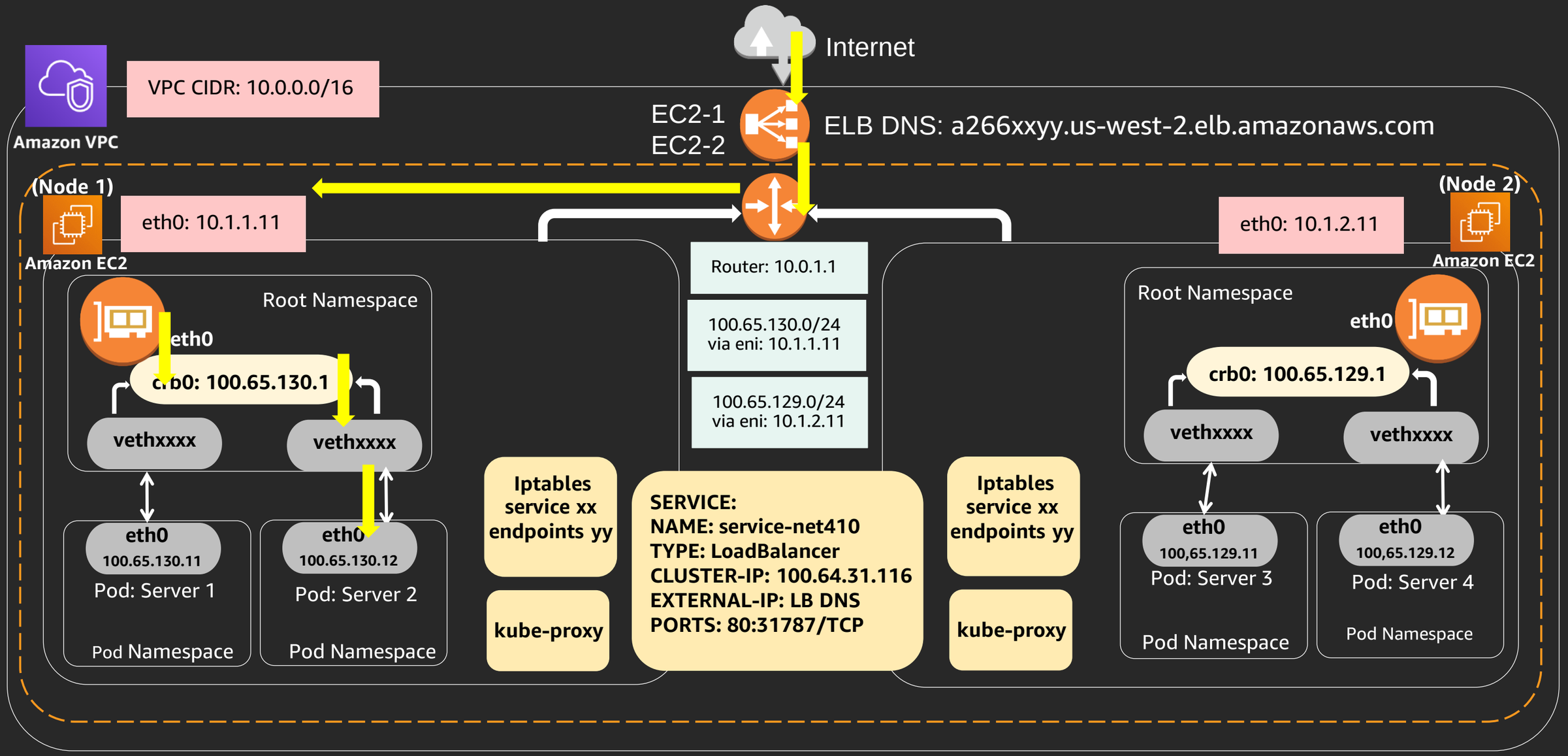
- Pods across EC2 instances:



# Pod-to-service communications (kops):



# External-to-internal communications (kops):



# Workshop activity module 0: Prerequisites

AWS re:Invent



aws

# Workshop activity module 1: Container networking

AWS re:Invent



aws

# Workshop activity module 2: Amazon EKS & kops cluster creation

AWS re:Invent



aws

# Workshop activity module 3: Amazon EKS cluster networking

AWS re:Invent



aws

# Workshop activity module 4: Kops cluster networking

AWS re:Invent



aws

# References:

- https://github.com/aws/amazon-vpc-cni-k8s
- https://aws.amazon.com/ec2/instance-types/
- https://kubernetes.io/
- https://kubernetes.io/docs/concepts/cluster-administration/networking/
- https://kubernetes.io/docs/concepts/services-networking/service/
- https://aws.amazon.com/blogs/compute/kubernetes-clusters-aws-kops/

# Learn networking with AWS Training and Certification

Resources created by the experts at AWS to help you build and validate networking skills

Free digital courses cover topics related to networking and content delivery, including Introduction to Amazon CloudFront and Introduction to Amazon VPC

Validate expertise with the
**AWS Certified Advanced Networking - Specialty** exam

Visit aws.amazon.com/training/paths-specialty



aws training and certification

# Thank you!

**Ikenna Izugbokwe**
ikeni@amazon.com

**Paavan Mistry**
paavan@amazon.co.uk

AWS re:Invent



aws

# Please complete the session survey in the mobile app.