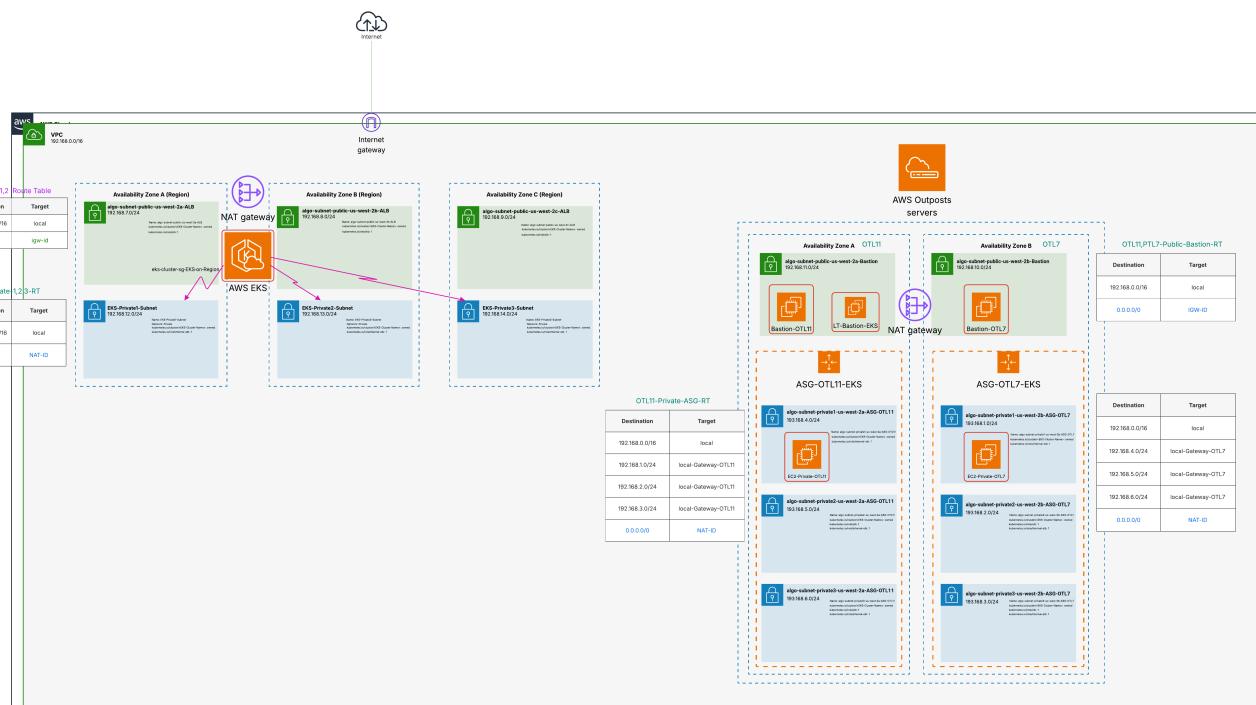
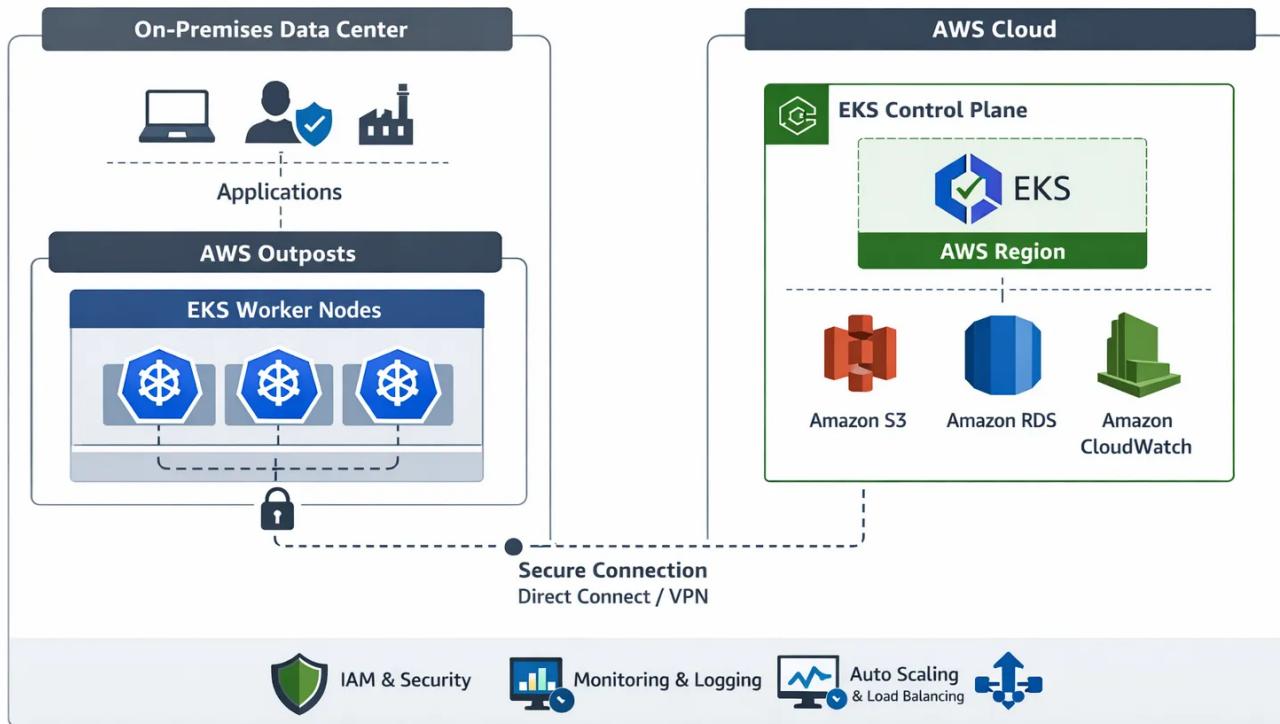


Amazon EKS (Self-managed)

E Architecture

Amazon EKS on AWS Outposts for Hybrid Cloud Operations



Amazon EKS (Control Plane) — Self-managed

1 Create IAM Roles

- **Cluster IAM Role** → Create → AWS Service → **Service: EKS (EKS - Cluster)** → **Add permissions (AmazonEKSClusterPolicy)**
- **Next**
 - **Role Name: EKSClusterRole**
 - **Description:** Allow the **K8s Cluster Control Plane** to manage AWS Resources on your behalf
- **Node Group** needs an **EC2 Instance IAM Role** to launch and register with a cluster.
 - **Node IAM Role** → AWS Service → **Service: EC2 (EC2)** → **Add permissions (AmazonSSMManagedInstanceCore, AmazonEKSCoComputePolicy, AmazonEC2ContainerRegistryReadOnly, AmazonEKS_CNI_Policy, AmazonEKSWorkerNodePolicy)**
 - **Role Name: EKSNodeRole**
 - **Description:** Allow **EC2 Instance** to call AWS services on your behalf.

2 Create the EKS Cluster

- Select **Custom Configuration**
- **Disable EKS Auto Mode**
 - ✓ You have now disabled AWS automatic **Node** management (**Self-Managed Nodes**)
- **Kubernetes control plane location: AWS Cloud**
- **Cluster name: EKS-on-Region**
- **Cluster IAM Role: EKSClusterRole**
- Keep the rest of the configurations default, and click on **next**
- **Networking**
 - Select <Your VPC>
 - Select **3 Private subnets (Have only one Route Table) & Created on Region & Tag them**
 - **Route them to NAT** (Allows the private subnet to access the internet)
 - **Sub-Name:** (Region)
 - **Tags for all Private Subnets of outpost**

```
Network: Private
Name: <Your-Subnet-Name>
kubernetes.io/cluster/<EKS-Cluster-Name>: shared
# ALB works inside the VPC only (Private Subnet)
kubernetes.io/role/internal-elb: 1
```

- **Security Group (Skip it)**

- **Security Group will be created by default — Control Plane**

The screenshot shows the AWS VPC Security Groups page with two entries:

Name	Security group ID	Security group name	VPC ID
eks-cluster-sg-EKS-on-Region-1946291870	sg-07120c86cc0c9b6f0	eks-cluster-sg-EKS-on-Region-1946291...	vpc-0779b6fdf8d788d1...
LT-SG-EKS-Worker-Node	sg-053c27e461caa0dc3	LT-SG-EKS-Worker-Node	vpc-0779b6fdf8d788d1...

Firewall**SG for Control Plane****SG for Worker Node**

**SG for
Inbound
&
Outbound**

The screenshot shows the AWS VPC Security Groups page with two sections:

- Edit inbound rules**: Contains three rules:
 - sg-053c27e461caa0dc3: All traffic to port 443 from 192.168.0.0/16.
 - sg-04441e088400c000: HTTPS to port 443 from 192.168.0.0/16.
 - sg-09c2319167d2014e: All traffic to port 443 from 192.168.0.0/16.
- Edit outbound rules**: Contains three rules:
 - sg-053c27e461caa0dc3: All traffic to port 443 to 192.168.0.0/16.
 - sg-0150a053211146e8d: All traffic to port 443 to 192.168.0.0/16.
 - sg-020f501ee03b7de: Custom TCP to port 10250 to 192.168.0.0/16.

The screenshot shows the AWS VPC Security Groups page with two sections:

- Edit inbound rules**: Contains seven rules:
 - sg-0395a23e7bd67a1: DNS (UDP) to port 53 to 192.168.0.0/16.
 - sg-0b127b30205d10d: Custom TCP to port 8443 to 192.168.0.0/16.
 - sg-00840200219ed5150: Custom TCP to port 40-3000 to 192.168.0.0/16.
 - sg-015aa0e18506501: HTTP to port 80 to 192.168.0.0/16.
 - sg-06814c0488ed7ac5: Custom TCP to port 10250 to 192.168.0.0/16.
 - sg-024494c11d23946: SSH to port 22 to 192.168.0.0/16.
 - sg-02d07949c1d80023: Custom TCP to port 23017 to 192.168.0.0/16.
- Edit outbound rules**: Contains two rules:
 - sg-02420741460524c: HTTPS to port 443 to 192.168.0.0/16.
 - sg-06d97f960212a27: All traffic to port 443 to 192.168.0.0/16.

- **Cluster endpoint access → Private (Access Cluster from inside VPC only)**

- Keep the rest of the configurations default, and click on **next**
- Skip Observability changes and click on **next**

- **Add the following add-ons:**

- **Amazon VPC CNI**
- **kube-proxy**
- **CoreDNS**
- **Amazon EBS CSI Driver**
- **Amazon EKS Pod Identity Agent**
- **Metrics Server**

- **Next → Next**

- Review and create the cluster.

The screenshot shows the AWS Elastic Kubernetes Service (EKS) Create EKS cluster page with the following details:

- Region:** United States (Oregon)
- Role:** AWSAdministratorAccess/Ahmed.abdelhamid@algo...
- Cluster Name:** proj682-algo-otl7andot11 (2270-4829-3427)
- CloudShell:** CloudShell
- Step 1: Configure cluster**
- Review and create**
- Progress:** Step 1 / 10

Step 2

- Step 2 Specify networking
- Step 3 Configure observability
- Step 4 Select add-ons
- Step 5 Configure selected add-ons settings
- Step 6 Review and create

Step 1: Cluster Edit

Cluster configuration

Name	Kubernetes version
EKS-on-Region	1.34
EKS Auto Mode	Upgrade policy
Disabled	Standard support
Cluster IAM role	Kubernetes cluster administrator access
arn:aws:iam::227048293427:role/EKSClusterRole	Allow cluster administrator access
Authentication mode	Control plane scaling tier
EKS API	Standard

ARC Zonal shift

ARC Zonal shift
Disabled

Deletion protection

Deletion protection
Disabled

Tags (0)

Tags that you've added. Each tag consists of a key and an optional value.

Key	Value
No tags	
This cluster does not have any tags.	

Step 2: Networking Edit

Networking

These properties cannot be changed after the cluster is created.

VPC	Subnets
vpc-0779b6fdf8d7882df	subnet-007120692f1211954 subnet-034f8bb5247adf32c subnet-02d4bc4a4d489638c
Cluster IP address family	
IPv4	

Cluster endpoint access

API server endpoint access
Private

Step 3: Observability Edit

Container network observability

Info

Network monitoring status	Included capabilities
Disabled	Service map Flow table
	Performance metric endpoint

Network flow monitor configuration

Network flow monitor name	Local resources	Remote resources
eks-EKS-on-Region-flow-monitor	EKS cluster	Everywhere in us-west-2

Control plane logs

API server	Authenticator	Scheduler
on	on	on
Audit	Controller manager	
on	on	

Step 4: Add-ons Edit

Selected add-ons (10)

Find add-on

Add-on name	Type	Status
amazon-cloudwatch-observability	observability	Ready to install
aws-ebs-csi-driver	storage	Ready to install
cert-manager	security	Ready to install
coredns	networking	Ready to install
eks-node-monitoring-agent	observability	Ready to install
eks-pod-identity-agent	security	Ready to install
external-dns	networking	Ready to install

Step 5: Versions

Selected add-ons version (10)

Add-on name	Version
amazon-cloudwatch-observability	v4.8.0-eksbuild.1
aws-ebs-csi-driver	v1.54.0-eksbuild.1
cert-manager	v1.19.2-eksbuild.1
coredns	v1.12.3-eksbuild.1
eks-node-monitoring-agent	v1.4.3-eksbuild.2
eks-pod-identity-agent	v1.3.10-eksbuild.2
external-dns	v0.20.0-eksbuild.2
kube-proxy	v1.34.0-eksbuild.2
metrics-server	v0.8.0-eksbuild.6
vpc-cni	v1.20.4-eksbuild.2

EKS Pod Identity (4)

Add-on name	IAM role	Service account
amazon-cloudwatch-observability	Not set	cloudwatch-agent
aws-ebs-csi-driver	Not set	ebs-csi-controller-sa
external-dns	Not set	external-dns
vpc-cni	Not set	aws-node

Actions: Cancel, Previous, Create

CloudShell Feedback © 2026, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

- Wait 8–10 minutes for provisioning

Amazon Elastic Kubernetes Service > Clusters > EKS-on-Region

EKS-on-Region

Cluster info

Status: Active	Kubernetes version: 1.34	Support period: Standard support until December 2, 2026	Provider: EKS
Cluster health: 0	Upgrade insights: 3	Node health issues: 0	Capability issues: 0

Overview Resources Compute Networking Add-ons 1 Capabilities Access Observability Update history &

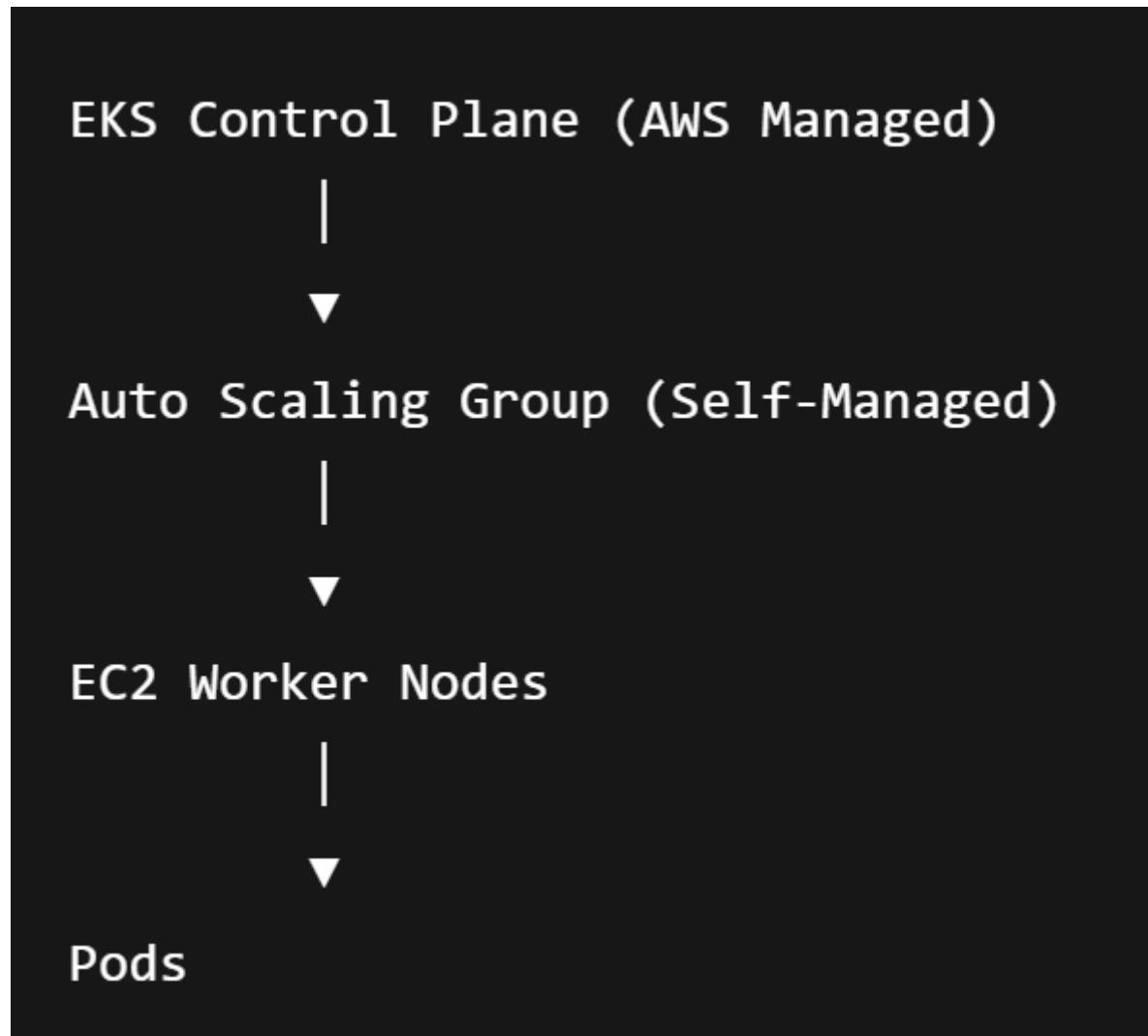
Details

API server endpoint: https://D2E489E4E5CCE7EC65EC8C71766183E7.s1.us-west-2.eks.amazonaws.com	OpenID Connect provider URL: https://oidc.eks.us-west-2.amazonaws.com/id/D2E489E4E5CCE7EC65EC8C71766183E7	Created: 35 minutes ago
Certificate authority	Cluster IAM role ARN	Cluster ARN: arn:aws:eks:us-west-2:227048293427:cluster/EKS-

3 Create a Node Group

- EKS Cluster is working (**without Node Group**)
- Managed **Nodegroups** are not supported on **Outposts**

Self-Managed Worker Nodes



🛠️ Create 3 Launch Templates

Launch Templates (3) [Info](#)

<input type="checkbox"/>	Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time	Created By
<input type="checkbox"/>	lt-0e051802d828a824d	LT-OTL7-ASG-EKS	1	1	2026-01-19T07:36:44.000Z	arn:aws:sts::22704
<input type="checkbox"/>	lt-0ad662a215fdadba2	LT-OTL11-ASG-EKS	1	1	2026-01-19T07:29:58.000Z	arn:aws:sts::22704
<input type="checkbox"/>	lt-0da60c1339cd65def	LT-Bastion-EKS	1	1	2026-01-19T08:31:21.000Z	arn:aws:sts::22704

- **Amazon Machine Image (AMI):**
 - Before choosing **AMI-xx**, ensure that the **Amazon-eks-node**
 - From **Community AMIs (500): Search for EKS Optimized**
 - Choose the right one
 - After choosing the right **ami**, check that **kubelet** installed through `sudo systemctl status kubelet` via **SSM**
- **Instance type — OTL11 m5.xlarge**
- **Instance type — OTL7 c5.xlarge**

- **Instance type — m5.xlarge (Bastion)**
- **Key pair name:**
- **Networking:**
 - **Subnet: Don't include in launch template**
 - **Availability Zone: Don't include in launch template**
 - **Security Group:**
 - **Name: LT-SG-EKS-Worker-Node**
 - **Security Group: Add → SGs — Worker Node**
- **EBS Volume: 20GB, gp2**
- **Advanced details**
 - **IAM instance profile**
 - **Add Role for Launch Template**
 - **Role Name: EKSNodeRole**
 - **Create a Role for Bastion**
 - **Use Case: EC2 (EC2)**
 - **Add permissions**
 - **Role Name: abdelhamid-SSM-Role**
 - **Attach it to EC2**
 - **Connect with EC2**
 - **AWS Systems Manager** → Session Manager → Start session
 - **Metadata response hop limit: 2**
 - ****User Data** for (Launch Template) — **NodeConfig****

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==BOUNDARY=="

====BOUNDARY==
Content-Type: text/x-shellscrip; charset="us-ascii"

#!/bin/bash
set -ex
# If any command fails → the script stops immediately ✘

====BOUNDARY==
Content-Type: application/node.eks.aws

---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
```

```

spec:
  cluster:
    name: <EKS-Cluster-Name>
    apiServerEndpoint: <EKS-endpoint>
    certificateAuthority: <EKS-certificate-Authority>
    # → The **IP range** from which the **Pods** will take an IP address
    cidr: "10.100.0.0/16"
  kubelet:
    config:
      maxPods: 110
  node:
    labels:
      app: my-app
      # Kubernetes will not be able to differentiate between OTL1 and OTL7
      # outpost: otl1
      # outpost: otl11
      eks.amazonaws.com/capacityType: ON_DEMAND

```

-----BOUNDARY-----

- **User Data for (BastionTemplate)**

```

#!/bin/bash
yum update -y
sudo yum install -y awscli
# Install kubectl
curl -LO https://dl.k8s.io/release/$(curl -Ls
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl
chmod +x kubectl
sudo mv kubectl /usr/local/bin/
kubectl version --client

```

🔧 Create 2 Auto Scaling Groups

We will create 2 Auto Scaling Groups; AWS does not allow a single instance to appear in more than one Outpost.

- **Name: ASG-OTL11-EKS**
- **Launch template: LT-OTL11-ASG-EKS**
- **Name: ASG-OTL7-EKS**
- **Launch template: LT-OTL7-ASG-EKS**
- **VPC: 'Your-VPC'**
- **Availability Zones and subnets:**
 - **3 Private Subnets (OTL11)**

- 3 Private Subnets (OTL7)
- Availability Zone distribution: Balanced best effort
- Load balancing: No load balancer
- VPC Lattice integration options: No VPC Lattice service
- Group size
 - Desired capacity: 3
 - Min desired capacity: 1
 - Max desired capacity: 7
- Automatic scaling - optional: No scaling policies
- Instance maintenance policy: No Policy
- Next
- After creating the **Auto Scaling Group**, → Check that **kubelet** is installed on **Worker Node**
 - Go to **AWS System Manager (SSM)** → Choose the **right ami** → Start Session → **sudo systemctl status kubelet**

Session ID: [REDACTED] Shortcuts Instance ID: i-[REDACTED]645b Terminate

```
sh-5.2$ sudo systemctl status kubelet
● kubelet.service - Kubernetes Kubelet
   Loaded: loaded (/etc/systemd/system/kubelet.service; disabled; preset: disabled)
     Active: active (running) since Mon 2026-01-19 07:46:26 UTC; 1 day 23h ago
       Docs: https://github.com/kubernetes/kubernetes
    Process: 2246 ExecStartPre=/sbin/iptables -P FORWARD ACCEPT -w 5 (code=exited, status=0/SUCCESS)
   Main PID: 2256 (kubelet)
     Tasks: 16 (limit: 18865)
    Memory: 141.8M
      CPU: 5h 52min 45.990s
     CGroup: /runtime.slice/kubelet.service
             └─2256 /usr/bin/kubelet --image-credential-provider-bin-dir=/etc/eks/image-credential-provider --image-credential-provider-config=/etc/eks/image-credential-p...
Jan 21 07:18:55 ip-192-168-6-14.us-west-2.compute.internal kubelet[2256]: I0121 07:18:55.909283 2256 ????:1] "http: TLS handshake error from 127.0.0.1:56328: no serving
Jan 21 07:18:55 ip-192-168-6-14.us-west-2.compute.internal kubelet[2256]: I0121 07:18:55.909786 2256 ????:1] "http: TLS handshake error from 127.0.0.1:56336: no serving
Jan 21 07:18:55 ip-192-168-6-14.us-west-2.compute.internal kubelet[2256]: I0121 07:18:55.910343 2256 ????:1] "http: TLS handshake error from 127.0.0.1:56346: no serving
Jan 21 07:18:55 ip-192-168-6-14.us-west-2.compute.internal kubelet[2256]: I0121 07:18:55.910854 2256 ????:1] "http: TLS handshake error from 127.0.0.1:56352: no serving
Jan 21 07:18:55 ip-192-168-6-14.us-west-2.compute.internal kubelet[2256]: I0121 07:18:55.939661 2256 ????:1] "Probe failed" probeType="Readiness" pod="kube-system"
Jan 21 07:18:55 ip-192-168-6-14.us-west-2.compute.internal kubelet[2256]: I0121 07:18:55.943505 2256 ????:1] "http: TLS handshake error from 127.0.0.1:56358: no serving
Jan 21 07:18:55 ip-192-168-6-14.us-west-2.compute.internal kubelet[2256]: I0121 07:18:55.983424 2256 ????:1] "http: TLS handshake error from 127.0.0.1:56368: no serving
Jan 21 07:18:56 ip-192-168-6-14.us-west-2.compute.internal kubelet[2256]: I0121 07:18:56.023442 2256 ????:1] "http: TLS handshake error from 127.0.0.1:56374: no serving
Jan 21 07:18:56 ip-192-168-6-14.us-west-2.compute.internal kubelet[2256]: I0121 07:18:56.073646 2256 ????:1] "http: TLS handshake error from 127.0.0.1:56378: no serving
Jan 21 07:18:56 ip-192-168-6-14.us-west-2.compute.internal kubelet[2256]: I0121 07:18:56.103380 2256 ????:1] "http: TLS handshake error from 127.0.0.1:56394: no serving
lines 1-22/22 (END)
```

⌚ Testing (Bastion)

Create an EC2 Instance from Launch Template (LT-Bastion-EKS)

Instances (1/1) Info Last updated 1 minute ago Connect Instance state ▾ Actions ▾ Launch instances ▾

Find Instance by attribute or tag (case-sensitive) All states ▾

Instance ID = i-009e5d97b7e9fd066 X Clear filters < 1 > ⌂

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input checked="" type="checkbox"/> Bastion-EKS	i-009e5d97b7e9fd066	Running ⓘ ⓘ	m5.xlarge	2/2 checks passed View alarms +		us-west-2a

Pre-Test Setup & Validation

Before Testing – EKS Access

Go to **EKS (Access) → IAM access entries → Create**

IAM principal ARN:

EKSNodeRole (Worker Node Role)

Policy name:

AmazonEKSClusterAdminPolicy

Click **Add policy**

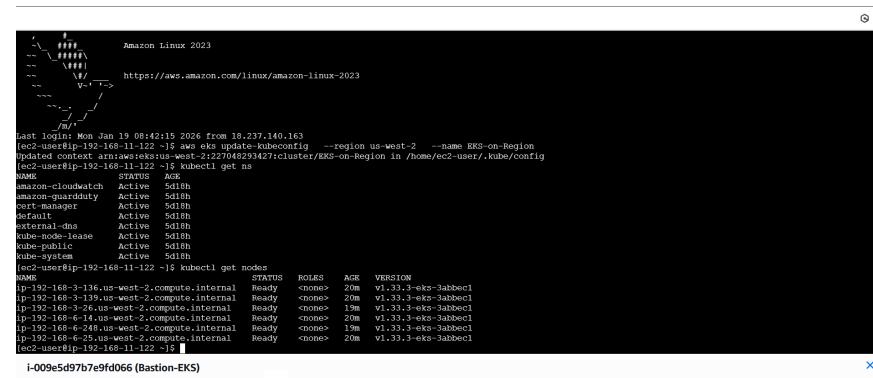
Bastion Access & Cluster Validation

SSH to Bastion host

```
aws eks update-kubeconfig --region <Region> --name <EKS-Name>
```

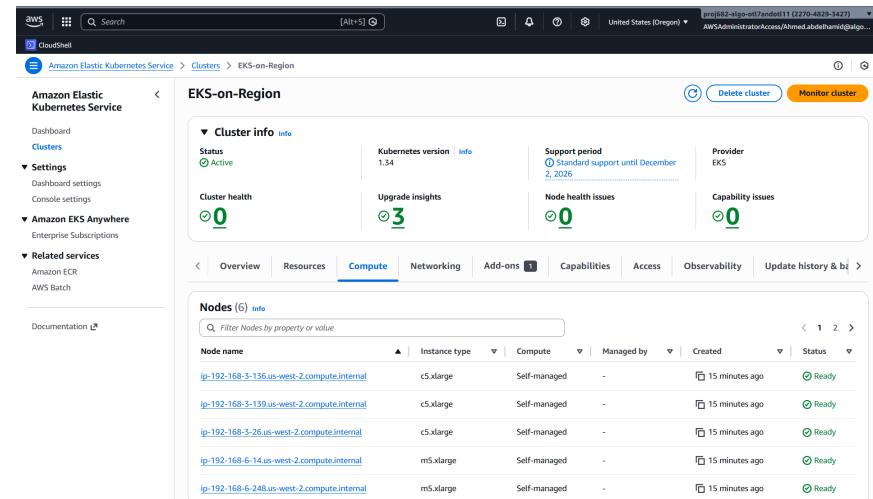
kubectl get ns

kubectl get nodes



```
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Mon Jan 19 08:42:15 2026 from 18.237.140.163
(ec2-user@ip-192-168-11-122 ~)$ aws eks update-kubeconfig --region us-west-2 --name EKS-on-Region
(ec2-user@ip-192-168-11-122 ~)$ kubectl get ns
NAME          STATUS
amazon-cloudwatch  Active
amazon-guard-duty  Active
cart-manager    Active
default        Active
external-dns    Active
kube-node-lease Active
kube-public     Active
kube-system    Active
(ec2-user@ip-192-168-11-122 ~)$ kubectl get nodes
NAME           STATUS   ROLES      AGE   VERSION
ip-192-168-3-136.us-west-2.compute.internal Ready   <none>   20m   v1.33.3-eks-3abbecl
ip-192-168-3-26.us-west-2.compute.internal Ready   <none>   20m   v1.33.3-eks-3abbecl
ip-192-168-6-14.us-west-2.compute.internal Ready   <none>   20m   v1.33.3-eks-3abbecl
ip-192-168-6-248.us-west-2.compute.internal Ready   <none>   19m   v1.33.3-eks-3abbecl
ip-192-168-11-122.us-west-2.compute.internal Ready   <none>   20m   v1.33.3-eks-3abbecl
(ip-192-168-11-122 ~)$ i-009e5d97b7e9fd066 (Bastion-EKS)
PublicIPs: 44.247.229.82 PrivateIP: 192.168.11.122
```

The screenshot shows the AWS EKS Cluster Overview page for the 'EKS-on-Region' cluster. Key details include:

- Cluster info:** Status: Active, Kubernetes version: 1.34, Support period: Standard support until December 2, 2026.
- Cluster health:** 0 issues.
- Nodes (6):**

Name	Instance type	Compute	Managed by	Created	Status
ip-192-168-3-136.us-west-2.compute.internal	c5.xlarge	Self-managed	-	15 minutes ago	Ready
ip-192-168-3-139.us-west-2.compute.internal	c5.xlarge	Self-managed	-	15 minutes ago	Ready
ip-192-168-3-26.us-west-2.compute.internal	c5.xlarge	Self-managed	-	15 minutes ago	Ready
ip-192-168-6-14.us-west-2.compute.internal	m5.xlarge	Self-managed	-	15 minutes ago	Ready
ip-192-168-6-248.us-west-2.compute.internal	m5.xlarge	Self-managed	-	15 minutes ago	Ready
i-009e5d97b7e9fd066 (Bastion-EKS)					Ready