1. What is Amazon EKS and how does it differ from self-managed Kubernetes?

Amazon EKS (Elastic Kubernetes Service) is a **fully managed Kubernetes service** by AWS.

✅ **EKS handles**:

- The **Kubernetes control plane**
- **Automatic patching**, upgrades, high availability
- **Integration** with AWS IAM, VPC, and services

🆚 **Differences from self-managed Kubernetes**:

Feature	Amazon EKS	Self-Managed Kubernetes
Control Plane	Managed by AWS	You install & manage it
HA & Scaling	Built-in	Manual
Security	Integrated with IAM	You configure RBAC manually
Upgrades	AWS does it	Manual patching
Cost	Pay for control plane + nodes	Pay only for nodes, but higher ops cost

---

2. What are the components of EKS architecture?

✅ **Key components of EKS architecture**:

Component	Role
EKS Control Plane	Manages etcd, scheduler, API server
Worker Nodes	EC2 or Fargate; run user workloads
Node Groups	Group of EC2/Fargate nodes
VPC/Subnets	Network layer for the cluster
IAM Roles	Authentication & permissions
kubectl + aws-auth	CLI access to cluster
Load Balancers	For Service exposure (ALB, NLB)

---

3. What is the EKS Control Plane and what does AWS manage?

The **EKS Control Plane** is the **brain of your cluster** — AWS manages this part so you don't have to.

🔧 **AWS Manages**:

- API server
- etcd (key-value store)
- Scheduler & controllers
- High availability across AZs

- Automatic patching & scaling

You only manage the **worker nodes and workloads**.

---

4. How do you create and configure an EKS cluster?

✅ **3 main options**:

1. **AWS Console** (GUI)
2. **eksctl CLI** – easiest
3. **Terraform / CloudFormation** – IaC

Using eksctl (recommended for quick setup):

bash

```
------
eksctl create cluster \
 --name my-cluster \
 --region ap-south-1 \
 --nodes 2 \
 --node-type t3.medium \
 --managed
```

🛠️ eksctl creates:

- EKS cluster
- VPC & networking
- IAM roles
- Node Group

---

5. What are EKS node groups and how do you manage them?

A **Node Group** is a group of EC2 instances that register as **worker nodes** with your EKS cluster.

✅ Used to:

- Host Pods
- Define instance type, scaling policies
- Separate workloads (e.g., prod vs dev)

You can create/manage them via:

- eksctl
- EKS Console
- Terraform / CloudFormation

🎯 You can have **multiple node groups** in a cluster (e.g., GPU, spot, on-demand).

---

6. What is the difference between managed node groups and self-managed nodes?

Here's a simple explanation of the difference between managed node groups and self-managed nodes in AWS:

Managed Node Groups

Think of this as the "easy mode" - AWS handles most of the heavy lifting for you.

What AWS does for you:

- Automatically provisions and configures the worker nodes (EC2 instances)
- Handles updates and patches to the operating system
- Manages the node lifecycle (adding/removing nodes as needed)
- Integrates seamlessly with Auto Scaling Groups

- Provides built-in monitoring and logging

You just need to:

- Choose your instance types and sizes
- Set minimum/maximum number of nodes
- Configure basic networking settings

Self-Managed Nodes

This is the "DIY mode" - you have full control but more responsibility.

What you need to handle:

- Launch and configure EC2 instances yourself
- Install and configure the Kubernetes node agent (kubelet)
- Set up networking and security groups
- Handle OS updates and patches manually
- Configure Auto Scaling Groups if you want them
- Set up monitoring and logging

Benefits:

- Complete control over the node configuration
- Can use custom AMIs (machine images)
- More flexibility in networking and security setup
- Potentially lower costs with careful optimization

When to Use Which?

Choose Managed Node Groups if:

- You're new to Kubernetes/EKS
- You want AWS to handle maintenance automatically
- You prefer simplicity over customization
- You're okay with some limitations in configuration

Choose Self-Managed Nodes if:

- You need specific custom configurations
- You want to use your own AMIs
- You have complex networking requirements
- You have experienced DevOps team to manage the infrastructure

---

7. How do you configure kubectl to work with EKS?

Use the **AWS CLI** to update your kubeconfig file:

bash

```
aws eks --region ap-south-1 update-kubeconfig --name my-cluster
```

🔐 This:

- Adds EKS cluster info to ~/.kube/config
- Enables you to use kubectl to interact with EKS

✅ Example:

bash

```
kubectl get nodes
kubectl get pods
```

---

8. What are the different ways to access EKS clusters?

Access Method	Use Case
kubectl + AWS CLI	Day-to-day CLI access
IAM Authenticator	Controls who can access EKS
AWS Console	GUI to monitor/manage
Bastion Host + kubectl	For private clusters
kubectl via CI/CD	For automation (use IAM roles or kubeconfig)

✅ Best Practice:

- Use **IAM Roles for Service Accounts (IRSA)** for secure Pod-level access to AWS services
- Use **RBAC** to restrict cluster access

1. What does Amazon EKS manage for you?

A. Worker nodes only
B. Application deployment logic
C. Kubernetes control plane (API server, etcd, etc.)
D. VPC and Subnets

✅ **Correct Answer:** C. Kubernetes control plane (API server, etcd, etc.)

---

2. In EKS architecture, what is the role of worker nodes?

A. Store control plane configuration
B. Expose services to the internet
C. Run user workloads (pods)
D. Manage IAM authentication

✅ **Correct Answer:** C. Run user workloads (pods)

---

3. Which tool provides the simplest way to create an EKS cluster quickly?

A. AWS CloudShell
B. Terraform
C. eksctl
D. AWS CodeBuild

✅ **Correct Answer:** C. eksctl

---

4. What does aws eks update-kubeconfig do?

A. Installs kubectl
B. Creates the EKS cluster
C. Adds EKS cluster credentials to your kubeconfig
D. Starts all pods in the cluster

✅ **Correct Answer:** C. Adds EKS cluster credentials to your kubeconfig

---

5. What is the main difference between Managed Node Groups and Self-Managed Nodes in EKS?

A. Managed nodes don't support autoscaling
B. Self-managed nodes can only run on t2.micro
C. AWS handles lifecycle and upgrades for managed nodes
D. Managed nodes require manual provisioning
✅ **Correct Answer:** C. AWS handles lifecycle and upgrades for managed nodes

---

6. What does the EKS control plane include?
A. Load Balancers and VPC routing
B. IAM policies and secrets
C. API server, etcd, scheduler, controllers
D. Node auto-scaling engine
✅ **Correct Answer:** C. API server, etcd, scheduler, controllers

---

7. Which of the following is NOT a valid way to access an EKS cluster?
A. AWS Console
B. IAM Authenticator
C. SSH into the control plane
D. kubectl with kubeconfig
✅ **Correct Answer:** C. SSH into the control plane

---

8. What is the purpose of an EKS Node Group?
A. Runs the Kubernetes API server
B. Authenticates users via IAM
C. Groups EC2/Fargate instances to run workloads
D. Stores the cluster configuration
✅ **Correct Answer:** C. Groups EC2/Fargate instances to run workloads

---

9. What are the benefits of using Managed Node Groups? (Choose one)
A. Lower EC2 instance cost
B. No need for IAM roles
C. AWS handles node replacement and upgrades
D. Unlimited control over kubelet configuration
✅ **Correct Answer:** C. AWS handles node replacement and upgrades

---

10. Which of the following is a best practice for accessing AWS services securely from EKS pods?
A. Use a hardcoded AWS secret in the Pod
B. Use IAM Roles for Service Accounts (IRSA)
C. Use SSH tunneling to access services
D. Allow unrestricted IAM access for all pods
✅ **Correct Answer:** B. Use IAM Roles for Service Accounts (IRSA)