**What is kubeconfig in Kubernetes cluster**

* kubeconfig file is a configuration file used by the kubectl command-line tool to interact with a Kubernetes cluster.
* A **kubeconfig** file contains information about:
* **Clusters** – the Kubernetes API servers to connect to.
* **Users** – authentication information for accessing the clusters.
* **Contexts** – a combination of cluster, user, and namespace.
* **Current-context** – the default context kubectl uses when executing commands.
* **Default Location** of kubeconfig file is “~/.kube/config”
* **Use Cases**
* Access multiple Kubernetes clusters from a single machine.
* Switch between different namespaces or users easily.
* Authenticate securely using certificates, bearer tokens, or cloud provider plugins.

**How to connect private endpoint EKS Cluster**

To access a **private EKS cluster**, you typically have **three main options**:

1. **Use a Bastion Host (Jumpbox)**

You set up an **EC2 instance** in the same VPC/subnet as the EKS cluster, then:

* SSH into the EC2 instance (bastion host).
* Run kubectl from inside that host **OR** port-forward the API server to your local machine.

1. **Use AWS Systems Manager (SSM)**

If you don't want to expose SSH ports, you can:

* Enable SSM Agent on a private EC2 instance in the cluster’s VPC.
* Use Session Manager to connect (without SSH).
* Run kubectl from there.

1. **Use VPN or Direct Connect**

Set up a **VPN connection** (e.g., AWS Client VPN or site-to-site VPN) or **AWS Direct Connect** so your local machine is inside the **same network (VPC)** as the EKS API endpoint.

Once connected, your kubectl can access the private endpoint as if you're in the VPC.

**Setup using SSM (Systems Manager)**

* **Ensure EKS Cluster Has a Private Endpoint Enabled**

When creating the cluster (via console or eksctl), ensure:

* API Server Endpoint Access: **Private Only** or **Private and Public (disabled later)**

You can verify using:

aws eks describe-cluster --name my-cluster --query "cluster.endpoint"

* **Setup EC2 Instance in the Same VPC**
* Same VPC and subnet as EKS
* IAM Role with SSM permissions (e.g., AmazonSSMManagedInstanceCore)
* SSM Agent installed and running
* No public IP needed
* **Update Your Kubeconfig**

Run this from your local machine:

aws eks update-kubeconfig --name my-cluster --region <your-region>

It will configure your ~/.kube/config with the private endpoint.

* **Connect to EC2 via SSM**

Start a session:

aws ssm start-session --target <instance-id>

Now you're inside the private EC2 instance.

* **Run kubectl from Inside the EC2 Instance**

Install kubectl and awscli on the EC2, then update kubeconfig **inside it**:

aws eks update-kubeconfig --name my-cluster --region <your-region>

kubectl get nodes

It should work now.