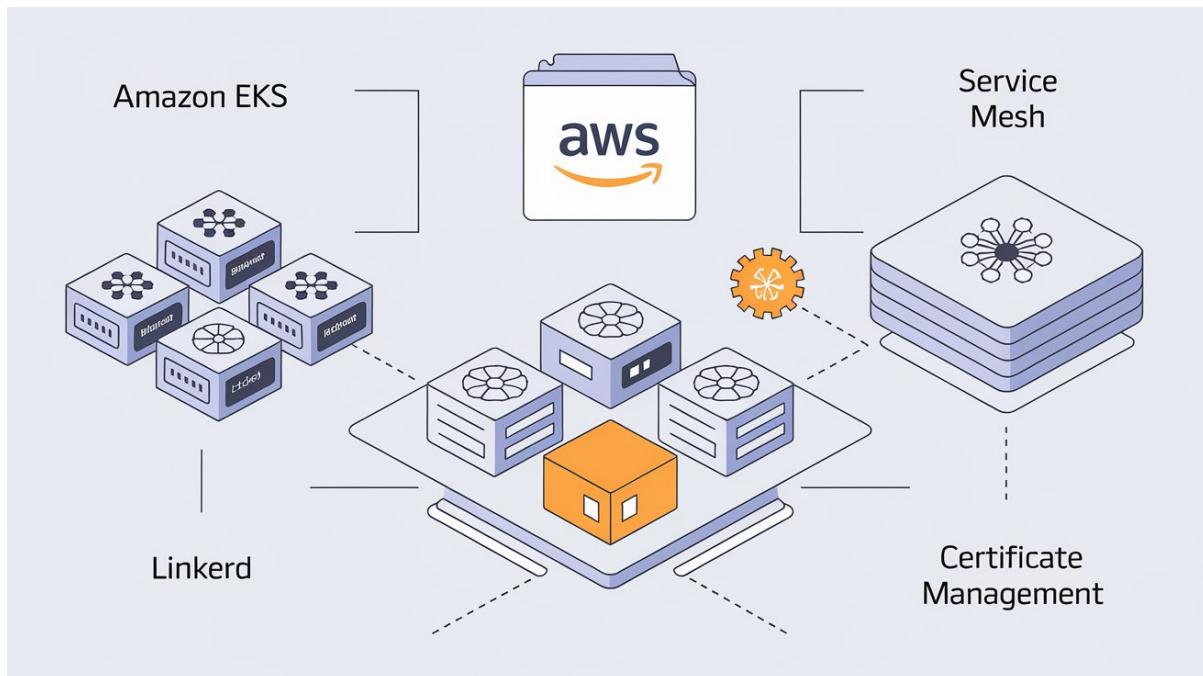




## Effortless Certificate Management for Linkerd Service Mesh on AWS EKS Using HCP Vault Dedicated



By

**Mahendran Selvakumar**

<https://devopstronaut.com/>



## What is Linkerd Service Mesh?

**Linkerd** is a lightweight, open-source service mesh specifically designed for Kubernetes environments. It focuses on providing **security, observability, and reliability** for cloud-native applications by managing service-to-service communication within a cluster. Developed as a CNCF (Cloud Native Computing Foundation) project, Linkerd is known for its simplicity, efficiency, and ease of use.

## Key Features of Linkerd

1. **Security:**
  - **mTLS (Mutual TLS):** Encrypts traffic between services automatically, ensuring secure communication without modifying application code.
  - Automatic certificate rotation and strong identity guarantees.
2. **Observability:**
  - Provides **golden metrics** (latency, success rate, and request volume) for services.
  - Tools for real-time inspection of service behavior, such as linkerd stat and linkerd tap.
  - Integration with observability tools like Prometheus, Grafana, and Jaeger for metrics and tracing.
3. **Reliability:**
  - Traffic policies for fine-grained control over communication between services.
  - Automatic retries and timeouts to handle transient failures.
  - Traffic splitting for blue-green or canary deployments.
4. **Lightweight:**
  - Minimal resource overhead compared to other service meshes like Istio.
  - Uses Rust-based proxies for efficiency and performance.
5. **Ease of Use:**
  - Simple installation and configuration process with a CLI (linkerd install).
  - Pre-configured defaults for Kubernetes clusters, making it beginner-friendly.

## What is HCP Vault Dedicated?

HCP Vault Dedicated is a managed service by HashiCorp that provides a private, secure instance of Vault in the cloud. It's designed to help organizations manage secrets and dynamic credentials without the complexity of self-hosting.

### Key Features:

- **Private Environment:** Ensures isolation and security
- **High Availability:** Guarantees uptime and reliability.



- **Simplified Operations:** HashiCorp handles maintenance and updates
- **Cloud Integration:** Works seamlessly with AWS, Azure, and other cloud platforms

HCP Vault Dedicated simplifies secret management, improves security, and reduces operational overhead, making it a reliable solution for cloud-based environments

## Step 1: Create the EKS Cluster Without Any Node Groups

Create an EKS cluster without a node group using the eksctl command **eksctl create cluster --name=eks-linkerd --region=eu-west-1 --without-nodegroup**. By default, eksctl creates a node group with m5.large instances, so we used the --without-nodegroup option to skip creating a default node group

```
mahendralselvakumar@Mahendrals-MBP ~ % eksctl create cluster --name=eks-linkerd --region=eu-west-1 --without-nodegroup
2024-12-31 14:04:56 [i] eksctl version 0.196.0
2024-12-31 14:04:56 [i] using region eu-west-1
2024-12-31 14:04:56 [i] setting availability zones to [eu-west-1a eu-west-1b eu-west-1c]
2024-12-31 14:04:56 [i] subnets for eu-west-1a - public:[192.168.0.0/19 private:192.168.96.0/19
2024-12-31 14:04:56 [i] subnets for eu-west-1b - public:[192.168.32.0/19 private:192.168.128.0/19
2024-12-31 14:04:56 [i] subnets for eu-west-1c - public:[192.168.64.0/19 private:192.168.160.0/19
2024-12-31 14:04:56 [i] using Kubernetes version 1.30
2024-12-31 14:04:56 [i] creating EKS cluster "eks-linkerd" in "eu-west-1" region with
2024-12-31 14:04:56 [i] If eksctl encounters any issues, check CloudFormation console or try "eksctl utils describe-stacks --region=eu-west-1 --cluster=eks-linkerd"
2024-12-31 14:04:56 [i] Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false} for cluster "eks-linkerd" in "eu-west-1"
2024-12-31 14:04:56 [i] CloudWatch logging will not be enabled for cluster "eks-linkerd" in "eu-west-1"
2024-12-31 14:04:56 [i] you can enable it with 'eksctl utils update-cluster-logging --enable-types=[SPECIFY-YOUR-LOG-TYPES-HERE (e.g. all)] --region=eu-west-1 --cluster=eks-linkerd'
2024-12-31 14:04:56 [i] default addons coredns, vpc-cni, kube-proxy were not specified, will install them as EKS addons
2024-12-31 14:04:56 [i]

2 sequential tasks: { create cluster control plane "eks-linkerd",
  2 tasks: { create addons },
    1 task: { create addons },
      wait for control plane to become ready,
  }
}

2024-12-31 14:04:56 [i] building cluster stack "eksctl-eks-linkerd-cluster"
2024-12-31 14:04:57 [i] deploying stack "eksctl-eks-linkerd-cluster"
2024-12-31 14:05:27 [i] waiting for CloudFormation stack "eksctl-eks-linkerd-cluster"
2024-12-31 14:05:57 [i] waiting for CloudFormation stack "eksctl-eks-linkerd-cluster"
2024-12-31 14:06:57 [i] waiting for CloudFormation stack "eksctl-eks-linkerd-cluster"
2024-12-31 14:07:58 [i] waiting for CloudFormation stack "eksctl-eks-linkerd-cluster"
2024-12-31 14:08:58 [i] waiting for CloudFormation stack "eksctl-eks-linkerd-cluster"
2024-12-31 14:09:59 [i] waiting for CloudFormation stack "eksctl-eks-linkerd-cluster"
2024-12-31 14:10:59 [i] waiting for CloudFormation stack "eksctl-eks-linkerd-cluster"
2024-12-31 14:40:24 [i] waiting for CloudFormation stack "eksctl-eks-linkerd-cluster"
2024-12-31 14:40:26 [i] creating addon
2024-12-31 14:40:26 [i] successfully created addon
2024-12-31 14:40:26 [i] recommended policies were found for "vpc-cni" addon, but since OIDC is disabled on the cluster, eksctl cannot configure the requested permissions; the recommended way to provide IAM permissions for vpc-cni addon is via pod identity associations; after addon creation is completed, add all recommended policies to the config file, under "addon.PodIdentityAssociations", and run eksctl update addon
2024-12-31 14:40:26 [i] creating addon
2024-12-31 14:40:27 [i] successfully created addon
2024-12-31 14:40:27 [i] creating addon
2024-12-31 14:40:27 [i] successfully created addon
2024-12-31 14:43:29 [i] waiting for the control plane to become ready
2024-12-31 14:43:29 [i] saved kubeconfig as "/Users/mahendralselvakumar/.kube/config"
2024-12-31 14:43:29 [i] no tasks
2024-12-31 14:43:29 [i] all EKS cluster resources for "eks-linkerd" have been created
2024-12-31 14:43:29 [i] kubectl command should work with "/Users/mahendralselvakumar/.kube/config", try 'kubectl get nodes'
2024-12-31 14:43:29 [i] EKS cluster "eks-linkerd" in "eu-west-1" region is ready
mahendralselvakumar@Mahendrals-MBP ~ %
```

Go to the EKS console to verify that the cluster was successfully created using eksctl

Clusters (1) <small>Info</small>							
<small>Filter clusters</small>							
Cluster name	Status	Kubernetes version	Support period	Upgrade policy	Created	Provider	
eks-linkerd	Active	1.30 <a href="#">Upgrade now</a>	<a href="#">Standard support until July 28, 2025</a>	Extended	<a href="#">an hour ago</a>	EKS	<a href="#">Delete</a> <a href="#">Create cluster</a>

## Step 2: Create a Managed Node Group

Add a node group using the following separate eksctl command **eksctl create nodegroup --name eks-linkerd-ng --cluster eks-linkerd --region eu-west-1 --nodes 2 --nodes-min 1 --nodes-max 3 --node-type t3.medium**

<https://www.linkedin.com/in/mahendran-selvakumar/>



```
mahendralselvakumar@Mahendrals-MBP ~ % eksctl create nodegroup --name eks-linkerd-ng --cluster eks-linkerd --region eu-west-1 --nodes 2 --nodes-min 1 --nodes-max 3 --node-type t3.medium
2024-12-31 14:47:45 [i] will use version 1.30 for new nodegroup(s) based on control plane version
2024-12-31 14:47:46 [i] nodegroup "eks-linkerd-ng" will use "" [Amazonlinux2/1.30]
2024-12-31 14:47:47 [i] 1 nodegroup (eks-linkerd-ng) was included (based on the include/exclude rules)
2024-12-31 14:47:47 [i] will create a CloudFormation stack for each of 1 managed nodegroups in cluster "eks-linkerd"
2024-12-31 14:47:47 [i]
2 sequential tasks: { fix cluster compatibility, 1 task: { 1 task: { create managed nodegroup "eks-linkerd-ng" } } }
2024-12-31 14:47:47 [i] checking cluster stack for missing resources
2024-12-31 14:47:47 [i] cluster stack has all required resources
2024-12-31 14:47:47 [i] building managed nodegroup stack "eksctl-eks-linkerd-nodegroup-eks-linkerd-ng"
2024-12-31 14:47:48 [i] deploying stack "eksctl-eks-linkerd-nodegroup-eks-linkerd-ng"
2024-12-31 14:47:48 [i] waiting for Cloudformation stack "eksctl-eks-linkerd-nodegroup-eks-linkerd-ng"
2024-12-31 14:48:18 [i] waiting for Cloudformation stack "eksctl-eks-linkerd-nodegroup-eks-linkerd-ng"
2024-12-31 14:48:48 [i] waiting for Cloudformation stack "eksctl-eks-linkerd-nodegroup-eks-linkerd-ng"
2024-12-31 14:50:05 [i] waiting for Cloudformation stack "eksctl-eks-linkerd-nodegroup-eks-linkerd-ng"
2024-12-31 14:50:05 [i] no tasks
2024-12-31 14:50:05 [v] created 0 nodegroup(s) in cluster "eks-linkerd"
2024-12-31 14:50:05 [i] nodegroup "eks-linkerd-ng" has 2 node(s)
2024-12-31 14:50:05 [i] node "ip-192-168-17-94.eu-west-1.compute.internal" is ready
2024-12-31 14:50:05 [i] node "ip-192-168-81-86.eu-west-1.compute.internal" is ready
2024-12-31 14:50:05 [i] waiting for at least 1 node(s) to become ready in "eks-linkerd-ng"
2024-12-31 14:50:05 [i] nodegroup "eks-linkerd-ng" has 2 node(s)
2024-12-31 14:50:05 [i] node "ip-192-168-17-94.eu-west-1.compute.internal" is ready
2024-12-31 14:50:05 [i] node "ip-192-168-81-86.eu-west-1.compute.internal" is ready
2024-12-31 14:50:05 [v] created 1 managed nodegroup(s) in cluster "eks-linkerd"
2024-12-31 14:50:05 [i] checking security group configuration for all nodegroups
2024-12-31 14:50:05 [i] all nodegroups have up-to-date Cloudformation templates
mahendralselvakumar@Mahendrals-MBP ~ %
```

## Explanation of the flags:

- **--cluster:** Specifies the name of the existing EKS cluster to which the node group will be added.
- **--name:** Names the node group for easy identification.
- **--region:** Specifies the AWS region.
- **--nodes:** Sets the initial desired number of nodes (in this case, 2).
- **--nodes-min and — nodes-max:** Define the minimum and maximum number of nodes for auto-scaling.
- **--node-type:** The EC2 instance type for the nodes (e.g., m5.large)

If you check in the console, you'll see that the node group has been created

**eks-linkerd-ng**

Your current IAM principal doesn't have access to Kubernetes objects on this cluster.  
This may be due to the current user or role not having Kubernetes RBAC permissions to describe cluster resources or not having an entry in the cluster's auth config map. [Learn more](#)

<b>Node group configuration</b> <a href="#">Info</a>	<b>AMI type</b> <a href="#">Info</a> AL2_x86_64	<b>Launch template</b> <a href="#">eksctl-eks-linkerd-nodegroup-eks-linkerd-ng</a>	<b>Status</b> <span>Active</span>
<b>Kubernetes version</b> 1.30	<b>AMI release version</b> <a href="#">Info</a> 1.30.7-20241225	<b>Instance types</b> t3.medium	<b>Launch template version</b> 1
<b>Disk size</b> Specified in launch template			

[Details](#) [Nodes](#) [Health issues](#) [Kubernetes labels](#) [Update config](#) [Kubernetes taints](#) [Update history](#) [Tags](#)

**Details**

<b>Node group ARN</b> <a href="#">arn:aws:eks:eu-west-1:038462791702:nodegroup/eks-linkerd-eks-linkerd-ng/e4ca0e98-1fb1-838a-c1a0-8cc07d96</a>	<b>Autoscaling group name</b> <a href="#">eks-eks-linkerd-ng-e4ca0e98-1fb1-838a-c1a0-2ce58cc07d96</a>	<b>Capacity type</b> On-Demand	<b>Subnets</b> <a href="#">subnet-057ce1247d4dec213</a> <a href="#">subnet-03ec1c23472112d14</a> <a href="#">subnet-0c14f9fe849188ed2</a>
<b>Created</b> <a href="#">5 minutes ago</a>	<b>Node IAM role ARN</b> <a href="#">arn:aws:iam::038462791702:role/eksctl-eks-linkerd-nodegroup-eks-l-NodeInstanceRole-gLcVFFF0mqj0</a>	<b>Desired size</b> 2 nodes	<b>Configure remote access to nodes</b> off
	<a href="#">View in IAM</a>	<b>Minimum size</b> 1 node	
		<b>Maximum size</b> 3 nodes	

**Node auto repair configuration** [Info](#)

**Node auto repair**  
Disabled



In the instances section of the EC2 console, you can view the Instances created for the node group

Instances (2) <a href="#">Info</a>		Last updated less than a minute ago							<a href="#">Connect</a>	<a href="#">Instance state</a>	<a href="#">Actions</a>	<a href="#">Launch instances</a>	
<a href="#">Find Instance by attribute or tag (case-sensitive)</a>		<a href="#">All states</a>											
<a href="#">Instance state = running</a> <a href="#">X</a> <a href="#">Clear filters</a>												< 1 > <a href="#">@</a>	
Name		Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS					
eks-linkerd-eks-linkerd-ng-Node		i-0f4fd3a6fb9260c5	Running	t3.medium	2/3 checks passed	<a href="#">View alarms +</a>	eu-west-1c	ec2-34-253-199-99.eu-west-1.					
eks-linkerd-eks-linkerd-ng-Node		i-01f72c68130b99599	Running	t3.medium	2/3 checks passed	<a href="#">View alarms +</a>	eu-west-1b	ec2-34-245-122-60.eu-west-1.					

## Step 3: Configure Context for EKS Cluster

Set the Kubernetes context for the EKS cluster using the following command: **aws eks --region eu-west-1 update-kubeconfig --name eks-linkerd**

```
mahendralselvakumar@Mahendrals-MBP ~ % aws eks --region eu-west-1 update-kubeconfig --name eks-linkerd
Updated context arn:aws:eks:eu-west-1:1038462791702:cluster/eks-linkerd in /Users/mahendralselvakumar/.kube/config
mahendralselvakumar@Mahendrals-MBP ~ %
```

Use **kubectl get ns** to view namespaces and **kubectl get nodes** to check the status of the nodes in the cluster, verifying that the setup is complete

```
mahendralselvakumar@Mahendrals-MBP ~ % kubectl get ns
NAME      STATUS   AGE
default    Active   41m
kube-node-lease  Active   41m
kube-public   Active   41m
kube-system   Active   41m
mahendralselvakumar@Mahendrals-MBP ~ % kubectl get nodes
NAME                           STATUS   ROLES   AGE     VERSION
ip-192-168-17-94.eu-west-1.compute.internal  Ready   <none>  2m6s  v1.30.7-eks-59bf375
ip-192-168-81-86.eu-west-1.compute.internal  Ready   <none>  2m4s  v1.30.7-eks-59bf375
mahendralselvakumar@Mahendrals-MBP ~ %
```

## Step 4: Create the HCP Vault Dedicated

Click **Vault Dedicated**

The screenshot shows the HashiCorp Devopstronaut project dashboard. On the left, a sidebar lists various services: Boundary, Consul, Packer, Terraform, Vagrant Registry, Vault Dedicated (which is currently selected), Vault Radar (Beta), Vault Secrets, and Waypoint. The main dashboard area displays the 'Devopstronaut' project status, which is active. It includes sections for 'Active resources' (2 active resources) and 'Billing summary'. A link at the bottom right points to 'View organization billing summary'.

<https://www.linkedin.com/in/mahendran-selvakumar/>



Click **Create Cluster** to create a cluster from scratch

Devopstronaut / Devopstronaut / Vault Dedicated

## Vault overview

**Set up your cluster**

Quickly deploy a Vault cluster using a pre-configured template or provision and set up a cluster manually with preferred settings. Template comes ready with sample secrets, policies and auth methods.

**Encryption**

Encrypt data from applications while still storing that encrypted data in some primary data store.

**Key-value secrets** Popular

Centrally store, access, and deploy secrets across applications, systems, and infrastructure.

**PKI**

Use Vault to quickly create X.509 certificates on demand and reduce the manual overhead.

OR

**Start from scratch**

Create a brand-new cluster and configure it to your needs to make your applications more secure. A fully managed cluster so you can focus on building.

Create cluster →

Choose **Amazon Web Services** as the provider and select the Vault tier based on your requirements

Devopstronaut / Devopstronaut / Vault Dedicated / Create cluster

## Create cluster

### Cluster configuration

**Provider** Not editable after creation

Select which cloud provider the cluster should be deployed to. Clusters will be able to easily communicate with applications deployed to the same cloud provider across private networks.

Amazon Web Services

Microsoft Azure

**Vault tier** [View full pricing](#)

**Development** Single node clusters for non-prod workflows | 24/7 Monitoring | 25 clients

**Cluster size**

**Extra Small** 2 vCPU | 1 GiB RAM **\$0.03/hr**

Enter the network ID and select the region



Network Not editable after creation

Your HashiCorp Virtual Network (HVN) will be utilized by your cluster. This will associate the cloud provider and region from the HashiCorp Virtual Network for your cluster's deployment.



#### Network ID

Must be a unique set of 3-36 characters. May include numbers, hyphens, and lowercase letters. Must start with a letter and end with a letter or number.

#### Region selection

Select the AWS region where you want to deploy your cluster. This must match the region where your application services are deployed.

### Network Configuration

The CIDR block value enables communication between your HVN and VPCs. It is an important step in the deployment process and the value cannot be changed.

If you plan to set up performance replication, the primary and secondary clusters must be created on HVNs with CIDR blocks that do not overlap. This ensures that a peering connection can be made between the two HVNs.

#### CIDR block

If you do not provide a value, we will use the default value shown to create your new network.

Show recommendations

Enter the cluster ID, choose the template, and click **Create Cluster**

### Basics

Cluster ID Not editable after creation

Must be a unique set of 3-36 characters. May include numbers, hyphens, and lowercase letters. Must start with a letter and end with a letter or number.

Templates Not editable after creation

Start from scratch or choose one of the templates below to set up your cluster with sample data to get started quickly.

#### Start from scratch

Create a brand new cluster and configure it to your needs to make your applications more secure.



#### Encryption

New

Template comes ready with a sample encryption key, policy, and auth method.



#### Key-value secrets

Template comes ready with a sample secret, policy, and auth method.



#### PKI

New

Template comes ready with a sample root certificate, policy, and auth method.



**Create cluster**

**Cancel**

Pay as you go!  
Cluster cost is based on hours used, billed at the end of each month.  
This estimate does not include client fees.

Hourly  
**\$0.03/hour**

Starting monthly cost  
**\$21.60/month**

The Vault is now being created, and it may take some time to complete the process



Devopstronaut / Devopstronaut / Vault Dedicated / vault-cluster / Overview



## vault-cluster

### Cluster is being initialized

We expect the process to take between 5-10 minutes. We will notify you when the process is complete. While waiting, you can explore the learning materials below.

Generating configuration

Creating cluster

Applying settings / bootstrapping cluster

Validating deployment

The Vault cluster has been successfully created, and you can view the cluster details, such as the version. Click **Command-line (CLI)** to view the Vault cluster and namespace details

Devopstronaut / Devopstronaut / Vault Dedicated / vault-cluster / Overview



## vault-cluster

[Launch web UI](#)

Manage ▾

This cluster's network configuration is set to public. Configure an IP Allowlist in the cluster's networking settings to limit network access to the cluster's public endpoint.

### Cluster Details

Status	<span>Running</span>
Vault Version	v1.18.1 <a href="#">🔗</a>
Version upgrades	Automatic
Cluster ID	vault-cluster
Cluster Tier	Development
Cluster Size	Extra Small
High Availability	No
Created	6 minutes ago

### Quick actions

How to access via  
[Command-line \(CLI\)](#) [API](#)

New admin token [🔗](#)  
[Generate token](#)

Read a secret  
[Tutorial](#) [🔗](#)

### Cluster URLs

Copy the address into your CLI or browser to access the cluster.

[Private](#)

[Public](#)

### In case of emergency

Vault data can be locked if an intrusion is detected.

[API Lock](#)

### Cluster networking

HVN	<a href="#">🔗</a> hvn
Provider/region	<a href="#">aws</a> AWS (us-west-2)
Cluster accessibility	<a href="#">🔗</a> Public
IP allowlist	Disabled

### Revoke all admin tokens

Admin tokens can be revoked before their expiration.

[Revoke](#)

### Learn more

Explore the features of Vault and learn best practices with the following tutorials and documentation.

#### Learn about Vault

[Introduction to Vault Dedicated](#) [🔗](#)

[What is a client?](#) [🔗](#)

#### Advanced topics

[Deploy Vault Dedicated with Terraform](#) [🔗](#)

[Codify management of Vault Dedicated](#) [🔗](#)



Copy the **Public URL** and **Namespace** to access the Vault cluster

The screenshot shows the HashiCorp Vault UI. On the left, there's a sidebar with 'Details' and 'Networking' sections. The 'Details' section includes fields like 'Status' (Running), 'Version' (v1.18.1), 'Deployment Type' (Automatic), 'Size' (Extra Small), 'Ability' (No), and 'Created' (1 hour ago). The 'Networking' section shows 'IP' (hvn), 'Region' (AWS (us-west-2)), and 'Security' (Public). At the bottom of the sidebar are 'Close', 'Learn about Vault', and 'Advanced topics' buttons. A central modal window titled 'Connect to your Vault cluster' contains instructions: 'If you haven't already, download [Vault](#) and install it locally to use the CLI.' Below this is a section titled 'Telling Vault where to find this cluster' with the sub-section 'Authenticating to Vault'. It provides the Public URL and Namespace for the vault-cluster instance: `export VAULT_ADDR="https://vault-cluster-public-vault-2b016eba.6effa500.z1.hashicorp.cloud:8200"; export VAULT_NAMESPACE="admin"`. There are also buttons for 'Use public URL' and 'Use private URL'. At the bottom of the modal is a 'Close' button.

## Set Environment Variable

Run this command `export VAULT_ADDR=https://vault-cluster-public-vault-2b016eba.6effa500.z1.hashicorp.cloud:8200` to export Vault Address

```
mahendranelvakumar@Mahendrans-MBP ~ % export VAULT_ADDR="https://vault-cluster-public-vault-2b016eba.6effa500.z1.hashicorp.cloud:8200"  
mahendranelvakumar@Mahendrans-MBP ~ %
```

Run this command `export VAULT_NAMESPACE="admin"` to export namespace

```
mahendranelvakumar@Mahendrans-MBP ~ % export VAULT_NAMESPACE="admin"  
mahendranelvakumar@Mahendrans-MBP ~ %
```

Run this command `export`

`VAULT_TOKEN=hvs.CAESIKYcXNkvRo7LQF2WUmdK1SshhcVPpqWZmXCPTJf_wxqI  
GicKImh2cy5PY1BsT1laVGhaTWp5WldUdW91WjVkJWFcuMHZpUmMQ0yM` to set the root token

```
mahendranelvakumar@Mahendrans-MBP ~ % export VAULT_TOKEN=hvs.CAESIKYcXNkvRo7LQF2WUmdK1SshhcVPpqWZmXCPTJf_wxqIGicKImh2cy5PY1BsT1laVGhaTWp5WldUdW91WjVkJWFcuMHZpUmMQ0yM  
mahendranelvakumar@Mahendrans-MBP ~ %
```

Run the **vault status** command to view the Vault's status. You can see details such as the seal type, initialization and unsealing status (initialized and unsealed automatically), Vault version (Enterprise), and High Availability (HA) mode (active)



```
mahendralselvakumar@Mahendrals-MBP ~ % vault status
Key          Value
---          -----
Seal Type    awsksms
Recovery Seal Type shamir
Initialized   true
Sealed       false
Total Recovery Shares 1
Threshold    1
Version      1.18.1+ent
Build Date   2024-10-29T14:21:43Z
Storage Type raft
Cluster Name 72047090-1af0-4f11-b6ce-7ccad412bef1
Cluster ID   4d322c8f-5d5b-8edf-352d-754e0d9cf158
HA Enabled   true
HA Cluster   https://172.25.16.49:8201
HA Mode      active
Active Since 2024-12-31T13:59:40.802758651Z
Raft Committed Index 11435
Raft Applied Index 11435
Last WAL     4648
mahendralselvakumar@Mahendrals-MBP ~ %
```

## Step 5: Configure Vault for Certificate Issuance

Enable the PKI secrets engine for issuing TLS certificates using this command **Vault secrets enable pki**

```
mahendralselvakumar@Mahendrals-MBP ~ % vault secrets enable pki
```

```
Success! Enabled the pki secrets engine at: pki/
mahendralselvakumar@Mahendrals-MBP ~ %
```

Run the following command to generate a root PKI certificate with a common name of **\*.cluster.local** and a validity of 8760 hours (1 year):

```
vault write pki/root/generate/internal common_name="*.cluster.local" ttl=8760h
```



```
mahendranelvakumar@Mahendrans-MBP Downloads % vault write pki/root/generate/internal \
    common_name="*.cluster.local" \
    ttl=8760h

WARNING! The following warnings were returned from Vault:

* This mount hasn't configured any authority information access (AIA)
  fields; this may make it harder for systems to find missing certificates
  in the chain or to validate revocation status of certificates. Consider
  updating /config/urls or the newly generated issuer with this information.

Key          Value
---          ---
certificate  -----BEGIN CERTIFICATE-----
MIIDQTCCAimgAwIBAgIUJPzHrwe1IJyReqWBSRt5dVVLE7cwDQYJKoZIhvcNAQEL
BQAwGjEYMBYGA1UEAwwPKi5jbHVzdGVyLmxvY2FsMB4XDTI0MTIxNTgxN1oX
DTI1MDIwMTIxNTg0N1owGjEYMBYGA1UEAwwPKi5jbHVzdGVyLmxvY2FsMIIBIjAN
BgkqhkiG9w0BAQEFAAOCAQ8AM1IBcGKCAQEayb1z0sQ4ydZfpJFPgTTi15yRMI
zQiQwDLHRL32Xd50E0t1DgLRqWb/kQ3jwLFZOTE/bhqp3fwWryVBN3PFqS4I03ug
n9eHn3GEKS6IckkcF2oBdVugbL7nfSjG/HsgGIMQwa7xe8pAwi.Zqnbc1t/Z1Cy9G
90QJV8qoUSqj1ZHHSzMj5WMBMTwNJIydc5Kyvkm0zoyVFKUQhw42BF6jZBwynGv
ndlfihBaMeP2GwDIKitpowwdZXEMl7jzVhurPsDz1REqTYeAT8rz0GRz+yHH1tw
CXzTR2yDU8tudIRNk+AMVQnCKDNijeZypw9u+cby5tED6M8F+Qi jIpNZUQIDAQAB
o38wfTAOBgNVHQ8BAf8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQU
ceFj3mZH65kZ2vmqLG4hKKKQ8LgwHwYDVR0jBBgwFoAUceFj3mZH65kZ2vmqLG4h
KKKQ8LgwGgYDVR0RBbmWEYIPK15jbHVzdGVyLmxvY2FsMA0GCSqGSIb3DQEBCwUA
A4IBAQAF9Bj7NxrPmAQngYU3P3SxjFRGG6nv1BruXeeTOFChi/VFVyN6YKHKpp8
EIqUXIJDr6fofvIBPYLT1pkBCDLP5eU1+WdsJgOYjvtGQ3qXjstJykwoNQdrR75I
qqdQ/yyHvyZg/bj9RqgcccIGEaTAJuPH+QrzNWqmFVU4j1odyXjQQQhevrv4kz2YN
ho1A3oX4jh9JQhDu/kTfK/bEwKjfksYbby/VNsDTt07ph9wDA3r2r3nRjDTfCIu1
WBgsedTFgRhAYMjH/5aAP3L0MaXCeuW0kmXMpmc3Ztf5fy/81baVYHwMVWzi63V
Xt0ke7n0Cdl3fQQU5xTsugleXgb
-----END CERTIFICATE-----
expiration    1738447127
issuer_id     39bc7375-58fa-31a5-48bf-d81bcaf2edc7
issuer_name   n/a
issuing_ca    -----BEGIN CERTIFICATE-----
MIIDQTCCAimgAwIBAgIUJPzHrwe1IJyReqWBSRt5dVVLE7cwDQYJKoZIhvcNAQEL
BQAwGjEYMBYGA1UEAwwPKi5jbHVzdGVyLmxvY2FsMB4XDTI0MTIxNTgxN1oX
DTI1MDIwMTIxNTg0N1owGjEYMBYGA1UEAwwPKi5jbHVzdGVyLmxvY2FsMIIBIjAN
BgkqhkiG9w0BAQEFAAOCAQ8AM1IBcGKCAQEayb1z0sQ4ydZfpJFPgTTi15yRMI
zQiQwDLHRL32Xd50E0t1DgLRqWb/kQ3jwLFZOTE/bhqp3fwWryVBN3PFqS4I03ug
n9eHn3GEKS6IckkcF2oBdVugbL7nfSjG/HsgGIMQwa7xe8pAwi.Zqnbc1t/Z1Cy9G
90QJV8qoUSqj1ZHHSzMj5WMBMTwNJIydc5Kyvkm0zoyVFKUQhw42BF6jZBwynGv
ndlfihBaMeP2GwDIKitpowwdZXEMl7jzVhurPsDz1REqTYeAT8rz0GRz+yHH1tw
CXzTR2yDU8tudIRNk+AMVQnCKDNijeZypw9u+cby5tED6M8F+Qi jIpNZUQIDAQAB
o38wfTAOBgNVHQ8BAf8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQU
ceFj3mZH65kZ2vmqLG4hKKKQ8LgwHwYDVR0jBBgwFoAUceFj3mZH65kZ2vmqLG4h
KKKQ8LgwGgYDVR0RBbmWEYIPK15jbHVzdGVyLmxvY2FsMA0GCSqGSIb3DQEBCwUA
A4IBAQAF9Bj7NxrPmAQngYU3P3SxjFRGG6nv1BruXeeTOFChi/VFVyN6YKHKpp8
EIqUXIJDr6fofvIBPYLT1pkBCDLP5eU1+WdsJgOYjvtGQ3qXjstJykwoNQdrR75I
qqdQ/yyHvyZg/bj9RqgcccIGEaTAJuPH+QrzNWqmFVU4j1odyXjQQQhevrv4kz2YN
ho1A3oX4jh9JQhDu/kTfK/bEwKjfksYbby/VNsDTt07ph9wDA3r2r3nRjDTfCIu1
WBgsedTFgRhAYMjH/5aAP3L0MaXCeuW0kmXMpmc3Ztf5fy/81baVYHwMVWzi63V
Xt0ke7n0Cdl3fQQU5xTsugleXgb
-----END CERTIFICATE-----
key_id        21fe4d94-cd68-6209-2f46-5828d15de54f
key_name      n/a
serial_number 24:fc:c7:af:07:b5:20:9c:91:7a:a5:81:49:1b:79:75:55:4b:13:b7
mahendranelvakumar@Mahendrans-MBP Downloads %
```

Set up URLs for issuing certificates and distributing the Certificate Revocation List (CRL) using this command

<https://www.linkedin.com/in/mahendran-selvakumar/>



```
vault write pki/config/urls issuing_certificates="$VAULT_ADDR/v1/pki/ca"
crl_distribution_points="$VAULT_ADDR/v1/pki/crl"
```

```
mahendralselvakumar@Mahendrans-MBP Downloads % vault write pki/config/urls \
    issuing_certificates="$VAULT_ADDR/v1/pki/ca" \
    crl_distribution_points="$VAULT_ADDR/v1/pki/crl"
Key          Value
---          -----
crl_distribution_points  [https://vault-cluster-public-vault-b94a7a79.1f214b29.z1.hashicorp.cloud:8200/v1/pki/crl]
enable_templating        false
issuing_certificates     [https://vault-cluster-public-vault-b94a7a79.1f214b29.z1.hashicorp.cloud:8200/v1/pki/ca]
oscp_servers             []
mahendralselvakumar@Mahendrans-MBP Downloads %
```

Create a role for Linkerd using this command **vault write pki/roles/linkerd**  
**allowed\_domains="identity.linkerd.cluster.local" allow\_subdomains=true**  
**allow\_any\_name=true max\_ttl="72h"**

```
mahendralselvakumar@Mahendrans-MBP Downloads % vault write pki/roles/linkerd \
    allowed_domains="identity.linkerd.cluster.local" \
    allow_subdomains=true \
    allow_any_name=true \
    max_ttl="72h"
Key          Value
---          -----
allow_any_name        true
allow_bare_domains   false
allow_glob_domains   false
allow_ip_sans         true
allow_localhost       true
allow_subdomains      true
allow_token_displayname  false
allow_wildcard_certificates  true
allowed_domains       [identity.linkerd.cluster.local]
allowed_domains_template  false
allowed_other_sans    []
allowed_serial_numbers  []
allowed_uri_sans      []
allowed_uri_sans_template  false
allowed_user_ids      []
basic_constraints_valid_for_non_ca  false
client_flag           true
cn_validations        [email hostname]
code_signing_flag     false
country               []
email_protection_flag  false
enforce_hostnames    true
ext_key_usage         []
ext_key_usage_oids    []
generate_lease        false
issuer_ref            default
key_bits              2048
key_type              rsa
key_usage             [DigitalSignature KeyAgreement KeyEncipherment]
locality              []
max_ttl               72h
no_store              false
no_store_metadata     false
not_after              n/a
not_before_duration   30s
organization           []
ou                    []
policy_identifiers    []
postal_code            []
province              []
require_cn             true
server_flag            true
signature_bits         256
street_address         []
ttl                   0s
use_csr_common_name   true
use_csr_sans           true
use_pss                false
mahendralselvakumar@Mahendrans-MBP Downloads %
```



Request a certificate for Linkerd's identity service using this command **vault write pki/issue/linkerd common\_name="identity.linkerd.cluster.local" ttl="72h"**

-----  
mahendransevakumar@Mahendrans-MBP Downloads % vault write pki/issue/linkerd  
common\_name="identity.linkerd.cluster.local" \  
ttl="72h"  
Key Value  
---  
ca\_chain [-----BEGIN CERTIFICATE-----  
MIIDQTCCAimgAwIBAgIUJPzHrwe1IjYreqWBSRt5dVVLE7cwDQYJKoZIhvcNAQEL  
BQAwGjEYMBYGA1UEAwwPKi5jbhvzdGvylmxvY2FsMB4XDTI0MTIzMTIxNTgxN1oX  
DTI1MD1wMTIxNg0N1owGjEYMBYGA1UEAwwPKi5jbhvzdGvylmxvY2FsMIIBiJAN  
BgkqhkiGw0BAQEFAAOCAQ8AMTIBCgKCAQEayb1lZ0sQ4ydzFpJFPgTFTi15yRMI  
zQiQwDLHLRl32Xd50E0t1DgLrQwB/k3jwlfZF0TE/bhqp3fwWryVBN3PFqS4I03ug  
n9eHn3GEKS6Ickkcf2oBdvugbl7nfSjG/HsgGIMQwa7xe8pAwizQnbclt/Z1Cy9G  
900Vq8uoSjz1ZHMSzMJ5WMBlmtwNJ1yc5K5yvkm0zoyVFKUohw42BF6jzbwymGv  
ndfihBaMeP2GwD1KtpowdZXMEl7jzVhurPsD21REqTYteAT8rz0GRz+yH1ltw  
CxZx7ryDy8tuDtrNk+AMVnCkDnijZeyw9u+cby5tED6M8F+QijIpNzUQIDAQAB  
o38wftAOBgNVHQ8BAf8EBCAMQyDwDVROTAQH/BAUwAeB/zAdBgNVHQ4EfGqU  
ceFj3mZH65kZ2vmlqLG4hKKKQ8LgwHwYDVR0jBBgwFoAUceFj3mZH65kZ2vmlqLG4h  
KKKQ8LgwGgYDVR0RBBlmWEYIPKi5jbhvzdGvylmxvY2FsMA0GCSqGSIb3DQEBCwUA  
A4IBAQF9B7J7NrPmuAqngYU3P3SxjFRGG6nv1lBrueXeTOFChi/VFVyN6YKHkp8  
EIqUXIJDr6f0fViBPyL1TpBkCDLP5eUl+WdsJgOYjtGQ3qXjsTjykwoNQdr75I  
qqd0/yHvYzB/bj9RqggcIGEaTAJuPh+QrzNWqmFVU4j1odyXjQQQghver4z2YN  
ho1A3o4Xjh9J0hDu/kTfk/BewkjfksYbvy/VnsDt07ph9wDA3r2r3nRjDTCfCu1  
WBgsedTfgRhaYMjH/5aAP3L0MaXceuW0kmXmpmc3Ztf5fy/8lbaVYHwMVWzi63V  
Xt0ke7n0Cd13fQQU5xtsugle1Xgb  
-----END CERTIFICATE-----]  
certificate [-----BEGIN CERTIFICATE-----  
MIIEtZCCAzegAwIBAgIUPlHmGwCaHvUWRlRlmGP9v3z20gwDQYJKoZIhvcNAQEL  
BQAwGjEYMBYGA1UEAwwPKi5jbhvzdGvylmxvY2FsMB4XDTI0MTIzMTIxMDYxOfOx  
DTI1MDewMzIyMDY0OfowKTEEnMCUGA1UEAxMeaWRlbnRpdkubGlua2VyzC5jbhvz  
dGvylmxvY2FsMIIBiJANBgkqhkiGw0BAQEFAAOCAQ8AMTIBCgKCAQEAv3C9L06d  
/iknwhxT7YjtTCSNzs0VWZHY2yhjW1+c3kL1m+CYen3UPTbY9wegxeuK6r1  
MySjC24cKEUTr1+huyxm0sDwffXG/Y1zRn+jjkaer7xPqphVjP5grU04aiu0X  
1cbpY5DJPsvksjzEfwQ0xLhh6dpKjtKst01AWVsJh0Exnhq3gr5LXDII/1y  
0r6q/lutshC3t8T5WQ+xgLmj06E3qgbzrW1rj10y2aq1EYztzMDWvEGjw6Auim  
ckLKuPSkDmYnFbwEZLehRGqf3K4VHUaYVvDn0uDwHAKgjqvagwnlB80WPkunajzx  
F9avejmj6sfqwIDQABo4IBFDCAXqwgDgYDVR0PAQH/BAQDAg0oMBA1udJQW  
MBQGCCsaQUFwMBBggRbgEPBCQdaJbdGvhNQH04EfGqUNrvcpENu+H1WL2gb7tP  
jtG1PqQwHwYDVR0jBggwFoAUceFj3mZH65kZ2vmlqLG4hKKKQ8LgwgcYIKwYBQQUH  
AQEEZjBkMGIGCCsGAQUFwBzACh1ZodHrwczolZ3hdWx0LNwsdXN0ZXItchvibglj  
Lxzhdxw0LWI5NGE3Ytc5ljFmMje0Yj15LnoxLmhcc2hpY29yc5jb91Zd04MjAw  
L3YxL3BraS9jYTApBgvNvREIjAgghSpZGVudG10e5saW5rZXjkLmNsdxN0ZXiu  
bg9jYWWwaAYDVR0FBGEwXzBdfugWYzxahr0cHMGLy92YXVsdc1jbhvzdGvylxb1  
YmxpYy1YXvsdc1i0trHNZE3054xzj1xNg1y0556MS5y0XNqaWVvncAuY2xvdwQ6  
0DIwCM92Ms9wa2kvY3jsMA0GCSqGSIb3DQEBCwUAA4IBAQcxnb2MXds0Q4Wx96rc  
c70ZAU1BpnftXTF4p43UbTxp2oshu+ZsrMuD+PfmFc669+Y2Tgvyen4qYtz0278t  
v37urbYk17lqsD1zd5geQPWeTTjMEzK414/+0xtc3lexYwiktByb6jcyfR7uox  
6f0eCITYD1Rvr5Edu0B8D52hqd9CzFBZqxyyR92LB+40XW8R4ty9+kz7A2ocqqP  
KICDIihCiuc3cdhHquYibYri0S1TyFmpjcpqCl2/Lqceurlc/EFlDM04F04LdtYsN  
QV8LLoQpWU34pZUTx0/KLGxgsNWWe62pJ90D1cvA8Pdz/cn4P0cZ4Y+B475me7Zh  
db4j  
-----END CERTIFICATE-----]  
expiration 1735942008  
issuing\_ca [-----BEGIN CERTIFICATE-----  
MIIDQTCCAimgAwIBAgIUJPzHrwe1IjYreqWBSRt5dVVLE7cwDQYJKoZIhvcNAQEL  
BQAwGjEYMBYGA1UEAwwPKi5jbhvzdGvylmxvY2FsMB4XDTI0MTIzMTIxNTgxN1oX  
DTI1MD1wMTIxNg0N1owGjEYMBYGA1UEAwwPKi5jbhvzdGvylmxvY2FsMIIBiJAN  
BgkqhkiGw0BAQEFAAOCAQ8AMTIBCgKCAQEayb1lZ0sQ4ydzFpJFPgTFTi15yRMI  
zQiQwDLHLRl32Xd50E0t1DgLrQwB/k3jwlfZF0TE/bhqp3fwWryVBN3PFqS4I03ug  
n9eHn3GEKS6Ickkcf2oBdvugbl7nfSjG/HsgGIMQwa7xe8pAwizQnbclt/Z1Cy9G

<https://www.linkedin.com/in/mahendran-selyakumar/>



Generate the **linkerd-issuer.pem** and **linkerd-cert.pem** files using the certificate and key details provided in the above output

```
└─── linkerd-cert.pem  
└─── linkerd-issuer.pem  
└─── private-key.pem  
└─── issuing_ca.pem  
└─── cert.pem  
└─── ca_chain.pem
```

Combine certificates(cert.pem and ca\_chain.pem) into a single PEM file(linkerd-issuer.pem)



```
mahendranelvakumar@Mahendrans-MBP Downloads % cat linkerd-issuer.pem
```

```
-----BEGIN CERTIFICATE-----
```

```
MIIETzCCAzeAwIBAgIUaIP1HmGWCaHVuWRILmGP9v3z20gwDQYJKoZIhvcNAQEL  
BQAwGjEYMBYGA1UEAwwPKi5jbHVzdGVyLmxvY2FsMB4XDTI0MTIzMThMDYx0FoX  
DTI1MDEwMzIyMDY00FowKTEnMCUGA1UEAxMeaWRlbnRpdkubGlua2VyzC5jbHVz  
dGVyLmxvY2FsMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAv3C9L06d  
/i knwhxAvYTJytTCSnkzs0VWZHY2yhjW1/c3K1Q1m+CYEn3UPTbY9wegXEuK64r1  
MySjC24cKEXuTR1+hu1yxmVs0DwwfXG/Y1ZrN+jjKaer7xPqphVjP5gRU04aiu0X  
1CbPY5DJPPsvkspjEzFwQQxLhh6pKjtKsto1AWsJHo0EnxHq3gR5LXDIIE/1y  
0r6q/lU4tshC3t8T5wQ+xgLmjg06E3gqbZrW1rj10y2aQIEYytzMDWvEGJw6AuIM  
ckLKuPSkDmYnFbwEZLehRGqf3K4VHUaYVvDN0uDwHAkGjqvagwnLB80WPUNajzx  
F9avejMjj6SfqwIDAQABo4IBfDCAXgwDgYDVR0PAQH/BAQDAgOoMB0GA1UdJQQW  
MBQGCCsGAQUFBwMBBgrBgeFBQcDAjAdBgNVHQ4EFgQUNrvCPEnu+H1WL2gb7tPM  
jTG1PqQwHwYDVR0jbBgwFoAUceFj3mZH65kZ2vmqLG4hKKKQ8LgwcfYIKwYBBQUH  
AQEEZjBkMGIGCCsGAQUFBzACH1ZodHRwczovL3ZhdWx0LWNsdXN0ZXItcHVibGlj  
LXZhdWx0LWI5NGE3YTc5LjFmMjE0YjI5LnoxFmhhc2hpY29ycC5jbG91ZDo4MjAw  
L3YxL3BraS9jYTApBgNVHREEIjAggh5pZGVudG10eS5saW5rZXJkLmNsdXN0ZXiu  
bG9jYWwwaAYDVR0fBGewXzBdoFugWYZXaHR0cHM6Ly92YXVsdc1jbHVzdGVyLXB1  
YmxpYy12YXVsdc1i0TRhN2E30S4xzjIxNGIyOS56MS5oYXNoaWNvcnAuY2xvdWQ6  
ODIwMC92MS9wa2kvY3JsmA0GCSqGSIb3DQEBCwUA4IBAQCxnb2MXdsQ4Wx96rc  
c70ZAUEIbpnfXTF4p43UbTXp2oshu+ZsrMUD+PFMcF669+Y2TGvyen4qYtz0278t  
v37urbYk17lqsD1zdg5eQPWeTTjMEzZk414/+0xtc3lexYwiKTBYbJ6jcYfR7uox  
6f0eCITYD1Rvr5Edu0B8D52hqdd9CzFBZqxyR92LB+40XW8R4ty9+Kz7A2ocqqP  
KICDIihCIu3cDhHquYibYri0S1TyFMpjcpqCl2/Lqceurlc/EFlDM04F04LdtYsN  
QV8LLoQpWU34pZUTx0/KLGxgsNWWe62pJ90DIcvA8Pdz/cn4P0cZ4Y+B475me7Zh  
db4J
```

```
-----END CERTIFICATE-----
```

```
-----BEGIN CERTIFICATE-----
```

```
MIIDQTCCAimgAwIBAgIUJPzHrwe1IJyReqWBSRt5dVVLE7cwDQYJKoZIhvcNAQEL  
BQAwGjEYMBYGA1UEAwwPKi5jbHVzdGVyLmxvY2FsMB4XDTI0MTIzMThMDYxNTgxN1oX  
DTI1MDIwMTIxNTg0N1owGjEYMBYGA1UEAwwPKi5jbHVzdGVyLmxvY2FsMIIBIjAN  
BgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAb1lzoS0q4ydZfpJFPgTFTi15yRMI  
zQiQwDLHRL32Xd50E0t1DgLRQwB/kQ3jwLFZOTE/bhqp3fwWryVBN3PFqS4I03ug  
n9eHn3GEKS6Ickkcf2oBdVugbl7nfSjG/HsgGIMQwa7xe8pAwizqNbclt/Z1Cy9G  
90QJV8qoUSqj1ZHHMSzMJ5WMGMTwNJIydc5Kvkm0zoyVFKUQhw42BF6jZBwynGv  
ndlfihBaMeP2GwDIKitpowwdZXEML7jzVhurPsDz1REqTYeAT8rz0GGRz+yHH1tw  
CXzTR2yDU8tudIRNk+AMVQnCKDNijeZypw9u+cby5tED6M8F+Qi jIpNZUQIDAQAB  
o38wfTAOBgNVHQ8BAf8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQU  
cefj3mZH65kZ2vmqLG4hKKKQ8LgwHwYDVR0jbBgwFoAUceFj3mZH65kZ2vmqLG4h  
KKKQ8LgwGgYDVR0RBwMwEYIPKi5jbHVzdGVyLmxvY2FsMA0GCSqGSIb3DQEBCwUA  
A4IBAQAF9BJ7NXrPmUAQngYU3P3SxjFRGG6nv1BruXeeTOFChi/VFVyN6YKHkpp8  
EIqUXIJDrf6fofViBPYLTIpkBCDLP5eUl+WdsJg0YjvtGQ3qXjstJykwoNQdrR75I  
qqdQ/yvHvyZg/bj9RqgcccIGEAoTAJuPH+QrzNWqmFVU4j1odyXjQQQqhevrv4kz2YN  
ho1A3oX4jh9JQhDu/kTfK/bEwKjfksYbbv/VNsDTt07ph9wDA3r2r3nRjDTfCIu1  
WBgseSdTfRhaYMjH/5aAP3L0MaXCeulWOkmXMpmc3Ztf5fy/81baVYHwMVWzi63V  
Xt0ke7n0Cd13fQQU5xTsuglelXgB
```

```
-----END CERTIFICATE-----%
```

```
mahendranelvakumar@Mahendrans-MBP Downloads %
```

Create linkerd namespace using this command **kubectl create namespace linkerd**



```
mahendralselvakumar@Mahendrans-MBP Downloads % kubectl create namespace linkerd  
namespace/linkerd created  
mahendralselvakumar@Mahendrans-MBP Downloads %
```

## Step 6: Create a Secret for the Issuer Certificate

Run this command **kubectl create secret tls linkerd-identity-issuer --cert=linkerd-issuer.pem --key=private-key.pem --namespace=linkerd** to create linkerd-identity-issuer

```
mahendralselvakumar@Mahendrans-MBP Downloads % kubectl create secret tls linkerd-identity-issuer \  
--cert=linkerd-issuer.pem \  
--key=private-key.pem \  
--namespace=linkerd  
secret/linkerd-identity-issuer created  
mahendralselvakumar@Mahendrans-MBP Downloads %
```

Run this command **kubectl create configmap linkerd-trust-anchor --from-file=ca.crt=linkerd-cert.pem --namespace=linkerd** to create linkerd-trust-anchor

```
mahendralselvakumar@Mahendrans-MBP Downloads % kubectl create configmap linkerd-trust-anchor \  
--from-file=ca.crt=linkerd-cert.pem \  
--namespace=linkerd  
  
configmap/linkerd-trust-anchor created  
mahendralselvakumar@Mahendrans-MBP Downloads %
```

Run the pre-check command **linkerd check -pre** to ensure your Kubernetes cluster meets Linkerd's requirements. Here you will get namespace already exist checks failed



```
mahendralselvakumar@Mahendrals-MBP Downloads % linkerd check --pre
kubernetes-api
-----
✓ can initialize the client
✓ can query the Kubernetes API

kubernetes-version
-----
✓ is running the minimum Kubernetes API version

pre-kubernetes-setup
-----
✖ control plane namespace does not already exist
  The "linkerd" namespace already exists
  see https://linkerd.io/2/checks/#pre-ns for hints
✖ can create non-namespaced resources
  cannot create Namespace/linkerd: namespaces "linkerd" already exists
  see https://linkerd.io/2/checks/#pre-k8s-cluster-k8s for hints
✓ can create ServiceAccounts
✓ can create Services
✓ can create Deployments
✓ can create CronJobs
✓ can create ConfigMaps
✓ can create Secrets
✓ can read Secrets
✓ can read extension-apiserver-authentication configmap
✓ no clock skew detected

linkerd-version
-----
✓ can determine the latest version
✓ cli is up-to-date

Status check results are ✖
mahendralselvakumar@Mahendrals-MBP Downloads %
```

## Step 7: Install Linkerd on your EKS cluster

Run this command **linkerd install --crds | kubectl apply -f -** to install Linkerd's Custom Resource Definitions (CRDs), which must be installed before the control plane

```
mahendralselvakumar@Mahendrals-MBP Downloads % linkerd install --crds | kubectl apply -f -
Rendering Linkerd CRDs...
Next, run `linkerd install | kubectl apply -f -` to install the control plane.

customresourcedefinition.apiextensions.k8s.io/authorizationpolicies.policy.linkerd.io created
customresourcedefinition.apiextensions.k8s.io/egressnetworks.policy.linkerd.io created
customresourcedefinition.apiextensions.k8s.io/httplocalratelimitpolicies.policy.linkerd.io created
customresourcedefinition.apiextensions.k8s.io/httproutes.policy.linkerd.io created
customresourcedefinition.apiextensions.k8s.io/meshtlsauthentications.policy.linkerd.io created
customresourcedefinition.apiextensions.k8s.io/networkauthentications.policy.linkerd.io created
customresourcedefinition.apiextensions.k8s.io/serverauthorizations.policy.linkerd.io created
customresourcedefinition.apiextensions.k8s.io/servers.policy.linkerd.io created
customresourcedefinition.apiextensions.k8s.io/serviceprofiles.linkerd.io created
customresourcedefinition.apiextensions.k8s.io/httproutes.gateway.networking.k8s.io created
customresourcedefinition.apiextensions.k8s.io/grpcroutes.gateway.networking.k8s.io created
customresourcedefinition.apiextensions.k8s.io/tlsrcoutes.gateway.networking.k8s.io created
customresourcedefinition.apiextensions.k8s.io/tcproutes.gateway.networking.k8s.io created
customresourcedefinition.apiextensions.k8s.io/externalworkloads.workload.linkerd.io created
mahendralselvakumar@Mahendrals-MBP Downloads %
```



Install the Linkerd control plane using this command **linkerd install | kubectl apply -f -**

```
mohendralselvakumar@Mahendrans-MBP Downloads % linkerd install | kubectl apply -f -\nWarning: resource namespaces/linkerd is missing the kubectl.kubernetes.io/last-applied-configuration annotation which is required by kubectl apply. kubectl apply should only be used on resources created declaratively by either kubectl create --save-config or kubectl apply. The missing annotation will be patched automatically.\nnamespace/linkerd configured\nclusterrole.rbac.authorization.k8s.io/linkerd-linkerd-identity created\nclusterrolebinding.rbac.authorization.k8s.io/linkerd-linkerd-identity created\nserviceaccount/linkerd-identity created\nclusterrole.rbac.authorization.k8s.io/linkerd-linkerd-destination created\nclusterrolebinding.rbac.authorization.k8s.io/linkerd-linkerd-destination created\nserviceaccount/linkerd-destination created\nsecret/linkerd-sp-validator-k8s-tls created\nvalidatingwebhookconfiguration.admissionregistration.k8s.io/linkerd-sp-validator-webhook-config created\nsecret/linkerd-policy-validator-k8s-tls created\nvalidatingwebhookconfiguration.admissionregistration.k8s.io/linkerd-policy-validator-webhook-config created\nclusterrole.rbac.authorization.k8s.io/linkerd-policy created\nclusterrolebinding.rbac.authorization.k8s.io/linkerd-destination-policy created\nrole.rbac.authorization.k8s.io/remote-discovery created\nrolebinding.rbac.authorization.k8s.io/linkerd-destination-remote-discovery created\nrole.rbac.authorization.k8s.io/linkerd-heartbeat created\nrolebinding.rbac.authorization.k8s.io/linkerd-heartbeat created\nclusterrole.rbac.authorization.k8s.io/linkerd-heartbeat created\nclusterrolebinding.rbac.authorization.k8s.io/linkerd-heartbeat created\nserviceaccount/linkerd-heartbeat created\nclusterrole.rbac.authorization.k8s.io/linkerd-linkerd-proxy-injector created\nclusterrolebinding.rbac.authorization.k8s.io/linkerd-linkerd-proxy-injector created\nserviceaccount/linkerd-proxy-injector created\nsecret/linkerd-proxy-injector-k8s-tls created\nmutatingwebhookconfiguration.admissionregistration.k8s.io/linkerd-proxy-injector-webhook-config created\nconfigmap/linkerd-config created\nrole.rbac.authorization.k8s.io/ext-namespace-metadata-linkerd-config created\nWarning: resource secrets/linkerd-identity-issuer is missing the kubectl.kubernetes.io/last-applied-configuration annotation which is required by kubectl apply. kubectl apply should only be used on resources created declaratively by either kubectl create --save-config or kubectl apply. The missing annotation will be patched automatically.\nsecret/linkerd-identity-issuer configured\nconfigmap/linkerd-identity-trust-roots created\nservice/linkerd-identity created\nservice/linkerd-identity-headless created\ndeployment.apps/linkerd-identity created\nservice/linkerd-dst created\nservice/linkerd-dst-headless created\nservice/linkerd-sp-validator created\nservice/linkerd-policy created\nservice/linkerd-policy-validator created\ndeployment.apps/linkerd-destination created\ncronjob.batch/linkerd-heartbeat created\ndeployment.apps/linkerd-proxy-injector created\nservice/linkerd-proxy-injector created\nsecret/linkerd-config-overrides created\nmohendralselvakumar@Mahendrans-MBP Downloads %
```

## Step 8: Verify Configuration

Use this command to inspect the details of a Kubernetes Secret named linkerd-identity-issuer in the linkerd namespace: **kubectl describe secret linkerd-identity-issuer -n linkerd**

```
mahendralselvakumar@Mahendrans-MBP Downloads % kubectl describe secret linkerd-identity-issuer -n linkerd\nName:      linkerd-identity-issuer\nNamespace: linkerd\nLabels:    linkerd.io/control-plane-component=identity\n           linkerd.io/control-plane-ns=linkerd\nAnnotations: linkerd.io/created-by: linkerd/cli edge-24.11.8\n\nType:  kubernetes.io/tls\n\nData\n=====\nkey.pem:  226 bytes\ntls.crt:  2741 bytes\ntls.key:  1674 bytes\ncrt.pem:  594 bytes
```

Use this command to inspect the details of a Kubernetes ConfigMap named linkerd-trust-anchor in the linkerd namespace: **kubectl describe configmap linkerd-trust-anchor -n linkerd**



Screenshot taken at 55.1 bytes



```
mahendralselvakumar@Mahendrans-MBP Downloads % kubectl describe configmap linkerd-trust-anchor -n linkerd
Name:      linkerd-trust-anchor
Namespace: linkerd
Labels:    <none>
Annotations: <none>

Data
=====
ca.crt:
-----
-----BEGIN CERTIFICATE-----
MIIDQTCCAimgAwIBAgIUJPzHrwe1IJyReqWBSRt5dVVL E7cwDQYJKoZIhvcNAQEL
BQAwGjEYMBYGA1UEAwPKi5jbHVzdGVyLmxvY2FsMB4XDTI0MTIxNTgxN1oX
DTI1MDIwMTIxNTg0N1owGjEYMBYGA1UEAwPKi5jbHVzdGVyLmxvY2FsMIIBIjAN
BgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAYb1lzM0sQ4ydZfpJFPgTFTi15yRMI
zQiQwDLHRL32Xd50E0t1DgLrQwB/kQ3jwLFZ0TE/bhqP3fwWryVBN3PFqS4I03ug
n9eHn3GEKS6IckkcF2oBdVugbL7nfSjG/HsgGIMQwa7xe8pAwiZqNbclt/Z1Cy9G
90QJV8qoUSqj1ZHMSzM5WMBMTwNJIydc5Kyvkm0zoyVFKUQhw42BF6jZBwynGv
ndlfihBaMeP2GwDIKitpowwdZXEML7jzVhurPsDz1REqTYeAT8rz0GGRz+yHH1tw
CXzTR2yDU8tudIRNk+AMVQnCKDNi.jeZypw9u+cbv5tED6M8F+QijIpNZQIDAQAB
o38wFTAOBgNVHQ8BAF8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQU
ceFj3mZH65kZ2vmqLG4hKKQ8LgwHwYDVR0jBBgwFoAUceFj3mZH65kZ2vmqLG4h
KKQ8LgwGgYDVR0RBBMwEYIPKi5jbHVzdGVyLmxvY2FsMA0GCSqGSIb3DQEBCwUA
A4IBAQAF9Bj7NxRpmUAQngYu3P3SxjFRGG6nv1BruXeeTOFChi/VFVyN6YKHkpp8
EIqUXIJDr6fofViBPYLTIpkBCDLP5eUl+WdsJgOYjvtGQ3qXjstJykwoNQdrR75I
qqdQ/yHvyZg/bj9RqgcccIGEaTAJuPH+QrzNWqmFVU4j1odyXjQQQqhevrv4kz2YN
ho1A3oX4jh9JQhDu/kTfK/bEwKjfksYbby/VNsDTt07ph9wDA3r2r3nRjDTfCIu1
WBgseSdTfgrhaYMjH/5aAP3L0MaXCeuW0kmXMpmc3Ztf5fy/8lbaVYHwMWzi63V
Xt0ke7n0Cd13fQQU5xTsuglelXgB
-----END CERTIFICATE-----

BinaryData
=====

Events: <none>
mahendralselvakumar@Mahendrans-MBP Downloads %
```

Run a health check to ensure Linkerd is properly configured using this command

**linkerd check**



```
mohendralselvakumar@Mahendrans-MBP Downloads % linkerd check
kubernetes-api
-----
✓ can initialize the client
✓ can query the Kubernetes API

kubernetes-version
-----
✓ is running the minimum Kubernetes API version

linkerd-existence
-----
✓ 'linkerd-config' config map exists
✓ heartbeat ServiceAccount exist
✓ control plane replica sets are ready
✓ no unschedulable pods
✓ control plane pods are ready
✓ cluster networks contains all pods
✓ cluster networks contains all services

linkerd-config
-----
✓ control plane Namespace exists
✓ control plane ClusterRoles exist
✓ control plane ClusterRoleBindings exist
✓ control plane ServiceAccounts exist
✓ control plane CustomResourceDefinitions exist
✓ control plane MutatingWebhookConfigurations exist
✓ control plane ValidatingWebhookConfigurations exist
✓ proxy-init container runs as root user if docker container runtime is used

linkerd-identity
-----
✓ certificate config is valid
✓ trust anchors are using supported crypto algorithm
✓ trust anchors are within their validity period
✓ trust anchors are valid for at least 60 days
✓ issuer cert is using supported crypto algorithm
✓ issuer cert is within its validity period
✓ issuer cert is valid for at least 60 days
✓ issuer cert is issued by the trust anchor

linkerd-webhooks-and-apisvc-tls
-----
✓ proxy-injector webhook has valid cert
✓ proxy-injector cert is valid for at least 60 days
✓ sp-validator webhook has valid cert
✓ sp-validator cert is valid for at least 60 days
✓ policy-validator webhook has valid cert
✓ policy-validator cert is valid for at least 60 days

linkerd-version
-----
✓ can determine the latest version
✓ cli is up-to-date

control-plane-version
-----
✓ can retrieve the control plane version
✓ control plane is up-to-date
✓ control plane and cli versions match

linkerd-control-plane-proxy
-----
✓ control plane proxies are healthy
✓ control plane proxies are up-to-date
✓ control plane proxies and cli versions match

linkerd-extension-checks
-----
✓ namespace configuration for extensions

Status check results are ✓
mohendralselvakumar@Mahendrans-MBP Downloads % -
```

## Conclusion

Using HCP Vault Dedicated with Linkerd provides a secure and efficient way to manage certificates and protect communication between services in your Kubernetes cluster. HCP Vault handles certificate issuance and renewal automatically, reducing manual effort and ensuring certificates are always up to date. Linkerd uses these certificates to enable secure, encrypted communication with mTLS, adding an extra layer of protection.

This combination simplifies security management and gives you tools like real-time metrics and traffic monitoring to keep your microservices running smoothly. Together, HCP Vault and Linkerd make it easier to build and maintain a secure, reliable, and scalable Kubernetes environment.



Keep Learning, Keep Mesh-ing!!

Feel free to reach out to me, if you have any other queries or suggestions Stay connected on LinkedIn: [Mahendran Selvakumar](https://www.linkedin.com/in/mahendran-selvakumar/)

Stay connected on Medium: <https://devopstronaut.com/>