OvalEdge

# EFS Volume Mount with Access Point

# Table of Contents

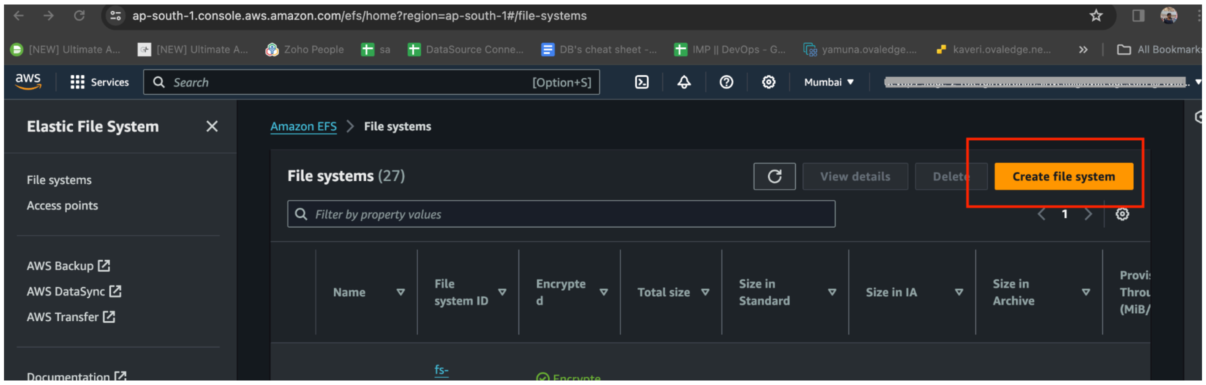
# Introduction

This guide outlines the steps to mount an Amazon Elastic File System (EFS) with access points to an Amazon OpenShift Service on AWS (ROSA). By following these steps, you'll be able to seamlessly integrate EFS storage into your OpenShift applications, allowing for shared access across multiple pods.

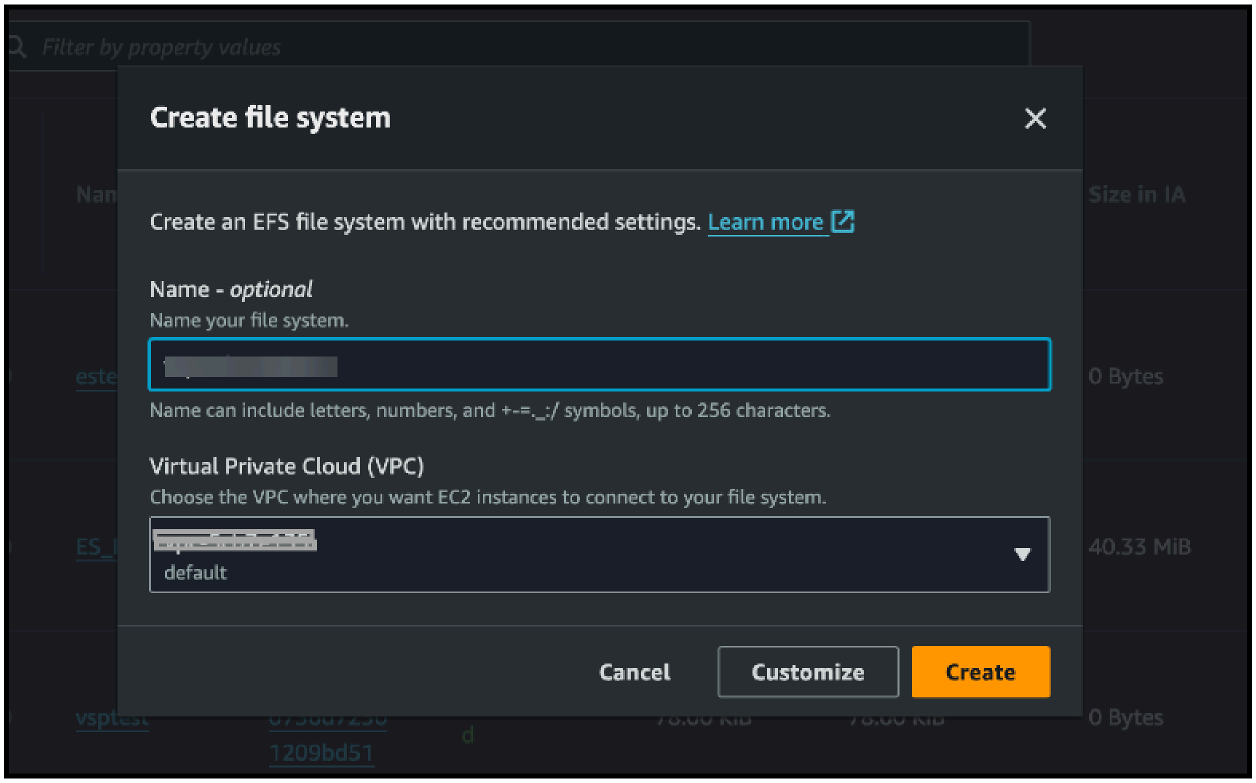
***Note**: If the EFS is already created in AWS, please skip the Create an EFS File System section and start from the Create Access points section.*

# Create an EFS File System

1. Click on "Create file system" in the Amazon EFS service in AWS.



2. Follow the prompts to create the EFS by selecting the appropriate VPC.



**Create file system**

Create an EFS file system with recommended settings. Learn more

Name - *optional*

Name your file system.

[redacted]

Name can include letters, numbers, and +-=._:/ symbols, up to 256 characters.

Virtual Private Cloud (VPC)

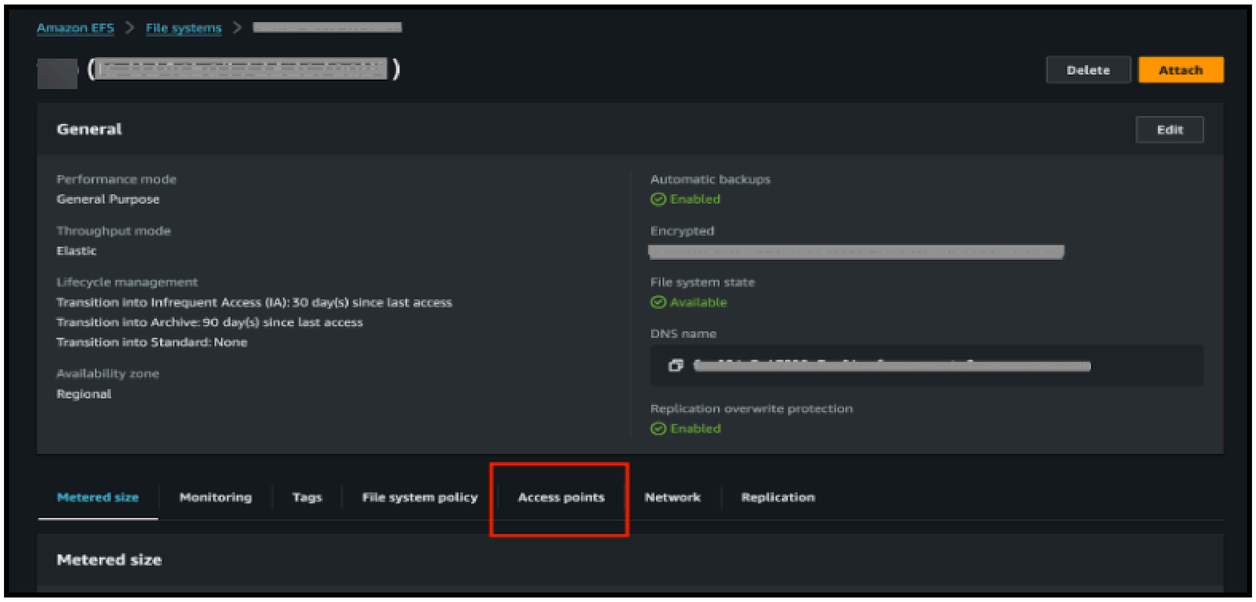
Choose the VPC where you want EC2 instances to connect to your file system.

[redacted]
default

Now EFS is created.

## Create Access points

1. Once the EFS is created, select the file system and click "**Access points**".



[redacted] ([redacted])

General

Performance mode
General Purpose

Throughput mode
Elastic

Lifecycle management
Transition into Infrequent Access (IA): 30 day(s) since last access
Transition into Archive: 90 day(s) since last access
Transition into Standard: None

Availability zone
Regional

Automatic backups
Enabled

Encrypted
[redacted]

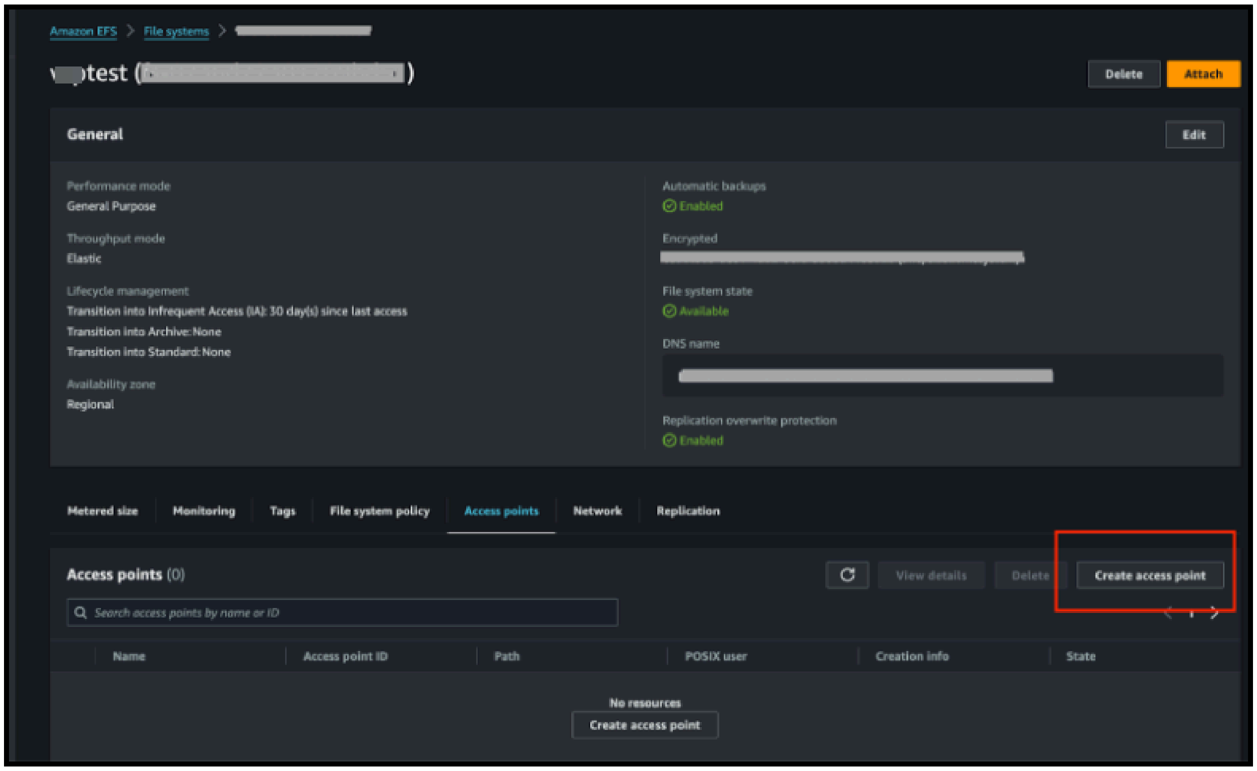
File system state
Available

DNS name
[redacted]

Replication overwrite protection
Enabled

Metered size

2. Click on "Create access point".



[redacted]test ([redacted])

General

Performance mode
General Purpose

Throughput mode
Elastic

Lifecycle management
Transition into Infrequent Access (IA): 30 day(s) since last access
Transition into Archive: None
Transition into Standard: None

Availability zone
Regional

Automatic backups
Enabled

Encrypted
[redacted]

File system state
Available

DNS name
[redacted]

Replication overwrite protection
Enabled

Access points (0)

| Name | Access point ID | Path | POSIX user | Creation info | State |
|---|---|---|---|---|---|

No resources

3. Provide a name for the Access point and specify the root directory path as "**/any-name**".
4. Under "POSIX user," set:

   USERID: 0

   Group ID: 0

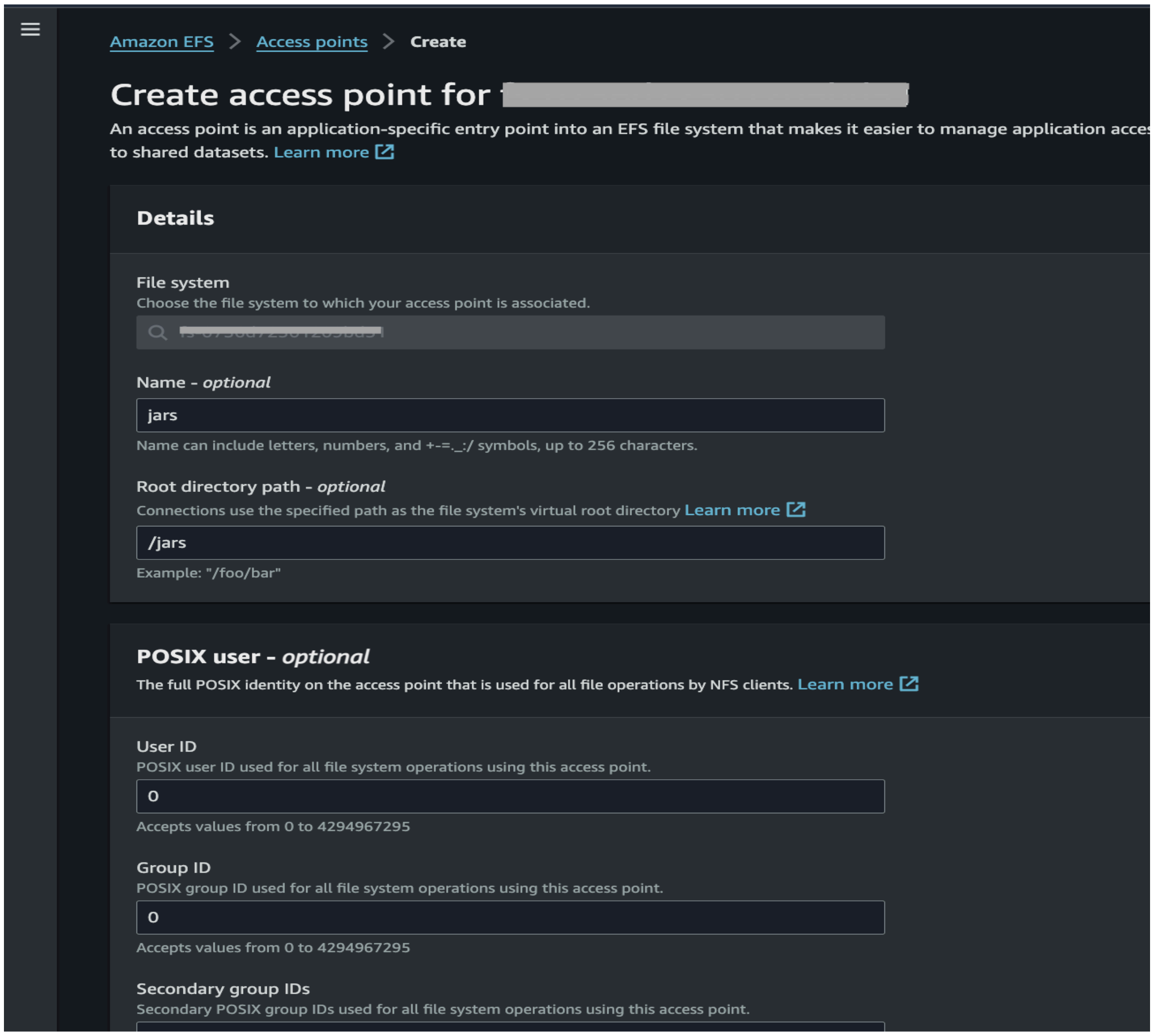
   Secondary GroupID: 0
5. Set permissions for the root directory:

   OWNER USER ID: 777

   OWNER GROUP ID: 777

   Access point: 777

   Follow the same configuration as shown in the screenshot below.



Create access point for [redacted]

An access point is an application-specific entry point into an EFS file system that makes it easier to manage application acce
to shared datasets. Learn more

**Details**

File system
Choose the file system to which your access point is associated.
[redacted]

Name - *optional*
jars
Name can include letters, numbers, and +-=._:/ symbols, up to 256 characters.

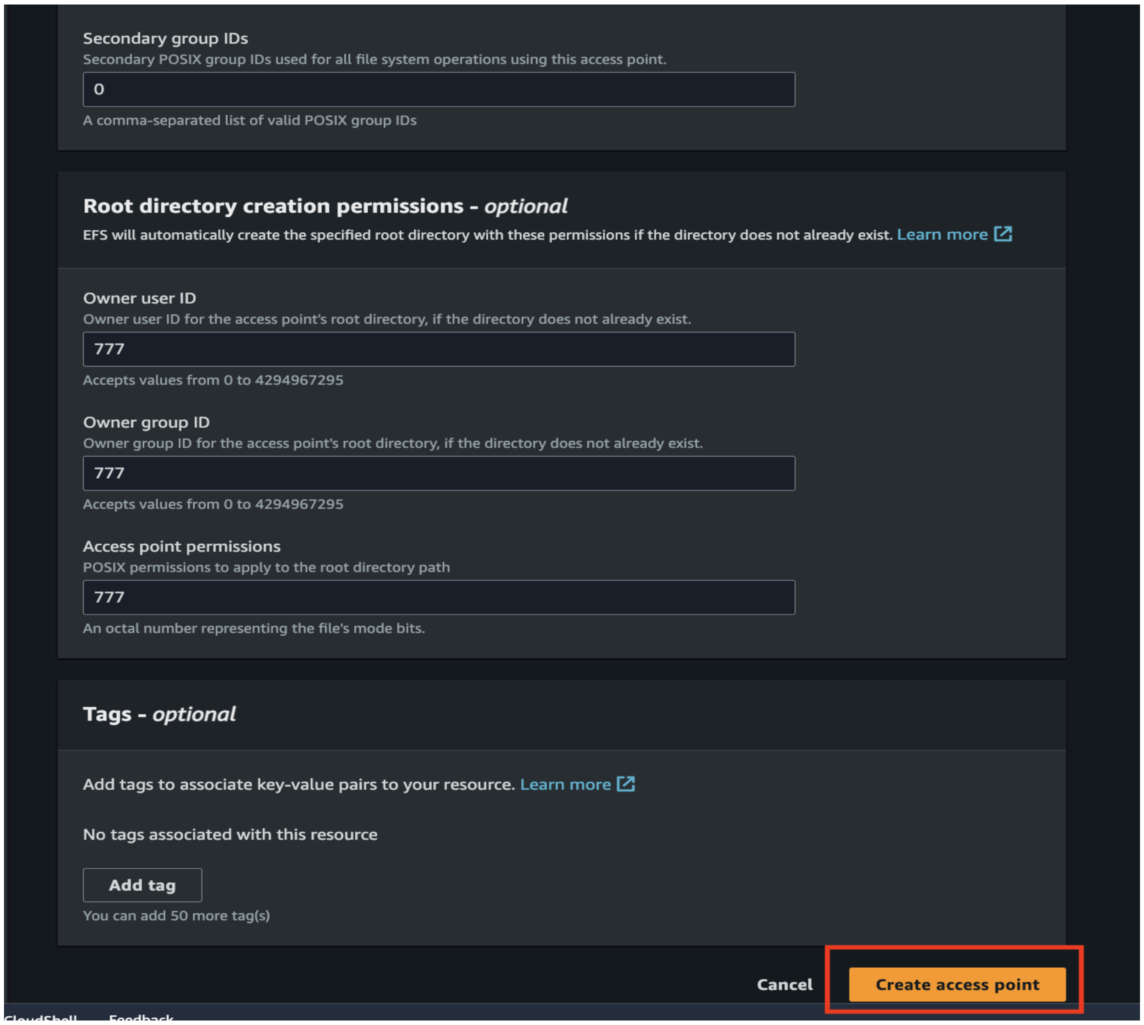
Root directory path - *optional*
Connections use the specified path as the file system's virtual root directory Learn more
/jars
Example: "/foo/bar"

**POSIX user - *optional***
The full POSIX identity on the access point that is used for all file operations by NFS clients. Learn more

User ID
POSIX user ID used for all file system operations using this access point.
0
Accepts values from 0 to 4294967295

Group ID
POSIX group ID used for all file system operations using this access point.
0
Accepts values from 0 to 4294967295

Secondary group IDs
Secondary POSIX group IDs used for all file system operations using this access point.



Secondary group IDs
Secondary POSIX group IDs used for all file system operations using this access point.
0
A comma-separated list of valid POSIX group IDs

**Root directory creation permissions - *optional***
EFS will automatically create the specified root directory with these permissions if the directory does not already exist. Learn more

Owner user ID
Owner user ID for the access point's root directory, if the directory does not already exist.
777
Accepts values from 0 to 4294967295

Owner group ID
Owner group ID for the access point's root directory, if the directory does not already exist.
777
Accepts values from 0 to 4294967295

Access point permissions
POSIX permissions to apply to the root directory path
777
An octal number representing the file's mode bits.

**Tags - *optional***

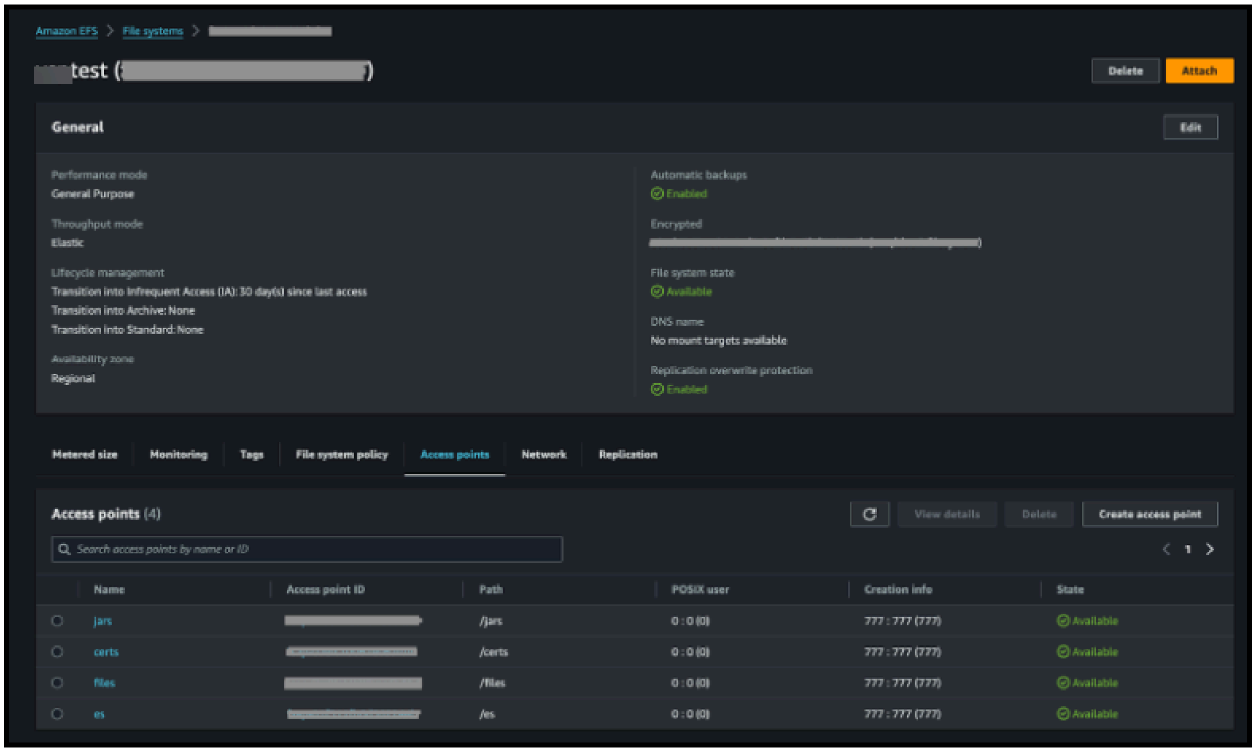
Add tags to associate key-value pairs to your resource. Learn more

No tags associated with this resource

You can add 50 more tag(s)

6. Repeat the process to create access points for the following directories:
    a. third-party-jars (**csp-lib.jar , lineage.jar, required jars**)
    b. certs
    c. files
    d. es

Now Access points are created.



Amazon EFS > File systems > [redacted]

[redacted]test ([redacted])

General

Performance mode
General Purpose

Throughput mode
Elastic

Lifecycle management
Transition into Infrequent Access (IA): 30 day(s) since last access
Transition into Archive: None
Transition into Standard: None

Availability zone
Regional

Automatic backups
✓ Enabled

Encrypted
[redacted]

File system state
✓ Available

DNS name
No mount targets available

Replication overwrite protection
✓ Enabled

Access points (4)

| | Name | Access point ID | Path | POSIX user | Creation info | State |
|---|---|---|---|---|---|---|
| ○ | jars | [redacted] | /jars | 0 : 0 (0) | 777 : 777 (777) | ✓ Available |
| ○ | certs | [redacted] | /certs | 0 : 0 (0) | 777 : 777 (777) | ✓ Available |
| ○ | files | [redacted] | /files | 0 : 0 (0) | 777 : 777 (777) | ✓ Available |
| ○ | es | [redacted] | /es | 0 : 0 (0) | 777 : 777 (777) | ✓ Available |

## Update Helm Charts

1. Navigate to the Helm charts for the necessary changes.
2. For the storage class, if it's already created in the templates, no additional action is required.
3. For each directory (jars, certs, files, es), update the PersistentVolume and PersistentVolumeClaim YAML files with the appropriate file system ID and access point ID obtained from the EFS.
    a. **JARS**
        i. **PersistentVolume**
        Replace <fileSystem-id> & <AccessPointID> with the actual EFS ID and access point ID of "jars" from EFS.

jars_pv.yaml

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: efs-pv-jars
spec:
  capacity:
    storage: 2Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  storageClassName: efs-sc
  csi:
    driver: efs.csi.aws.com
    volumeHandle: <fs-filesystem_ID>::<AccessPointID>
```

ii. **PersistentVolumeClaim**

```yaml
jars_pvc.yaml
```

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: efs-claim-jars
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: efs-sc
  resources:
    requests:
      storage: 2Gi
```

b. **Certs**

i. **PersistentVolume**
Replace <fileSystem-id> & <AccessPointID> with the actual file system ID and access point ID of "certs" from EFS.

```yaml
certs_pv.yaml
```

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: efs-pv-certs
spec:
  capacity:
    storage: 1Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  storageClassName: efs-sc
  csi:
    driver: efs.csi.aws.com
    volumeHandle: <fs-filesystem_ID>::<AccessPointID>
```

ii. **PersistentVolumeClaim**

```
certs_pvc.yaml
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: efs-claim-certs
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: efs-sc
  resources:
    requests:
      storage: 1Gi
```

c. **Files**

i. **PersistentVolume**
Replace <fileSystem-id> & <AccessPointID> with the actual file system ID and access point ID of "files" from EFS.

**files_pv.yaml**

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: efs-pv-files
spec:
  capacity:
    storage: 7Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  storageClassName: efs-sc
  csi:
    driver: efs.csi.aws.com
    volumeHandle: <fs-filesystem_ID>::<AccessPointID>
```

ii. **PersistentVolumeClaim**

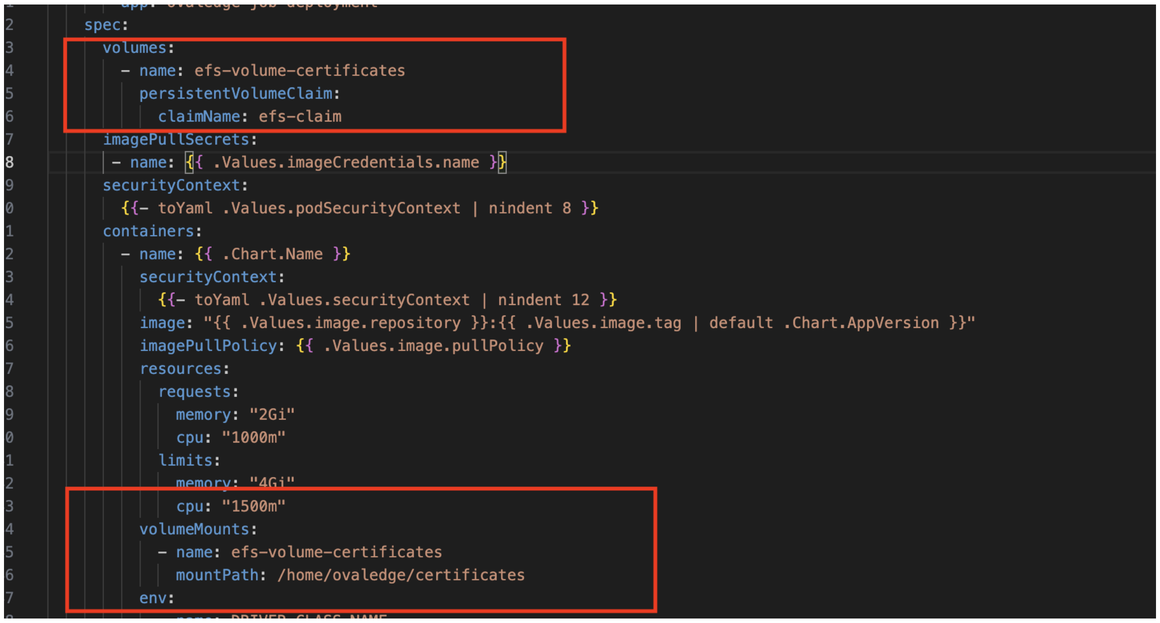
**files_pvc.yaml**

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: efs-claim-files
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: efs-sc
  resources:
    requests:
      storage: 7Gi
```

d. Attach the previously created PersistentVolumeClaim as a volume to the desired ui & job pods.

```yaml
volumes:
  - name: efs-volume-jars
    persistentVolumeClaim:
      claimName: efs-claim-jars
  - name: efs-volume-certs
    persistentVolumeClaim:
      claimName: efs-claim-certs
  - name: efs-volume-files
    persistentVolumeClaim:
      claimName: efs-claim-files
volumeMounts:
  - name: efs-volume-jars
    mountPath: /home/ovaledge/third_party_jars
  - name: efs-volume-certs
    mountPath: /home/ovaledge/certificates
  - name: efs-volume-files
    mountPath: /home/ovaledgefiles
```

Please find below a screenshot for your reference



```
      app: ovaledge-job-deployment
  spec:
    volumes:
      - name: efs-volume-certificates
        persistentVolumeClaim:
          claimName: efs-claim
    imagePullSecrets:
     - name: {{ .Values.imageCredentials.name }}
    securityContext:
      {{- toYaml .Values.podSecurityContext | nindent 8 }}
    containers:
      - name: {{ .Chart.Name }}
        securityContext:
          {{- toYaml .Values.securityContext | nindent 12 }}
        image: "{{ .Values.image.repository }}:{{ .Values.image.tag | default .Chart.AppVersion }}"
        imagePullPolicy: {{ .Values.image.pullPolicy }}
        resources:
          requests:
            memory: "2Gi"
            cpu: "1000m"
          limits:
            memory: "4Gi"
            cpu: "1500m"
        volumeMounts:
          - name: efs-volume-certificates
            mountPath: /home/ovaledge/certificates
        env:
```

Install the updated Helm Chart

*helm install ovaledge ovaledge*

This will deploy your application with the updated configurations including the EFS volume mounts.

***End of the Document***

OVALEDGE

Govern your data smartly.

www.ovaledge.com