DARSHAN NIKAM DATE: 01/02/2024

ELASTIC FILE SYSTEM

EFS allows you to create and manage scalable file storage in the cloud.

EFS Features

- **File Storage in the Cloud***: AWS EFS provides a way for you to store your files in the cloud, just like you would on your computer's hard drive.*
- **Scalable:** ***It's designed to scale automatically based on your needs. Whether you have a small amount of data or a large amount, AWS EFS can handle it without you needing to worry about managing the underlying infrastructure.***
- **Shared File Storage:** ***With EFS, multiple instances (servers) can access the same files simultaneously. This makes it great for applications that need shared access to data.***
- **Compatibility:** ***EFS works with a variety of AWS services and tools, making it easy to integrate into your existing workflows.***
- **Pay-as-You-Go Pricing:** ***You only pay for the storage you use, with no upfront costs or long-term commitments. This makes it cost-effective, especially for growing businesses.***
- **Performance:** ***EFS is designed to provide low-latency performance for a wide range of workloads. It can handle both small, random read/write operations and large, sequential data transfers.***
- **Durability and Availability:** ***Data stored in EFS is distributed across multiple Availability Zones within a region to provide high availability and durability. This ensures that your data is resilient to failures and remains accessible even if an Availability Zone experiences an outage.***
- **Security:** ***EFS supports AWS Identity and Access Management (IAM) for controlling access to file systems. You can define fine-grained permissions to restrict who can access and modify files within your file systems.***
- **Integration:** ***EFS integrates seamlessly with other AWS services such as Amazon EC2, AWS Lambda, and AWS Backup. This allows you to easily incorporate file storage into your existing AWS workflows and architectures.***
- **Performance Modes***: EFS offers two performance modes, General Purpose and Max I/O. General Purpose mode is suitable for a wide range of workloads, while Max I/O mode is optimized for high-performance, latency-sensitive applications.*
- **Lifecycle Management:** ***EFS supports lifecycle management policies, allowing you to automatically move files between storage classes (Standard and Infrequent Access) based on access patterns and retention policies. This helps optimize storage costs without sacrificing performance.***
- **Data Transfer Acceleration:** ***EFS offers data transfer acceleration, which can improve the speed of data transfers between your on-premises environment and EFS by leveraging the AWS Global Accelerator service.***

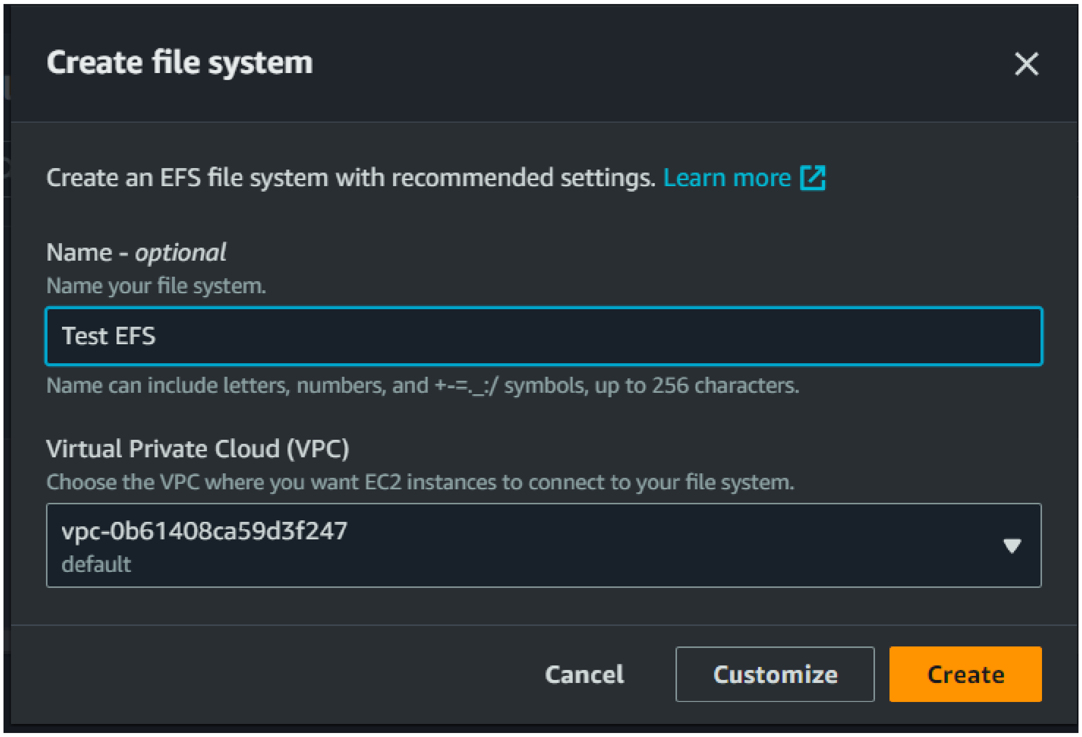
Important Things to keep in mind when attaching EFS to the instance or other resources.

1) ***EC2 instances (or other resources) and your EFS file system are in the same Virtual Private Cloud (VPC) and subnet. EFS is accessible only within the same VPC and cannot be accessed over the public internet.***
2) ***Configure security groups and network ACLs to allow traffic between your EC2 instances and the EFS file system. You need to allow inbound and outbound traffic on the necessary ports for NFS (Network File System), typically port 2049.***
3) ***Mount Targets: EFS uses mount targets to provide access to the file system within specific subnets. You need to create mount targets in each subnet where you want to access the EFS file system from EC2 instances. Ensure that your EC2 instances are launched in these subnets.***
4) ***IAM Permissions: Ensure that the IAM role associated with your EC2 instances has the necessary permissions to access the EFS file system. You can use IAM policies to control access to specific EFS resources and actions.***

❖ Now we create the EFS and attach it to the Instance

Step 1) **Go to ESF Service and click on the Create File System Button.**

Step 2) **Give the file system name that you want, Select the VPC same as your instance, and click on the Create button.**



Create file system

Create an EFS file system with recommended settings. Learn more

Name - *optional*
Name your file system.

Test EFS

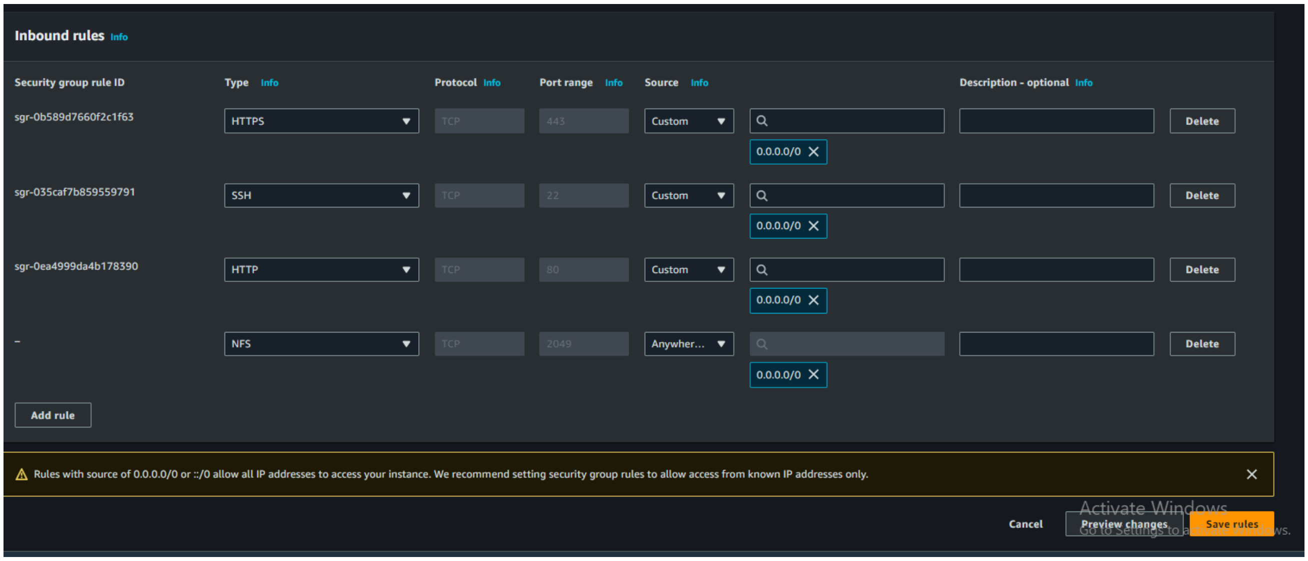
Name can include letters, numbers, and +-=._:/ symbols, up to 256 characters.

Virtual Private Cloud (VPC)
Choose the VPC where you want EC2 instances to connect to your file system.

vpc-0b61408ca59d3f247
default

Now your EFS is created.

Step 3) **Go to the Instance Security option, Select Security Group, and check in the inbound rule is the NFS rule is added or not if not then add the NFS rule and Save rule.**

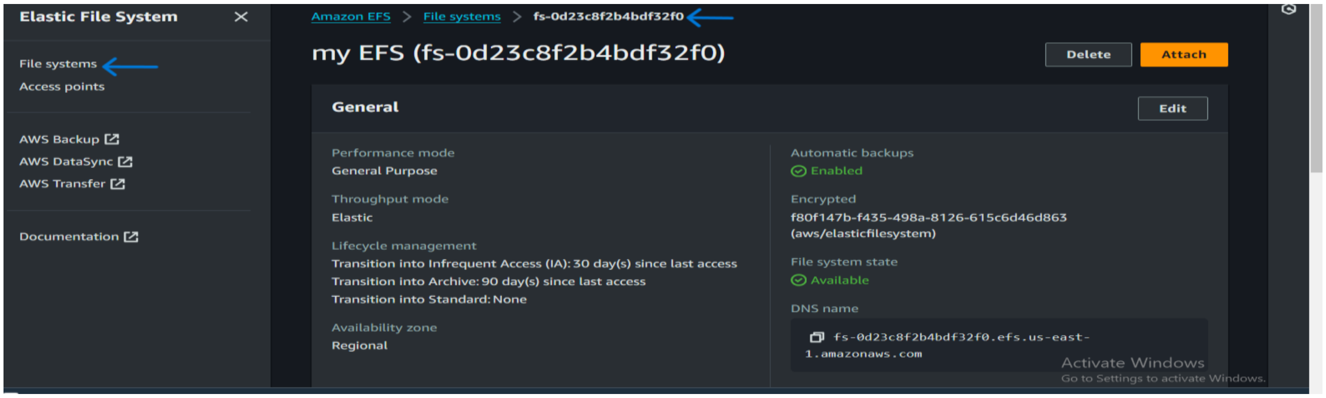


Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-0b589d7660f2c1f63	HTTPS	TCP	443	Custom	0.0.0.0/0		Delete
sgr-035caf7b859559791	SSH	TCP	22	Custom	0.0.0.0/0		Delete
sgr-0ea4999da4b178390	HTTP	TCP	80	Custom	0.0.0.0/0		Delete
-	NFS	TCP	2049	Anywher...	0.0.0.0/0		Delete

⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Step 4) **Now go to the EFS, Click on the File system ID that you created, and click on the Attach button.**



my EFS (fs-0d23c8f2b4bdf32f0)

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

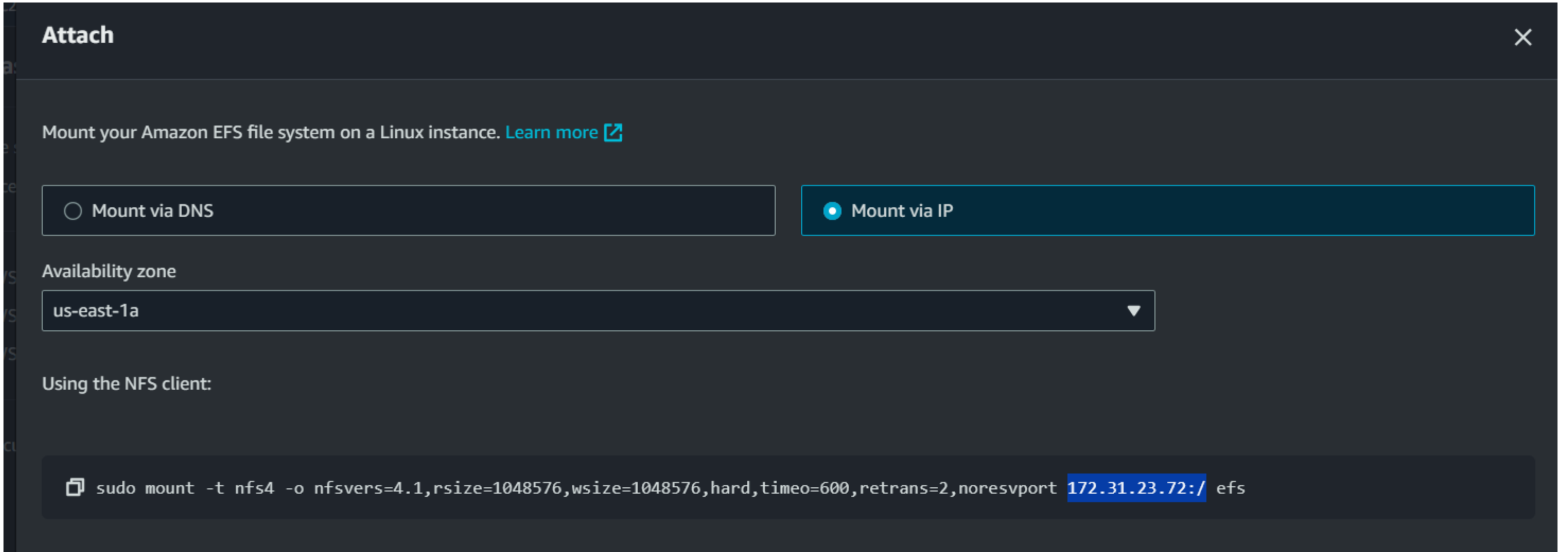
Automatic backups
Enabled

Encrypted
f80f147b-f435-498a-8126-615c6d46d863
(aws/elasticfilesystem)

File system state
Available

DNS name
fs-0d23c8f2b4bdf32f0.efs.us-east-1.amazonaws.com

Step 5) **Select Mount via IP option and copy IP address.**



Attach
Mount your Amazon EFS file system on a Linux instance. Learn more
○ Mount via DNS
● Mount via IP
Availability zone

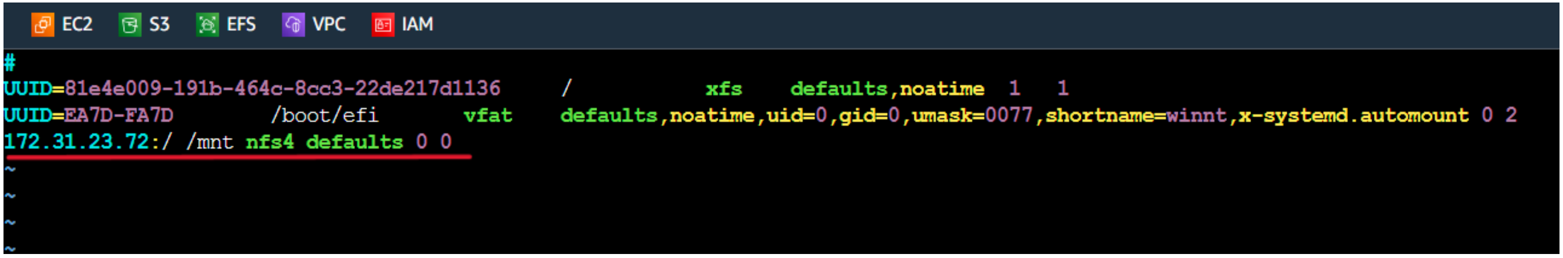
us-east-1a

Using the NFS client:

```
sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport 172.31.23.72:/ efs
```

Step 6) **Connect to your Instance, switch to the root user, and run the [vim /etc/fstab] command.**

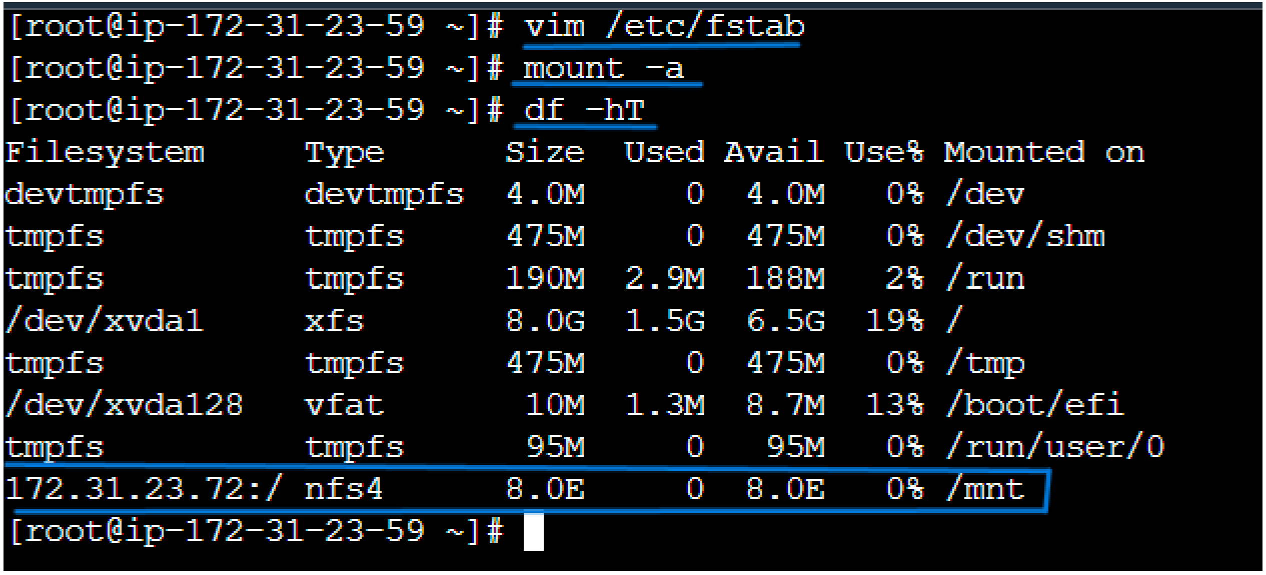
Step 7) **Now add a [<copied.IP.address> mount.path nfs4 defaults 0 0] entry in the file to mount the EFS and save the file.**



```
#
UUID=81e4e009-191b-464c-8cc3-22de217d1136     /           xfs    defaults,noatime  1   1
UUID=EA7D-FA7D         /boot/efi       vfat    defaults,noatime,uid=0,gid=0,umask=0077,shortname=winnt,x-systemd.automount 0 2
172.31.23.72:/ /mnt nfs4 defaults 0 0
~
~
~
~
```

Step 8) **Run [mount –a] command to mount the file system.**

Step 9) **Run [df-tH] command to check your file is mounted or not.**



```
[root@ip-172-31-23-59 ~]# vim /etc/fstab
[root@ip-172-31-23-59 ~]# mount -a
[root@ip-172-31-23-59 ~]# df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  4.0M     0  4.0M   0% /dev
tmpfs          tmpfs     475M     0  475M   0% /dev/shm
tmpfs          tmpfs     190M  2.9M  188M   2% /run
/dev/xvda1     xfs       8.0G  1.5G  6.5G  19% /
tmpfs          tmpfs     475M     0  475M   0% /tmp
/dev/xvda128   vfat       10M  1.3M  8.7M  13% /boot/efi
tmpfs          tmpfs      95M     0   95M   0% /run/user/0
172.31.23.72:/ nfs4      8.0E     0  8.0E   0% /mnt
[root@ip-172-31-23-59 ~]#
```

Now we will see our file system is successfully mounted.