

Project Title – Highly available and optimized web server architecture on AWS.

Overview

This project demonstrates the design and deployment of resilient, secure and cost-effective web server infrastructure using Amazon EC2 and complementary AWS services.

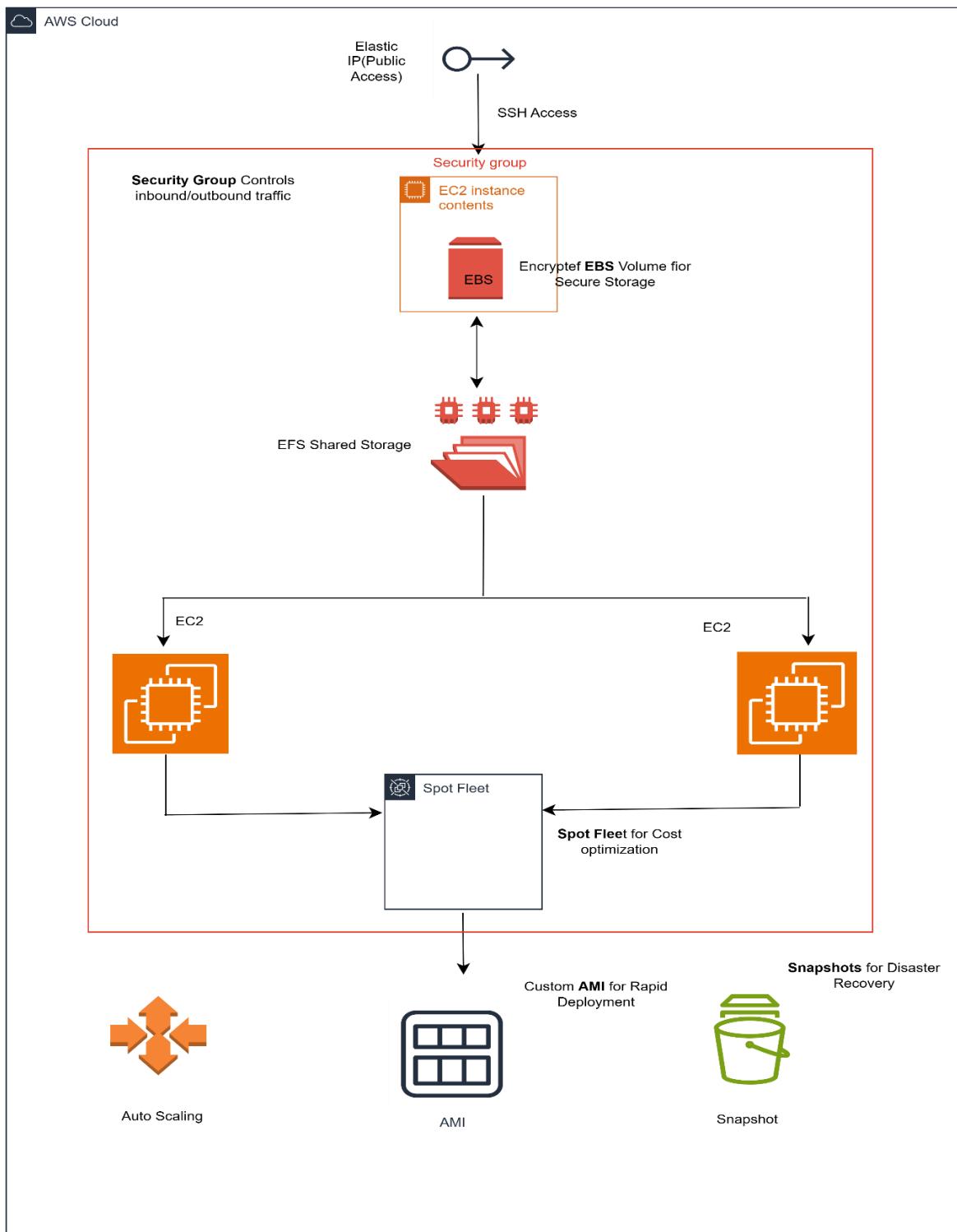
This objective was to build a production-grade architecture that showcases the full lifecycle of EC2-based workloads from instance deployment, optimization, backups and scaling, to cost management and security.

As a Solutions Architect, this project reflects how to think beyond simply “running a server”, focusing instead on availability, performance security, and efficiency.

Project Goals

- Host a functional web server on AWS EC2
- Implement persistent and shared storage (EBS + EFS)
- Demonstrate backup, recovery, and scaling using snapshots and custom AMIs
- Improve availability and latency using Placement Groups
- Secure the environment with encryption, IAM roles, and Security Groups
- Optimize cost with Spot Instances, Spot Fleets, and Hibernation
- Maintain consistent access using Elastic IP
- Document and visualize the architecture for professional presentation

Architecture Illustration



This architecture illustrates a secure, scalable, and cost-optimized EC2 environment using Elastic IP for SSH access, encrypted EBS and EFS for storage, Spot Fleets for efficiency, Auto Scaling for availability, and custom AMIs and Snapshots for rapid recovery and deployment.

- Using encrypted EBS Volumes ensures data-at-rest protection. Security groups enforce least privilege network access

- Elastic IP gives your server a permanent, static public address, perfect for web hosting or when instances are stopped/started frequently

```
aws [Q] Search [Alt+5] Europe (Stockholm) * aws-settable-01 (1034-4132-2345)

Verifying : libbrotli-1.0.9-1.amzn2023.0.2.x86_64
Verifying : mailcap-2.1.49-3.amzn2023.0.3.noarch
Verifying : mod_http2-2.0.27-1.amzn2023.0.3.x86_64
Verifying : mod_lua-2.4.65-1.amzn2023.0.1.x86_64

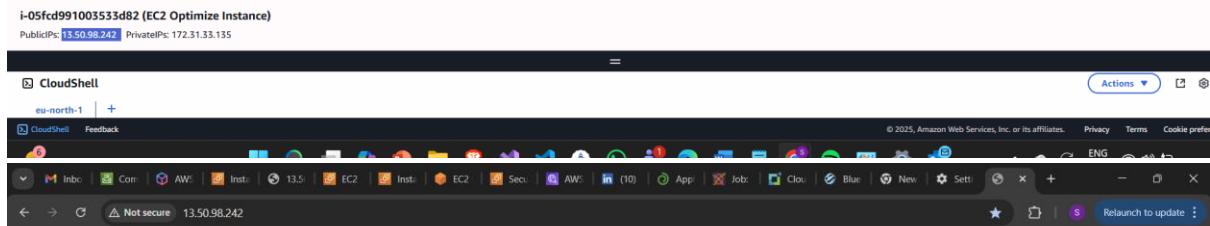
Installed:
apr-1.7.5-1.amzn2023.0.4.x86_64           apr-util-1.6.3-1.amzn2023.0.1.x86_64           apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64           generic-logos-https-18.0.0-12.amzn2023.0.3.noarch
httpd-2.4.65-1.amzn2023.0.1.x86_64        httpd-core-2.4.65-1.amzn2023.0.1.x86_64        httpd-fs-2.4.65-1.amzn2023.0.1.noarch             httpd-tools-2.4.65-1.amzn2023.0.1.x86_64
libbrotli-1.0.9-4.amzn2023.0.2.x86_64      mailcap-2.1.49-3.amzn2023.0.3.noarch          mod_http2-2.0.27-1.amzn2023.0.3.x86_64           mod_lua-2.4.65-1.amzn2023.0.1.x86_64

Complete!
[ec2-user@ip-172-31-33-135 ~]$ sudo yum install httpd -y
Last metadata expiration check: 01:01 ago on Wed Oct 22 08:38:20 2025.
Package httpd-2.4.65-1.amzn2023.0.1.x86_64 is already installed.
Nothing else to do.
Nothing to resolve.

[ec2-user@ip-172-31-33-135 ~]$ sudo systemctl start httpd
[ec2-user@ip-172-31-33-135 ~]$ sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.

[ec2-user@ip-172-31-33-135 ~]$ echo "<h1>Hello from my EC2 web server!</h1>" | sudo tee /var/www/html/index.html
bash: !!: event not found

[ec2-user@ip-172-31-33-135 ~]$ ^C
[ec2-user@ip-172-31-33-135 ~]$ ^C
[ec2-user@ip-172-31-33-135 ~]$ ^C
[ec2-user@ip-172-31-33-135 ~]$ echo "<h1>Hello from my EC2 web server!</h1>" | sudo tee /var/www/html/index.html
<h1>Hello from my EC2 web server!</h1>
[ec2-user@ip-172-31-33-135 ~]$ [
```



Hello from my EC2 web server

3. Using EC2 Instance Connect, I accessed my instance's terminal and installed the Apache web server, I then created a simple HTML file using the echo command to host a custom “Hello from my EC2 web server” page, demonstrating how to deploy and serve web content directly from an EC2 instance.

The screenshot shows the AWS EFS console under the 'File systems' section. A green success message at the top states: 'Success! File system (fs-0d1faa74da1f132868) is available.' Below this, a table lists the created file system 'efs_optimizer_file' with details such as Name (efs_optimizer_file), File system ID (fs-0d1faa74da1f132868), Encrypted (Yes), Total size (6.00 KB), Size in Standard (6.00 KB), Size in IA (0 Bytes), Size in Archive (0 Bytes), Provisioned Throughput (MiB/s) (-), File system state (Available), Creation time (Wed, 22 Oct 2025 13:19:03 GMT), Availability Zone (Regional), and Replication on overwrite protection (Enabled).

```

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

[ec2-user@ip-172-31-33-175 ~]$ sudo yum install -y amazon-efs-utils
Last metadata expiration check: 0:13:16 ago on Wed Oct 22 15:05:09 2025.
Package amazon-efs-utils-2.3.0-1.amzn2023.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
[ec2-user@ip-172-31-33-175 ~]$ sudo mkdir /mnt/efs
mkdir: cannot create directory '/mnt/efs': File exists
[ec2-user@ip-172-31-33-175 ~]$ sudo mount -t efs
[ec2-0d9dfaf4d7edb1c1:/] No such file or directory
[ec2-user@ip-172-31-33-175 ~]$ sudo mount -t efs -o tls fs-0bd9dfaf4d7edb1c:/ efs
[ec2-user@ip-172-31-33-175 ~]$ mount_point efs does not exist
[ec2-user@ip-172-31-33-175 ~]$ sudo mount -t nfs4 -o nfsv4=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,moresvport fs-0bd9dfaf4d7edb1c.efs.eu-north-1.amazonaws.com:/ efs
[ec2-user@ip-172-31-33-175 ~]$ mount_point efs does not exist
[ec2-user@ip-172-31-33-175 ~]$ sudo -l
[ec2-user@ip-172-31-33-175 ~]$ df -hT
Filesystem      Type  Size  Used Avail Use% Mounted on
tmpfs           tmpfs  1.0M   0K  1.0M  0% /dev
tmpfs           tmpfs  453M   0  453M  0% /dev/shm
tmpfs           tmpfs  181M  492K 181M  1% /run
/dev/svme0n1p1  xfs   8.0G  1.6G  6.4G  21% /home
tmpfs           tmpfs  9.0G  9.0G  0     100% /tmp
/dev/svme0n1p28 vfat  1.0M  1.3M  0.7M  13% /boot/efi
127.0.0.1:/    nfs4   8.0E   0  8.0E  0% /mnt/efs/fsl
tmpfs           tmpfs  91M   0  91M  0% /run/user/1000
[ec2-user@ip-172-31-33-175 ~]$ ls /mnt/efs
[ec2-user@ip-172-31-33-175 ~]$ echo "Hello EFS" > /mnt/efs/fsl/test.txt
[ec2-user@ip-172-31-33-175 ~]$ cat /mnt/efs/fsl/test.txt

```

4. EFS allows multiple instances to access shared data concurrently, great for scaling horizontally

The screenshot shows the AWS Placement Groups console with one entry: 'high-perform...' (Group name: pg-002af9561c..., Group Id: pg-002af9561c..., Strategy: cluster, State: available, Partition: -, Group ARN: arn:aws:ec2:eu-north-1:103441322304:placement-group/high-performance).

5. Placement Groups are used in performance-critical systems to reduce network latency between instances — such as in real-time or parallel processing environments.

6. Custom AMIs ensure consistency, speed of deployment, and disaster recovery readiness, allowing identical environments to be launched quickly.

Key Learning and Observations:

1. Designing with Purpose:

- Each AWS service plays a defined role, Elastic IP for stability, EBS for persistence and EFS for scalability.

2. Security by Default:

- a. Using encryption and Security groups ensures that security is built into the architecture from the start, not added later.

3. Cost Optimization:

- a. Spot Instances, Hibernations and efficient storage management demonstrate the importance of balancing performance with affordability.

4. Scalability and Automation:

- a. Custom AMIs and Launch Templates make scaling automatic and predictable, showing real-world deployment readiness.

5. Architectural Thinking:

- a. The project encourages thinking beyond “setting up a server”, focusing on availability, recovery, and optimization, Key Solution Architect principles

Conclusion

This project encapsulates the mindset and practices of an AWS Solution Architect, combining compute, storage, security and cost optimization into a cohesive and practical design.

By using EC2 with complementary AWS services (EBS, EFS, Elastic IP, AMIs, Placement Groups, etc.), the resulting environment demonstrates

- High performance and reliability.
- Strong data protection and access control
- Reduced operational costs
- Scalability and reusability

In essence this project transforms a simple EC2 web server into a robust, production grade cloud architecture, ready for real-world enterprise workloads.