



BUILDING A SCALABLE AND SECURE WEB APPLICATION ON AWS WAF



Introduction

- **Amazon VPC** allows you to create an isolated network environment within AWS for hosting applications securely
- **Subnets** divide the VPC into logical segments (public/private) for better organization and security.
- **Internet Gateway (IGW)** enables communication between resources in the VPC and the internet.
- **Route Tables** control traffic flow between subnets, the internet, and other networks.
- **EC2 Instances** act as compute resources where your application runs.
- **Application Load Balancer (ALB)** distributes incoming traffic across multiple EC2 instances, ensuring scalability and availability.
- **AWS WAF (Web Application Firewall)** protects the application from common web exploits like SQL injection, XSS, and malicious bots.

Benefits

Security:

Fine-grained network control with VPC, subnets, and security groups.
WAF filters malicious requests, reducing attack surface.

High Availability:

ALB distributes traffic across multiple AZs, preventing single points of failure.

Scalability:

Add or remove EC2 instances easily based on load, supported by ALB.

Centralized Control:

One Web ACL can protect multiple resources (ALBs, API Gateways).

Visibility:

WAF provides request logs, CloudWatch metrics, and insights into threats.



Drawbacks / Considerations

⚠ Cost:

VPC components are free, but ALB, EC2, and WAF rules incur hourly and per-request charges.

⚠ Complexity:

Proper planning is required for CIDR ranges, routing, and security group rules.

⚠ Latency:

WAF inspection can add a slight delay to requests, especially with many rules.

⚠ Rule Tuning:

False positives may block legitimate users unless rules are tested in COUNT mode first.

Real-World Use Case

E-commerce Platform:

A company hosting its online store in AWS deploys EC2 instances in private subnets, uses an ALB in public subnets to distribute traffic, and applies WAF rules to protect against attacks like SQL injection, XSS, and DDoS attempts.

Customers access the site through the ALB's DNS, and malicious requests are blocked at the WAF level before reaching the application.



Workflow

VPC – Creates an isolated network environment for your application.

Subnets – Divide the VPC into segments (public for ALB, private for EC2 if needed).

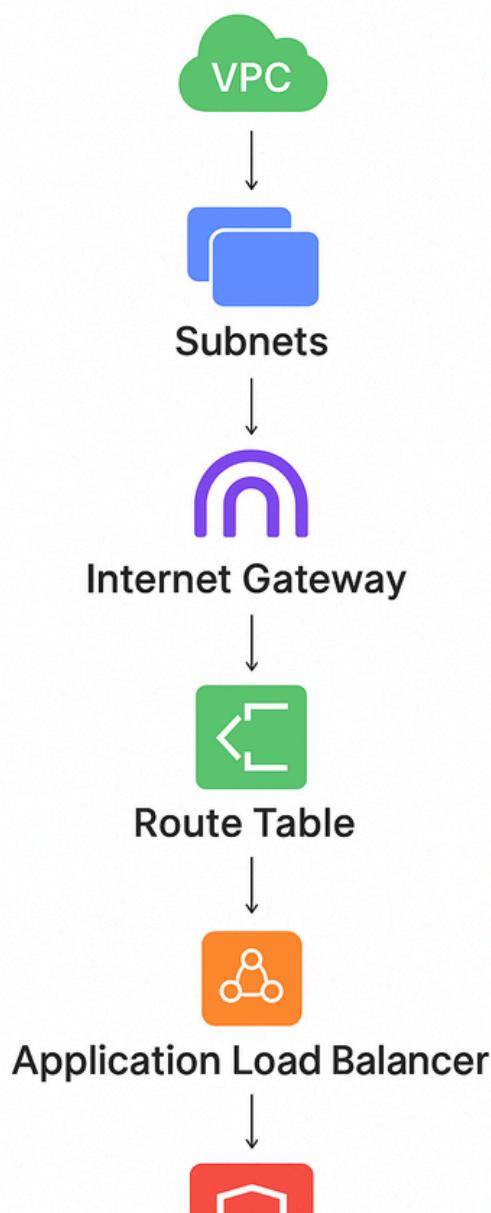
Internet Gateway – Connects the VPC to the internet.

Route Table – Defines routing rules so public subnets can send traffic to the IGW.

EC2 Instances – Run your application code or web server.

Application Load Balancer (ALB) – Distributes incoming traffic across EC2 instances, improving availability.

AWS WAF – Filters and blocks malicious traffic before it reaches the ALB/EC2.





Console Steps

1) Create VPC

VPC → Create VPC

The screenshot shows the AWS Home page with a search bar at the top containing 'vpc'. Below the search bar are several service cards: 'VPC' (Isolated Cloud Resources), 'AWS Global View' (Provides a global dashboard and search functionality), and 'AWS Firewall Manager' (Central management of firewall rules). On the right side, there's a 'Create application' button and a 'Find applications' search field.

Name: my-vpc,

IPv4 CIDR: 10.0.0.0/16

Enable DNS hostnames/support.

The screenshot shows the 'Create VPC' wizard. In the 'VPC settings' step, under 'Resources to create', the 'VPC only' option is selected. A 'Name tag - optional' field contains 'my-vpc'. Under 'IPv4 CIDR block', the value '10.0.0.0/16' is entered. Under 'IPv6 CIDR block', the 'No IPv6 CIDR block' option is selected.

Check VPC sucessfully created

The screenshot shows the VPC dashboard for the 'my-vpc' VPC. The 'Details' section displays the following information:

Details	Value
VPC ID	vpc-0d4eb78e282eb0209
State	Available
DNS resolution	Enabled
Main network ACL	act-0382125a87f31d50d
IPv6 CIDR (Network border group)	-
Tenancy	default
Default VPC	No
Network Address Usage metrics	Disabled
Block Public Access	Off
DHCP option set	dhcp-0d69ff7d72f091c7f
IPv4 CIDR	10.0.0.0/16
Route 53 Resolver DNS Firewall rule groups	-
DNS hostnames	Disabled
Main route table	rtb-0adcf3fb1385eb1ab6
IPv6 pool	-
Owner ID	355487205405



2) Create Subnets (2 public)

VPC >> Subnets >> Create Subnet

Select VPC ID {my-vpc}

The screenshot shows the 'Create subnet' wizard. In the 'VPC' section, the VPC ID 'vpc-0d4eb78e282eb0209 (my-vpc)' is selected. Below it, under 'Associated VPC CIDRs', the IPv4 CIDR '10.0.0.0/16' is listed.

Subnet 1:

Subnet name : public-vpc-1a

Zone : (ap-south-1a)

IPv4 VPC CIDR : 10.0.0.0/16

IPv4 subnet : 10.0.1.0/24

The screenshot shows the 'Subnet 1 of 1' configuration screen. It includes fields for 'Subnet name' (public-vpc-1a), 'Availability Zone' (Asia Pacific (Mumbai) / aps1-az1 (ap-south-1a)), 'IPv4 VPC CIDR block' (10.0.0.0/16), 'IPv4 subnet CIDR block' (10.0.1.0/24), and a 'Tags - optional' section with a single tag 'Name: public-vpc-1a'.

Subnet 2:

Subnet name : public-vpc-1b

Zone : (ap-south-1b)

IPv4 VPC CIDR : 10.0.0.0/16

IPv4 subnet : 10.0.2.0/24

The screenshot shows the 'Subnet 2 of 2' configuration screen, mirroring the setup for Subnet 1. It includes fields for 'Subnet name' (public-vpc-1b), 'Availability Zone' (Asia Pacific (Mumbai) / aps1-az3 (ap-south-1b)), 'IPv4 VPC CIDR block' (10.0.0.0/16), 'IPv4 subnet CIDR block' (10.0.2.0/24), and a 'Tags - optional' section with a single tag 'Name: public-vpc-1b'.



Check Subnets successfully created

The screenshot shows the AWS VPC Subnets page. A success message at the top states: "You have successfully created 2 subnets: subnet-061f19f8871e248ab, subnet-0aa41a0a6fa264bb4". Below this, there is a table with the following data:

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
public-vpc-1a	subnet-061f19f8871e248ab	Available	vpc-0d4eb78e282e60209	Off	10.0.1.0/24
public-vpc-1b	subnet-0aa41a0a6fa264bb4	Available	vpc-0d4eb78e282e60209	Off	10.0.2.0/24

3) Internet Gateway (IGW)

VPC > Internet Gateways > Create IGW

The screenshot shows the AWS VPC Internet Gateways page. A success message at the top states: "You have successfully created 1 internet gateway: igw-0482dd0684c1fb9c9". Below this, there is a table with the following data:

Name	Internet gateway ID	State	VPC ID	Owner
-	igw-0482dd0684c1fb9c9	Attached	vpc-097c7cb2debd7298	355487203405

Create internet gateway

name : my-IG

The screenshot shows the "Create internet gateway" wizard. The "Internet gateway settings" section has a "Name tag" input field containing "my-IG". The "Tags - optional" section contains a single tag "Name" with value "my-IG". At the bottom right are "Cancel" and "Create internet gateway" buttons.

Attach to VPC

The screenshot shows the "Attach to VPC" wizard. It displays a message: "The following internet gateway was created: igw-06b20b8cf1365ff5f - my-IG. You can now attach to a VPC to enable the VPC to communicate with the internet." Below this, it says "Attach to VPC (igw-06b20b8cf1365ff5f)". The "Available VPCs" dropdown is set to "vpc-0d4eb78e282e60209". At the bottom right are "Cancel" and "Attach internet gateway" buttons.



Internet Gateway created

Name	Internet gateway ID	State	VPC ID	Owner
-	gw-0482dd06884c1f8c9	Attached	vpc-097c7db2ede8d7298	355487203405
my-IG	gw-06b20bbcf1365ff5f	Attached	vpc-0d4eb78e282eb0209	355487203405

4) Route Table (public)

Route Tables >> Create >> Name: my-RT (VPC = my-vpc)

Subnet associations >> Select both public subnets.



Edit routes → Add 0.0.0.0/0 → Target: Internet Gateway

The screenshot shows the 'Edit routes' dialog for route table rtb-0a2455556b7aad051. A new route is being added with the following details:

Destination	Target	Status	Propagated	Route Origin
10.0.0.0/16	local	Active	No	CreateRouteTable
0.0.0.0/0	Internet Gateway	-	No	CreateRoute
	igw-06b20b8cf1365ff5	-		

Buttons at the bottom include 'Cancel', 'Preview', and 'Save changes'.

The screenshot shows the 'Details' page for route table rtb-0a2455556b7aad051. The 'Routes' tab displays the following routes:

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-06b20b8cf1365ff5	Active	No	Create Route
10.0.0.0/16	local	Active	No	Create Route Table

Route table created

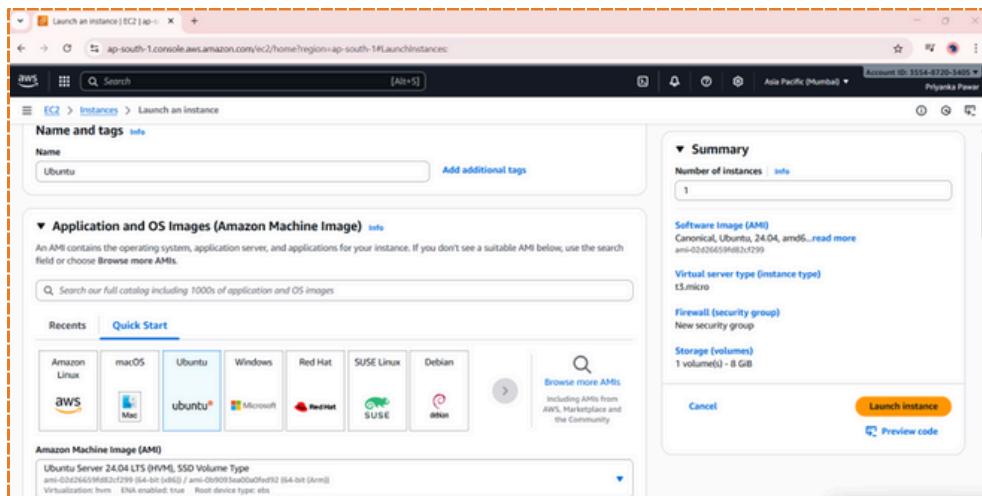
The screenshot shows the 'Route tables' list page. The table displays the following route tables:

Name	Route table ID	Explicit subnet assoc...	Edge associations	Main	VPC
my-RT	rtb-0a2455556b7aad051	-	-	No	vpc-0d4eb78e282eb0209
-	rtb-08c8bb49eb79065ec	-	-	Yes	vpc-097c7cb2ede8d7298
-	rtb-0adck3fb1985eb1ab6	-	-	Yes	vpc-0d4eb78e282eb0209



5) Launch EC2 Instances (web servers)

EC2 → Launch instance (Ubuntu).



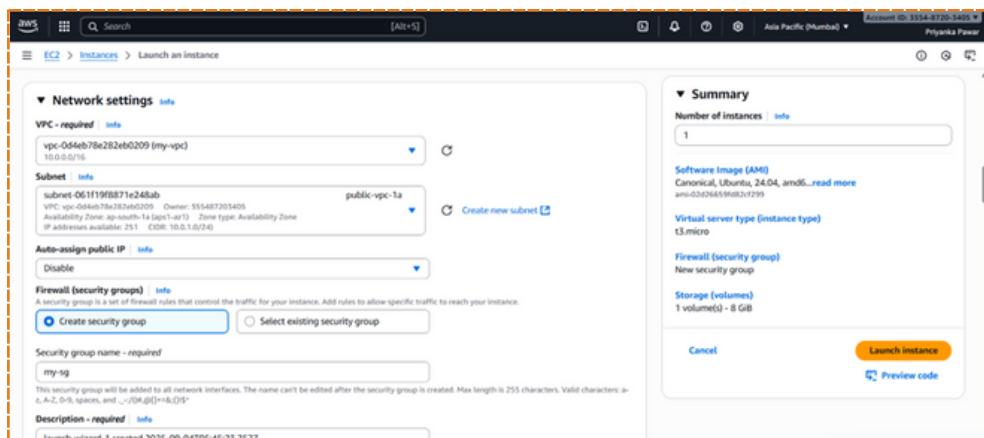
Network Settings :

VPC : (my-vpc)

Subnet : public-vpc-1a

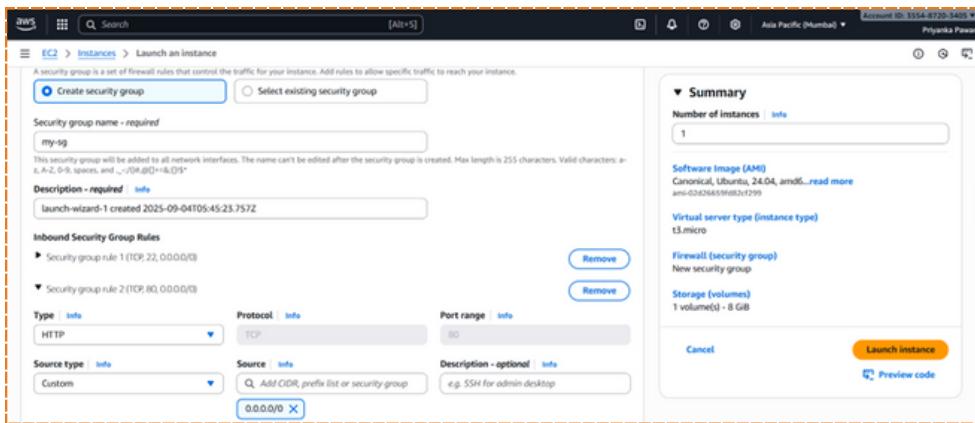
Create security group

name : my-sg



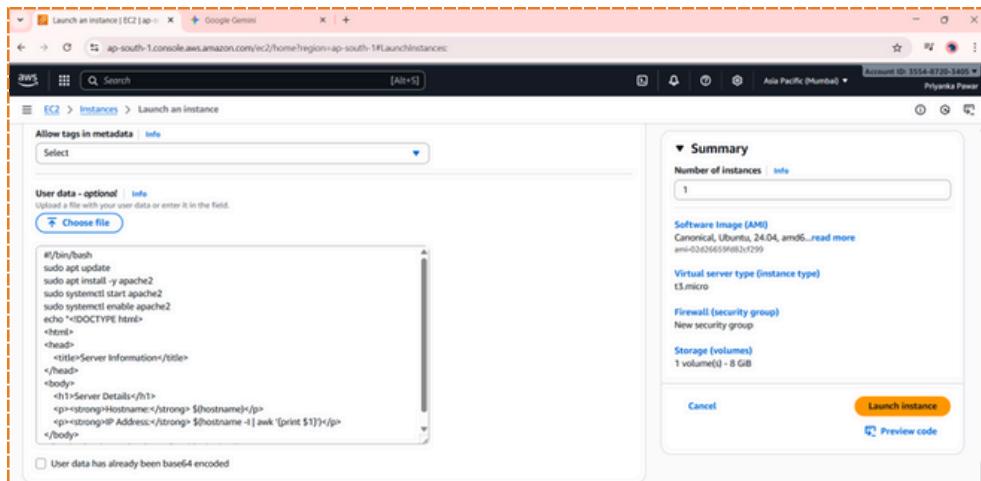


Inbound Security Group add Type : HTTP
Source Type : Custom
Source : 0.0.0.0/0



Advanced Settings add these in Execute Shell

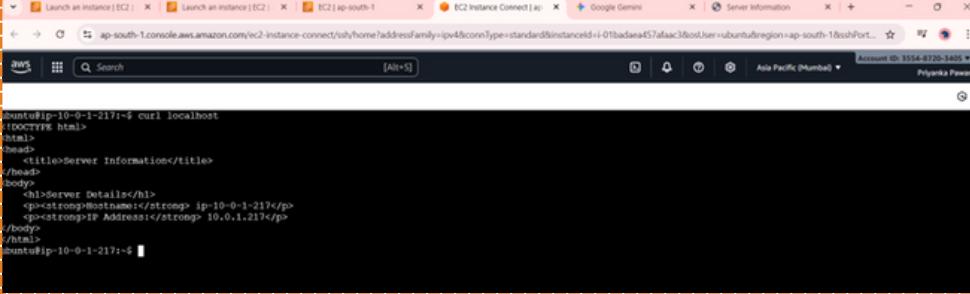
```
#!/bin/bash
sudo apt update
sudo apt install -y apache2
sudo systemctl start apache2
sudo systemctl enable apache2
echo "<!DOCTYPE html>
<html>
<head>
<title>Server Information</title>
</head>
<body>
<h1>Server Details</h1>
<p><strong>Hostname:</strong> $(hostname)</p>
<p><strong>IP Address:</strong> $(hostname -I | awk '{print $1}')</p>
</body>
</html>" | sudo tee /var/www/html/index.html
```





The end result is a running web server (Apache) that hosts a simple, single-page website. This website, which can be accessed from any web browser, displays the hostname and IP address of the server it's running on.

Connect to instance and do curl localhost



```
ubuntu@ip-10-0-1-217:~$ curl localhost
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 112
Connection: keep-alive
Date: Mon, 10 Dec 2018 10:45:17 GMT
Server: Apache/2.4.29 (Ubuntu)
<!DOCTYPE html>
<html>
  <head>
    <title>Server Information</title>
  </head>
  <body>
    <h3>Server Details</h3>
    <p><strong>Hostname:</strong> ip-10-0-1-217</p>
    <p><strong>IP Address:</strong> 10.0.1.217</p>
  </body>
</html>
ubuntu@ip-10-0-1-217:~$
```

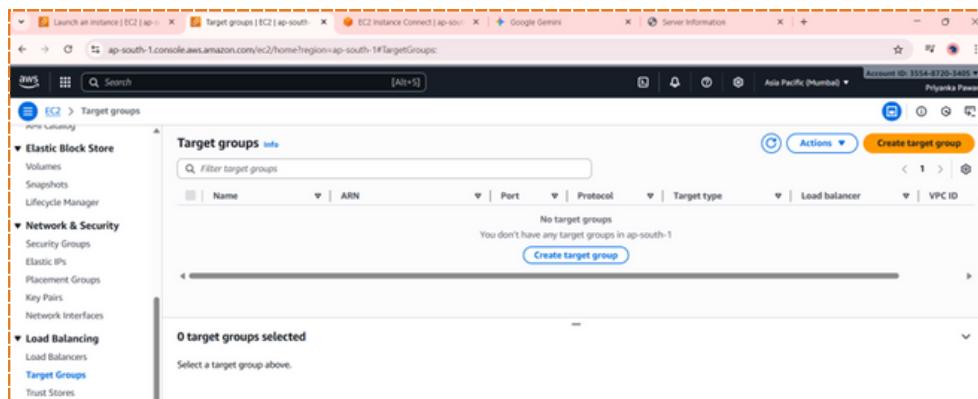
Copy instance public IP and past it in browser

<http://13.232.75.159>



6) Target Group (for ALB)

EC2 >> Load Balancing >> Target Groups >> Create





Type: Instances, Protocol: HTTP:80, VPC: my-vpc, Health check: /

Step 1
Specify group details
Step 2
Register targets

Specify group details
Your load balancer routes requests to the targets in a target group and performs health checks on the targets.

Basic configuration
Settings in this section can't be changed after the target group is created.

Choose a target type

Instances

- Supports load balancing to instances within a specific VPC.
- Facilitates the use of [Amazon EC2 Auto Scaling](#) to manage and scale your EC2 capacity.

IP addresses

- Supports load balancing to VPC and on-premises resources.
- Facilitates routing to multiple IP addresses and network interfaces on the same instance.
- Offers flexibility with microservice based architectures, simplifying inter-application communication.
- Supports IPv6 targets, enabling end-to-end IPv6 communication, and IPv6-to-IPv4 NAT.

Lambda function

- Facilitates routing to a single Lambda function.
- Accessible to Application Load Balancers only.

Target group name: my-tg

Protocol: HTTP Port: 80

IP address type: IPv4

VPC: vpc-0d4eb78e282eb0209 (my-vpc)

Health check

my-tg

Details

Target type	Protocol : Port	Protocol version
Instance	HTTP:80	HTTP1

Distribution of targets by Availability Zone (AZ)

Total targets	Healthy	Unhealthy	Unused	Initial	Draining
1	0	0	1	0	0



7) Application Load Balancer (ALB)

EC2 → Load Balancers → Create Load Balancer → Application LB

The screenshot shows the AWS EC2 Load Balancers page. On the left, there's a navigation sidebar with 'Elastic Block Store' and 'Network & Security' sections. The main area is titled 'Load balancers' with a sub-instruction: 'Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.' A search bar and a 'Create load balancer' button are at the top right. Below, a table header includes columns for Name, State, Type, Scheme, IP address type, VPC ID, and Availability Zones. A message states 'No load balancers' and 'You don't have any load balancers in ap-south-1'. A 'Create load balancer' button is located at the bottom right of the table.

Scheme: Internet-facing, IP type: IPv4

The screenshot shows the 'Create Application Load Balancer' wizard, Step 1: Basic configuration. It has a title 'Create Application Load Balancer' with a 'info' link. Below it is a sub-instruction: 'The Application Load Balancer distributes incoming HTTP and HTTPS traffic across multiple targets such as Amazon EC2 instances, microservices, and containers, based on request attributes. When the load balancer receives a connection request, it evaluates the listener rules in priority order to determine which rule to apply, and if applicable, it selects a target from the target group for the rule action.' There are two tabs: 'How Application Load Balancers work' and 'Basic configuration'. Under 'Basic configuration', there's a 'Load balancer name' field containing 'my-lb', a note about character limits, and a 'Scheme' section. The 'Internet-facing' option is selected, showing its details: 'Serves internet-facing traffic', 'Has public IP addresses', 'DNS name resolves to public IPs', and 'Requires a public subnet'. The 'Internal' option is also listed. Below this is a 'Load balancer IP address type' section with 'IPv4' selected.

Subnets: select both public subnets

The screenshot shows the 'Create Application Load Balancer' wizard, Step 2: Network mapping. It has a title 'Network mapping' with a 'info' link. Below it is a 'VPC' section with a note: 'The load balancer routes traffic to targets in the selected subnets, and in accordance with your IP address settings.' A dropdown menu shows 'vpc-05d6eb78e282eb0209 (my-vpc)' and '10.0.0.0/16'. There's also an 'IP pools' section with a note: 'You can optionally choose to configure an IPAM pool as the preferred source for your load balancer IP addresses. Create or view Pools in the Amazon VPC IP Address Manager console.' A checkbox 'Use IPAM pool for public IPv4 addresses' is checked. The 'Availability Zones and subnets' section contains two entries: 'ap-south-1a (ap-s1-a2)' and 'ap-south-1b (ap-s1-a2)'. Each entry has a 'Subnet' dropdown showing 'subnet-061f198887e24ab' and 'subnet-0aa41a096fa2646b4' respectively, with 'public-vpc-1a' and 'public-vpc-1b' selected.



Security group: my-sg

Listener: HTTP:80 – Forward to your Target Group

The screenshot shows the 'Create Application Load Balancer' wizard. In the 'Security groups' step, 'my-sg' is selected. In the 'Listeners and routing' step, a new listener for 'HTTP:80' is being configured, pointing to a target group named 'my-tg'. The 'Create target group' button is visible.

After creation, hit the ALB DNS name in a browser.

The screenshot shows the 'Load balancers' page with one entry: 'my-lb' (Active, application, Internet-facing, IPv4, VPC ID: vpc-0d4eb7be282eb0209, 2 Availability Zones).

The screenshot shows the 'Server Details' page for 'my-lb'. It displays the Hostname: ip-10-0-1-217 and IP Address: 10.0.1.217.



8) AWS WAF (protect the ALB)

IP sets >> Create IP set

name : my-ip-set

Region : Asia Pacific (Mumbai)

IP Version : IPv4

IP address : 172.17.6.0/24 {this ip will block}

Create IP set Info
An IP set is a collection of IP addresses.

IP set details

IP set name The name must have 1-128 characters. Valid characters: A-Z, a-z, 0-9, - (hyphen), and _ (underscore).
Description - optional

The description can have 1-256 characters.

Region Choose the AWS region to create this IP set in.

IP version IPv4 IPv6

IP addresses

IP set Created

Success
You successfully created the IP set my-ip-set.

AWS WAF > IP sets

IP sets (1)

IP sets that you have defined in the selected region.

Name	Description	ID
my-ip-set	-	9028feb7-3f8a-473e-85f5-0fd96605769a



AWS WAF → Web ACLs → Create

Web ACLs (0)

No web ACLs found

You don't have any web ACLs in the Asia Pacific (Mumbai) Region created with this version of AWS WAF.

If you are looking for web ACLs created in the past, please check the AWS WAF Classic console. Please click [here](#) for more information.

Create web ACL

Scope: REGIONAL (same Region as ALB)

Default action: Allow

Step 2
Add rules and rule groups

Step 3
Set rule priority

Step 4
Configure metrics

Step 5
Review and create web ACL

Resource type

Choose the type of resource to associate with this web-ACL. Changing this setting will reset the page.

Regional resources (Application Load Balancers, Amazon API Gateway REST APIs, Amazon App Runner services, AWS AppSync APIs, Amazon Cognito user pools and AWS Verified Access Instances)

Region

Choose the AWS Region to create this web-ACL in. Changing this setting will reset the page.

Asia Pacific (Mumbai)

Name

my-acl

The name must have 1-128 characters. Valid characters: A-Z, a-z, 0-9, - (hyphen), and _ (underscore).

Description - optional

The description can have 1-256 characters.

CloudWatch metric name

my-acl

The name must have 1-128 characters. Valid characters: A-Z, a-z, 0-9, - (hyphen), and _ (underscore).

Add AWS resources

Resource type = Application Load Balancer

Resource type

Select the resource type and then select the resource you want to associate with this web-ACL.

Application Load Balancer

Amazon API Gateway REST API

Amazon App Runner service

AWS AppSync API

Amazon Cognito user pool

AWS Verified Access

Resources (1)

Select the resource you want to associate with the web-ACL.

Find AWS resources to associate

Name

my-lb

Add



Add IP set

The screenshot shows the 'Add my own rules and rule groups' step of the 'Create web ACL' wizard. Under 'Rule type', 'IP set' is selected. A new rule is being defined with the name 'my-test-ip'. The 'IP set' section is visible at the bottom.

action: Block

Add rule

The screenshot shows the 'IP set' configuration page. For the rule 'my-test-ip', the 'Action' dropdown is set to 'Block'. Other options like 'Allow', 'Count', and 'Challenge' are also listed.

Configure metrics

Enable sampled requests

The screenshot shows the 'Configure metrics' step of the 'Create web ACL' wizard. It displays CloudWatch metrics configuration for the rule 'my-test-ip'. Under 'Request sampling options', the 'Enable sampled requests' option is selected.



WAF tuning: start rules in Count mode, review logs, then switch to Block.

The screenshot shows the AWS WAF & Shield interface. On the left, a sidebar lists options like AWS WAF, Web ACLs, Bot control dashboard, Application integration, IP sets, Regex pattern sets, Rule groups, and Add-on protections. The 'Web ACLs' option is selected. The main area displays a success message: 'You successfully created the web ACL my-acl.' Below this, a table lists the created Web ACL, with one entry shown:

Name	Description	ARN	ID
my-acl	-	arn:aws:wafv2:ap-south-1:1355487203405:regional/weba... 8002-66d2b21aa75b	c347f055-b2ca-4186-8002-66d2b21aa75b

Conclusion

Building a secure and scalable application on AWS involves carefully designing a VPC, configuring public and private subnets, setting up an ALB for load distribution, and adding AWS WAF to protect against threats. This architecture is production-grade, offering security, high availability, and scalability.

When properly tuned, it minimizes downtime and protects applications from common vulnerabilities, making it ideal for businesses that need a secure, internet-facing application environment.



Thank
You