

ASG

1. Create one VPC in N. Virginia region.

- Open aws console
- Navigate to vpc

The screenshot shows the AWS VPC creation page. At the top, a green success message says "You successfully created vpc-0a39d7c4a4792149d / ASG-task-vpc". Below it, the VPC ID is listed as "vpc-0a39d7c4a4792149d". The "Details" section contains various configuration parameters:

VPC ID vpc-0a39d7c4a4792149d	State Available	Block Public Access Off	DNS hostnames Disabled
DNS resolution Enabled	Tenancy default	DHCP option set dopt-05c5f706803d0d90e	Main route table rtb-0004078bd986192aa
Main network ACL acl-014491767cc904a00	Default VPC No	IPv4 CIDR 10.0.0.0/24	IPv6 pool -
IPv6 CIDR (Network border group) -	Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -	Owner ID 414691912691

Below the details, there are tabs for "Resource map", "CIDRs", "Flow logs", "Tags", and "Integrations". The "Resource map" tab is selected, showing a summary of resources: "VPC", "Subnets (0)", and "Route tables (1)".

1.

2 . Create two subnets: one public subnet and one private subnet.

- In vpc
- One public subnet
- One private subnet

The screenshot shows the AWS Subnet creation page. A green success message at the top says "You have successfully created 2 subnets: subnet-08e25989ee6c2a873, subnet-07c7d65419aad7e70". Below it, the "Subnets (2)" section displays the following table:

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
asg-private	subnet-07c7d65419aad7e70	Available	vpc-0a39d7c4a4792149d ASG...	Off	10.0.0.128/25
asg-public	subnet-08e25989ee6c2a873	Available	vpc-0a39d7c4a4792149d ASG...	Off	10.0.0.0/25

At the bottom, there is a "Select a subnet" dropdown menu.

3 . Attach an IGW to the VPC.

- Create a internetgate way and attach it to VPC

The screenshot shows the AWS CloudFormation console with a success message: "Internet gateway igw-07470312083595646 successfully attached to vpc-0a39d7c4a4792149d". Below this, the resource details for "igw-07470312083595646 / ASG-IGW" are shown. The "Details" tab is selected, displaying the Internet gateway ID (igw-07470312083595646), State (Attached), VPC ID (vpc-0a39d7c4a4792149d | ASG-task-vpc), and Owner (414691912691). A "Tags (1)" section shows a single tag: "Search tags".

4. Create one public route table (RT) and one private route table.

The screenshot shows the AWS CloudFormation console with a success message: "Route table rtb-0b77eab20a0e583fa | asg-private-rt was created successfully". Below this, the "Route tables (4)" list is displayed. The table includes columns for Name, Route table ID, Explicit subnet associations, Edge associations, Main, and VPC. The rows show four route tables: "-" (two entries), "asg-public-rt" (with RT ID rtb-0c8cdb8d7c33aa5d6), and "asg-private-rt" (with RT ID rtb-0b77eab20a0e583fa).

5. Deploy a NAT gateway in the public subnet and attach the NAT gateway to the private subnet.

- NAT gateway created in public subnet while creating
-

NAT gateway ID: nat-0edf3fff2e85a9bc5

NAT gateway ARN: arn:aws:ec2:us-east-1:414691912691:natgateway/nat-0edf3fff2e85a9bc5

VPC: vpc-0a39d7c4a4792149d / ASG-task-vpc

Connectivity type: Public

State: Pending

Primary public IPv4 address: -

Primary private IPv4 address: -

Subnet: subnet-07c7d65419aad7e70 / asg-private

Created: Tuesday, November 4, 2025 at 11:44:40 GMT+5:30

Secondary IPv4 addresses | Monitoring | Tags

Secondary IPv4 addresses

Search:

Private IP address: Network interface ID: Status: Failure message:

- Attaching NAT to private subnet in private route table
- Add private subnet to private route table and edit route sand add NAT gateway

You have successfully updated subnet associations for rtb-0b77eab20a0e583fa / asg-private-rt.

rtb-0b77eab20a0e583fa / asg-private-rt

Details Info

Route table ID: rtb-0b77eab20a0e583fa

VPC: vpc-0a39d7c4a4792149d | ASG-task-vpc

Main: No

Owner ID: 414691912691

Explicit subnet associations: subnet-07c7d65419aad7e70 / asg-private

Edge associations: -

Routes | Subnet associations | Edge associations | Route propagation | Tags

Routes (2)

Destination	Target	Status	Propagated	Route ID
0.0.0.0/0	nat-0edf3fff2e85a9bc5	Active	No	Create
10.0.0.0/24	local	Active	No	Create

6. Create two instances, one in the public subnet and one in the private subnet.

- Navigate ec2 and edit network setting add vpc and public subnet and security groups

▼ Network settings [Info](#)

VPC - required | [Info](#)

vpc-0a39d7c4a4792149d (ASG-task-vpc)
10.0.0.0/24



Subnet | [Info](#)

subnet-08e25989ee6c2a873
VPC: vpc-0a39d7c4a4792149d Owner: 414691912691
Availability Zone: us-east-1a (use1-az4) Zone type: Availability Zone
IP addresses available: 123 CIDR: 10.0.0.0/25

asg-public



[Create new subnet](#) ↗

Auto-assign public IP | [Info](#)

Enable

Firewall (security groups) | [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group

Select existing security group



[Compare security group rules](#)

Common security groups | [Info](#)

Select security groups

default sg-07f90e1219dc5b3ad X
VPC: vpc-0a39d7c4a4792149d

Security groups that you add or remove here will be added to or removed from all your network interfaces.

- Second instance private
- Select vpc
- Select private subnet and default security groups

▼ Network settings [Info](#)

VPC - required | [Info](#)

vpc-0a39d7c4a4792149d (ASG-task-vpc)
10.0.0.0/24



Subnet | [Info](#)

subnet-07c7d65419aad7e70
VPC: vpc-0a39d7c4a4792149d Owner: 414691912691
Availability Zone: us-east-1a (use1-az4) Zone type: Availability Zone
IP addresses available: 122 CIDR: 10.0.0.128/25

asg-private



[Create new subnet](#) ↗

Auto-assign public IP | [Info](#)

Disable

Firewall (security groups) | [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group

Select existing security group



[Compare security gr...](#)

Common security groups | [Info](#)

Select security groups

default sg-07f90e1219dc5b3ad X
VPC: vpc-0a39d7c4a4792149d

- Security groups that you add or remove here will be added to or removed from all your network interfaces.
- **Public instance and private instance**

<input type="checkbox"/>	asg-public	i-0345700c9381ead39	<input type="checkbox"/> Stopped			t3.micro	-	View alarms +	us-east-1a
<input type="checkbox"/>	asg-private	i-05fd47955fca46195	<input checked="" type="checkbox"/> Running			t3.micro	<input type="checkbox"/> Initializing	View alarms +	us-east-1a

- 7. Deploy Apache server on both EC2 instances with a sample index.html file.

- Using jump server,
- Acces to public subnet instance
- Next, give chmod 400 (pem.key name)
- Now do ssh , using private instance ssh paste iit in the same terminal

```
[ec2-user@ip-10-0-0-114 ~]$ ssh -i "all-key.pem" ec2-user@10.0.0.197
The authenticity of host '10.0.0.197 (10.0.0.197)' can't be established
ED25519 key fingerprint is SHA256:a5gy+SRDTxycyUU98HiLDXCTbDKQNIQPTYha/n
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.0.197' (ED25519) to the list of known
          ,
          #_
  ~\_\_ #####_      Amazon Linux 2023
  ~~ \_\#####\_
  ~~   \###|_
  ~~     \|/_,_--> https://aws.amazon.com/linux/amazon-linux-2023
  ~~       V~,'`->
  ~~~
  ~~.-.  /`/
  ~~ /`/`/
  ~~ /`/`/
[ec2-user@ip-10-0-0-197 ~]$ exit
logout
```

- Install httpd
- Sudo yum install httpd -y

Installed:

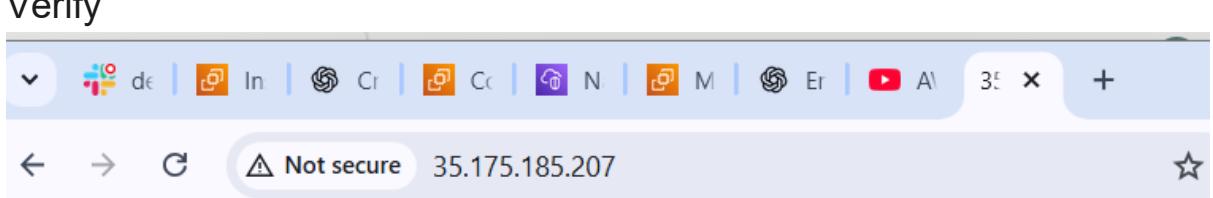
```
apr-1.7.5-1.amzn2023.0.4.x86_64
apr-util-1.6.3-1.amzn2023.0.2.x86_64
apr-util-lmdb-1.6.3-1.amzn2023.0.2.x86_64
apr-util-openssl-1.6.3-1.amzn2023.0.2.x86_64
generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
httpd-2.4.65-1.amzn2023.0.2.x86_64
httpd-core-2.4.65-1.amzn2023.0.2.x86_64
httpd-filesystem-2.4.65-1.amzn2023.0.2.noarch
httpd-tools-2.4.65-1.amzn2023.0.2.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.amzn2023.0.2.x86_64
```

Complete!

- Creating simple webpage
- echo "<h1>This is the PUBLIC Apache Server (Jump Server)</h1>" | sudo tee /var/www/html/index.html

```
Complete!
[ec2-user@ip-10-0-0-197 ~]$ sudo systemctl start httpd
sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-10-0-0-197 ~]$ cd /var/www/html/index.html
-bash: cd: /var/www/html/index.html: No such file or directory
[ec2-user@ip-10-0-0-197 ~]$ echo "<h1>This is the PUBLIC Apache Server (Jump Server)</h1>" | sudo tee /var/www/html/index.html
<h1>This is the PUBLIC Apache Server (Jump Server)</h1>
[ec2-user@ip-10-0-0-197 ~]$ curl localhost
<h1>This is the PUBLIC Apache Server (Jump Server)</h1>
[ec2-user@ip-10-0-0-197 ~]$
```

- Verify



Apache on Jump Server

- Jump to private ip instance

```
[ec2-user@ip-10-0-0-114 ~]$ ssh -i "all-key.pem" ec2-user@10.0.0.197
      #_
      ~\_ #####_          Amazon Linux 2023
      ~~ \#####\
      ~~   \###|
      ~~     \#/,_-->
      ~~       V~'.'-
      ~~~
      ~~.~.
      ~~/_ _/
      _/m/'
```

- Last login: Tue Nov 4 09:19:20 2025 from 10.0.0.114

- Install httpd in private instance also

```
.
Package httpd-2.4.65-1.amzn2023.0.2.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
```

- Complete!

- Create a web page

- echo "<h1>Welcome! This is the PRIVATE Apache Server</h1>" |
sudo tee /var/www/html/index.html
- now from private instance curl public ip so you can see apache
working page

```
[ec2-user@ip-10-0-0-197 ~]$ curl http://35.175.185.207
<h1>Apache on Jump Server</h1>
```

- [ec2-user@ip-10-0-0-197 ~]\$ | █

8. Create one application load balancer and attach it to both EC2 instances.

- Open aws console
- Navigate to ec2 > target groups
- Give name, select target as instance
- Select vpc of existing subnets
- Select public subnet and private subnet

- Now navigate to load balancer
- Create new
- Select application load balancer

Step 1
Create target group
Step 2
Register targets
Step 3
Review and create

Register targets

This is an optional step to create a target group. However, to ensure that your load balancer routes traffic to this target group you must register your targets.

Available instances (2/2)

Instance ID	Name	State	Security groups	Zone
i-01f12a13e564aefbb	autoscaling-private	Running	default	us-east-1a
i-0bc2cb72c8ca6f030	autoscaling-ec2	Running	default	us-east-1a

2 selected

Ports for the selected instances

Ports for routing traffic to the selected instances.

80

1-65535 (separate multiple ports with commas)

[Include as pending below](#)

Select internet facing in scheme

Scheme | [Info](#)

Scheme can't be changed after the load balancer is created.

Internet-facing

- Serves internet-facing traffic.
- Has public IP addresses.
- DNS name resolves to public IPs.
- Requires a public subnet.

Internal

- Serves internal traffic.
- Has private IP addresses.
- DNS name resolves to private IPs.
- Compatible with the **IPv4** and **Dualstack IP** address types.

Load balancer IP address type | [Info](#)

Select the front-end IP address type to assign to the load balancer. The VPC and subnets mapped to this load balancer must include the selected IP address types. Public IPv4 addresses have an additional cost.

IPv4

Includes only IPv4 addresses.

Dualstack

Includes IPv4 and IPv6 addresses.

Dualstack without public IPv4

Includes a public IPv6 address, and private IPv4 and IPv6 addresses. Compatible with **internet-facing** load balancers only.

Network mapping | [Info](#)

- Click [Include as pending below](#)

target group

80
1-65535 (separate multiple ports with commas)

Include as pending below

2 selections are now pending below. Include more or register targets when ready.

Review targets

Targets (2)

Instance ID	Name	Port	State	Security groups	Zone	Private IPv4 address	Subnet ID
i-01f12a13e564aefbb	autoscaling-private	80	Running	default	us-east-1a	10.0.3.19	subnet-0a56bacc97
i-0bc2cb72c8ca6f030	autoscaling-ec2	80	Running	default	us-east-1a	10.0.1.87	subnet-0e890e6d48

2 pending

Cancel Previous Next

- Target created successfully
- Now , create a application nload balancer
- Navigasge to load balancer in ec2
- Select vpc and 2 subnet s from different availability zone

The load balancer routes traffic to targets in the selected subnets, and in accordance with your IP address settings.

VPC | Info

The load balancer will exist and scale within the selected VPC. The selected VPC is also where the load balancer targets must be hosted unless routing to Lambda or on-premises targets, or if using VPC peering. To confirm the VPC for your targets, view [target groups](#).

vpc-0ad3c0b33fecdd285e (autoscaling)
10.0.0.16



IP pools | Info

You can optionally choose to configure an IPAM pool as the preferred source for your load balancers IP addresses. Create or view [Pools](#) in the [Amazon VPC IP Address Manager console](#).

Use IPAM pool for public IPv4 addresses

The IPAM pool you choose will be the preferred source of public IPv4 addresses. If the pool is depleted IPv4 addresses will be assigned by AWS.

Availability Zones and subnets | Info

Select at least two Availability Zones and a subnet for each zone. A load balancer node will be placed in each selected zone and will automatically scale in response to traffic. The load b.

us-east-1a (use1-az4)

Subnet

Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.

subnet-0e890e6d482e183aa
IPv4 subnet CIDR: 10.0.1.0/24

us-east-1b (use1-az6)

Subnet

Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.

subnet-01e8c875b370cd0aa
IPv4 subnet CIDR: 10.0.2.0/24

[EC2](#) > [Load balancers](#) > Create Application Load Balancer

▼ Listener HTTP:80

Protocol	Port
HTTP	80 1-65535

Default action | [Info](#)
The default action is used if no other rules apply. Choose the default action for traffic on this listener.

Routing action

- Forward to target groups
- Redirect to URL
- Return fixed response

Forward to target group | [Info](#)
Choose a target group and specify routing weight or [create target group](#).

Target group

asg Target type: Instance, IPv4 Target stickiness: Off	HTTP	Weight	Percent
	▼	1 0-999	100%

[+ Add target group](#)
You can add up to 4 more target groups.

Target group stickiness | [Info](#)
Enables the load balancer to bind a user's session to a specific target group. To use stickiness the client must support cookies. If you want to bind a user's session to a specific target, turn on the Target

● **autoscaling-alb**

● **Successfully created load balancer: autoscaling-alb**
It might take a few minutes for your load balancer to fully set up and route traffic. Targets will also take a few minutes to complete the registration process and pass initial health checks.

● **Introducing URL rewrite for Application Load Balancer**
Modify host headers and URL paths of incoming requests before they reach your targets. To get started, add a rule to your listener and configure a transform. [Learn more](#)

autoscaling-alb

● **Details**

Load balancer type	Status	VPC	Load balancer IP address type
--------------------	--------	-----	-------------------------------

- **Everything is healthy**

asg

Details

arn:aws:elasticloadbalancing:us-east-1:414691912691:targetgroup/asg/2ec37f88654679a3	Protocol : Port	Protocol version	VPC
Target type Instance	HTTP: 80	HTTP1	vpc-0ad3c0b33fecdd285e
IP address type IPv4	Load balancer autoscaling-alb		
2 Total targets	2 Healthy	0 Unhealthy	0 Unused
	0 Anomalous		0 Initial
			0 Draining

► **Distribution of targets by Availability Zone (AZ)**
Select values in this table to see corresponding filters applied to the Registered targets table below.

-
- **Copy dns name from load balancer**

VPC
[vpc-0ad3c0b33fecd285e](#)

Availability Zones
[subnet-0e890e6d482e183aa](#) us-east-1a (use1-az4)
[subnet-01e8c875b370cd0aa](#) us-east-1b (use1-az6)

Load balancer IP address type
IPv4

Date created
November 4, 2025, 17:02 (UTC+05:30)

DNS name [Info](#)
 autoscaling-alb-1425481021.us-east-1.elb.amazonaws.com (A Record)

(✓) DNS name copied

Resource map Security Monitoring Integrations Attributes Ca >

- Paste in local browser

← → ⌂ △ Not secure autoscaling-alb-1425481021.us-east-1.elb.amazonaws.com

Apache on Public Instance

- It is on load balancer
- 9 . Store application load balancer logs in S3.
- Navigate to s3 bucket
- Create a bucket enable block all ip
- Open bucket and click permission
- Edit and give script

✓ Successfully edited bucket policy.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AWSApplicationLoadBalancerAccessLogs",  
            "Effect": "Allow",  
            "Principal": {  
                "Service": "logdelivery.elasticloadbalancing.amazonaws.com"  
            },  
            "Action": "s3:PutObject",  
            "Resource": "arn:aws:s3:::asg-autoscaling/AWSLogs/414691912691/*"  
        }  
    ]  
}
```

- Now navigate to , ec2 > load balancer,
- Edit attributes
- Click access logs
- Click s3 bucket add existing bucket
- Can monitoring logs in > load balancer > below the attributes

- Auto Scaling
- Can find the logs in s3 buckets

- 10 . Store the VPC flow logs in a CloudWatch log group.
- Navigate to vpc
- Select vpc
- Below find flow logs
- And create and add , destination and time

Flow log settings

Name - optional

autoscaling-flow-logs

Filter

The type of traffic to capture (accepted traffic only, rejected traffic only, or all traffic).

- Accept
- Reject
- All

Maximum aggregation interval | Info

The maximum interval of time during which a flow of packets is captured and aggregated into a flow log record.

- 10 minutes
- 1 minute

Destination

The destination to which to publish the flow log data.

- Send to CloudWatch Logs
- Send to an Amazon S3 bucket
- Send to Amazon Data Firehose in the same account
- Send to Amazon Data Firehose in a different account

- Navigate to cloud watch and create a log group
- And return back to vpc and refresh you can see cloudwatch log group
- Select log group
- Select new role
- And then create

512ec5c0b9daa913



⌚ Successfully created flow log for the following resource:

vpc-0ad3c0b33fecfd285e

fl-0512ec5c0b9daa913 / autoscaling-flow-logs

Active

Details

Flow Log ID
 fl-0512ec5c0b9daa913

Destination Type
cloud-watch-logs

Traffic Type
All

File Format

-

Name
 autoscaling-flow-logs

Destination Name
 autoscaling-logs

Max Aggregation Interval
10 minutes

Hive Compatible Partitions

-

State
 Active

IAM Role
 arn:aws:iam::414691912691:role/service-role/VPCFlowLogs-Cloudwatch-1762258515835

Log Format
Default

Partition Logs

-

Creation Time
Tuesday, November 4, 2025 at
17:50:27 GMT+5:30

Cross Account IAM Role
-

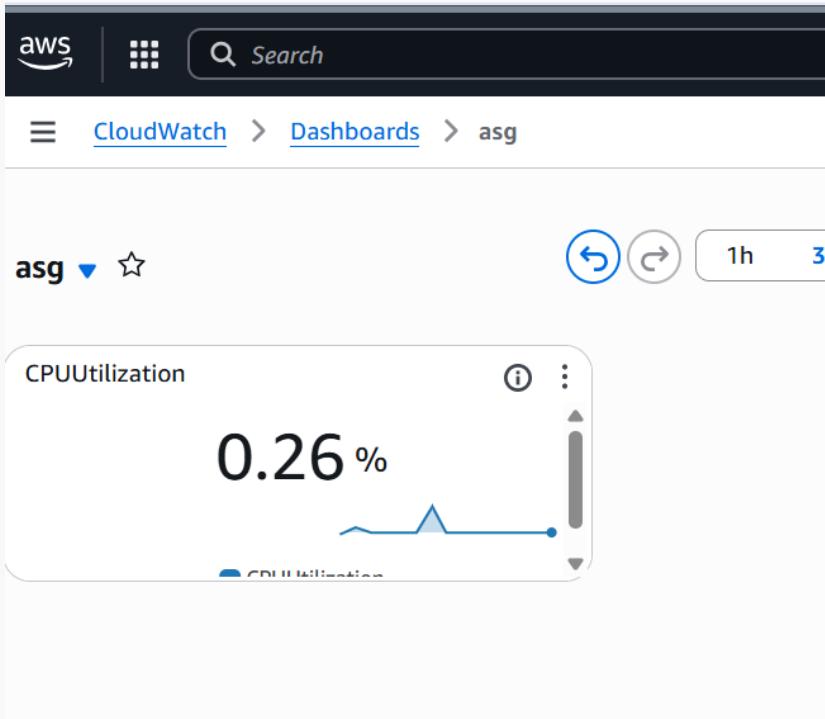
- Verify in cloud watch

11. Create monitoring dashboards to monitor CPU utilization and to monitor the Apache service.

- Cloud watch
 - Create new dashboard
 - Add ec2 in metrics
 - Pre instance select cpu utilization add
 - Add instance id in graphed metrics

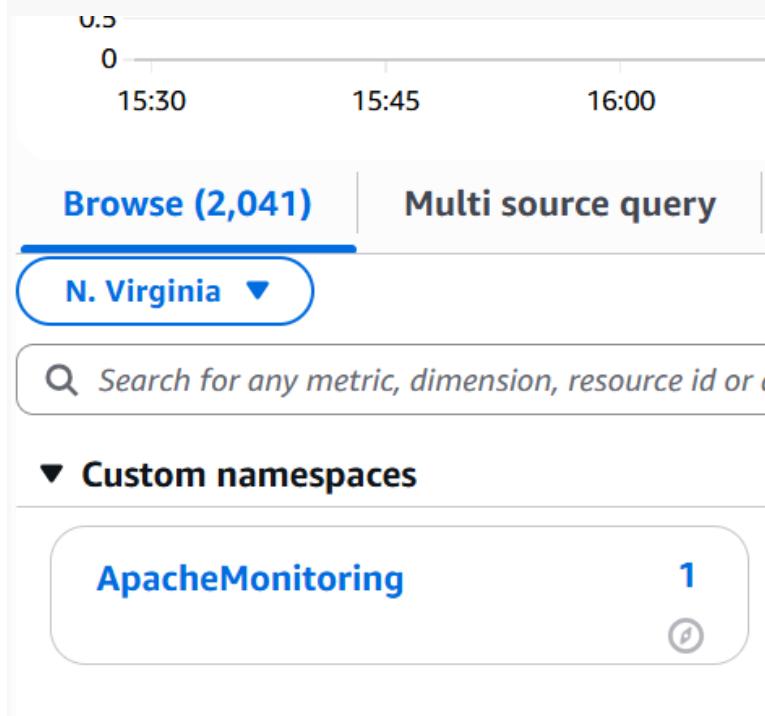
The screenshot shows the AWS CloudWatch Metrics Insights interface. On the left, there's a visualization of CPUUtilization over time, with a value of 0.26% displayed. Below the visualization are tabs for 'Options' and 'Source'. A modal window titled 'Edit details' is open in the center. The modal contains fields for AccountId (414691912691), Region (us-east-1), Namespace (AWS/EC2), Metric name (CPUUtilization), and InstanceId (i-0bc2cb72c8ca6f030). At the bottom of the modal are 'Cancel' and 'Update' buttons. To the right of the modal, there's a 'Show all actions (6)' button. At the very bottom right, there are 'Cancel' and 'Create widget' buttons.

- Created a dashboard, for instance

- 
- Monitor apache is running or not
- Open cli check aws is configured or not
- Now create file and add script for monitor apache running or not
 - #!/bin/bash
 -
 - # Check Apache (httpd) status
 - status=\$(systemctl is-active httpd)
 -
 - if ["\$status" == "active"]; then
 - metric_value=1
 - else
 - metric_value=0
 - fi
 -
 - # Push the metric to CloudWatch
 - aws cloudwatch put-metric-data \
• --metric-name ApacheServiceStatus \
• --namespace "ApacheMonitoring" \
• --value \$metric_value \
• --region us-east-1
 - Install crontab and edit it
 - By sudo crontab -e
 - Add */5 * * * *

- Run this command in CLI to create a custom name space in gui

```
aws cloudwatch put-metric-data \
--metric-name ApacheServiceStatus \
--namespace "ApacheMonitoring" \
--value 1 \
--region us-east-1
```

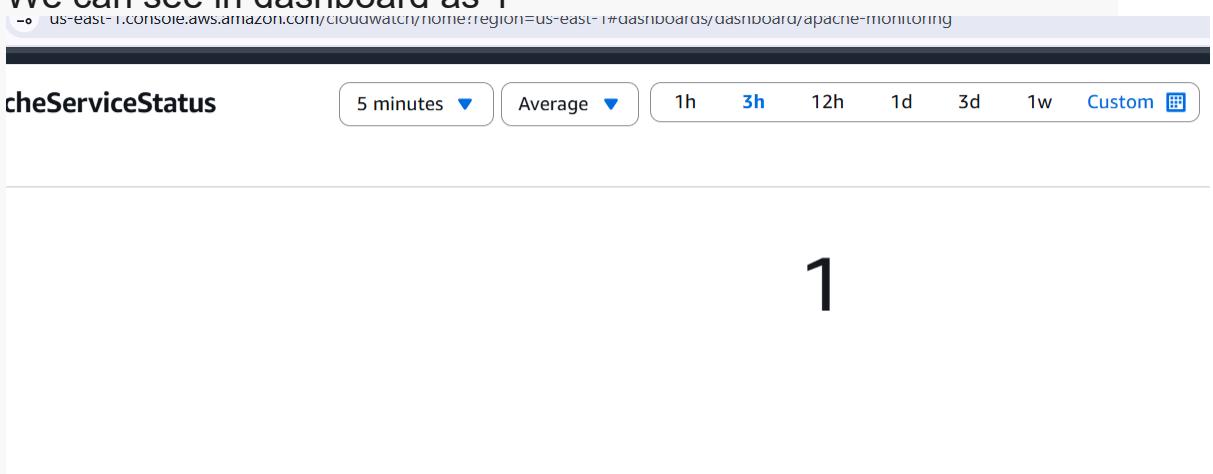


- As apache running in cli
-

```
[ec2-user@ip-10-0-1-87 ~]$ aws cloudwatch put-metric-data \
--metric-name ApacheServiceStatus \
--namespace "ApacheMonitoring" \
--value 1 \
--region us-east-1
[ec2-user@ip-10-0-1-87 ~]$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: Loaded (/usr/lib/systemd/system/httpd.service; enabled; preset:>)
   Active: active (running) since Tue 2025-11-04 11:46:20 UTC; 1h 15min ago
     Docs: man:httpd.service(8)
   Main PID: 25951 (httpd)
      Status: "Total requests: 332; Idle/Busy workers 100/0;Requests/sec: 0.0>
        Tasks: 177 (limit: 1053)
       Memory: 16.2M
         CPU: 4.273s
      CGroup: /system.slice/httpd.service
              ├─25951 /usr/sbin/httpd -DFOREGROUND
              ├─25952 /usr/sbin/httpd -DFOREGROUND
              ├─25954 /usr/sbin/httpd -DFOREGROUND
              ├─25955 /usr/sbin/httpd -DFOREGROUND
              └─25956 /usr/sbin/httpd -DFOREGROUND

Nov 04 11:46:20 ip-10-0-1-87.ec2.internal systemd[1]: Starting httpd.service>
Nov 04 11:46:20 ip-10-0-1-87.ec2.internal systemd[1]: Started httpd.service >
Nov 04 11:46:20 ip-10-0-1-87.ec2.internal httpd[25951]: Server configured, 1>
[lines 1-19/19 (END)]
```

- We can see in dashboard as 1



- 12. If CPU utilization is more than 70%, then it should trigger auto scaling and launch new instance.
 - Open aws console
 - Navigate to launch template
 - Give configuration as needed for autoscaling

▼ Network settings [Info](#)

Subnet | [Info](#)

subnet-0e890e6d482e183aa
VPC: vpc-0ad3c0b33fec285e Owner: 414691912691
Availability Zone: us-east-1a (use1-az4) Zone type: Availability Zone
IP addresses available: 248 CIDR: 10.0.1.0/24

Public-1A-k



[Create new subnet](#)

When you specify a subnet, a network interface is automatically added to your template.

Availability Zone | [Info](#)

us-east-1a

use1-az4



[Enable additional zones](#)

Firewall (security groups) | [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Select existing security group

Create security group

Common security groups | [Info](#)

Select security groups



default sg-0afe7e3379e26dce8

VPC: vpc-0ad3c0b33fec285e

[Compare security group rules](#)

Security groups that you add or remove here will be added to or removed from all your network interfaces

- Now navigate autoscaling and create new
- Add launch template to this
- Add vpc and 2 subnets atleast

For most applications, you can use multiple Availability Zones and let EC2 Auto Scaling default subnets are suitable for getting started quickly.

VPC

Choose the VPC that defines the virtual network for your Auto Scaling group.

vpc-0ad3c0b33fec285e (autoscaling)
10.0.0.0/16



[Create a VPC](#)

Availability Zones and subnets

Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC.

Select Availability Zones and subnets



use1-az4 (us-east-1a) | subnet-0e890e6d482e183aa (Public-1A-k)
10.0.1.0/24

use1-az6 (us-east-1b) | subnet-01e8c875b370cd0aa (public-av-1b-k)
10.0.2.0/24

[Create a subnet](#)

Availability Zone distribution - new

Auto Scaling automatically balances instances across Availability Zones. If launch failures occur in a zone

- Add existing load balancer
-

EC2 > Auto Scaling groups > Create Auto Scaling group

Configure group size and scaling

Step 5 - optional

Add notifications

Step 6 - optional

Add tags

Step 7

Review

Select Load Balancing options

No load balancer
Traffic to your Auto Scaling group will not be fronted by a load balancer.

Attach to an existing load balancer
Choose from your existing load balancers.

Attach to a new load balancer
Quickly create a basic load balancer to attach to your Auto Scaling group.

Attach to an existing load balancer

Select the load balancers to attach

Choose from your load balancer target groups
This option allows you to attach Application, Network, or Gateway Load Balancers.

Choose from Classic Load Balancers

Existing load balancer target groups

Only instance target groups that belong to the same VPC as your Auto Scaling group are available for selection.

Select target groups

asg | HTTP Application Load Balancer: autoscaling-alb

VPC Lattice integration options [Info](#)

roups > Create Auto Scaling group

1

Scaling [Info](#)

You can resize your Auto Scaling group manually or automatically to meet changes in demand.

Scaling limits

Set limits on how much your desired capacity can be increased or decreased.

Min desired capacity	Max desired capacity
<input type="text" value="1"/>	<input type="text" value="1"/>
Equal or less than desired capacity	Equal or greater than desired capacity

Automatic scaling - optional

Choose whether to use a target tracking policy | [Info](#)

You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

No scaling policies
Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand.

Target tracking scaling policy
Choose a CloudWatch metric and target value and let the desired capacity in proportion to the metric's value.

Scaling policy name

Target Tracking Policy

- Load increased in monitor is working

PUUTI 5 minutes ▾ Average ▾ 3h 1d 1w 3m Local timezone ▾ X

C 1 minute ▾

78.3 %

```
[36] 31271 [ec2-user@ip-10-0-1-87 ~]$ yes > /dev/null &
[37] 31298 [ec2-user@ip-10-0-1-87 ~]$ while :; do ; done &
while :; do ; done &
[38] 31350 [39] 31351 [ec2-user@ip-10-0-1-87 ~]$ while :; do ; done &
[40] 31352 [ec2-user@ip-10-0-1-87 ~]$ echo $((RANDOM % 900000 +
117669 [ec2-user@ip-10-0-1-87 ~]$ while true; do
echo "Your OTP is: $((RANDOM % 900000 + 100000))"
sleep 5
done
Your OTP is: 103165
Your OTP is: 112376
Your OTP is: 125217
Your OTP is: 122222
Your OTP is: 109721
Your OTP is: 127571
Your OTP is: 109969
Your OTP is: 107523
Your OTP is: 130880
```

- So new instance launched automatically as its cpu usage increased above 50%

Last updated less than a minute ago

Connect Instance state Actions Launch instances

Find Instance by attribute or tag (case-sensitive)

Running

1

Name Instance ID Instance state Instance type Status check Alarm status Availability Zone

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
autoscaling-ec2	i-0bc2cb72c8ca6f030	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a
autoscaling-pr...	i-01f12a13e564aefbb	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a
	i-058024f967ab29267	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a