

PROJECT ASSIGNMENT

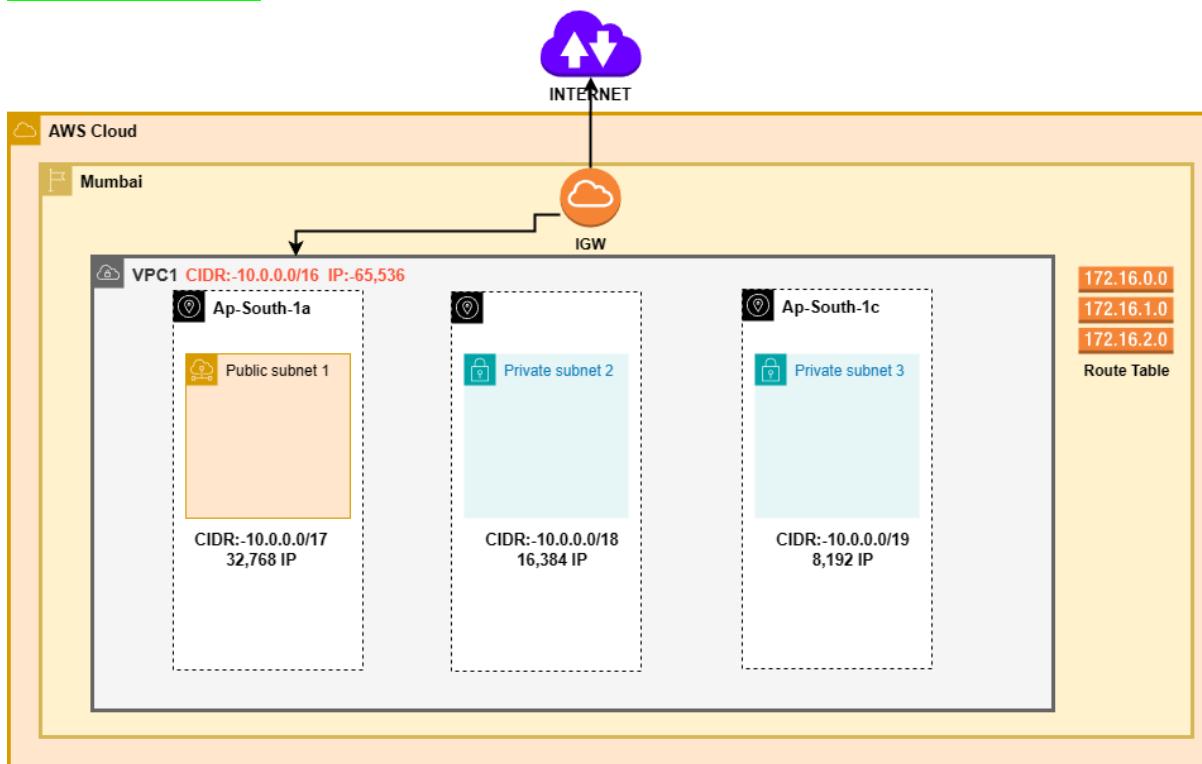
BATCH ID: UA003AWS

Name: -Deepak Patra

Ref No: - UA/BLR/000013

1. Create a VPC, e.g., VPC1, and create three subnets within it: one that is half the size and two that are quarter-sized. The larger subnet should be public, while the other two should be private.

Architecture: -



VPC Screen: -

Screenshot of the AWS VPC console showing the details of VPC-1 (vpc-06c20dde071c8c58a). The VPC is in an **Available** state with a CIDR of 10.0.0.0/16. It has three subnets: ap-South-1a, ap-South-1b, and ap-South-1c. The Route Table contains route entries for each subnet. The Network connections section shows a connection to test-igw.

Details	Info
VPC ID	vpc-06c20dde071c8c58a
Tenancy	Default
Default VPC	No
Network Address Usage metrics	Disabled
State	Available
DHCP option set	dopt-026a891be098a94e1
IPv4 CIDR	10.0.0.0/16
Route 53 Resolver DNS Firewall rule groups	-
DNS hostnames	Disabled
Main route table	rtb-05545043e263ee6b1
IPv6 pool	-
Owner ID	751762612495
DNS resolution	Enabled
Main network ACL	acl-001dc627059bee6b8
IPv6 CIDR (Network border group)	-

Subnet Screen: -

Subnets Screen:

The screenshot shows the AWS VPC Subnets page. On the left, a sidebar lists various VPC-related services like Route tables, Internet gateways, and Security. The main area displays a table of subnets with columns for Name, Subnet ID, State, VPC, and IPv4 CIDR. Three subnets are listed: 'Public-subnet' (subnet-08b92b7216a060d2f), 'Private-subnet-1' (subnet-0fadcb64d129bf03a), and 'Private-subnet-2' (subnet-0fe2b0523d6ee8844). All are in an 'Available' state and belong to VPC-1.

RTB Screen:

The screenshot shows the AWS VPC Route Table Details page for route-table-0df9cbd7c6384ab5f. The sidebar shows the same VPC-related services as the previous screen. The main content shows the details of the route table, including its ID (rtb-0df9cbd7c6384ab5f), VPC (vpc-0df802af8a90f5f7b | VPC-1), and associations. It also lists two routes: one to an endpoint (igw-0407894a559f01f43) and one to 'local'.

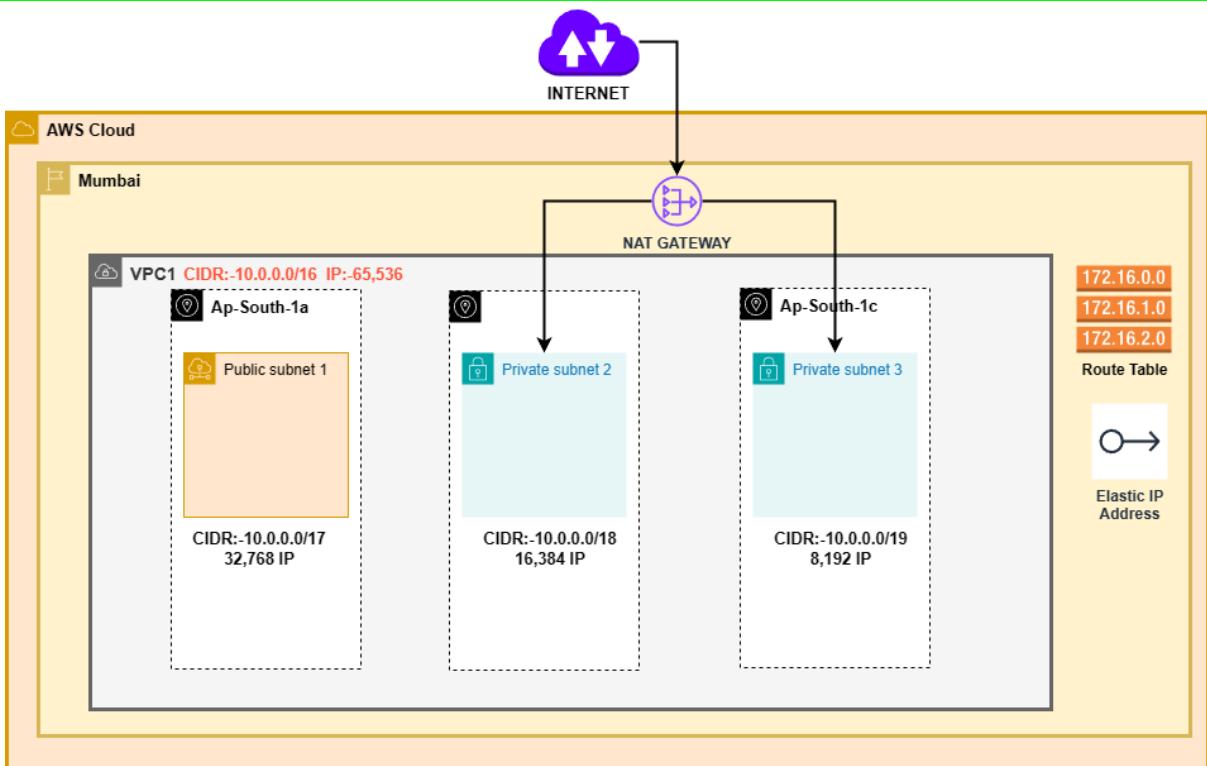
IGW Screen:

The screenshot shows the AWS VPC Internet Gateway Details page for igw-09bc18435ff3b7a14. The sidebar includes the VPC-related services. The main area shows the details of the internet gateway, including its ID (igw-09bc18435ff3b7a14), VPC ID (vpc-06c20dde071c8c58a | VPC-1), and owner information. A 'Tags' section shows a single tag named 'test-igw'.

VPC RESOURCES MAP:

Screenshot of the AWS VPC Console showing the details of a VPC named 'VPC-1'. The VPC has an IPv4 CIDR of 10.0.0.0/16 and an IPv6 pool. It contains three subnets: 'ap-south-1a' (Public-subnet), 'ap-south-1b' (Private-subnet-1), and 'ap-south-1c' (Private-subnet-2). There are two route tables: 'test-route' (with route IDs rtb-0f2b077f9907081cc) and 'rtb-0f2b077f9907081cc'. A single network connection 'test-igw' is listed.

2. Establish a NAT gateway connection for subnet 2 and subnet 3 in VPC1.



Resources map

The screenshot shows the AWS VPC Resource Map interface. It displays a network diagram with the following components and their associations:

- VPC:** VPC-1 (Your AWS virtual network)
- Subnets (3):** ap-south-1a (Public-subnet), ap-south-1b (Private-subnet-1), ap-south-1c (Private-subnet-2)
- Route tables (3):** test-route (rtb-0f2b077f9907081cc), private-route (rtb-0613642633675f652)
- Network connections (2):** test-igw, test-nat-gateway

Associations shown in the diagram:

- ap-south-1a is associated with test-route
- ap-south-1b is associated with private-route
- ap-south-1c is associated with private-route
- test-route is associated with test-igw and test-nat-gateway
- private-route is associated with test-nat-gateway

RTB SCREEN

The screenshot shows the AWS Route Tables page. The table lists four route tables, with one selected:

Name	Route Table ID	Explicit subnet assoc...	Edge associations	Main	VPC
-	rtb-0f2b077f9907081cc	-	-	Yes	vpc-0df802af8a90f5f7b
-	rtb-042548ef6e084036	-	-	Yes	vpc-0eae1957345edce4c
test-route	rtb-0df9cd7c6384ab5f	subnet-0afff561f041ca7...	-	No	vpc-0df802af8a90f5f7b
private-route	rtb-0613642633675f652	2 subnets	-	No	vpc-0df802af8a90f5f7b

The selected route table, **rtb-0613642633675f652 / private-route**, is shown in detail:

- Details:** Route table ID: rtb-0613642633675f652, Main: No, Explicit subnet associations: 2 subnets.
- Routes:** (not visible in the screenshot)
- Subnet associations:** (not visible in the screenshot)
- Edge associations:** (not visible in the screenshot)
- Route propagation:** (not visible in the screenshot)
- Tags:** (not visible in the screenshot)

NAT SCREEN

The screenshot shows the AWS NAT Gateways page. The table lists one NAT gateway, with one selected:

NAT gateway ID	Connectivity type	State	State message
nat-0df1ad53b94a93dc1	Public	Available	Info

The selected NAT gateway, **nat-0df1ad53b94a93dc1 / test-nat-gateway**, is shown in detail:

- Details:** NAT gateway ID: nat-0df1ad53b94a93dc1, Connectivity type: Public, State: Available, State message: Info.
- Secondary IPv4 addresses:** (not visible in the screenshot)
- Monitoring:** (not visible in the screenshot)
- Tags:** (not visible in the screenshot)

Below the details, it says: "Secondary IPv4 addresses are not available for this nat gateway."

EIP SCREEN

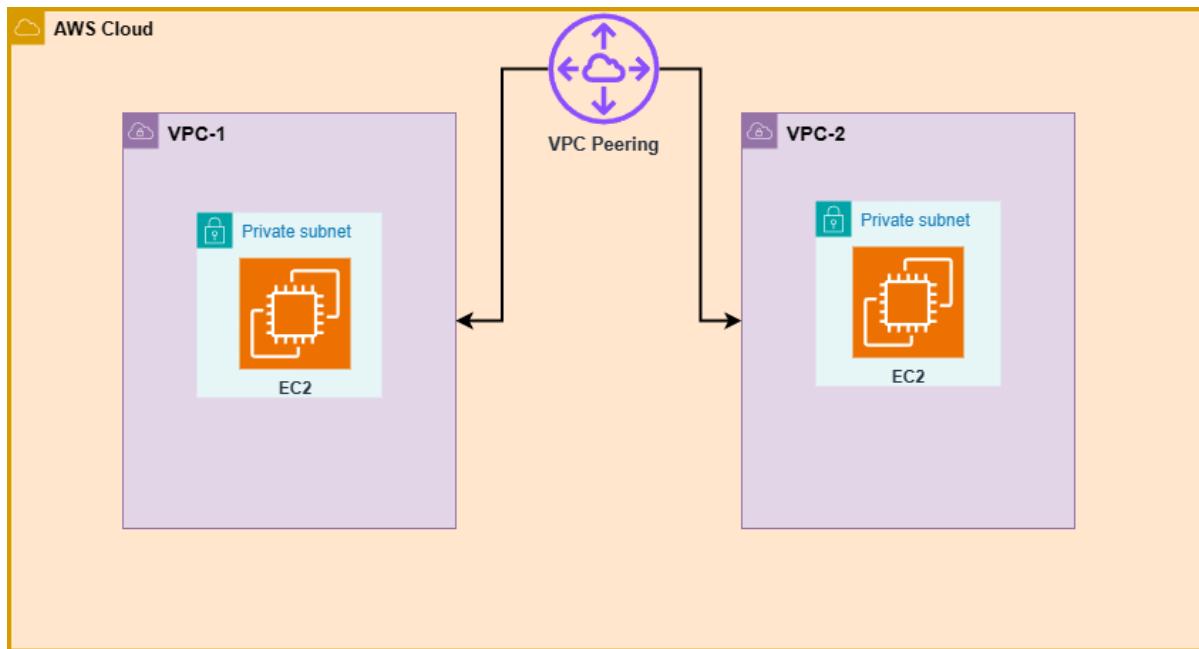
The screenshot shows the AWS VPC console with the path: VPC > Elastic IP addresses > 13.200.58.115. The main view displays the summary of the assigned EIP. Key details include:

Allocated IPv4 address	Type	Allocation ID	Reverse DNS record
13.200.58.115	Public IP	eipalloc-0e53974430460d82	-
Association ID	Scope	Associated instance ID	Private IP address
eipassoc-0f04fc1cf2e47a44	VPC	-	10.0.176.58
Network interface ID	Network interface owner account ID	Public DNS	NAT Gateway ID
eni-0b9ead0203ae59ada	751762612495	-	nat-0df1ad5b94a93dc1 (test-nat-gateway)
Address pool	Network border group		
Amazon	ap-south-1		

Below the summary, there is a section for Tags (1) with one entry: Name (EIP). The bottom of the page includes standard AWS navigation links and a footer with copyright information.

3. Create another VPC, e.g., VPC2, and set up one subnet within it without internet connectivity. Connect to the Windows server running in the VPC2 subnet using a Peering connection from Subnet 2 of VPC1.

Architecture: -



PCX Screen

UA003AWS_PROJECT-ASSIGNMENT | PeerConnectionDetails | VPC | +

ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#PeeringConnectionDetails:peeringConnectionId=pxc-01e94046a089d6e44

aws Services Search [Alt+S]

Your VPC peering connection (pxc-01e94046a089d6e44) has been established. To send and receive traffic across this VPC peering connection, you must add a route to the peered VPC in one or more of your VPC route tables.

Info

VPC > Peering connections > pxc-01e94046a089d6e44

pxc-01e94046a089d6e44 / VPC-peering-connection-between-vpc-1-and-vpc-2 Actions ▾

Details **Info**

Requester owner ID 751762612495	Acceptor owner ID 751762612495	VPC Peering connection ARN arn:aws:ec2:ap-south-1:751762612495:vpc-peering-connection/pxc-01e94046a089d6e44
Peering connection ID pxc-01e94046a089d6e44	Requester VPC vpc-0e18655f7c527dfe9 / VPC-2	Acceptor VPC vpc-0dc06971a87a1abc9 / VPC-1
Status Active	Requester CIDRs 16.0.0.0/16	Acceptor CIDRs 10.0.0.0/16
Expiration time -	Requester Region Mumbai (ap-south-1)	Acceptor Region Mumbai (ap-south-1)

DNS Route tables Tags

Edit DNS settings

https://ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#vpcs.VpcId=vpc-0e18655f7c527dfe9

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

ENG IN 8:28 PM 3/14/2024

RTB-1 Screen

UA003AWS_PROJECT-ASSIGNMENT | RouteTableDetails | VPC Console | VpcDetails | VPC Console | +

ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#RouteTableDetails:RouteTableId=rtb-0acf4eb443811ac18

aws Services Search [Alt+S]

VPC > Route tables > rtb-0acf4eb443811ac18

rtb-0acf4eb443811ac18 / route-table-vpc-1 Actions ▾

Details **Info**

Route table ID rtb-0acf4eb443811ac18	Main No	Explicit subnet associations subnet-0ce6bbba70f4c087 / VPC-1-subnet-1	Edge associations -
VPC vpc-0dc06971a87a1abc9 VPC-1	Owner ID 751762612495		

Routes Subnet associations Edge associations Route propagation Tags

Routes (2)

Destination	Target	Status	Propagated
10.0.0.0/16	local	Active	No
16.0.0.0/16	pxc-01e94046a089d6e44	Active	No

Filter routes Both Edit routes

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

ENG IN 8:34 PM 3/14/2024

RTB-2-Screen

UA003AWS_PROJECT-ASSIGNMENT | RouteTableDetails | VPC Console | VpcDetails | VPC Console | +

ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#RouteTableDetails:RouteTableId=rtb-09c3584b3c22ab6bf

aws Services Search [Alt+S]

VPC > Route tables > rtb-09c3584b3c22ab6bf

rtb-09c3584b3c22ab6bf / route-table-vpc-2 Actions ▾

Details **Info**

Route table ID rtb-09c3584b3c22ab6bf	Main No	Explicit subnet associations subnet-07ee6862ad386872b / VPC-2-subnet-2	Edge associations -
VPC vpc-0e18655f7c527dfe9 VPC-2	Owner ID 751762612495		

Routes Subnet associations Edge associations Route propagation Tags

Routes (2)

Destination	Target	Status	Propagated
10.0.0.0/16	pxc-01e94046a089d6e44	Active	No
16.0.0.0/16	local	Active	No

Filter routes Both Edit routes

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

ENG IN 8:35 PM 3/14/2024

Task: Network Security

1. Launch two instances: one with Windows and the other with Amazon Linux 2.

Linux: -

The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays a table titled "Instances (1) Info" with one row. The row details an instance named "Amazon-Linux-2" with instance ID "i-08813ce981c7fd20a", which is currently "Running". Other columns include Instance state, Instance type (t2.micro), Status check (2/2 checks passed), Alarm status, Availability Zone (ap-south-1a), and Public IPv4 (ec2-13-233-). A search bar at the top allows filtering by instance name or tag. A "Launch instances" button is visible in the top right corner.

The screenshot shows the AWS EC2 Instance details page for the instance "Amazon-Linux-2". The left sidebar is collapsed. The main area is titled "Instance details" and contains various configuration settings. Key details include:

- Platform: Amazon Linux (Inferred)
- AMI ID: ami-013168dc3850ef002
- AMI name: al2023-ami-2023.3.20240312.0-kernel-6.1-x86_64
- Launch time: Sun Mar 17 2024 01:06:55 GMT+0530 (India Standard Time) (7 minutes)
- Stop protection: Disabled
- Instance auto-recovery: Default
- AMI Launch index: 0
- Credit specification: standard
- Usage operation: RunInstances
- Enclaves Support: -
- Allow tags in instance metadata: Disabled
- Key pair assigned at launch: test
- Kernel ID: -
- RAM disk ID: -
- Boot mode: uefi-preferred
- Use RBN as guest OS hostname: Disabled
- Monitoring: disabled
- Termination protection: Disabled
- AMI location: amazon/al2023-ami-2023.3.20240312.0-kernel-6.1-x86_64
- Stop-hibernate behavior: Disabled
- State transition reason: -
- State transition message: -
- Owner: 751762612495
- Current instance boot mode: legacy-bios
- Answer RBN DNS hostname IPv4: Disabled

Windows: -

EC2 Dashboard | EC2 Global View | Instances | Instance Types | Launch Templates | Spot Requests | Savings Plans | Reserved Instances | Dedicated Hosts | Capacity Reservations | New | Images | AMIs | AMI Catalog | Elastic Block Store | Volumes | Snapshots | Lifecycle Manager

Instance details

Platform	AMI ID	Monitoring
Windows	ami-09b9e25b6db1d130c	disabled
Platform details	AMI name	Termination protection
Windows	Windows_Server-2022-English-Full-Base-2024.02.14	Disabled
Stop protection	Launch time	AMI location
Disabled	Sun Mar 17 2024 01:20:58 GMT+0530 (India Standard Time) (less than a minute)	amazon/Windows_Server-2022-English-Full-Base-2024.02.14
Instance auto-recovery	Lifecycle	Stop-hibernate behavior
Default	normal	Disabled
AMI Launch index	Key pair assigned at launch	State transition reason
0	test	-
Credit specification	Kernel ID	State transition message
standard	-	-
Usage operation	RAM disk ID	Owner
RunInstances0002	-	751762612495
Enclaves Support	Boot mode	Current instance boot mode
-	-	legacy-bios
Allow tags in instance metadata	Use RBN as guest OS hostname	Answer RBN DNS hostname IPv4
Disabled	Disabled	Disabled
Host and placement group		
Host ID	Affinity	Placement group
-	-	-

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 121 AM 3/17/2024

2. Install an httpd web server on the Amazon Linux 2 instance.

```
[ec2-user@ip-10-0-5-139 ~]$ sudo yum install httpd
Last metadata expiration check: 0:01:10 ago on Fri Mar 22 03:09:56 2024.
Dependencies resolved.

=====
Package           Architecture Version       Repository   Size
=====
Installing:
httpd             x86_64      2.4.58-1.amzn2023 amazonlinux 47 k
Installing dependencies:
apr               x86_64      1.7.2-2.amzn2023.0.2   amazonlinux 129 k
apr-util          x86_64      1.6.3-1.amzn2023.0.1   amazonlinux 98 k
generic-logos-httpd noarch     18.0.0-12.amzn2023.0.3  amazonlinux 19 k
httpd-core        x86_64      2.4.58-1.amzn2023    amazonlinux 1.4 M
httpd-filesystem noarch     2.4.58-1.amzn2023    amazonlinux 14 k
httpd-tools       x86_64      2.4.58-1.amzn2023    amazonlinux 81 k
libbrotli         x86_64      1.0.9-4.amzn2023.0.2  amazonlinux 315 k
mailcap           noarch     2.1.49-3.amzn2023.0.3 amazonlinux 33 k
Installing weak dependencies:
apr-util-openssl x86_64      1.6.3-1.amzn2023.0.1   amazonlinux 17 k
mod_http2         x86_64      2.0.11-2.amzn2023    amazonlinux 150 k
mod_lua           x86_64      2.4.58-1.amzn2023    amazonlinux 61 k

Transaction Summary
=====
Install 12 Packages

Total download size: 2.3 M
Installed size: 6.9 M
Is this ok [y/N]: y
Downloaded packages:
(1/12): httpd-tools-2.4.58-1.amzn2023.x86_64.rpm 1.3 MB/s | 81 kB 00:00
(2/12): mod_lua-2.4.58-1.amzn2023.x86_64.rpm 523 kB/s | 61 kB 00:00
(3/12): apr-1.7.2-2.amzn2023.0.2.x86_64.rpm 1.8 MB/s | 129 kB 00:00
(4/12): httpd-2.4.58-1.amzn2023.x86_64.rpm 3.3 MB/s | 47 kB 00:00
(5/12): apr-util-1.6.3-1.amzn2023.0.1.x86_64.rpm 6.2 MB/s | 98 kB 00:00
(6/12): mod_http2-2.0.11-2.amzn2023.x86_64.rpm 9.1 MB/s | 150 kB 00:00
(7/12): apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64.rpm 1.3 MB/s | 17 kB 00:00
```

3. Log in to the Windows machine and check if the httpd server is running by sending requests to the IP address of the Linux server through a browser and ensure that the proper ports are opened in the security groups.

Way to follow: -

1. Log in to the Windows Machine:

- Access the AWS Management Console.
- Navigate to the EC2 service.
- Locate and select the Windows EC2 instance you want to log in to.
- Click on the "Connect" button.
- Follow the instructions provided to connect to the Windows instance using Remote Desktop Protocol (RDP).

2. Check if the HTTPD Server is Running on the Linux Server request to windows:

- Once logged in to the Windows instance, open a web browser (e.g., Internet Explorer, Google Chrome).
- Enter the IP address of the Linux server in the address bar of the browser.
- If the HTTPD server is running on the Linux server and accessible from the Windows machine, you should see the default Apache web page or any other content hosted by the server.

3. Ensure Proper Ports are Opened in Security Groups:

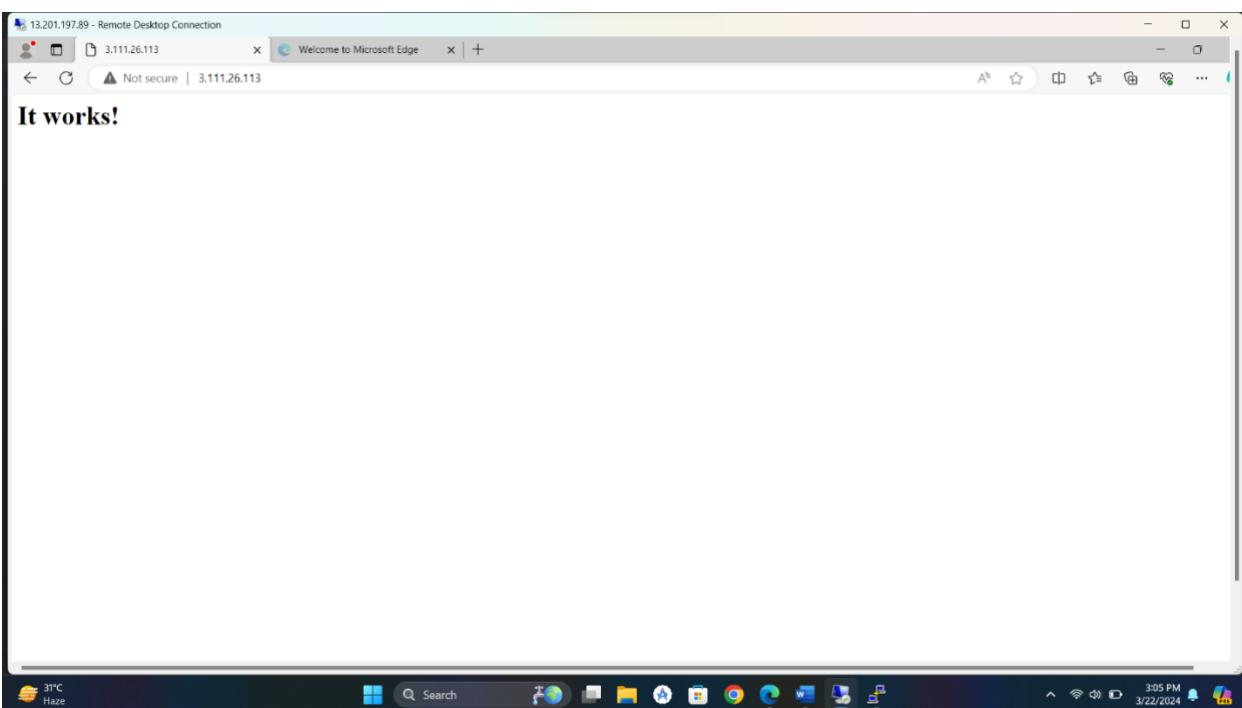
- Go to the AWS Management Console.
- Navigate to the EC2 service.
- Click on "Security Groups" in the navigation pane.
- Find and select the security group associated with your Linux EC2 instance.
- Click on the "Inbound Rules" tab.
- Ensure that port 80 (HTTP) is open for incoming traffic from the Windows instance's IP address or from any source (0.0.0.0/0) if you want to allow access from any location.
- If port 80 is not open, click on "Edit inbound rules" and add a new rule allowing HTTP traffic (port 80) from the desired source.

Httpd server installed on Linux

```
ec2-user@ip-10-0-4-159:~$ rpm -q httpd
httpd-2.4.58-1.amzn2023.x86_64
ec2-user@ip-10-0-4-159:~$ curl http://127.0.0.1
curl: (7) Failed connect to 127.0.0.1:80; Connection refused
ec2-user@ip-10-0-4-159:~$
```

```
[ec2-user@ip-10-0-4-159 ~]$ cd /var/www/html
[ec2-user@ip-10-0-4-159 html]$ curl http://127.0.0.1
curl: (7) Failed connect to 127.0.0.1:80; Connection refused
[ec2-user@ip-10-0-4-159 html]$
```

Httpd server successfully sending requests to Windows Browser,



Proper port 80 are opened on Amazon Linux Instances

The screenshot shows the AWS Management Console interface for managing security groups. The URL is ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#ModifyInboundSecurityGroupRules:securityGroupId=sg-08df2b2cc55f7cf54. The page title is "Edit inbound rules". It displays two security group rules:

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-0a32c42fa2d1c8bb5	HTTP	TCP	80	Custom	0.0.0.0/0
sgr-06f876afa13c41b5a	SSH	TCP	22	Custom	0.0.0.0/0

A warning message at the bottom states: "⚠ 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." There are buttons for "Cancel", "Preview changes", and "Save rules".

Task: EC2, Snapshot

1. Launch an EC2 instance and install the Tomcat server on it.

Path to be followed:

- Log in to the AWS Management Console.
- Navigate to the EC2 dashboard.
- Click on "Launch Instance".
- Choose an Amazon Machine Image (AMI) based on your requirements (e.g., Amazon Linux, Ubuntu, etc.).
- Select an instance type.
- Configure instance details (e.g., network, subnet, security group).
- Add storage as needed.
- Add tags if required.

- Configure security groups to allow inbound traffic on port 8080 (default Tomcat port).
- Review and launch the instance.
- Connect to your instance using SSH.
- Install Tomcat using appropriate package management commands based on the chosen Linux distribution.
- Use wget command with Apache Tomcats Zip link for downloading Tomcat.
- Yum install java -y (install java because building Tomcat it requires JDK).

Apache Tomcat

```

ec2-user@ip-10-0-12-220:~$ inflating: apache-tomcat-10.1.19/webapps/examples/jsp/xml/xml.jsp.html
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/cookies.html
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/helloworld.html
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/images/code.gif
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/images/execute.gif
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/images/icon/return.gif
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/images/icon/icon.html
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/images/icon/icon.htm
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/nonblocking/bytectorner.html
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/reheaders.html
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/reinfo.html
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/reparams.html
inflating: apache-tomcat-10.1.19/webapps/examples/servlets/sessions.html
inflating: apache-tomcat-10.1.19/webapps/examples/websocket/chat.xhtml
inflating: apache-tomcat-10.1.19/webapps/examples/websocket/drawboard.xhtml
inflating: apache-tomcat-10.1.19/webapps/examples/websocket/echo.xhtml
inflating: apache-tomcat-10.1.19/webapps/examples/websocket/index.xhtml
inflating: apache-tomcat-10.1.19/webapps/examples/websocket/snake.xhtml
inflating: apache-tomcat-10.1.19/webapps/host-manager/META-INF/context.xml
inflating: apache-tomcat-10.1.19/webapps/host-manager/META-INF/jsp/401.jsp
inflating: apache-tomcat-10.1.19/webapps/host-manager/META-INF/jsp/404.jsp
inflating: apache-tomcat-10.1.19/webapps/host-manager/META-INF/jsp/4044.jsp
inflating: apache-tomcat-10.1.19/webapps/host-manager/META-INF/manager.xml
inflating: apache-tomcat-10.1.19/webapps/host-manager/META-INF/web.xml
inflating: apache-tomcat-10.1.19/webapps/host-manager/css/manager.css
inflating: apache-tomcat-10.1.19/webapps/host-manager/images/asf-logo.svg
inflating: apache-tomcat-10.1.19/webapps/host-manager/images/tomcat.svg
inflating: apache-tomcat-10.1.19/webapps/host-manager/index.jsp
inflating: apache-tomcat-10.1.19/webapps/manager/META-INF/context.xml
inflating: apache-tomcat-10.1.19/webapps/manager/META-INF/jsp/401.jsp
inflating: apache-tomcat-10.1.19/webapps/manager/META-INF/jsp/403.jsp
inflating: apache-tomcat-10.1.19/webapps/manager/META-INF/jsp/404.jsp
inflating: apache-tomcat-10.1.19/webapps/manager/META-INF/jsp/connectorCerts.jsp
inflating: apache-tomcat-10.1.19/webapps/manager/META-INF/jsp/connectorIphost.jsp
inflating: apache-tomcat-10.1.19/webapps/manager/META-INF/jsp/connectorTrustedCerts.jsp
inflating: apache-tomcat-10.1.19/webapps/manager/META-INF/jsp/sessionDetail.jsp
inflating: apache-tomcat-10.1.19/webapps/manager/META-INF/jsp/sessionsList.jsp
inflating: apache-tomcat-10.1.19/webapps/manager/META-INF/web.xml
inflating: apache-tomcat-10.1.19/webapps/manager/css/manager.css
inflating: apache-tomcat-10.1.19/webapps/manager/images/asf-logo.svg
inflating: apache-tomcat-10.1.19/webapps/manager/images/tomcat.svg
inflating: apache-tomcat-10.1.19/webapps/manager/index.jsp
inflating: apache-tomcat-10.1.19/webapps/manager/status.xsd
inflating: apache-tomcat-10.1.19/webapps/manager/xform.xsl
[ec2-user@ip-10-0-12-220 ~]$ apache-tomcat-10.1.19.zip
[ec2-user@ip-10-0-12-220 ~]$ rm -rf apache-tomcat-10.1.19.zip
[ec2-user@ip-10-0-12-220 ~]$ apache-tomcat-10.1.19
[ec2-user@ip-10-0-12-220 ~]$ 

```

2. Create a volume for the instance and take a snapshot.

Path to be followed:

- Navigate to the EC2 dashboard.
- Under "Elastic Block Store", click on "Volumes".
- Click on "Create Volume".
- Configure the volume settings (size, availability zone).
- Once created, select the volume, and click on "Actions" -> "Create Snapshot".
- Give the snapshot a name and description.
- Confirm and create the snapshot.

Volume

The screenshot shows the AWS EC2 Volumes page. A green success message at the top states: "Successfully created snapshot snap-087479081e58e88cf from volume vol-0225953b94f82053c. If you need your snapshot to be immediately available consider using Fast Snapshot Restore." Below this, a table lists one volume: "demo-volume" with Volume ID "vol-0225953b94f82053c". The volume is of type "gp3", size "20 GiB", IOPS "3000", Throughput "125", and was created on "2024/03/24 08:07 GMT+5....".

Snapshot

The screenshot shows the AWS EC2 Snapshots page. A table lists one snapshot: "demo-snapshot" with Snapshot ID "snap-087479081e58e88cf". The snapshot is of size "20 GiB", description "demo-snapshot", storage tier "Standard", and status "Completed". It was started on "Sun Mar 24 2024 08:08:51 GMT+0530 (India Standard Time)".

3. Create and attach the volume through the snapshot to another instance launched in a different availability zone but within the same security group.

Path to be followed:

- Navigate to the EC2 dashboard.
- Under "Elastic Block Store", click on "Snapshots".
- Select the snapshot you created in the previous step.
- Click on "Actions" -> "Create Volume".
- Configure the volume settings (e.g., size, availability zone).
- Once created, select the new volume and click on "Actions" -> "Attach Volume".
- Choose the instance to attach the volume to.

- SSH into the instance and mount the attached volume.(doubt)

Volume Attached

The screenshot shows the AWS EC2 Volumes page with a single volume listed:

vol-0957da692d85011a1 (new-snap-volume)			
Volume ID vol-0957da692d85011a1 (new-snap-volume)	Size 20 GiB	Type gp3	Volume status Insufficient data
AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. Learn more	Volume state In-use	IOPS 3000	Throughput 125
Encryption Encrypted	KMS key ID 569227e2-7a3a-4c14-8346-35b51b45a08b	KMS key alias aws/ebs	KMS key ARN arn:aws:kms:ap-south-1:751762612495:key/569227e2-7a3a-4c14-8346-35b51b45a08b
Fast snapshot restored No	Snapshot snap-087479081e58e88cf	Availability Zone ap-south-1a	Created Sun Mar 24 2024 08:14:16 GMT+0530 (India Standard Time)
Multi-Attach enabled No	Attached resources i-0dbf6e73978cf215b (demo-Ec2); /dev/sdb (attached)	Outposts ARN -	

4. Create an AMI backup of that instance.

Path to be followed:

- Navigate to the EC2 dashboard.
- Select the instance you want to create an AMI backup of.
- Click on "Actions" -> "Image and templates" -> "Create image".
- Provide a name and description for the AMI.
- Configure any additional settings if needed.
- Confirm and create the AMI backup.

AMI

The screenshot shows the AWS EC2 AMIs page with one AMI listed:

Image summary for ami-05f4a2c469c60a3d6 (AMI)			
AMI ID ami-05f4a2c469c60a3d6 (AMI)	Image type machine	Platform details Linux/UNIX	Root device type EBS
AMI name demo-img-ami	Owner account ID 751762612495	Architecture x86_64	Usage operation RunInstances
Root device name /dev/xvda	Status Pending	Source 751762612495/demo-img-ami	Virtualization type hvm
Boot mode uefi-preferred	State reason -	Creation date Sun Mar 24 2024 08:29:07 GMT+0530 (India Standard Time)	Kernel ID -
Description This is Demo AMI	Product codes -	RAM disk ID -	Deprecation time -
Last launched time -	Block devices /dev/xvda=8:true:gp3 /dev/sdb=snap-087479081e58e88cf:20:false:gp3:encrypted		

EBS Storage Problem Statement:

Scenario: You work for XYZ Corporation, and your corporation wants to launch a new web-based application using AWS Virtual Machines. Configure the resources accordingly with appropriate storage for the tasks.

Answer:

To configure AWS resources for launching a new web-based application for XYZ Corporation using Virtual Machines, we'll need to consider various components such as Virtual Machines, storage, networking, and security. Below is a basic setup to get started:

1. EC2 Instances (Virtual Machines):

- First, I Determine the appropriate EC2 instance types based on the application's requirements (e.g., CPU, memory, network performance).
- Then I Choose the suitable operating system for the instances (e.g., Amazon Linux, Ubuntu, Windows).
- Then I Configure the number of instances required for scalability and redundancy.
- I will definitely Ensure that the instances are launched in the customers desired AWS region for optimal performance and compliance.

Storage:

- I Utilize Amazon Elastic Block Store (EBS) for block storage that can be attached to EC2 instances.
- Then I will Determine the size and type (e.g., General Purpose SSD, Provisioned IOPS SSD) of EBS volumes based on the application's storage requirements and performance needs.
- I will Consider utilizing Amazon Elastic File System (EFS) for scalable and elastic file storage if the application requires shared file storage across multiple instances.
- I need to Set up snapshots for EBS volumes to enable backup and disaster recovery capabilities.

Networking:

- I need to Configure Virtual Private Cloud (VPC) to launch EC2 instances in an isolated network environment.
- I Define subnets within the VPC to logically segment resources.
- -Then I Set up security groups to control inbound and outbound traffic to EC2 instances.
- Then I Configure Network Access Control Lists (NACLs) for additional network-level security.
- Then I Assign Elastic IP addresses (EIPs) to EC2 instances for static public IP addresses if needed.
- Then I Implement AWS CloudFront for content delivery and caching to improve latency and global reach.

Security:

- I Utilize AWS Identity and Access Management (IAM) to manage user access and permissions.

Monitoring and Logging:

- I need to Set up Amazon CloudWatch to monitor EC2 instances, EBS volumes, and other AWS services for metrics and logs.
- Then I Configure alarms and notifications for monitoring key performance indicators and thresholds.
- Then I Utilize AWS CloudTrail for auditing the AWS resources.

High Availability and Scalability:

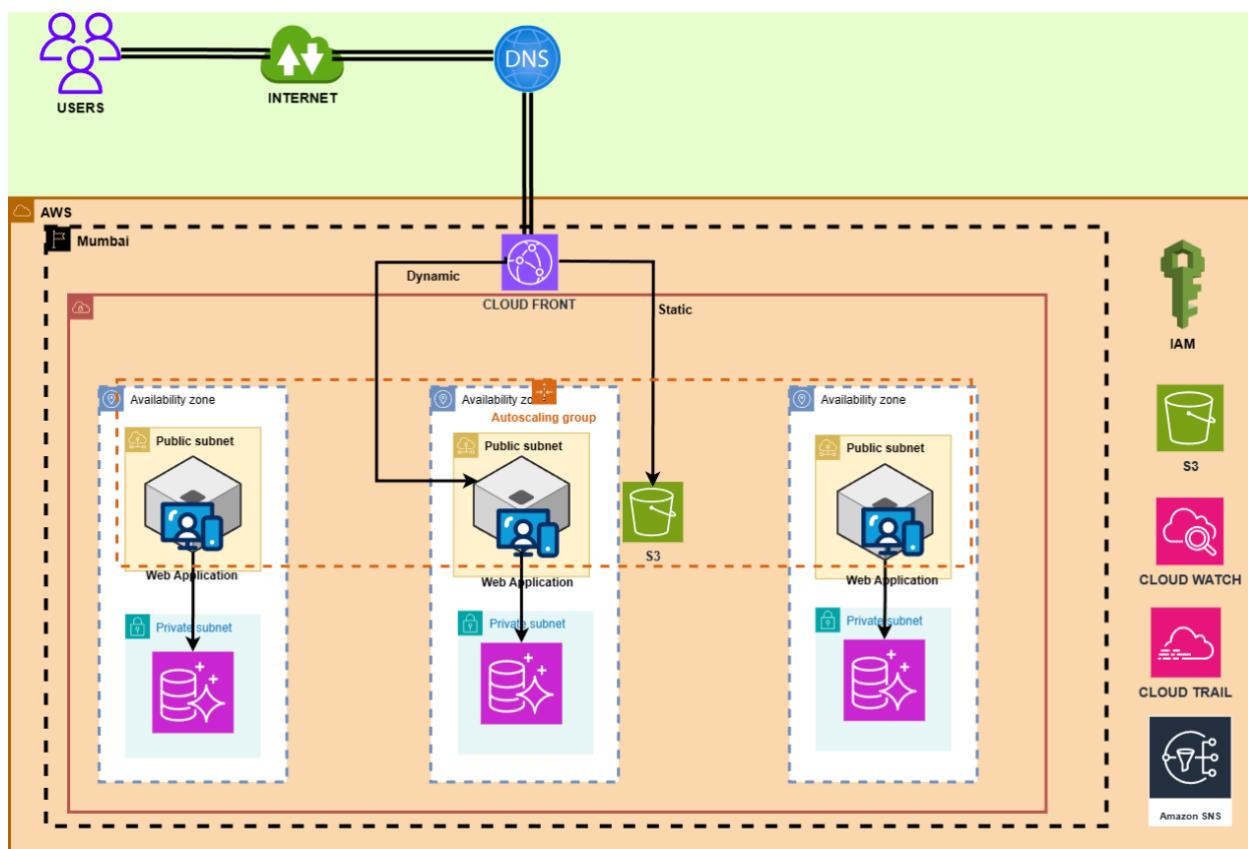
- I Utilize Auto Scaling groups to automatically adjust the number of EC2 instances based on demand.
- Then I Configure Elastic Load Balancing (ELB) to distribute incoming traffic across multiple EC2 instances for high availability and fault tolerance.
- I need to Utilize AWS Route 53 for DNS management and to route traffic to the application's endpoints.

Additionally, thing I will do:

- I Enable WAF (web Application Firewall). Why Because I will disallow some specific ip address as per my requirement.
- I Use ACM (Amazon Certificate Manager). Why Because when request comes from **HTTP**, I will redirect it to **HTTPS port 443**.
- I Use Route 53 for define multiple Route Policy's like: - **simple Route, Weighted Route, Failover Route, Geolocation Route.**

By configuring these AWS resources appropriately, XYZ Corporation can launch their new web-based application with the necessary scalability, reliability, and security on AWS Virtual Machines.

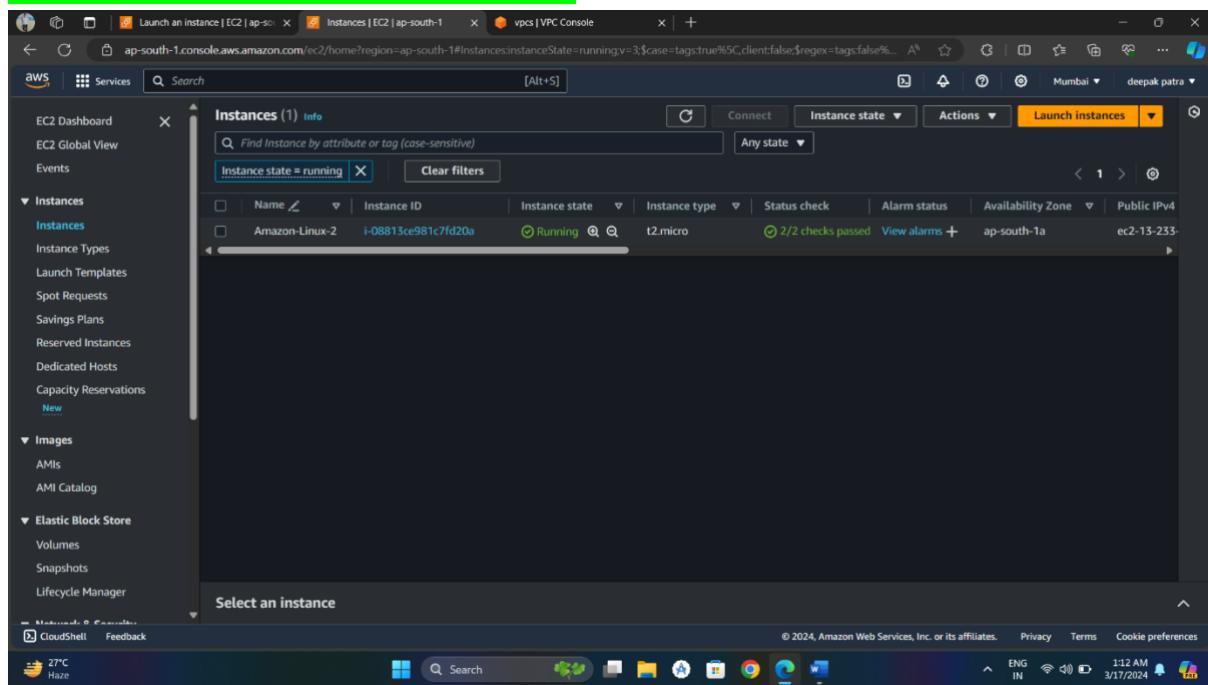
Architecture: -



Demo-Screenshots

Tasks To Be Performed:

1. Launch a Linux EC2 instance.



2. Create an EBS volume with 20 GB of storage and attach it to the created EC2 instance.

The screenshot shows the AWS Cloud Console interface for managing EC2 instances. The left sidebar lists various services like EC2 Dashboard, EC2 Global View, Events, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, AMI Catalog, and Elastic Block Store. The main content area is focused on the 'Storage' tab for the instance 'public1-ap-south-1a'. Under 'Root device details', it shows the root device name as /dev/xvda and the root device type as EBS, with EBS optimization disabled. The 'Block devices' section lists two volumes: 'vol-0431dce1ea5d8ede9' (8 GiB) and 'vol-0957da692d85011a1' (20 GiB), both of which are attached and marked as 'Attached'. The 'Recent root volume replacement tasks' section indicates 'No recent replace root volume tasks'.

3. Resize the attached volume from 20 GB to 30 GB and ensure it reflects in the connected instance.

Path to be followed

- Configure AWS CLI using Access key. (AWS Configure)
- aws ec2 modify-volume --volume-id **\$volume_id** --size 30 (fire this command).
- aws ec2 describe-volumes --volume-ids **\$volume_id** --query 'Volumes[].Size' (fire this for checking size).

EBS volume Change by CLI using Access key

```
ec2-user@ip-10-0-10-168:~$ aws error: argument --volume-id: expected one argument
[ec2-user@ip-10-0-10-168 ~]$ aws ec2 modify-volume --volume-id vol-0957da692d85011a1 --size 30
Unable to locate credentials. You can configure credentials by running "aws configure".
[ec2-user@ip-10-0-10-168 ~]$ aws configure
AWS Access Key ID [None]: AKIA2GCELEHR3D3WE45
AWS Secret Access Key [None]: /WGVrc1lhft2z2ohuDX0UhqrvvhxqdqFccWThx
Default region name [None]:
Default output format [None]:
[ec2-user@ip-10-0-10-168 ~]$ aws --version
aws-cli/2.14.5 Python/3.9.16 Linux/6.1.79-99.167.amzn2023.x86_64 source/x86_64.amzn.2023 prompt/off
[ec2-user@ip-10-0-10-168 ~]$ volume_id="vol-0957da692d85011a1"
[ec2-user@ip-10-0-10-168 ~]$ aws ec2 modify-volume --volume-id $vol-0957da692d85011a1 --size 30
usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:
  aws help
  aws <command> help
  aws <command> <subcommand> help
aws: error: argument --volume-id: expected one argument
[ec2-user@ip-10-0-10-168 ~]$ aws ec2 modify-volume --volume-id vol-0957da692d85011a1 --size 30
{
    "VolumeModification": {
        "VolumeId": "vol-0957da692d85011a1",
        "ModificationState": "modifying",
        "TargetSize": 30,
        "TargetIops": 3000,
        "TargetVolumeType": "gp3",
        "TargetThroughput": 125,
        "OriginalMultiAttachEnabled": false,
        "OriginalSize": 20,
        "OriginalIops": 3000,
        "OriginalVolumeType": "gp3",
        "OriginalThroughput": 125,
        "OriginalMultiAttachEnabled": false,
        "Progress": 0,
        "StartTime": "2024-03-24T03:19:25+00:00"
    }
}
[ec2-user@ip-10-0-10-168 ~]$ aws ec2 describe-volumes --volume-ids vol-0957da692d85011a1 --query 'Volumes[].Size'
[
    30
]
[ec2-user@ip-10-0-10-168 ~]$
```

Problem Statements:

Task 1:

1. Create a Classic Load Balancer (CLB) and attach it to an Auto Scaling Group (ASG).

Load-Balancer

The screenshot shows the AWS CloudFront distribution configuration. Under the 'Distribution' tab, there is a table titled 'Distribution of targets by Availability Zone (AZ)' which lists three availability zones: us-east-1a, us-east-1b, and us-east-1c. Each zone has a count of 1 target instance.

Availability Zone (AZ)	Count
us-east-1a	1
us-east-1b	1
us-east-1c	1

Auto-Scaling-Group

The screenshot shows the AWS Auto Scaling Groups configuration for the 'demo-asg' group. Under the 'Launch template' section, it displays the launch template 'lt-025d3e09c155b7667' with the version 'demo-template'. The AMI ID is listed as 'ami-05295b6e6c790593e', the instance type as 't2.micro', and the owner as 'arn:aws:iam::751762612495:root'.

Task 2:

1. Create an Application Load Balancer (ALB).

The screenshot shows the AWS CloudFront console with a distribution named 'alb'. The 'Details' tab is selected, displaying the following information:

- Load balancer type: Application
- Status: Active
- VPC: vpc-055a50e8d136a30aa
- IP address type: IPv4
- Scheme: Internet-facing
- Hosted zone: ZP97RAFLTNZK
- Availability Zones:
 - subnet-06c8974d46ed21bf ap-south-1a (aps1-a1)
 - subnet-09b4584b1a808ca54 ap-south-1b (aps1-a2)
- Date created: March 24, 2024, 14:15 (UTC+05:30)
- Load balancer ARN: arn:aws:elasticloadbalancing:ap-south-1:751762612495:loadbalancer/app/alb/2aa7dc0651ea7721
- DNS name: alb-1259881611.ap-south-1.elb.amazonaws.com (A Record)

Below the details, there are tabs for 'Listeners and rules', 'Network mapping', 'Resource map - new', 'Security', 'Monitoring', 'Integrations', 'Attributes', and 'Tags'. The 'Resource map' tab is currently active.

2. Attach Target Groups to the ALB.

The screenshot shows the AWS Target Groups console with a target group named 'demo-Target-group'. The 'Details' tab is selected, displaying the following information:

- Target type: Instance
- Protocol: Port
- Protocol version: HTTP
- VPC: vpc-055a50e8d136a30aa
- IP address type: IPv4
- Load balancer: alb

Below the details, there is a table showing target status:

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

A message box at the top right states: "Introducing Automatic Target Weights (ATW) to increase application availability. Automatic Target Weights is achieved by turning on anomaly mitigation, which provides responsive, dynamic distribution of traffic to targets based on anomaly detection results. All HTTP/HTTPS target groups now include anomaly detection by default. [Learn more](#)".

3. Traffic should be served through the ALB and if you are purchasing any domain then point it to ALB DNS.

Ans: Currently I have not purchased any DNS.

The screenshot shows the AWS CloudWatch Metrics Insights interface. A search bar at the top contains the query "CloudWatch Metrics Insights". Below the search bar, there are two tabs: "Metrics" and "Logs". The "Metrics" tab is selected, displaying a table of metrics. The table has columns for "Metric Name", "Dimensions", "Unit", and "Last Value". One row is highlighted, showing "CloudWatch Metrics Insights" with dimensions "Region=ap-south-1" and "MetricName=CloudWatch Metrics Insights". The "Logs" tab is also visible.

Problem Statement:

Task 1:

Set up a Network Load Balancer for an Apache web server where the load balancer port should be 25. And configure Nginx Web Server on load balancer port 58. Additionally, set up a Tomcat App server on load balancer port 90.

1. Create EC2 Instances:

- Launch EC2 instances for Apache, Nginx, and Tomcat servers, ensuring they are configured properly with the required software.

The screenshot shows the AWS EC2 Instances page. The left sidebar includes options like EC2 Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity, and Reservations. The main area displays a table of instances. The table has columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IPv4 DNS. Three instances are listed: "Tomcat-instance" (Instance ID: i-06e4ed35022ee8b70, State: Running, Type: t2.micro), "Nginx-instance" (Instance ID: i-0c7e1a73f18777f64, State: Running, Type: t2.micro), and "Apache-instance" (Instance ID: i-00c64d5e34fd7b82, State: Running, Type: t2.micro). The instances are located in the "ap-south-1b" availability zone.

2. Configure Security Groups:

- Create a security group that allows inbound traffic on ports 25, 58, and 90 from the load balancer to the respective instances.
- Attach this security group to all the EC2 instances.

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-06b95c8d63f324dd	SSH	TCP	22	Custom	0.0.0.0/0
sgr-00040ab58940b8dc	SMTP	TCP	25	Custom	0.0.0.0/0
sgr-0d876c4c3abeea74	Custom TCP	TCP	90	Custom	0.0.0.0/0
sgr-00a7154929b233df3	Custom TCP	TCP	58	Custom	0.0.0.0/0
sgr-0af966223ae2f9925	HTTP	TCP	80	Custom	0.0.0.0/0

3. Set Up the Network Load Balancer:

- Navigate to the EC2 dashboard and select "Load Balancers" from the sidebar.
- Click on "Create Load Balancer" and choose "Network Load Balancer".
- Configure the NLB settings, such as name, scheme, IP address type, and listeners:
 - For port 25, set the listener protocol to **TCP** and port to 25, then configure the target group for Apache.
 - For port 58, set the listener protocol to **TCP** and port to 58, then configure the target group for Nginx.
 - For port 90, set the listener protocol to **TCP** and port to 90, then configure the target group for Tomcat.
- Configure the availability zones where your EC2 instances are located.
- Configure security settings and tags as needed.
- Create the NLB.

The screenshot shows the AWS EC2 Load Balancers console. On the left sidebar, under the 'Load Balancing' section, 'Target Groups' is selected. In the main pane, the 'Load balancers' section displays one entry: 'test-nlb'. Below it, the 'Load balancer: test-nlb' configuration shows three listeners:

Protocol:Port	Action	Target Group
TCP:58	Forward to target group	test-tg-nginx
TCP:90	Forward to target group	test-tg-tomcat
TCP:25	Forward to target group	test-sg

4. Create Target Groups:

- Navigate to the EC2 dashboard and select "Target Groups" from the sidebar.
- Click on "Create target group" and configure the target group settings:
 - For Apache, set the target type to instance and choose the instances running Apache.
 - For Nginx, set the target type to instance and choose the instances running Nginx.
 - For Tomcat, set the target type to instance and choose the instances running Tomcat.
- Create the target groups.

The screenshot shows the AWS EC2 Target groups console. On the left sidebar, under the 'Load Balancing' section, 'Target Groups' is selected. In the main pane, the 'Target groups' section displays three entries:

Name	ARN	Port	Protocol	Target type	Load balancer
test-tg-tomcat	arn:aws:elasticloadbalancing:ap-south-1:123456789012:targetgroup/test-tg-tomcat/79ec985...	90	TCP	Instance	(None associated)
test-tg-nginx	arn:aws:elasticloadbalancing:ap-south-1:123456789012:targetgroup/test-tg-nginx/79ec985...	58	TCP	Instance	(None associated)
test-sg	arn:aws:elasticloadbalancing:ap-south-1:123456789012:targetgroup/test-sg/79ec985...	80	TCP	Instance	(None associated)

5. Register Targets:

- After creating the target groups, register the corresponding EC2 instances (Apache, Nginx, and Tomcat servers) with their respective target groups.

6. Update Load Balancer Listener Rules:

- Once the target groups are set up, update the load balancer listener rules to forward traffic on ports 25, 58, and 90 to their respective target groups.

Protocol:Port	Default action	ARN	Security policy	Default SSL/TLS certificate	ALPN policy
TCP-58	Forward to target group • test-tg-nginx	ARN	Not applicable	Not applicable	None
TCP-90	Forward to target group • test-tg-tomcat	ARN	Not applicable	Not applicable	None
TCP-25	Forward to target group • test-sg	ARN	Not applicable	Not applicable	None

7. Test the Configuration:

- Verify that the NLB distributes traffic correctly to the Apache, Nginx, and Tomcat servers by accessing the NLB endpoint with the specified ports.
- for testing that NLB route traffic perfectly to each instance I go to config files of all web-server's and updated Listening port of all web-servers.**

Assignment:

1. Create a private Route 53 domain and point it to an Application Load Balancer.

Complete path:

- Go to the Route 53 console.
- Choose "Create Hosted Zone."
- Enter the domain name for your private zone (e.g., example.com).
- Choose the type "Private hosted zone for Amazon VPC."
- Select the VPC(s) where you want to associate this private hosted zone.

Create a Record Set:

- Inside your private hosted zone, create a new record set.
- Choose the record type appropriate for your use case (e.g., A or CNAME).
- Configure the record to point to the DNS name of your ALB.
- Configure the Application Load Balancer:
- Go to the EC2 console and navigate to the "Load Balancers" section.
- Select your ALB.

private Route 53 domain

Route 53 > Hosted zones > example.com

Hosted zone details

Hosted zone name	Type	Associated VPCs
example.com	Private hosted zone	vpc-008e4b6290e2a7ff0 ap-south-1

Records (3)

Record name	Type	Value
example.com	A	dualstack.test-alb-1513774134.ap-south-1.elb.amazonaws.com.
example.com	NS	simple
example.com	SOA	simple

point it to an Application Load Balancer

Records (1/3) Info

Record name	Type	Routing policy	Alias
example.com	A	Simple	Yes
example.com	NS	Simple	
example.com	SOA	Simple	

Record details

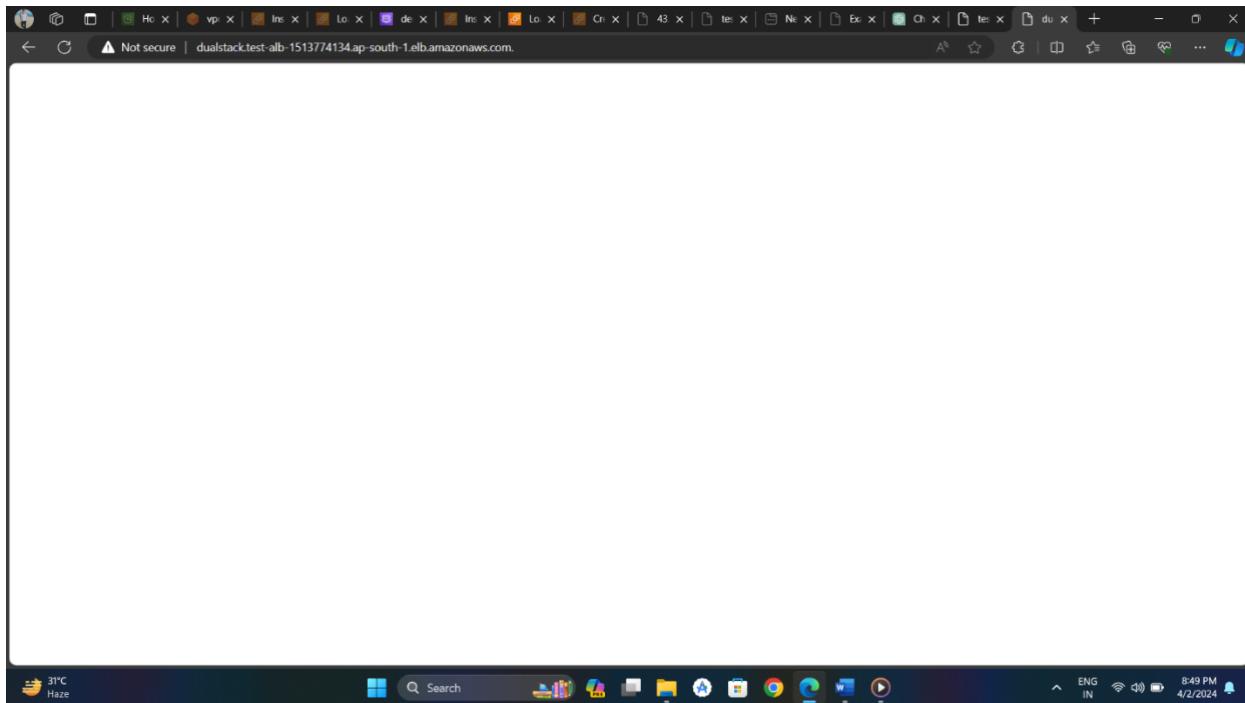
Record name	Value
example.com	dualstack.test-alb-1513774134.ap-south-1.elb.amazonaws.com.

Record type: A
Value: dualstack.test-alb-1513774134.ap-south-1.elb.amazonaws.com.
Alias: Yes
TTL (seconds):
Routing policy: Simple

2. Verify the domain through another machine in the same network.

Create a Private Hosted Zone in Route 53:

NB:- I didn't install any web server inside this. so for that it shows like this Empty White Screen.



S3:

- 1. Create an S3 bucket and restrict access to specific users.**
- 2. Create a bucket accessible through an EC2 instance.**
- 3. Create a bucket, copy, and sync data from a local source to the S3 bucket.**
- 4. Hosting a static website in S3.**

1. Create an S3 Bucket:

- Sign in to the AWS Management Console.**
- Go to the S3 service.**
- Click on "Create Bucket".**
- Choose a unique bucket name and select the region.**
- Do ACLs enabled.**
- Disable Block *all* public access**
- Click "Create".**

2. Upload Website Files:

- In the bucket you created, click on the "Upload" button.**
- Upload your website files (HTML, CSS, JavaScript, images, etc.).**
- Ensure that your main page is named according to the index document specified earlier (e.g., index.html).**

3. Configure Bucket for Website Hosting:

- Select the bucket you just created.**
- Go to the "Properties" tab.**
- Click on "Static website hosting enable it".**
- Choose "Host a static website".**

- Enter the index document (e.g., index.html).
- Optionally, enter an error document (e.g., error.html) for custom error pages.
- Click "Save".

Test Your Website:

- Once you've uploaded your files and set permissions, go back to the bucket's static website hosting configuration.
- You'll find the endpoint URL where your website is hosted. It will be something like "<http://YOUR-BUCKET-NAME.s3-website-REGION.amazonaws.com>."
- Open this URL in a web browser to verify that your website is working correctly.

Demo

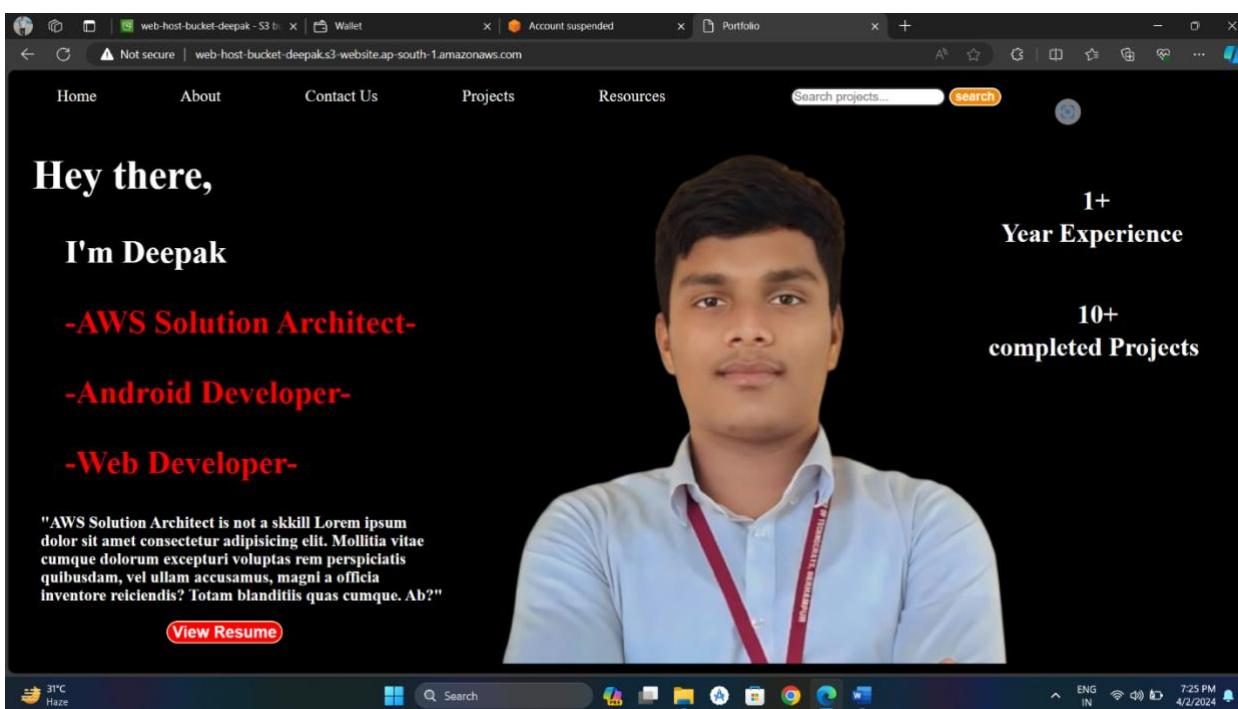
Bucket Screen

Objects (3) [Info](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	deepak.png	png	April 2, 2024, 19:21:52 (UTC+05:30)	171.1 KB	Standard
<input type="checkbox"/>	index.html	html	April 2, 2024, 19:21:52 (UTC+05:30)	1.9 KB	Standard
<input type="checkbox"/>	style.css	css	April 2, 2024, 19:21:53 (UTC+05:30)	1.6 KB	Standard

Website



CloudWatch:

Task 1:

Create a CloudWatch Alarm for an EC2 instance. Trigger the alarm to send notifications to end-users via SNS when it crosses a predefined threshold.

Cpu_spike code Running

Link of Python code:- https://github.com/deepak-401/cpu_spike/blob/main/cpu_spike

```
ec2-user@ip-10-0-0-114:~$ ps -ef | grep python3
root      20  0 105152 16416 10092 S  0.0  1.7 0:00.94 systemd
root      20  0      0  0  0 0  0.0  0.0 0:00.00 kthread
root      0 -20      0  0  0 0  0.0  0.0 0:00.00 rch_gp
root      4 -20      0  0  0 0  0.0  0.0 0:00.00 rch_gp
root      9 -20      0  0  0 0  0.0  0.0 0:00.00 slab_flushwq
root      6 -20      0  0  0 0  0.0  0.0 0:00.00 netns
root      8 -20      0  0  0 0  0.0  0.0 0:00.00 kworker/0:0H-events_highpri
root     10 -20      0  0  0 0  0.0  0.0 0:00.00 mm_percpu_wq
root     11 -20      0  0  0 0  0.0  0.0 0:00.00 rCU_tasks_kthread
root     12 -20      0  0  0 0  0.0  0.0 0:00.00 rCU_tasks_rude_kthread
root     13 -20      0  0  0 0  0.0  0.0 0:00.00 rCU_tasks_trace_kthread
root     14 -20      0  0  0 0  0.0  0.0 0:00.00 ksoftirqd/0
root     15 -20      0  0  0 0  0.0  0.0 0:00.00 rCU_prempt
root     16 -20      0  0  0 0  0.0  0.0 0:00.00 migration/0
root     18 -20      0  0  0 0  0.0  0.0 0:00.00 cpuhp/0
root     20 -20      0  0  0 0  0.0  0.0 0:00.00 kdevtmpfs
root     21 -20      0  0  0 0  0.0  0.0 0:00.00 Intel_iommu_wq
root     22 -20      0  0  0 0  0.0  0.0 0:00.00 ksoftirqd
root     23 -20      0  0  0 0  0.0  0.0 0:00.00 khungtaskd
root     24 -20      0  0  0 0  0.0  0.0 0:00.00 com_reaper
root     25 -20      0  0  0 0  0.0  0.0 0:00.13 kworker/u30:1-events_unbound
root     27 -20      0  0  0 0  0.0  0.0 0:00.00 writeback
root     28 -20      0  0  0 0  0.0  0.0 0:00.04 kcompactd0
root     29 -20      0  0  0 0  0.0  0.0 0:00.00 khugepaged
root     30 -20      0  0  0 0  0.0  0.0 0:00.00 kintegrityd
root     31 -20      0  0  0 0  0.0  0.0 0:00.00 kblockd
root     32 -20      0  0  0 0  0.0  0.0 0:00.00 blkcg_punt_bio
root     33 -20      0  0  0 0  0.0  0.0 0:00.00 xen-balloon
root     34 -20      0  0  0 0  0.0  0.0 0:00.00 tpm_dev_wq
root     35 -20      0  0  0 0  0.0  0.0 0:00.00 edac-poller
root     36 -20      0  0  0 0  0.0  0.0 0:00.00 watchdogd
root     -51  0      0  0  0 0  0.0  0.0 0:00.00 kwatchdogd
root     38 -20      0  0  0 0  0.0  0.0 0:00.01 kworker/0:1H-kblockd
root     72 -20      0  0  0 0  0.0  0.0 0:00.01 kswapd0
root     75 -20      0  0  0 0  0.0  0.0 0:00.00 xsalloc
root     76 -20      0  0  0 0  0.0  0.0 0:00.00 xfs_mru_cache
root     79 -20      0  0  0 0  0.0  0.0 0:00.00 kthrotid
root     94 -20      0  0  0 0  0.0  0.0 0:00.00 xenbus
root     95 -20      0  0  0 0  0.0  0.0 0:00.00 xenwatch
root    133 -20      0  0  0 0  0.0  0.0 0:00.00 nvme-wq
root    135 -20      0  0  0 0  0.0  0.0 0:00.00 nvme-reset-wq
root    137 -20      0  0  0 0  0.0  0.0 0:00.00 nvme-delete-wq
[ec2-user@ip-10-0-0-114 ~]$ python3 cpu_spike.py
Simulating CPU spike at 80%...
```

Screenshot of the AWS CloudWatch Home page showing an alarm for EC2 CPU utilization.

Alarms by AWS service

Recent alarms

Danger cpu utilization is reached maximum

Percent

100
50.1
0.164

12:30 13:30 14:30

Default Dashboard

Name any CloudWatch dashboard **CloudWatch-Default** to display it here. Create a new default dashboard

Application Insights

CloudShell Feedback 30°C Haze © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 9:01 PM 4/3/2024

In Alaram State

Screenshot of the AWS CloudWatch Alarms page for the "Danger cpu utilization is reached maximum" alarm.

CloudWatch > Alarms > Danger cpu utilization is reached maximum

Graph

CPUUtilization

CPUUtilization ≥ 50 for 1 datapoints within 1 minute

In alarm

Percent

100
50.1
0.164

12:30 13:00 13:30 14:00 14:30 15:00

Click timeline to see the state change at the selected time.

Actions

Danger cpu utilization is reached maximum

Metric alarm

In alarm

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 30°C Haze ENG IN 8:54 PM 4/3/2024

E-mail

Inbox - Gmail 2

Search ... Try the new Outlook

Inbox

- AN AWS Notifications** ALARM: "Danger cpu utilization is reached maximum" in Asia Pacific (Mumbai) 8:53 PM You are receiving this email because you have chosen to subscribe to the topic.
- SL Sanghamitra Satpathy via LinkedIn** Deepak Kumar, message your (3) 8:50 PM See Sanghamitra's connections, experience, and more.
- FA Font Awesome** 50% Off Font Awesome Pro. Forever. 8:46 PM Did you know that you can upgrade to the Pro version?
- AN AWS Notifications** AWS Notification - Subscription Confirmation 8:46 PM You have chosen to subscribe to the topic.
- CT Cutvette Tech** Job Guarantee Course | CTC upto 25 | 7:35 PM Hey Deepak, After placing 3150+ students.
- NE Neha at Edureka** Limited-Time Offer! Save on PRINCE 7:29 PM Price Drop Alert. Hi Learner, We have a limited-time offer for you!
- J Jobs@S&PGlobal** S&P Global is hiring Data Analysts | A 7:26 PM Hi Unstoppable, S&P Global is hiring!
- KE Kate Wilson, EDUCBA** [90% Off] Just for \$159 - Upgrade to EDUCBA 6:41 PM @ just \$159 \$1999 ENROLL NOW View course details

30°C Haze

8:53 PM 4/3/2024 ENG IN WiFi Battery

ALARM: "Danger cpu utilization is reached maximum" in Asia Pacific (Mumbai)

AN AWS Notifications <no-reply@sns.amazonaws.com> 8:53 PM To: deepak65862@gmail.com

You are receiving this email because your Amazon CloudWatch Alarm "Danger cpu utilization is reached maximum" in the Asia Pacific (Mumbai) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [84.66666666666666 (03/04/24 15:21:00)] was greater than or equal to the threshold (50.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Wednesday 03 April, 2024 15:23:26 UTC".

View this alarm in the AWS Management Console: <https://ap-south-1.console.aws.amazon.com/cloudwatch/deeplink.js?region=ap-south-1#alarmsV2:alarm/Danger%20cpu%20utilization%20is%20reached%20maximum>

Alarm Details:

- Name: Danger cpu utilization is reached maximum
- Description: Hey Team,
- Danger!: cpu utilization is reached maximum. review it as soon as possible.
- State Change: OK -> ALARM
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [84.66666666666666 (03/04/24 15:21:00)] was greater than or equal to the threshold (50.0) (minimum 1 datapoint for OK -> ALARM transition).
- Timestamp: Wednesday 03 April, 2024 15:23:26 UTC
- AWS Account: 588880959538
- Alarm Arn: arn:aws:cloudwatch:ap-south-1:588880959538:alarm:Danger cpu utilization is reached maximum

Threshold:

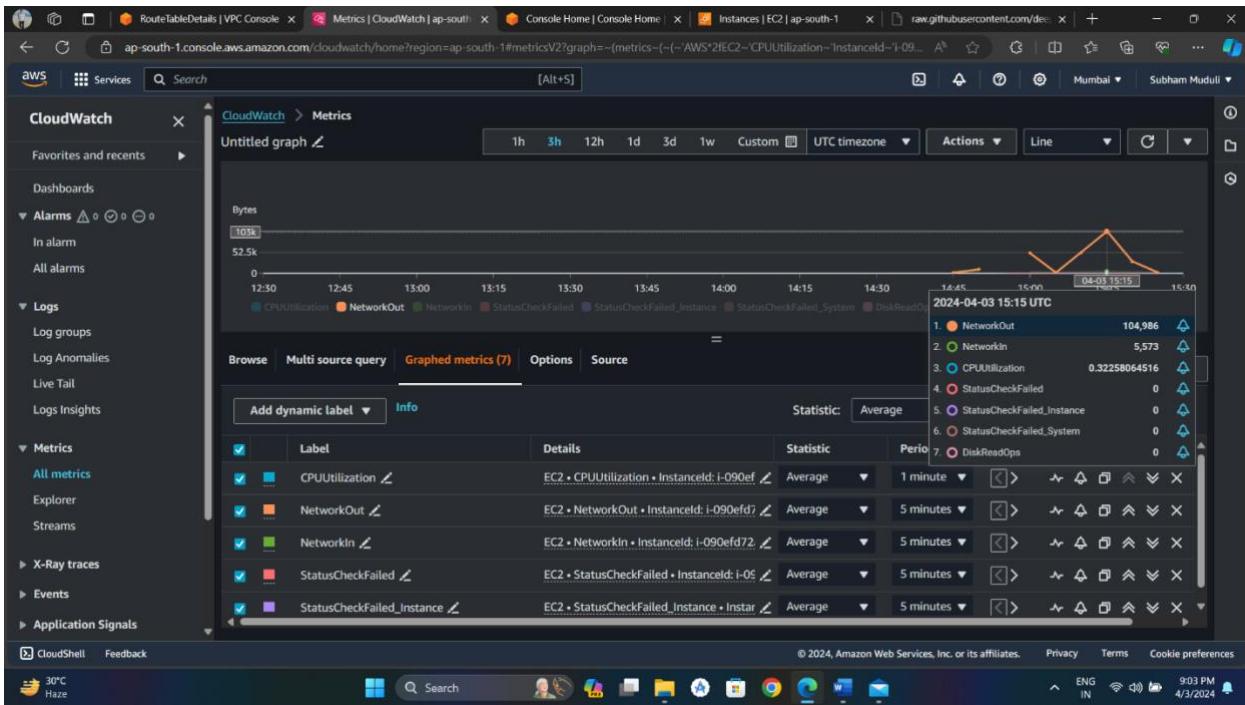
- The alarm is in the ALARM state when the metric is GreaterThanOrEqualToThreshold 50.0 for at least 1 of the last 1 period(s) of 60 seconds.

Monitored Metric:

- MetricNamespace: AWS/EC2
- MetricName: CPUUtilization
- Dimensions: [InstanceId = i-090efd72aaef8a4d0]
- Period: 60 seconds
- Statistic: Maximum
- Unit: not specified
- TreatMissingData: missing

Monitor the following metrics:

- 1. CPU Utilization**
- 2. Memory Utilization**
- 3. Disk Utilization**
- 4. Network Out and In**
- 5. Instance and System Status Check Failed**



Task 2:

Create a dashboard to monitor CPU utilization and Memory Utilization for a specific EC2 instance.

