

TASK 1

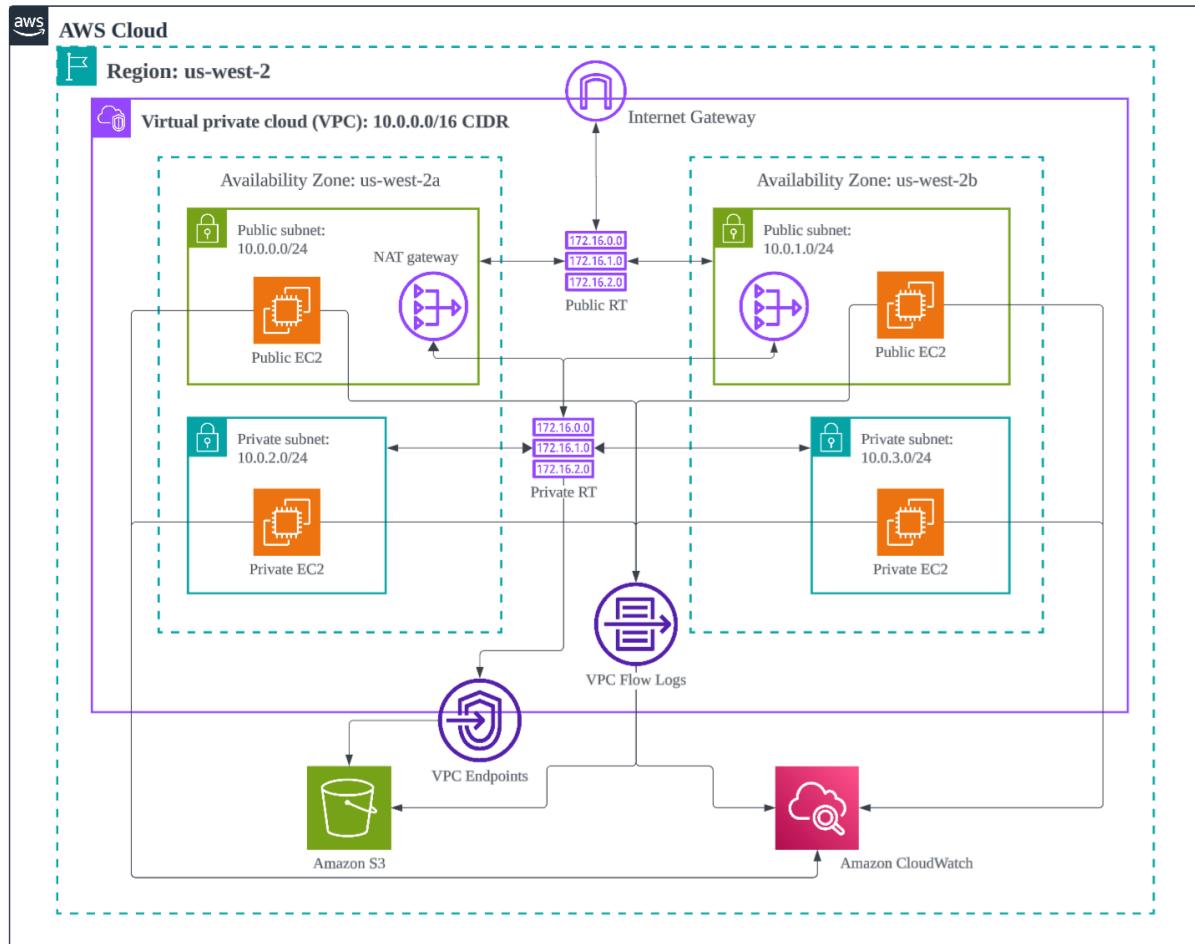
Create VPC Infrastructure

Umar Satti

Task Description

Custom Virtual Private Cloud (VPC) setup to host secure and scalable AWS resources across public and private subnets with proper routing, NAT, and security configurations.

VPC Architecture Diagram



Task 1.1: Create a custom VPC with a specified CIDR block

Steps:

1. Navigate to the “VPC” service from the AWS Management Console search bar
2. In the left navigation panel, select “Your VPCs”
3. Click on Create VPC button located in the top right corner
4. Under Resources to create, select “VPC only”
5. Enter a Name and specify an IPv4 CIDR block range
6. Click “Create VPC” button

The screenshot shows the AWS VPC Details page for a VPC named 'umar-vpc'. The 'CIDRs' tab is selected, displaying a single IPv4 CIDR block: 10.0.0.0/16, which is associated and currently active.

The VPC name is umar-vpc and the IPv4 CIDR block range is 10.0.0.0/16.

Task 1.2: Define public and private subnets across multiple Availability Zones

Steps:

1. Click on “Subnets” section in the left navigation panel
2. Click on “Create Subnet” button and select the VPC created in previous task
3. Define each subnet with a name, Availability Zone, and IPv4 CIDR block
4. Repeat the process to create both public and private subnets

The screenshot shows the AWS Subnets (4) page. It lists four subnets: private-subnet-a, public-subnet-b, public-subnet-a, and private-subnet-b. Each subnet is associated with a specific Subnet ID, State (Available), VPC (umar-vpc), Block Public Access (Off), and an IPv4 CIDR range. The subnets are distributed across different Availability Zones.

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
private-subnet-a	subnet-08ad52da6fb1a9cd7	Available	vpc-0728cd553c4f9f0c5 umar...	Off	10.0.2.0/24
public-subnet-b	subnet-021e36d5e0399af9	Available	vpc-0728cd553c4f9f0c5 umar...	Off	10.0.1.0/24
public-subnet-a	subnet-00f9f0b04e6e84e14	Available	vpc-0728cd553c4f9f0c5 umar...	Off	10.0.0.0/24
private-subnet-b	subnet-04c840d2484a6c611	Available	vpc-0728cd553c4f9f0c5 umar...	Off	10.0.3.0/24

The following subnets were created as illustrated above:

Public Subnets

- public-subnet-a, AZ is us-west-2a, and CIDR range is 10.0.0.0/24
- public-subnet-b, AZ is us-west-2b, and CIDR range 10.0.1.0/24

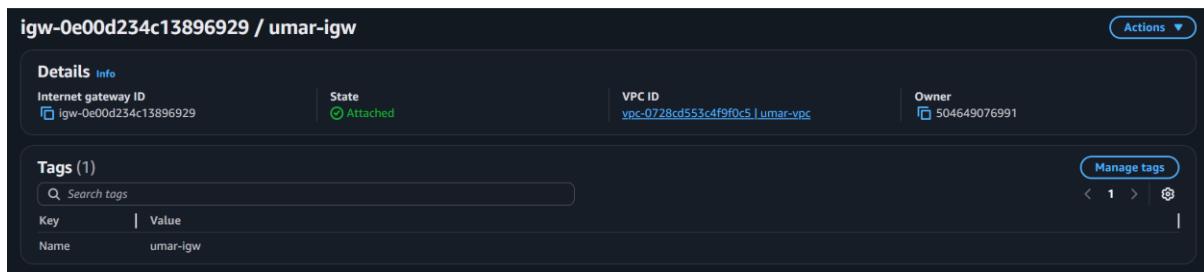
Private Subnets

- private-subnet-a, AZ is us-west-2a, and CIDR range is 10.0.2.0/24
- private-subnet-b, AZ is us-west-2a, and CIDR range is 10.0.3.0/24

Task 1.3: Set up an Internet Gateway and attach it to the VPC

Steps:

1. Click on “Internet gateways” section in the left navigation panel
2. Click “Create internet gateway” button on the top right
3. Enter a name tag and click Create Internet Gateway
4. After creating the internet gateway, attach it to the VPC created earlier
5. Click “Actions” button and select “Attach to VPC”. Choose the VPC ID and click “Attach internet gateway”



The Internet gateway name is umar-igw and it is attached to the VPC as shown by the status “Attached”

Task 1.4: Create route tables and associate them with appropriate subnets

Steps:

1. Click on “Route Tables” section in the left navigation panel
2. Click the “Create route table” button on the top right
3. Choose a name and select the VPC created in Task 1.1
4. Create one route table for public subnets and one for private subnets
5. Add the appropriate “Routes” and “Subnet associations” to these subnets

I created one route table for public subnet with the name “umar-public-route-table” and one route table for private subnet with the name “umar-private-route-table.”

Public Subnet Route Table:

The screenshot shows the AWS Route Tables page with the route table 'rtb-0f6d1b5491736beed / umar-public-route-table'. The 'Details' tab is selected. The table details are as follows:

Route table ID	Main	Explicit subnet associations	Edge associations
rtb-0f6d1b5491736beed	No	-	-
VPC	Owner ID		
vpc-0728cd553c4f9f0c5 umar-vpc	504649076991		

Private Subnet Route Table:

The screenshot shows the AWS Route Tables page with the route table 'rtb-0441f1d8e13754829 / umar-private-route-table'. The 'Details' tab is selected. The table details are as follows:

Route table ID	Main	Explicit subnet associations	Edge associations
rtb-0441f1d8e13754829	No	-	-
VPC	Owner ID		
vpc-0728cd553c4f9f0c5 umar-vpc	504649076991		

Configure the public route table to direct traffic to the internet gateway.

1. Select the public route table, click on “Routes” and then click “Edit routes”
2. Click “Add route”, select Destination and Target, and then click “Save changes”

I chose Destination as 0.0.0.0/0 and the Target as Internet Gateway (umar-igw)

rtb-0f6d1b5491736becd / umar-public-route-table

Details Info

Route table ID rtb-0f6d1b5491736becd	Main No	Explicit subnet associations 2 subnets	Edge associations —
VPC vpc-0728cd553c4f9f0c5 umar-vpc	Owner ID 504649076991		

Routes **Subnet associations** **Edge associations** **Route propagation** **Tags**

Routes (2)

Q Filter routes		Target	Status	Propagated	Route Origin
Destination	▼	igw-0e00d234c13896929	Active	No	Create Route
0.0.0.0/0	▼	local	Active	No	Create Route Table
10.0.0.0/16	▼				

Both [Edit routes](#) < 1 > ⚙

Configure the route tables to add subnet associations

1. Select the route table.
2. Click on “Subnet associations”
3. Click the “Edit subnet associations” button and choose the appropriate subnets

Public Route Table Subnet Association:

- public-subnet-a with the IPv4 CIDR range 10.0.0.0/24
- public-subnet-b with the IPv4 CIDR range 10.0.1.0/24

rtb-0f6d1b5491736becd / umar-public-route-table

Details Info

Route table ID rtb-0f6d1b5491736becd	Main No	Explicit subnet associations 2 subnets	Edge associations —
VPC vpc-0728cd553c4f9f0c5 umar-vpc	Owner ID 504649076991		

Subnet associations **Edge associations** **Route propagation** **Tags**

Explicit subnet associations (2)

Q Find subnet association		IPv4 CIDR	IPv6 CIDR
Name	Subnet ID	▼	▼
public-subnet-b	subnet-021e36d5e03999af9	10.0.1.0/24	—
public-subnet-a	subnet-00f9f0b04e6e84e14	10.0.0.0/24	—

[Edit subnet associations](#) < 1 > ⚙

Subnets without explicit associations (0)

The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:

Q Find subnet association		IPv4 CIDR	IPv6 CIDR
Name	Subnet ID	▼	▼
No subnets without explicit associations			
All your subnets are associated with a route table.			

[Edit subnet associations](#) < 1 > ⚙

Private Route Table Subnet Association:

- private-subnet-a with the IPv4 CIDR range 10.0.2.0/24
- private-subnet-b with the IPv4 CIDR range 10.0.3.0/24

The screenshot shows the AWS Route Table details page for route table ID rtb-0441f1d8e13754829. The 'Subnet associations' tab is selected. Under 'Explicit subnet associations', there are two entries:

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
private-subnet-a	subnet-08ad52da6fb1a9cd7	10.0.2.0/24	-
private-subnet-b	subnet-04c840d2484a6c611	10.0.3.0/24	-

Under 'Subnets without explicit associations', it says 'No subnets without explicit associations' and 'All your subnets are associated with a route table.'

Task 1.5: Configure NAT Gateway in a public subnet for private subnet internet access

Steps:

1. Click on “NAT Gateways” section in the left navigation panel
2. Click “Create NAT gateway” button on the top right
3. Provide a name, select a public subnet, and allocate an Elastic IP
4. Choose Connectivity Type as “Public”
5. Click “Create NAT Gateway”

I configured a NAT Gateway in a public subnet to allow instances in private subnets to access the internet

- NAT Gateway Name is “umar-nat-gateway”, Subnet is “public-subnet-a”, Elastic IP is “eipalloc-0d07b3272e0c6b5a2”
- Destination as 0.0.0.0/0 to Target as “umar-nat-gateway”

nat-025b042a1c1981a0d / umar-nat-gateway

Details | Secondary IPv4 addresses | Monitoring | Tags

Details

NAT gateway ID nat-025b042a1c1981a0d	Connectivity type Public	State Available	State message -
NAT gateway ARN arn:aws:ec2:us-west-2:504649076991:natgateway/nat-025b042a1c1981a0d	Primary public IPv4 address 16.145.201.55	Primary private IPv4 address 10.0.0.5	Primary network interface ID eni-0464224cd37f480f
VPC vpc-0728cd553c4f9f0c5 / umar-vpc	Subnet subnet-00f9f0b04e6e84e14 / public-subnet-a	Created Wednesday, November 5, 2025 at 20:35:23 GMT+5	Deleted -

Configure the private route table to direct traffic to NAT gateway.

1. Select the private route table, click on “Routes” and then click “Edit routes”
2. Click “Add route”, select Destination and Target, and then click “Save changes”

Edit routes

Destination	Target	Status	Propagated	Route Origin
10.0.0.16	local	Active	No	CreateRouteTable
0.0.0.0/0	NAT Gateway	-	No	CreateRoute
0.0.0.0/0	nat-025b042a1c1981a0d	-	-	-

Add route

Cancel | Preview | Save changes

I chose Destination as 0.0.0.0/0 and Target as NAT Gateway (umar-nat-gateway) as shown below:

rtb-0441f1d8e13754829 / umar-private-route-table

Details

Route table ID rtb-0441f1d8e13754829	Main No	Explicit subnet associations 2 subnets	Edge associations -
VPC vpc-0728cd553c4f9f0c5 umar-vpc	Owner ID 504649076991		

Routes (2)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	nat-025b042a1c1981a0d	Active	No	Create Route
10.0.0.16	local	Active	No	Create Route Table

Task 1.6: Implement security groups and network ACLs for traffic control

Steps:

1. Navigate to the “EC2” service from the AWS Management Console search bar
2. In the left navigation panel, select “Security Groups”
3. Click the “Create security group” button on the top right
4. Choose a security group name, description, and VPC
5. Add Inbound and Outbound rules

For public resources, I chose the following security group configuration:

- Security group name: umar-public-sg
- VPC: umar-vpc
- Inbound Rules:
 - All ICMP - IPv4 from Anywhere-IPv4 (0.0.0.0/0)
 - HTTP from Anywhere-IPv4 (0.0.0.0/0)
 - SSH from Anywhere-IPv4 (0.0.0.0/0)
- Outbound Rules:
 - All Traffic to Destination 0.0.0.0/0

The screenshot shows the AWS Management Console interface for a security group named "umar-public-sg". The "Inbound rules" tab is selected. There are three rules listed:

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
-	sgr-046b6739846f6c3c	IPv4	All ICMP - IPv4	ICMP	All	0.0.0.0/0
-	sgr-0ecd9fa3c3f0a7644	IPv4	HTTP	TCP	80	0.0.0.0/0
-	sgr-0668f5f1cc03403c6	IPv4	SSH	TCP	22	0.0.0.0/0

For private resources, I chose the following security group configuration:

- Security group name: umar-private-sg
- VPC: umar-vpc
- Inbound Rules:
 - All ICMP - IPv4 traffic from umar-public-sg
 - HTTP traffic from umar-public-sg
 - SSH traffic from umar-public-sg
- Outbound Rules:
 - All Traffic to Destination 0.0.0.0/0

sg-019d80eb41f69a055 - umar-private-sg																																												
Details		Inbound rules		Outbound rules		Sharing - new		VPC associations - new																																				
Inbound rules (3)																																												
<table border="1"> <thead> <tr> <th>Q Search</th> <th>Name</th> <th>Security group rule ID</th> <th>IP version</th> <th>Type</th> <th>Protocol</th> <th>Port range</th> <th>Source</th> <th>De</th> </tr> </thead> <tbody> <tr> <td><input type="checkbox"/></td> <td>-</td> <td>sgr-0afdb1b00742d082b</td> <td>-</td> <td>SSH</td> <td>TCP</td> <td>22</td> <td>sgr-00345f4ab165745ff / umar-public-sg</td> <td>-</td> </tr> <tr> <td><input type="checkbox"/></td> <td>-</td> <td>sgr-04b41ab69cdb3985</td> <td>-</td> <td>All ICMP - IPv4</td> <td>ICMP</td> <td>All</td> <td>sgr-00345f4ab165745ff / umar-public-sg</td> <td>-</td> </tr> <tr> <td><input type="checkbox"/></td> <td>-</td> <td>sgr-085544f9793d4b2d3</td> <td>-</td> <td>HTTP</td> <td>TCP</td> <td>80</td> <td>sgr-00345f4ab165745ff / umar-public-sg</td> <td>-</td> </tr> </tbody> </table>									Q Search	Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	De	<input type="checkbox"/>	-	sgr-0afdb1b00742d082b	-	SSH	TCP	22	sgr-00345f4ab165745ff / umar-public-sg	-	<input type="checkbox"/>	-	sgr-04b41ab69cdb3985	-	All ICMP - IPv4	ICMP	All	sgr-00345f4ab165745ff / umar-public-sg	-	<input type="checkbox"/>	-	sgr-085544f9793d4b2d3	-	HTTP	TCP	80	sgr-00345f4ab165745ff / umar-public-sg	-
Q Search	Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	De																																				
<input type="checkbox"/>	-	sgr-0afdb1b00742d082b	-	SSH	TCP	22	sgr-00345f4ab165745ff / umar-public-sg	-																																				
<input type="checkbox"/>	-	sgr-04b41ab69cdb3985	-	All ICMP - IPv4	ICMP	All	sgr-00345f4ab165745ff / umar-public-sg	-																																				
<input type="checkbox"/>	-	sgr-085544f9793d4b2d3	-	HTTP	TCP	80	sgr-00345f4ab165745ff / umar-public-sg	-																																				

Task 1.7: Launch EC2 instances in private and public subnets as needed

Steps:

1. Navigate to the “EC2” service from the AWS Management Console search bar
2. In the left navigation panel, select “Instances”
3. Click the “Launch instances” button on the top right
4. Choose a name, OS image, AMI, Instance type, Key pair, Network Settings, and Storage
5. Click “Launch Instance” button

I launched two EC2 instances, one in a public subnet and another in a private subnet.

Public EC2 Instance:

- Name: ec2-public-a
- AMI: Ubuntu 24.04 LTS
- Instance Type: t3.micro
- Key Pair: umarsatti.pem
- Security Group: umar-public-sg

i-0654d82fb22de16a3 (ec2-public-a)		
Details Status and alarms Monitoring Security Networking Storage Tags		
Instance summary Info		
Instance ID	Public IPv4 address	Private IPv4 addresses
i-0654d82fb22de16a3	18.237.254.34 open address	10.0.0.138
IPv6 address	Instance state	Public DNS
-	Running	-
Hostname type	Private IP DNS name (IPv4 only)	Elastic IP addresses
IP name: ip-10-0-0-138.us-west-2.compute.internal	ip-10-0-0-138.us-west-2.compute.internal	-
Answer private resource DNS name	Instance type	AWS Compute Optimizer finding
-	t3.micro	No recommendations available for this instance.
Auto-assigned IP address	VPC ID	Auto Scaling Group name
18.237.254.34 [Public IP]	vpc-0728cd553c4f9f0c5 (umar-vpc)	-
IAM Role	Subnet ID	Managed
-	subnet-00f9f0b04e6e84e14 (public-subnet-a)	false
IMDSv2	Instance ARN	
Required	arn:aws:ec2:us-west-2:504649076991:instance/i-0654d82fb22de16a3	

Private EC2 Instance:

- Name: ec2-private-a
- AMI: Ubuntu 24.04 LTS
- Instance Type: t3.micro
- Key Pair: umarsatti.pem
- Security Group: umar-private-sg

The screenshot shows the AWS EC2 instance details page for 'ec2-private-a'. The instance ID is i-01c6d4ed4bdffd103. The instance is running (state: Running) and has a private IP of ip-10-0-2-30.us-west-2.compute.internal. It is an t3.micro instance type, associated with VPC ID vpc-0728cd553c4f9f0c5 and subnet subnet-08ad52da6fb1a9cd7. The instance ARN is arn:aws:ec2:us-west-2:504649076991:instance/i-01c6d4ed4bdffd103. The public IP is 10.0.2.30 and the public DNS is ip-10-0-2-30.us-west-2.compute.internal. The instance is not part of an Auto Scaling group and is managed by the user.

Task 1.8: Create IAM Role and Policy for VPC Flow Logs

Steps:

1. Navigate to the IAM service from the AWS Management Console search bar
2. Create a new policy that allows the VPC Flow Logs service to write to CloudWatch Logs
3. Create an IAM role and attach the custom policy to it
4. Add a trust relationship that allows the VPC Flow logs to assume the IAM role.

I created an IAM policy and IAM role to grant permission for VPC Flow Logs to send logs to CloudWatch

- Policy Permissions:
 - logs:CreateLogGroup
 - logs:CreateLogStream
 - logs:PutLogEvents
 - logs:DescribeLogGroups
 - logs:DescribeLogStreams
- IAM Role: umar-vpc-flowlogs-role
- Trusted Entity: vpc-flow-logs.amazonaws.com

```

Permissions defined in this policy Info
Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it.

1 - {
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Resource": "*"
    }
  ]
}

```

Permissions	Trust relationships	Tags	Last Accessed	Revoke sessions
	Trusted entities Entities that can assume this role under specified conditions. 1 - { "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Principal": { "Service": "vpc-flow-logs.amazonaws.com" }, "Action": "sts:AssumeRole" }] }			Edit trust policy

Task 1.9: Create CloudWatch Logs to publish VPC Flow Logs

Steps:

1. Navigate to “CloudWatch” service from the AWS Management Console search bar
2. Click on “Log groups” under “Logs” in the left navigation panel
3. Click on “Create log group” button
4. Choose a Log group name, Retention setting, and Log class.
5. Click on “Create” button

I created a CloudWatch Log group with the name “umar-vpc-logs” and retention period of “Never Expire”. This Log group will publish the logs generated from the VPC flow logs.

umar-vpc-logs		Actions ▾	View in Logs Insights	Start tailing	Search log group																																										
Log group details Log class: Info Standard ARN: arn:aws:logs:us-west-2:504649076991:log-group:umar-vpc-logs: Creation time: 18 hours ago Retention: Never expire Stored bytes: 652.35 KB		Metric filters: 0	Subscription filters: 0	Contributor Insights rules: -	KMS key ID: -																																										
		Anomaly detection: Configure		Data protection: -	Sensitive data count: -																																										
				Custom field indexes: Configure	Transformer: Configure																																										
Log streams Tags Anomaly detection Metric filters Subscription filters Contributor Insights Data protection Field indexes Transformer																																															
Log streams (10) By default, we only load the most recent log streams. <table border="1"> <thead> <tr> <th colspan="2">Filter log streams or try prefix search</th> <th><input type="checkbox"/> Exact match</th> <th><input type="checkbox"/> Show expired Info</th> <th>Delete</th> <th>Create log stream</th> <th>Search all log streams</th> </tr> </thead> <tbody> <tr> <td colspan="2"></td> <td><input type="checkbox"/> Log stream</td> <td><input type="checkbox"/> Last event time</td> <td colspan="3"></td> </tr> <tr> <td colspan="2">eni-0464224cd37f480f-all</td> <td></td> <td>2025-11-05 19:54:17 (UTC)</td> <td colspan="3"></td> </tr> <tr> <td colspan="2">eni-02294a427ac1b74f8-all</td> <td></td> <td>2025-11-05 19:50:36 (UTC)</td> <td colspan="3"></td> </tr> <tr> <td colspan="2">eni-0c752accef344a9-all</td> <td></td> <td>2025-11-05 19:50:33 (UTC)</td> <td colspan="3"></td> </tr> <tr> <td colspan="2">eni-09f320f31616e161a-all</td> <td></td> <td>2025-11-05 19:19:28 (UTC)</td> <td colspan="3"></td> </tr> </tbody> </table>						Filter log streams or try prefix search		<input type="checkbox"/> Exact match	<input type="checkbox"/> Show expired Info	Delete	Create log stream	Search all log streams			<input type="checkbox"/> Log stream	<input type="checkbox"/> Last event time				eni-0464224cd37f480f-all			2025-11-05 19:54:17 (UTC)				eni-02294a427ac1b74f8-all			2025-11-05 19:50:36 (UTC)				eni-0c752accef344a9-all			2025-11-05 19:50:33 (UTC)				eni-09f320f31616e161a-all			2025-11-05 19:19:28 (UTC)			
Filter log streams or try prefix search		<input type="checkbox"/> Exact match	<input type="checkbox"/> Show expired Info	Delete	Create log stream	Search all log streams																																									
		<input type="checkbox"/> Log stream	<input type="checkbox"/> Last event time																																												
eni-0464224cd37f480f-all			2025-11-05 19:54:17 (UTC)																																												
eni-02294a427ac1b74f8-all			2025-11-05 19:50:36 (UTC)																																												
eni-0c752accef344a9-all			2025-11-05 19:50:33 (UTC)																																												
eni-09f320f31616e161a-all			2025-11-05 19:19:28 (UTC)																																												

Task 1.10: Enable VPC Flow Logs for network traffic monitoring

Steps:

1. Navigate to “VPC” from the AWS Management Console search bar
2. Choose the target VPC and open the “Flow Logs” tab
3. Click “Create Flow Log”
4. Select “Destination” as CloudWatch Logs or S3
5. Choose the IAM role created in Task 1.8
6. Click “Create Flow Log”

I enabled VPC Flow Logs to capture detailed network traffic information for analysis and monitoring.

- VPC Flow Logs Name: umar-vpc-flowlogs
- Destination: CloudWatch Logs
- Log Group: umar-vpc-logs
- IAM Role: VPC-flowlogs-IAM-role

The screenshot shows the AWS VPC Flow Log configuration page for flow log ID fl-05c2542015588a617. The page is titled "fl-05c2542015588a617 / umar-vpc-flowlogs". It displays the following details:

Details	Destination Type	Traffic Type	File Format
Flow Log ID: fl-05c2542015588a617	cloud-watch-logs	ALL	-
Name: umar-vpc-flowlogs	Destination Name: umar-vpc-logs Edit	Max Aggregation Interval: 10 minutes	Hive Compatible Partitions
State: Active	IAM Role: arn:aws:iam::504649076991:role/VPC-flowlogs-IAM-role Edit	Log Format: Default	Partition Logs
Creation Time: Wednesday, November 5, 2025 at 21:37:20 GMT+5	Cross Account IAM Role: -		

Below the details section, there are two tabs: "Tags" (selected) and "Integrations". The "Tags" tab shows one tag: "Name" with value "umar-vpc-flowlogs". There is also a "Manage tags" button and a navigation bar with page 1 of 1.

The following image shows the CloudWatch Log Stream of public EC2 instance with Elastic Network Interface ID eni-02294a427ac1b74f8

The following image shows the CloudWatch Log Stream of private EC2 instance with Elastic Network Interface ID eni-0c7523accefe344a9

Task 1.11: Create S3 bucket to store VPC Flow logs

Steps:

1. Navigate to “S3” from the AWS Management Console search bar
2. Click on “Create bucket” button
3. Choose bucket type, name, and encryption settings
4. Click on “Create bucket” button

I created an S3 bucket to store VPC Flow logs

- **Bucket Name:** umar-vpc-flowlogs-s3-bucket
- **Object Ownership:** ACLs disabled
- **Block All Public Access:** Enabled
- **Versioning:** Disabled
- **Encryption:** SSE-S3 enabled

Task 1.12: Create VPC Endpoints for secure private connectivity to AWS services

Steps:

1. Navigate to “VPC” from the AWS Management Console search bar
2. Click on “Endpoints” in the left navigation panel
3. Click “Create endpoint”
4. Choose Name, Type, and Services, and VPC
5. Click on “Create endpoint” button

6. Additionally, create a VPC flow log with the same steps defined in Task 1.10 but change the Destination to “Send to an Amazon S3 bucket” by adding the bucket ARN

I created a VPC Endpoint for Amazon S3 to allow private communication between the VPC and S3 without using the public internet

- Endpoint Name: umar-vpc-endpoint-s3
- Service: com.amazonaws.us-west-2.s3
- Endpoint type: Gateway

The screenshot shows the AWS VPC Endpoints console. At the top, there is a search bar and a table listing two endpoints:

Name	VPC endpoint ID	Endpoint type	Status	Service name
project-vpce-s3	vpce-06946064ca33da0a6	Gateway	Available	com.amazonaws.us-west-2.s3
umar-vpc-endpoint-s3	vpce-0c0158e014e311101	Gateway	Available	com.amazonaws.us-west-2.s3

Below the table, a modal window displays the details for the selected endpoint:

Details / **umar-vpc-endpoint-s3**

Details

Endpoint ID vpce-0c0158e014e311101	Status Available	Creation time Wednesday, November 5, 2025 at 21:47:44 GMT+5	Endpoint type Gateway
VPC ID vpc-0728cd553c4f9f0c5 (umar-vpc)	Status message -	Service name com.amazonaws.us-west-2.s3	Private DNS names enabled No
DNS record IP type service-defined	IP address type ipv4	Service region us-west-2	

VPC Flow logs using S3 as the Destination type:

The screenshot shows the AWS VPC Flow Logs console. At the top, there is a search bar and a table listing one flow log:

Flow Log ID	Destination Type	Traffic Type	File Format
fl-03351a2e3d6697cb1	s3	All	Plain text

Below the table, a modal window displays the details for the selected flow log:

fl-03351a2e3d6697cb1 / umar-vpc-flowlogs-s3

Details

Flow Log ID fl-03351a2e3d6697cb1	Destination Type s3	Traffic Type All	File Format Plain text
Name umar-vpc-flowlogs-s3	Destination Name umar-vpc-flowlogs-s3-bucket	Max Aggregation Interval 10 minutes	Hive Compatible Partitions <input checked="" type="radio"/> Not enabled
State Active	IAM Role -	Log Format Default	Partition Logs Daily
Creation Time Wednesday, November 5, 2025 at 22:03:12 GMT+5	Cross Account IAM Role -		

Tags

Key	Value
Name	umar-vpc-flowlogs-s3

VPC Flow logs are stored inside the newly created S3 bucket as shown below:

Name	Type	Last modified	Size	Storage class
504649076991_vpcflowlogs_us-west-2_03351a2e3d6697cb1_2025110517002_b.log.gz	gz	November 5, 2025, 22:08:21 (UTC+05:00)	1.1 KB	Standard
504649076991_vpcflowlogs_us-west-2_03351a2e3d6697cb1_2025110517052_1.log.gz	gz	November 5, 2025, 22:13:21 (UTC+05:00)	2.2 KB	Standard
504649076991_vpcflowlogs_us-west-2_03351a2e3d6697cb1_2025110517052_e2742b1ff.log.gz	gz	November 5, 2025, 22:08:21 (UTC+05:00)	2.8 KB	Standard
504649076991_vpcflowlogs_us-west-2_03351a2e3d6697cb1_2025110517102_6.log.gz	gz	November 5, 2025, 22:13:21 (UTC+05:00)	2.5 KB	Standard

Task 1.13: Test connectivity between subnets and to the internet

Steps:

1. Navigate to the “EC2” service from the AWS Management Console search bar
2. In the left navigation panel, select “Instances”
3. Select the instance and click “Connect” button
4. Choose “EC2 Instance Connect” to SSH into the instance
5. Make sure the key pair has the necessary permissions by using “chmod 400 <keypair.pem> command”
6. Connect to the EC2 instance using the command SSH command.
7. For Ubuntu instances, use “ssh -I <keypair.pem> ubuntu@<ip address>” command
8. For Amazon Linux instances, use “ssh -I <keypair.pem> ec2-user@<ip address>” command

I downloaded the keypair and changed its permission by using “chmod 400 umarsatti.pem” command. Instead of using EC2 Instance Connect or a local terminal on my computer, I used an SSH Client (MobaXterm) to connect to the EC2 instance.

The image below shows the ping command from the EC2 in public subnet A with the internal IP 10.0.0.62 to EC2 in private subnet A with the internal IP 10.0.2.229:

```

ubuntu@ip-10-0-0-62:~$ whoami
ubuntu
ubuntu@ip-10-0-0-62:~$ ping google.com
PING google.com (142.250.73.110) 56(84) bytes of data.
64 bytes from pnseaa-an-in-f14.1e100.net (142.250.73.110): icmp_seq=1 ttl=117 time=5.52 ms
64 bytes from pnseaa-an-in-f14.1e100.net (142.250.73.110): icmp_seq=2 ttl=117 time=5.57 ms
64 bytes from pnseaa-an-in-f14.1e100.net (142.250.73.110): icmp_seq=3 ttl=117 time=5.59 ms
64 bytes from pnseaa-an-in-f14.1e100.net (142.250.73.110): icmp_seq=4 ttl=117 time=5.54 ms
64 bytes from pnseaa-an-in-f14.1e100.net (142.250.73.110): icmp_seq=5 ttl=117 time=5.60 ms
64 bytes from pnseaa-an-in-f14.1e100.net (142.250.73.110): icmp_seq=6 ttl=117 time=5.56 ms
^C
--- google.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5009ms
rtt min/avg/max/mdev = 5.517/5.563/5.601/0.027 ms
ubuntu@ip-10-0-0-62:~$ ping 10.0.2.229
PING 10.0.2.229 (10.0.2.229) 56(84) bytes of data.
64 bytes from 10.0.2.229: icmp_seq=1 ttl=127 time=0.560 ms
64 bytes from 10.0.2.229: icmp_seq=2 ttl=127 time=0.173 ms
64 bytes from 10.0.2.229: icmp_seq=3 ttl=127 time=0.185 ms
64 bytes from 10.0.2.229: icmp_seq=4 ttl=127 time=0.182 ms
64 bytes from 10.0.2.229: icmp_seq=5 ttl=127 time=0.164 ms
64 bytes from 10.0.2.229: icmp_seq=6 ttl=127 time=0.176 ms
64 bytes from 10.0.2.229: icmp_seq=7 ttl=127 time=0.183 ms
^C
--- 10.0.2.229 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6156ms
rtt min/avg/max/mdev = 0.164/0.231/0.560/0.134 ms
ubuntu@ip-10-0-0-62:~$ █

```

The image below shows the ping command from the EC2 in private subnet A with the internal IP 10.0.2.229 to EC2 in public subnet A with the internal IP 10.0.0.62:

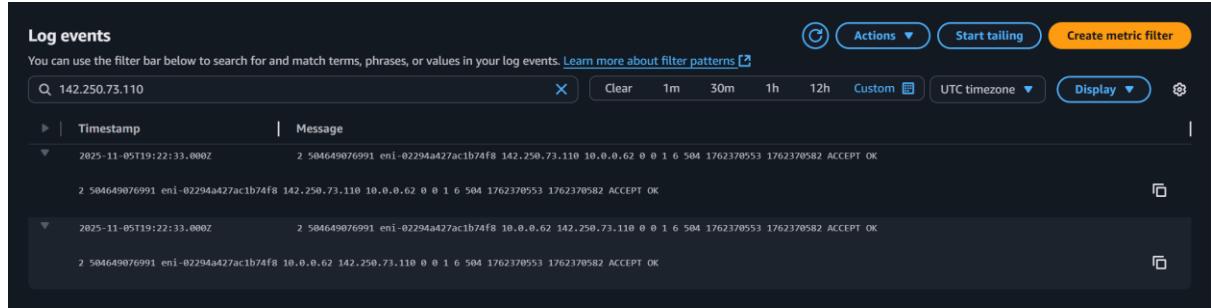
```

[ec2-user@ip-10-0-2-229 ~]$ whoami
ec2-user
[ec2-user@ip-10-0-2-229 ~]$ ls
[ec2-user@ip-10-0-2-229 ~]$ ping google.com
PING google.com (142.250.73.78) 56(84) bytes of data.
64 bytes from pnseaa-am-in-f14.1e100.net (142.250.73.78): icmp_seq=1 ttl=116 time=6.14 ms
64 bytes from pnseaa-am-in-f14.1e100.net (142.250.73.78): icmp_seq=2 ttl=116 time=5.65 ms
64 bytes from pnseaa-am-in-f14.1e100.net (142.250.73.78): icmp_seq=3 ttl=116 time=5.65 ms
64 bytes from pnseaa-am-in-f14.1e100.net (142.250.73.78): icmp_seq=4 ttl=116 time=5.71 ms
64 bytes from pnseaa-am-in-f14.1e100.net (142.250.73.78): icmp_seq=5 ttl=116 time=5.70 ms
64 bytes from pnseaa-am-in-f14.1e100.net (142.250.73.78): icmp_seq=6 ttl=116 time=5.65 ms
64 bytes from pnseaa-am-in-f14.1e100.net (142.250.73.78): icmp_seq=7 ttl=116 time=5.65 ms
^C
--- google.com ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6007ms
rtt min/avg/max/mdev = 5.645/5.735/6.137/0.165 ms
[ec2-user@ip-10-0-2-229 ~]$ ping 10.0.0.62
PING 10.0.0.62 (10.0.0.62) 56(84) bytes of data.
64 bytes from 10.0.0.62: icmp_seq=1 ttl=64 time=0.219 ms
64 bytes from 10.0.0.62: icmp_seq=2 ttl=64 time=0.193 ms
64 bytes from 10.0.0.62: icmp_seq=3 ttl=64 time=0.280 ms
64 bytes from 10.0.0.62: icmp_seq=4 ttl=64 time=0.192 ms
64 bytes from 10.0.0.62: icmp_seq=5 ttl=64 time=0.191 ms
64 bytes from 10.0.0.62: icmp_seq=6 ttl=64 time=0.216 ms
64 bytes from 10.0.0.62: icmp_seq=7 ttl=64 time=0.202 ms
^C
--- 10.0.0.62 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6264ms
rtt min/avg/max/mdev = 0.191/0.213/0.280/0.029 ms
[ec2-user@ip-10-0-2-229 ~]$ █

```

Both the public and private instances can ping google.com which determines that both instances have access to the internet.

CloudWatch Logs to indicate successful ping to google.com (142.250.73.110)



The screenshot shows a CloudWatch Log Events interface. The search bar at the top contains the query "142.250.73.110". Below the search bar are various filter and display options: Actions, Start tailing, Create metric filter, Clear, 1m, 30m, 1h, 12h, Custom, UTC timezone, Display, and a gear icon. The log entries are listed in a table with two columns: Timestamp and Message. The first message is timestamped "2025-11-05T19:22:33.000Z" and has the message "2 504649076991 eni-02294a427ac1b74f8 142.250.73.110 10.0.0.62 0 0 1 6 504 1762370553 1762370582 ACCEPT OK". The second message is timestamped "2025-11-05T19:22:33.000Z" and has the message "2 504649076991 eni-02294a427ac1b74f8 10.0.0.62 142.250.73.110 0 0 1 6 504 1762370553 1762370582 ACCEPT OK". The third message is timestamped "2025-11-05T19:22:33.000Z" and has the message "2 504649076991 eni-02294a427ac1b74f8 10.0.0.62 142.250.73.110 0 0 1 6 504 1762370553 1762370582 ACCEPT OK".

Timestamp	Message
2025-11-05T19:22:33.000Z	2 504649076991 eni-02294a427ac1b74f8 142.250.73.110 10.0.0.62 0 0 1 6 504 1762370553 1762370582 ACCEPT OK
2025-11-05T19:22:33.000Z	2 504649076991 eni-02294a427ac1b74f8 10.0.0.62 142.250.73.110 0 0 1 6 504 1762370553 1762370582 ACCEPT OK
2025-11-05T19:22:33.000Z	2 504649076991 eni-02294a427ac1b74f8 10.0.0.62 142.250.73.110 0 0 1 6 504 1762370553 1762370582 ACCEPT OK

CloudWatch Logs indicating successful ping from public EC2 instance (10.0.0.62) to private EC2 instance (10.0.2.229)



The screenshot shows a CloudWatch Log Events interface. The search bar at the top contains the query "There are older events to load. Load more.". Below the search bar are various filter and display options: Actions, Start tailing, Create metric filter, Clear, 1m, 30m, 1h, 12h, Custom, UTC timezone, Display, and a gear icon. The log entries are listed in a table with two columns: Timestamp and Message. The first message is timestamped "2025-11-05T19:27:09.000Z" and has the message "2 504649076991 eni-02294a427ac1b74f8 10.0.2.229 10.0.0.62 22 54248 6 5 1709 1762370829 1762370849 ACCEPT OK". The second message is timestamped "2025-11-05T19:27:09.000Z" and has the message "2 504649076991 eni-02294a427ac1b74f8 10.0.2.229 10.0.0.62 22 54248 6 5 1709 1762370829 1762370849 ACCEPT OK". The third message is timestamped "2025-11-05T19:27:09.000Z" and has the message "2 504649076991 eni-02294a427ac1b74f8 10.0.0.62 119.73.124.211 22 35067 6 539 54520 1762370829 1762370849 ACCEPT OK".

Timestamp	Message
2025-11-05T19:27:09.000Z	2 504649076991 eni-02294a427ac1b74f8 10.0.2.229 10.0.0.62 22 54248 6 5 1709 1762370829 1762370849 ACCEPT OK
2025-11-05T19:27:09.000Z	2 504649076991 eni-02294a427ac1b74f8 10.0.2.229 10.0.0.62 22 54248 6 5 1709 1762370829 1762370849 ACCEPT OK
2025-11-05T19:27:09.000Z	2 504649076991 eni-02294a427ac1b74f8 10.0.0.62 119.73.124.211 22 35067 6 539 54520 1762370829 1762370849 ACCEPT OK