

Enterprise Secure Network Architecture on AWS

Author: Adedayo

Specialization: Cloud Security & Network Architecture

Platform: Amazon Web Services (AWS)

## 1. Introduction

This document describes the design and implementation of a secure, multi-tier AWS network architecture where only load balancers are publicly accessible, while application and database resources remain fully isolated in private networks.

I designed this architecture to support high availability, strong network segmentation, least-privilege access, and continuous security monitoring.

## 2. Objectives

The primary objectives of this implementation were to:

- Isolate application and database resources from the public internet
- Enforce strong network segmentation
- Support high availability across Availability Zones
- Minimize attack surface
- Enable secure internal communication
- Maintain continuous monitoring and audit visibility

## 3. VPC and Subnet Architecture

### 3.1 VPC Design

I created a Virtual Private Cloud (VPC) with a CIDR range of 10.0.0.0/16 to support long-term scalability.

### 3.2 Subnet Layout

Six subnets were created across two Availability Zones:

- Two public subnets for internet-facing resources
- Two private subnets for application workloads
- Two private subnets for database workloads

The screenshot shows the AWS VPC Subnets page. On the left, there's a navigation sidebar with sections like 'Virtual private cloud' (Your VPCs, Subnets, Route tables, Internet gateways, Egress-only Internet gateways, Carrier gateways, DHCP option sets, Elastic IPs, Managed prefix lists, NAT gateways, Peering connections, Route servers), 'Security' (Network ACLs, Security groups), and 'PrivateLink and Lattice'. The main area has a green banner at the top stating 'You have successfully deleted subnet-04b4d68defaa53fb'. Below it is a table titled 'Subnets (6) Info' with columns: Name, Subnet ID, State, VPC, Block Public..., and IPv4 CIDR. The subnets listed are: Fortinet Public Subnet A, Fortinet Public Subnet B, Fortinet Private Subnet A, Fortinet Private Subnet B, Fortinet DB private Subnet A, and fortinet DB private Subnet B. All subnets are marked as 'Available' and have their respective VPC IDs and IPv4 CIDRs.

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
Fortinet Public Subnet A	subnet-039a5a9c1f73c5b38	Available	vpc-04b6ad3ee924cda7a   fort...	Off	10.0.1.0/24
Fortinet Public Subnet B	subnet-0b82cc553403e22ac	Available	vpc-04b6ad3ee924cda7a   fort...	Off	10.0.2.0/24
Fortinet Private Subnet A	subnet-0e41f37070efb38ab	Available	vpc-04b6ad3ee924cda7a   fort...	Off	10.0.11.0/24
Fortinet Private Subnet B	subnet-0e1cde4d4bb411aba	Available	vpc-04b6ad3ee924cda7a   fort...	Off	10.0.12.0/24
Fortinet DB private Subnet A	subnet-0c3bed553e2e79a48	Available	vpc-04b6ad3ee924cda7a   fort...	Off	10.0.21.0/24
fortinet DB private Subnet B	subnet-09767994e3b2d7fea	Available	vpc-04b6ad3ee924cda7a   fort...	Off	10.0.22.0/24

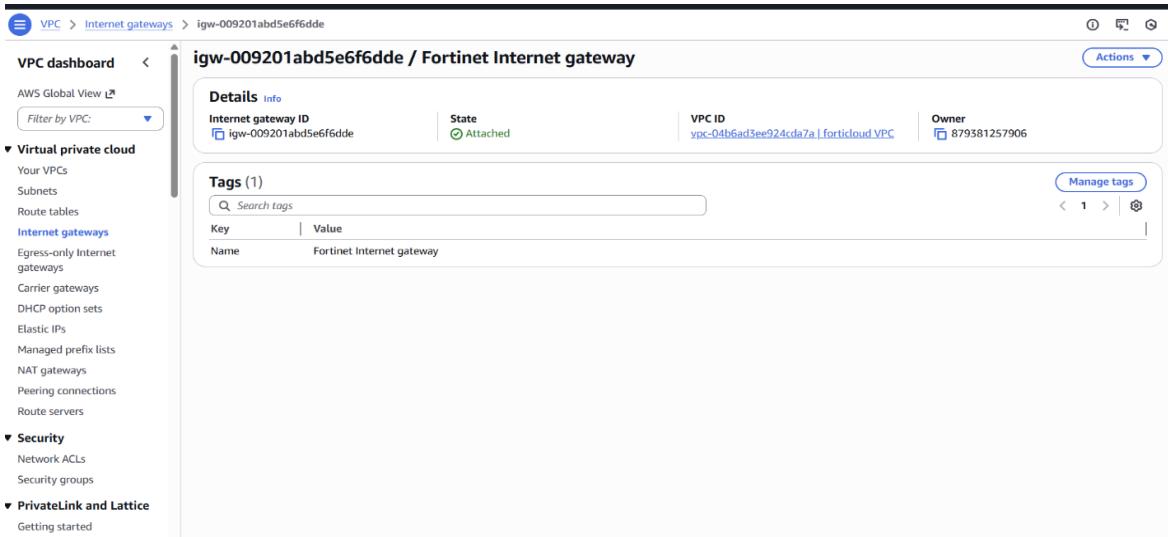
Each Availability Zone contains:

- One public subnet
- One private application subnet
- One private database subnet

This design ensures fault tolerance and strong isolation between tiers.

#### 4. Internet Gateway and Routing

An Internet Gateway was created and attached to the VPC.



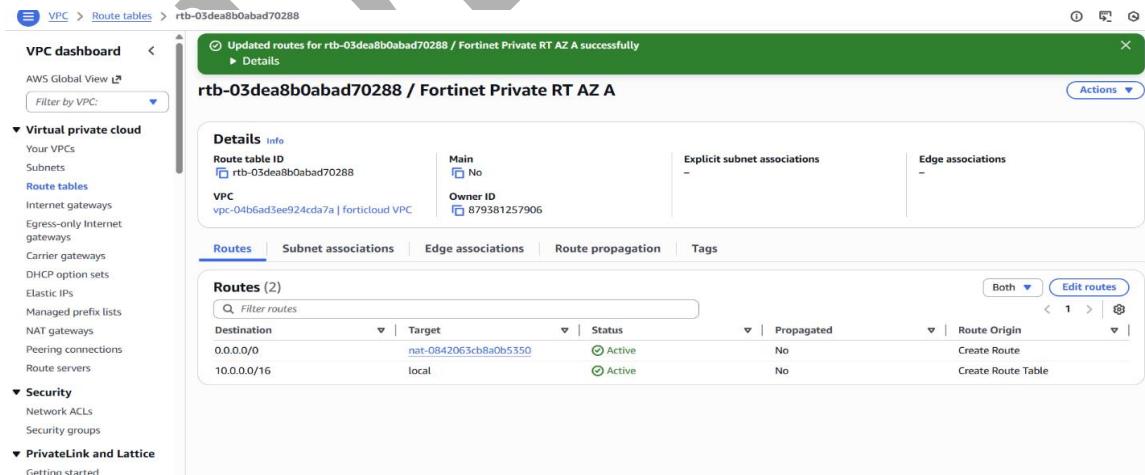
The screenshot shows the AWS VPC console. In the left sidebar, under 'Virtual private cloud', 'Internet gateways' is selected. The main pane displays 'igw-009201abd5e6f6dde / Fortinet Internet gateway'. The 'Details' tab is active, showing the Internet gateway ID (igw-009201abd5e6f6dde), State (Attached), VPC ID (vpc-04b6ad3ee924cda7a | forticloud VPC), and Owner (879381257906). A 'Tags (1)' section contains a single tag: 'Name' with the value 'Fortinet Internet gateway'. An 'Actions' dropdown menu is visible in the top right.

Public route tables were configured to direct outbound internet traffic to the Internet Gateway and associated with public subnets.

This ensured that only approved public resources had direct internet access.

## 5. NAT Gateway Configuration

I deployed a NAT Gateway in each Availability Zone and routed private subnet traffic to the local NAT Gateway.



The screenshot shows the AWS VPC console. In the left sidebar, under 'Virtual private cloud', 'Route tables' is selected. The main pane displays 'rtb-03dea8b0abad70288 / Fortinet Private RT AZ A'. A green success message at the top states: 'Updated routes for rtb-03dea8b0abad70288 / Fortinet Private RT AZ A successfully' with a 'Details' link. The 'Details' tab is active, showing the Route table ID (rtb-03dea8b0abad70288), Main (No), Owner ID (879381257906), and Explicit subnet associations and Edge associations (both empty). The 'Routes' tab is selected, showing two routes:

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	nat-0842063cb0b0b5350	Active	No	Create Route
10.0.0.0/16	local	Active	No	Create Route Table

**Details Info**

Route table ID rtb-0e0fde5d344d5b8e3	Main No	Explicit subnet associations 2 subnets	Edge associations -
Owner ID vpc-04b6ad5ee924cda7a   forticloud VPC			

**Routes** (2)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	nat-068ff4d6ca61012f3	Active	No	Create Route
10.0.0.16	local	Active	No	Create Route Table

This design:

- Prevented cross-AZ dependencies
- Improved fault isolation
- Increased availability
- Followed AWS best practices

Private resources were able to access required external services without being directly exposed.

## 6. Security Group Architecture

Security groups were implemented for each architectural tier.

### 6.1 Load Balancer Security Group

- Inbound: HTTPS (443) from 0.0.0.0/0
- Outbound: Application Security Group only

VPC > Security Groups > sg-03e23fa0f68bd471e - Fortinet ALB SG

**sg-03e23fa0f68bd471e - Fortinet ALB SG**

**Details**

Security group name Fortinet ALB SG	Security group ID sg-03e23fa0f68bd471e	Description Public Subnets security group	VPC ID vpc-04b6ad3ee924cda7a
Owner 879381257906	Inbound rules count 1 Permission entry	Outbound rules count 1 Permission entry	

**Inbound rules (1)**

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-00a41fd2f83fdae5b	IPv4	HTTPS	TCP	443

## 6.2 Application Security Group

- Inbound: Application traffic from ALB Security Group
- Outbound: PostgreSQL (5432) to Database Security Group

VPC > Security Groups > sg-09a6ff30b6513890a - Fortinet App SG

**sg-09a6ff30b6513890a - Fortinet App SG**

**Details**

Security group name Fortinet App SG	Security group ID sg-09a6ff30b6513890a	Description allow traffic from Fortinet Public Subnet.	VPC ID vpc-04b6ad3ee924cda7a
Owner 879381257906	Inbound rules count 1 Permission entry	Outbound rules count 1 Permission entry	

**Inbound rules (1)**

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-08a15ea7ff0bd7247	-	Custom TCP	TCP	8080

## 6.3 Database Security Group

**sg-09d808af526ba9711 - Fortinet DB security Group**

**Details**

Security group name Fortinet DB security Group	Security group ID sg-09d808af526ba9711	Description allow traffic from App Security Group.	VPC ID vpc-04b6ad3ee924cd7a
Owner 879381257906	Inbound rules count 1 Permission entry	Outbound rules count 0 Permission entries	

**Inbound rules (1)**

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-0f88ef47c686c9a0d	-	PostgreSQL	TCP	5432

- Inbound: PostgreSQL (5432) from Application Security Group
- Outbound: None

This structure enforced strict tier-to-tier communication and least privilege.

## 7. Network Access Control Lists (NACLs)

Network ACLs were configured to provide an additional layer of subnet-level protection.

### 7.1 Public Subnet NACL

Inbound:

- HTTPS (443) from internet
- Ephemeral ports (1024–65535)

Outbound:

- HTTPS (443)
- Ephemeral ports

All other traffic was denied.

The screenshot shows the AWS VPC Network ACLs interface. On the left, there's a navigation sidebar with options like VPC dashboard, Virtual private cloud, Security, and PrivateLink and Lattice. The main area displays a table titled "Network ACLs (1/5) Info". The table has columns for Name, Network ACL ID, Associated with, Default, VPC ID, and Inbound. It lists four entries: "acl-04970c3d76dfcf683" (Default Yes, VPC ID vpc-07026deb7f5daebc5, 2 Inbx), "acl-069af0d0e664287d9" (Default Yes, VPC ID vpc-04b6ad3ee924cda7a / forticloud VPC, 2 Inbx), "acl-09002fa1cede13ef9a" (Default Yes, VPC ID vpc-0d0014fc2012c7b67, 2 Inbx), and "Fortinet Public NACLs" (Default No, VPC ID vpc-04b6ad3ee924cda7a / forticloud VPC, 3 Inbx). Below this, a specific Network ACL is selected: "acl-0bdadb6916846af28 / Fortinet Public NACLs". The "Inbound rules" tab is active, showing a table with three rules:

Rule number	Type	Protocol	Port range	Source	Allow/Deny
100	HTTPS (443)	TCP (6)	443	0.0.0.0/0	Allow
110	Custom TCP	TCP (6)	1024 - 65535	0.0.0.0/0	Allow
*	All traffic	All	All	0.0.0.0/0	Deny

## 7.2 Private Subnet NACL

Inbound:

- Application traffic from public subnets
- PostgreSQL from application subnets
- Ephemeral ports

The screenshot shows the AWS VPC Network ACLs interface. The left sidebar is identical to the previous one. The main area displays a table titled "Network ACLs (1/5) Info". The table has columns for Name, Network ACL ID, Associated with, Default, VPC ID, and Inbound. It lists four entries: "acl-04970c3d76dfcf683" (Default Yes, VPC ID vpc-07026deb7f5daebc5, 2 Inbx), "acl-069af0d0e664287d9" (Default Yes, VPC ID vpc-04b6ad3ee924cda7a / forticloud VPC, 2 Inbx), "acl-09002fa1cede13ef9a" (Default Yes, VPC ID vpc-0d0014fc2012c7b67, 2 Inbx), and "Fortinet Public NACLs" (Default No, VPC ID vpc-04b6ad3ee924cda7a / forticloud VPC, 3 Inbx). Below this, a specific Network ACL is selected: "acl-04b469beb5ef5f765 / Fortinet Private NACLs". The "Inbound rules" tab is active, showing a table with six rules:

Rule number	Type	Protocol	Port range	Source	Allow/Deny
100	HTTP* (8080)	TCP (6)	8080	10.0.1.0/24	Allow
105	HTTP* (8080)	TCP (6)	8080	10.0.2.0/24	Allow
110	PostgreSQL (5432)	TCP (6)	5432	10.0.11.0/24	Allow
115	PostgreSQL (5432)	TCP (6)	5432	10.0.12.0/24	Allow
120	Custom TCP	TCP (6)	1024 - 65535	0.0.0.0/0	Allow
*	All traffic	All	All	0.0.0.0/0	Deny

Outbound:

- HTTPS via NAT Gateway
- Database communication
- Ephemeral ports

All other traffic was denied.

The screenshot shows the AWS VPC dashboard with the 'Network ACLs' section selected. A success message at the top indicates that outbound rules for 'Fortinet Private NACLs' have been updated. The main table lists five Network ACLs:

Name	Network ACL ID	Associated with	Default	VPC ID	Inbound rules
acl-069af0d0e664287d9	-	-	Yes	vpc-04b6ad3ee924cda7a / forticloud VPC	2 Inbx
acl-09002fa1ede13ef9a	-	-	Yes	vpc-0d0014fc2012e7b67	2 Inbx
Fortinet Public NACLs	acl-0b0dab6916846af28	2 Subnets	No	vpc-04b6ad3ee924cda7a / forticloud VPC	3 Inbx
Fortinet Private NACLs	acl-04b469beb5ef5f765	4 Subnets	No	vpc-04b6ad3ee924cda7a / forticloud VPC	6 Inbx

Below the table, a specific Network ACL named 'acl-04b469beb5ef5f765 / Fortinet Private NACLs' is selected. The 'Outbound rules' section shows five entries:

Rule number	Type	Protocol	Port range	Destination	Allow/Deny
100	HTTPS (443)	TCP (6)	443	0.0.0.0/0	Allow
110	PostgreSQL (5432)	TCP (6)	5432	10.0.21.0/24	Allow
115	PostgreSQL (5432)	TCP (6)	5432	10.0.22.0/24	Allow
120	Custom TCP	TCP (6)	1024 - 65535	0.0.0.0/0	Allow
*	All traffic	All	All	0.0.0.0/0	Deny

This configuration enforced hard boundaries between network zones.

## 8. Compute and Database Deployment

EC2 instances were deployed in private application subnets with restricted security groups.

An RDS PostgreSQL database was deployed in private database subnets using a dedicated subnet group.

No database resources were exposed to the public internet.

## 9. Secure Connectivity Testing

Connectivity testing was performed using AWS Session Manager.

An IAM role was attached to EC2 instances to enable Session Manager access.

PostgreSQL client tools were installed to validate database connectivity.

A screenshot of a terminal window titled "Shortcuts" with an Instance ID of i-077de260224b80b74. The session ID is root-9yr8i85pbva4s8oabt6h8jobgu. The terminal shows a successful connection to a PostgreSQL database on port 5432. The output includes PostgreSQL version information (15.19, server 17.4), a warning about SSL connection features, and a note about compression settings. The prompt "postgres=>" is visible at the bottom.

Successful tests confirmed:

- Correct routing
- Proper security group configuration
- Valid NACL rules
- No direct internet exposure

## 10. Monitoring and Visibility

To maintain continuous visibility, the following services were enabled:

- VPC Flow Logs to CloudWatch Logs
- AWS Security Hub
- AWS Config

These services provided insight into network activity, misconfigurations, and compliance status.

## 11. Incident Response Procedures

An incident response process was designed to address potential exposure.

Detection:

- Security Hub findings
- AWS Config change history
- VPC Flow Log analysis

Investigation:

- Review affected resources
- Analyze configuration changes
- Examine access patterns

Containment:

- Tighten security group rules
- Update NACL configurations
- Remove unintended public access

Validation:

- Re-test connectivity
- Confirm isolation
- Verify compliance

## 12. Outcomes and Impact

This implementation delivered the following results:

- Secure multi-tier network architecture
- Strong isolation of sensitive resources
- Reduced attack surface
- High availability design
- Improved audit readiness
- Continuous security visibility

### 13. Conclusion

I designed and implemented a secure, highly available AWS network architecture that enforces least privilege, strong segmentation, and continuous monitoring.

This solution supports secure operations, protects sensitive workloads, and aligns with enterprise security standards.

ADEDAYO