Networking

The Digital networking infrastructure will be the primary target for exploitation and external threats as it is the portal where users interact with the financial application . With the increase in threat complexity, it is important to harden these financial networks as they continuously expand due to the need for commerce and data.The figure below represents the network structure without any additional services for better visualisation.

The following table depicts the various services as well as their role they have in securing the network.

Service/feature name	Function and role
Elastic load balancer	ELB manages incoming traffic to applications that aids in distributing traffic between applications over multiple availability zones. This ensures each zone is not overloaded by an overwhelming amount of customers thus maintaining availability and performance
Virtual private cloud	The VPC that contains all Network, service and processes used for financial business function within an isolated and secure environment. Through the VPC customers can access the financial application and interact with the database.
API gateway	This service allows the management of apis calls from administrators and users. It is through the API gateway the administrators can update user settings and customers access various aspects of the vpc.

VPC and subnets

Virtual Private Cloud

The VPC is where the cloud solution will be based; it is a virtualized cloud that is isolated from other VPCs running on the same shared hardware. This is where the PaaS and other services such as EC2 will be deployed and run from; along with the various security measures such as security groups, route tables and subnets.

Implementation

1. In **AWS dashboard** select **VPC**
2. Tag accordingly with a meaningful description that represents the purpose of the **VPC**
3. Create **VPC** and **apply IPV4**
    a. To simplify the **CIDAR block usage**, a default ipv4 example provided by AWS will be used.
4. Leave tenancy as default
5. Select create

Subnets

AWS recommends having both a mixture of public, private and isolated subnets. This is to enhance the security of the VPC as an isolated subnet can only be accessed from within the network. This is where databases and important functions are kept. Therefore a primary public subnet is used to access a private subnet that communicates with the subnet holding the databases. The first network structure is cloned into identical availability zones in case one zone has downtime.

These subnets make up the network where the various security, IAM and storage components will be implemented in. It is through the subnets that the customers will interact with the product.

Implementation

1. In **AWS console** select **VPC** and then the subnet
2. Create **Subnet** and link it to the respective **VPC**
3. Ensure you place the correct **IPV4 block** and **availability zone**
4. Add **tag** and **name** and press create

Routing tables

Traffic will be manually routed via predefined source and destination addresses. These are the routes that user api calls and data will be routed through.

Implementation

1. In the **VPC** module within the **AWS console**, select **routing tables option**
2. Create **table** and assign the **appropriate subnet/s** to it
3. Select the relevant **VPC** and provide a name

Internet Gateway	
Dest	**Route**
0.0.0.0/0	**Igw-id**

Main route table	
Dest	**Route**
10.0.0.0/16	**local**
0.0.0.0/0	**Nat gateway**

custom internet route table	
Dest	**Route**
10.0.0.0/16	**local**
0.0.0.0/0	**lgw-id**

Public route table 1	
Dest	**Route**
10.0.0.0/24	**local**
0.0.0.0/0	**igw-id**

Private subnet route table 1	
Dest	**Route**
10.0.1.0/24	**local**
0.0.0.0/0	**Nat gateway**

private subnet route table 3	
Dest	**Route**
10.0.2.0/24	**local**
0.0.0.0/0	**Nat gateway**

Public route table 2	
Dest	**Route**
10.0.3.0/24	**local**
0.0.0.0/0	**igw-id**

Private subnet route table 2	
Dest	**Route**

10.0.4.0/24	**local**
0.0.0.0/0	**Nat gateway**

private subnet route table 4	
Dest	**Route**
10.0.5.0/24	**local**
0.0.0.0/0	**Nat gateway**

Gateways

NAT gateways are the primary method of giving the network public internet access. VPN access on the other hand only allows specific users with the right authentication credentials to privately access the network (primarily used for administration).

Implementation

1. In the **VPC module** within the **AWS console**, select **NAT Gateway settings**.
2. Assign the necessary **subnet**.
3. Ensure that connectivity type is **public** and the right **elastic ip ID** is allocated.
4. Provide name and tag then press create.

NACL

NACLs are stateless rules that control traffic based on pre-configured rulesets for both inbound and outbound connections.

Implementation

1. In the **VPC module** within the **AWS console**, select **network ACL.**
2. When creating an **ACL**, assign the **relevant VPC** and provide a name for the **ACL**.
3. Click create.

Security groups

These are stateless firewalls that allow/block certain inbound and outbound connections e.g. allow HTTPS connections but block SSH. Unlike ACLs, security groups are stateful meaning these rules save previous network information. The users information will be received through primarily https connection adding a layer of security. If https connections are not possible, a fallback http option is present. SSH will also be the primary access protocol for

administrators of the product to manage and change user permissions, resource use and product settings.

Implementation

1. In the **VPC module** within the **AWS console**, select **security groups**
2. Click create group and select the **applicable VPC**
3. Create both **inbound** and **outbound rule** for **ssh** and **https connections**
4. Write **description** of **security group rules** and provide optional tags
5. Click create.

Two main security groups will be created:

SSH Inbound	IP Version	Type	Protocol	port	dest
	IPv4	SSH	TCP	22	0.0.0.0/0

SSH Outbound	IP Version	Type	Protocol	port	dest
	IPv4	SSH	TCP	22	0.0.0.0/0

HTTPS Inbound	IP Version	Type	Protocol	port	dest
	IPv4	HTTPS	TCP	443	0.0.0.0/0

HTTPS Outbound	IP Version	Type	Protocol	port	dest
	IPv4	HTTPS	TCP	443	0.0.0.0/0

HTTP Inbound	IP Version	Type	Protocol	port	dest
	IPv4	HTTP	TCP	80	0.0.0.0/0

HTTP Outbound	IP Version	Type	Protocol	port	dest
	IPv4	HTTP	TCP	80	0.0.0.0/0