3 Tier Architecture

Objective: - Web Tier hosting a IIS Server website with a button which triggers the connection to backend api server and then backend api server logs that record in SQL DB Implement 3 tier application architecture considering 3 Virtual Networks and required subnets as a tier, in addition to this, provision 1 windows iis server and 1 apache linux server for Web tier, backend api for App tier and a SQL Database in DB tier following the below conditions: 1. No tier should have outbound access to internet (outbound internet access can be restricted after installing iis and apache server, packages required to host backend api or installing SQL DB) 2. Only Web and App tier should have inbound internet access 3. No private connectivity should be allowed between any tier without a must-have requirement 4. Individual NSG must be associated with subnets of each tier

To implement a 3-tier architecture using three Virtual Networks (VNets) and the specified conditions, you'll need to provision and configure several components. Below is a detailed plan to set up this architecture.

1. Create VNets and Subnets

Create three separate VNets for each tier:

- **Web Tier VNet**
  - Subnet: `WebSubnet`

- **App Tier VNet**
  - Subnet: `AppSubnet`

- **DB Tier VNet**
  - Subnet: `DBSubnet`

2. Provision Resources

Provision the necessary resources in each VNet:

- **Web Tier VNet**
  - Windows VM with IIS Server
  - Linux VM with Apache Server

- **App Tier VNet**

  - VM to host backend API (this can be a Windows or Linux VM, depending on your API requirements)

- **DB Tier VNet**

  - SQL Database Server VM (likely a Windows VM with SQL Server installed)

3. Network Security Groups (NSGs)

Create and associate individual NSGs with each subnet:

- **WebSubnet NSG**

  - Allow inbound HTTP (80) and HTTPS (443) from the internet

  - Deny outbound internet access

- **AppSubnet NSG**

  - Allow inbound HTTP (80) and HTTPS (443) only from WebSubnet

  - Deny outbound internet access

- **DBSubnet NSG**

  - Allow inbound SQL traffic (e.g., port 1433 for SQL Server) only from AppSubnet

  - Deny outbound internet access

4. Route Tables and UDRs (User Defined Routes)

Configure routing to ensure proper communication flow between the tiers and prevent any unnecessary outbound internet access:

- **Web Tier Route Table**

  - Define UDR to direct traffic to AppSubnet

- **App Tier Route Table**
  - Define UDR to direct traffic to DBSubnet

5. Install and Configure Software

Before restricting outbound internet access, ensure the necessary software is installed on each VM:

- **Web Tier VMs**
  - Install IIS on the Windows VM
  - Install Apache on the Linux VM

- **App Tier VM**
  - Install the required backend API software

- **DB Tier VM**
  - Install SQL Server or your preferred SQL database software

6. Security Rules and NSG Configuration

Define NSG rules to restrict access as specified:

- **Web Tier NSG**
  - Inbound: Allow HTTP/HTTPS from any source (internet)
  - Outbound: Deny all

- **App Tier NSG**
  - Inbound: Allow HTTP/HTTPS from WebSubnet
  - Outbound: Deny all

- **DB Tier NSG**
  - Inbound: Allow SQL traffic from AppSubnet

- Outbound: Deny all

7. Testing and Validation

- Ensure that the web tier can serve the website.

- Test the button on the website to confirm it triggers the backend API.

- Verify that the backend API can log records in the SQL database.

- Validate that no tier has outbound internet access.

8. Maintenance

- Regularly update NSG rules to ensure security.

- Monitor network traffic to identify and resolve potential security breaches.

By following these steps, you can implement a 3-tier architecture that meets the given conditions using Azure VNets and NSGs.