***Network Security Groups (NSGs)***

**Definition:**  
A Network Security Group (NSG) is a set of security rules that allow or deny network traffic to and from Azure resources in an Azure virtual network. NSGs can be associated with subnets or individual network interfaces of virtual machines.

**Key Features:**

1. **Security Rules:**
   * NSGs contain inbound and outbound security rules that control network traffic based on source and destination IP addresses, ports, and protocols.
   * Each rule specifies the direction (inbound or outbound), source and destination, port numbers, and action (allow or deny).
2. **Associations:**
   * NSGs can be associated with:
     + **Subnets:** All resources in the subnet inherit the NSG rules.
     + **Network Interfaces:** Specific virtual machines can have unique NSG settings.
3. **Default Rules:**
   * NSGs come with default rules that allow or deny certain traffic, such as allowing all outbound traffic and denying inbound traffic from any source unless explicitly allowed.
4. **Prioritization:**
   * Rules in an NSG are evaluated based on their priority (lower numbers have higher priority). The first rule that matches the traffic is applied.
5. **Logging and Monitoring:**
   * NSGs can be monitored using Azure Monitor and logs can be enabled to track allowed and denied traffic.

**Use Cases:**

* Secure virtual machines by controlling inbound and outbound traffic.
* Create tiered security models by applying NSGs at both the subnet and VM levels.

***Application Security Groups (ASGs)***

**Definition:**  
Application Security Groups (ASGs) provide a way to group and manage network security rules for applications deployed in Azure. They allow for more granular control of traffic flow between resources, making it easier to manage security.

**Key Features:**

1. **Group Management:**
   * ASGs enable the grouping of virtual machines or other resources to simplify NSG management. You can create security rules that apply to all resources within the group instead of applying rules to each individual resource.
2. **Dynamic Membership:**
   * Resources can be dynamically added to an ASG based on tags or other criteria, which helps manage security at scale.
3. **Integration with NSGs:**
   * ASGs can be used in NSG rules to specify source or destination. This allows you to define rules that apply to all members of an ASG without needing to reference individual IP addresses.
4. **Simplified Security Management:**
   * ASGs help reduce the complexity of NSG rules by allowing you to create rules that reference groups of resources, making it easier to manage security in complex environments.

**Use Cases:**

* Grouping VMs by application role (e.g., frontend, backend) to simplify network security management.
* Managing security for microservices architectures, where different services communicate with each other.

**Summary**

* **NSGs** are essential for defining security rules that control network traffic to and from Azure resources, providing a way to secure your Azure environment at various levels (subnet and VM).
* **ASGs** simplify the management of security rules by allowing you to group resources and define rules for the group, making it easier to maintain security in dynamic and complex environments.

Together, NSGs and ASGs provide powerful tools for securing applications deployed in Azure, ensuring that network traffic is controlled and managed effectively.

**NSG flow:-**

+--------------------+

| |

| Internet |

| |

+---------+----------+

|

| (HTTP/HTTPS)

| (Port 80/443)

|

+---------v----------+

| |

| Nginx Frontend |

| |

+---------+----------+

|

(HTTP Requests to Backend)

(Port 8000 for Django)

|

+---------v----------+

| |

|Django Backend |

| |

+---------+----------+

|

(MySQL Queries)

(Port 3306 for MySQL)

|

+---------v----------+

| |

|MySQL Server |

| |

+--------------------+

### ***Scenario: Web Application with Frontend, Backend, and Database***

Imagine you are deploying a web application that consists of the following components:

1. **Frontend Web Server:** Serves static content and interacts with users.
2. **Backend Application Server:** Processes business logic and handles API requests.
3. **Database Server:** Stores user data and application data.

### Objective

You want to secure the communication between these components while simplifying security rule management. Instead of manually defining IP addresses for each server in your Network Security Groups (NSGs), you can use ASGs to group them logically.

### Step-by-Step Example

#### 1. Create Application Security Groups

You will create three ASGs for your components:

* **ASG-Web:** For the frontend web server.
* **ASG-API:** For the backend application server.
* **ASG-DB:** For the database server.

#### 2. Assign Resources to ASGs

* **Frontend Web Server:** Assign to **ASG-Web**
* **Backend Application Server:** Assign to **ASG-backend**
* **Database Server:** Assign to **ASG-DB**

#### 3. Create Network Security Groups (NSGs)

You will create two NSGs:

* **NSG-Frontend:** To control inbound and outbound traffic to the frontend web server.
* **NSG-Backend:** To control traffic to the backend application server and database.

#### 4. Configure NSG Rules Using ASGs

**NSG-Frontend Rules:**

* **Inbound Rules:**
  + **Allow HTTP/HTTPS from Internet to ASG-Web**
    - Source: Internet
    - Destination: ASG-Web (frontend server)
    - Ports: 80, 443
    - Action: Allow
* **Outbound Rules:**
  + **Allow traffic to ASG-API**
    - Source: ASG-Web (frontend server)
    - Destination: ASG-API (backend server)
    - Port: 8000 (or any port your API uses)
    - Action: Allow

**NSG-Backend Rules:**

* **Inbound Rules:**
  + **Allow traffic from ASG-backend to ASG-DB**
    - Source: ASG-API (backend server)
    - Destination: ASG-DB (database server)
    - Port: 3306 (default MySQL port)
    - Action: Allow
* **Outbound Rules:**
  + **Allow traffic to ASG-Web**
    - Source: ASG-backend (backend server)
    - Destination: ASG-Web (frontend server)
    - Port: 8080 (or any port your API uses)
    - Action: Allow

#### 5. Diagram Representation

Here’s a simplified representation of the architecture:

**Internet**

**|**

**| (HTTP/HTTPS)**

**|**

**+-----v-----+**

**| ASG-Web | <--- Frontend Web Server**

**+-----------+**

**|**

**| (backend Requests)**

**|**

**+-----v-----+**

**|ASG-backend| <--- Backend Application Server**

**+-----------+**

**|**

**| (Database Queries)**

**|**

**+-----v-----+**

**| ASG-DB | <--- Database Server**

**+-----------+**

### Benefits of Using ASGs

* **Simplified Management:** Instead of creating multiple rules for IP addresses, you can reference ASGs in your NSG rules. This makes it easier to manage and update security rules as your architecture changes.
* **Dynamic Scalability:** If you need to add more servers (e.g., scaling out the backend), you can simply add them to the **ASG-API** without modifying the NSG rules.
* **Logical Grouping:** ASGs help maintain a clean and logical separation of different application components, which enhances security posture.

### Verification:-

sysadmin@fe-vmsslaNZB6CM:~$ curl <http://10.0.2.5:8000>

sysadmin@be-vmsslaL5SSO9:~$ curl <http://10.0.1.4>

Also check chat\_apps is working or not.

### Summary:-

By using ASGs, you can efficiently manage network security rules in Azure, ensuring that your application components can communicate securely while simplifying the overall configuration. This is especially useful in complex environments or when deploying microservices, where services need to communicate with one another while being secured from unauthorized access.