**Cloud & IoT Security**

**Assignment 1: Configuring Cloud Firewall to Allow Only HTTP and SSH Access**

**Objective**

The aim of this assignment is to create a virtual machine (VM) in a cloud environment (AWS, Azure, or Google Cloud) and configure its firewall to allow **only HTTP (port 80)** and **SSH (port 22)** access while blocking all other inbound network traffic. Verification is conducted through testing allowed and denied port accessibility.

**Platform Used**

For this assignment, **Amazon Web Services (AWS)** EC2 was used due to its simplicity in managing security groups and network configurations.  
(You can perform equivalent actions in **Azure NSG** or **Google Cloud VPC firewall**.)

**Step 1: Launch a Virtual Machine Instance**

1. Log in to the **AWS Management Console**.
2. Go to **Services → EC2 → Instances → Launch Instance**.
3. Choose an Amazon Machine Image (AMI) such as **Ubuntu 22.04 LTS**.
4. Select an instance type (e.g., t2.micro, free tier eligible).
5. Configure network settings:
   * Select the default VPC.
   * Choose **Create new security group**.
6. Enter instance name and key pair for SSH authentication.

**Step 2: Configure the Firewall (Security Group) Rules**

In AWS, **Security Groups** act as a virtual firewall to control inbound and outbound network traffic.

Create or edit the security group associated with the VM as follows:

**Inbound Rules**

| **Type** | **Protocol** | **Port Range** | **Source** | **Description** |
| --- | --- | --- | --- | --- |
| SSH | TCP | 22 | Your IP (or CIDR) | Allow SSH access |
| HTTP | TCP | 80 | 0.0.0.0/0 | Allow web access |

Remove any other pre-existing inbound rules such as HTTPS (443) or ICMP.

**Outbound Rules**

Keep the default outbound rule (allow all traffic) to enable updates and communications initiated by the VM.

Save and attach this security group to the instance.

**Step 3: Connect to the Instance via SSH**

From a terminal on your local system, connect to the VM using its public IP address:

ssh -i yourkey.pem ubuntu@<public-ip-address>

If successful, this confirms SSH (port 22) access is working correctly.

**Step 4: Enable and Test HTTP Access**

1. Install Apache web server on the VM:

sudo apt update

sudo apt install apache2 -y

1. Once installed, confirm the server is running:

systemctl status apache2

1. Open a browser and navigate to:

*text*

*http://<instance-public-ip>*

You should see the default Apache web page, confirming HTTP (port 80) access is open.

**Step 5: Verify Other Ports Are Blocked**

Use nmap or telnet to scan the instance for open ports:

Example command from your local system:

nmap -Pn <public-ip-address>

Expected output should show only:

* Port 22/tcp open (SSH)
* Port 80/tcp open (HTTP)

All other ports should appear as **filtered or closed**, confirming that your firewall settings are enforced.

**Step 6: Verification and Evidence**

| **Test Description** | **Expected Result** | **Status** |
| --- | --- | --- |
| SSH connection (port 22) | Connection successful | Passed |
| HTTP access (port 80) | Apache page loads successfully | Passed |
| Other ports (e.g., 21, 25, 8080) | Connection refused or filtered | Passed |

**Step 7: Azure and Google Cloud Alternatives**

* **Azure:** Configure inbound rules in the **Network Security Group (NSG)** for the VM. Allow ports 22 and 80 only; block all others.
* **Google Cloud:** Create **VPC firewall rules** to allow only TCP ports 22 and 80 to the VM network tag.

Google Cloud firewall verification commands:

gcloud compute firewall-rules list

nmap <vm-external-ip>

**Conclusion**

The assignment successfully demonstrated the setup of a secure virtual machine instance with restricted network access. The security group/firewall was effectively configured to allow only **SSH (22)** and **HTTP (80)** traffic while preventing unauthorized inbound connections across all other ports. This principle enhances cloud instance security by reducing the exposed attack surface.