Domain: Network Security Question 1: Faulty Firewall

Firewalls are put in place to protect servers from unsecure and unwanted connections. Having a faulty firewall that allows SSH connections instead of blocking them could cause issues for the machines involved. There could be multiple ways to debug this situation, such as checking if the machines are only accepting the correct form of connection, such as an SSH key instead of a generic password, or checking the inbound rules within the Network Security Group.

In my ELK Stack Project, I created 4 virtual machines; 1 Jump Box Provisioner machine, 2 Web Server machines, and 1 ELK server machine. Only the Jump Box Provisioner machine allows SSH connection using an outside public IP address. If I tried connecting to any of the other 3 with my personal computer, I would get an error saying “Connection timed out.” This is because not only does the Network Security Group stop the other machines from being reached by outside the virtual network, they also require a specific SSH key that is only available within the Jump Box Provisioner machine.

If one of my VMs accepted SSH connections without permission, I would first check my inbound rules on the network security group. If the rules were faulty, it could cause multiple VMs to accept SSH connections, which could lead to a bigger problem if it isn’t fixed. Another thing I would check is if the Virtual Machines were using an SSH public key as a source of connection. The virtual machine might have not been set up with the SSH key from the Jump Box Provisioner, so changing it to accept the right SSH private key would fix the problem.

Using the Azure UI, I would first navigate to the Network Security Group that these machines fall under. I would then check to make sure there is a DenyAllInBound rule set up. If not made already, I would make a rule with higher priority that allows SSH connection from my workstation to the Jump Box Provisioner machine. This means that the DenyAllInbound rule will automatically block all incoming traffic, while the new rule will override it and allow only my workstation to connect. I would next navigate to the rest of the virtual machines under the Connect tab. I would make sure that the SSH public key is set up to match the private key from the ansible container only available within the Jump Box Provisioner machine.

Changing these settings would guarantee to fix any faulty firewall issues and allow only the right workstation to access the correct VMs, while leaving others hidden from outside access. Using Kibana is also useful in monitoring if there were any failed logins attempted, with information such as the time and ports used.

Question 2: Unsecured Web Server

If compliance guidelines required encryption in motion and a server was running HTTP on port 80, I would have it redirect traffic to port 443, which is HTTPS.

During the ELK Stack Project, I had 2 web servers running HTTP on port 80 through a load balancer. This means that the traffic wasn’t going directly to my Web Server machines, as they did not have their own public IP. They used a public IP address given to them by the load balancer to manage inbound traffic and only allowed them to see the frontend pool and not the backend pool. During a real deployment, the load balancer would be configured to use port 443 instead of port 80, and could have a rule redirecting port 80 to port 443.

Running HTTP on port 80 could lead to potential problems because it is not encrypted, meaning the packets containing private information could be captured and viewed by anyone with malicious intent. To fix this, all connections should be made to port 443 instead.. This fixes the problem because HTTPS encrypts the communication, making it harder to access information within the packets.

To implement this solution in my ELK Stack Project, I would configure my load balancer to use port 443 instead of port 80 as default. If I had to keep port 80 open, then I would have traffic be redirected to port 443, making all unsecure connections change to secure connections before continuing with communication. This could be done by implementing a Front Door or an Application Gateway through Microsoft Azure.

The original solution to only use port 443 (HTTPS) would prevent clients who use port 80 to communicate with the server to connect. The only solution to give access to both port 80 users and port 443 users would be the rerouting solution so all connections were encrypted. I don’t believe you can just “set it and forget it”, as you might need to update rules and load balancers depending on the amount of traffic coming into web servers.