**Network Security and how to deal with a faulty Firewall**

Suppose you have a firewall that's supposed to block SSH connections, but instead lets them through. How would you debug it?

1) Firewalls are put into place to help insure the integrity of the network. If even one machine is accepting SSH requests when they are not supposed to, the entire network could be compromised.

2) One of the first steps that was taken when setting up my web servers was to enact a rule within the Network Security Group (NSG) that denied all inbound traffic to my WebVMs. By denying all inbound traffic I was essentially preventing the initial TCP handshake protocol with the server which would negotiate the secure connection between host and server.

Once I confirmed that my NSG rule was working properly, I set up a secondary rule with a higher priority to override the initial ‘Deny All’ rule to allow SSH access from my private IP address to my Jumpbox. Along with allowing only 1 IP address to connect I removed the option for a password connection and elected to go a more secure route by requiring a privately generated SSH key. This ensured that even if someone was able to spoof my personal IP address, they would still be unable to access the Jumpbox without my private SSH key.

Once these rules were put into place, attempts were made to connect to the web servers via both SSH, HTTP and Ping requests. All attempts to connect to the servers resulted in time out errors and lost packets due to the server not sending and ACK to my host machines SYN. This then confirmed my NSG was working properly.

3) Now if my initial NSG rules failed and my web servers were still allowing SSH connections, then it would be safe to assume that the error is coming from the servers themselves. In this case, my first step would be to check the /etc/hosts.(deny/allow) configuration files.

The first step would be to check /etc/hosts.allow and add the static IP address of the Jumpbox I was using to connect to the web servers. As a secondary precaution, I also added my private IP address in there to prevent locking myself out of the server. The second step would be to access /etc/hosts.deny and add the syntax ALL : ALL. Once done, a restart of SSH services would need to be done (systemctl restart sshd.service) for the changes to take effect.

Now that the systems have been configured, a test was ran to attempt to secure a connection to the web server from the jumpbox. Once successful, I would then need to go back into the /etc/hosts.allow file and remove my private IP address. This is done primarily to ensure the integrity of the system and remove an unnecessary vulnerability.

4) As stated in the first question, If one of my servers was accepting unauthorized SSH connections, I would need to investigate the NSG panel and check to see which rules are in place and what their priority is. Secondly I would need to ensure that only the jumpbox ansible public key was being used to connect to my servers.

Specifically speaking, checking the inbound traffic rules is the only place that is going to allow or deny SSH connections. There should be a rule in there that denies all traffic. If this rule isn’t there then it needs to be added. If there is a rule that allows ssh traffic (that does not specify my IP address) that rule then needs to be removed. Afterwards, the ansible public key would need to be added to the web server via the ‘reset users’ tab. This will prevent unauthorized access to the web servers even if someone was logged into my network.

Once completed, this could all be tested in the ‘Connection Troubleshooting’ tab of the Azure UI. If successful, the connection should fail.

5) This solution does not make the system immune to unauthorized access. Although the connections to the web servers can only be obtained via the jumpbox, the host machine that accesses the jumpbox is still a single point of failure. Meaning that if someone was to access the host machine, the entire network could be compromised. What this solution does is ensure that the integrity of the server by removing the threat of outside forces. The host machine and the people who have access to said machine will in theory be the only vulnerability to the system.

To help with this vulnerability, the use of Kibana is implemented to help monitor and track any and all suspicious login attempts.