**Domain: Cloud Security**

**Question 1: Cloud Access Control**

During our Cloud Security chapter for weeks 12 and 13 during this course, we learned the ins and outs of deploying and securing a cloud virtual network. Through architecture design and configuration options, we were able to securely deploy our network consisting of firewalls, load balancers, and various virtual machines.

In Weeks 12 and 13, we were able to deploy a cloud virtual network. Cloud networks differ from physical, on-premises networks in that all physical components in a physical network are virtualized from software in the cloud. For example, NICs, hard drives, and CPUs all exist in virtualized form when it comes to virtual machines. Connections between nodes are also virtualized, as opposed to requiring physical cabling to connect one machine to another. This allows cloud networks to be deployed quickly and with flexibility. Machines can be deployed, deleted, and rearranged with the click of a few buttons, rather than physically re-connecting the wires as you would on a physical network.

Control access is also virtualized, as there are no physical firewalls. Instead, a security group acting as a firewall is configured in order to safeguard a virtual network, and load balancers are also virtualized. Various rules were created in order to maintain security within our VM. Examples include our “Default-Deny” rule, which blocked all inbound traffic by default, following the principle of least privilege. Later on, other inbound rules were created in order to make exceptions to the Default-Deny rule. For example, we created inbound rules that allowed for us to SSH into our Jump Box Provisioner VM from our at-home workstation IP. The purpose of this configuration was to design the network in such a way that the Jump Box Provisioner VM acted as a gateway, meaning that the only way to access the rest of the network (i.e. our DVWA container Web-VMs) was to gain access to our Jump Box Provisioner first. Seeing as we set up access in such a way that the only way one could SSH into our Jump Box was through the IP of my home workstation, and also the fact that this computer was the only machine where the public key to access the Jump Box would be stored, I believe that this architecture was very secure due to only creating a single point of access.

We also created a rule that allowed for the Jump Box Provisioner VM to ssh into, and access, our various (in my case, three) DVWA Web-VM containers. Once again, the purpose of this was to be able to allow the Jump Box Provisioner VM to act as a gateway, and also to allow the Ansible Provisioning tool to deploy our DVWA containers to each Web-VM using our pentest.yml playbook. In addition, none of our three Web-VMs had a public IP address. Setting up a load balancer, and using that load balancer in conjunction with the RedTeam Security Group firewall to allow port 80 traffic into the virtual network, was another security control that allowed only authorized personnel to access our DVWA application and web servers.

For Project 1, we also deployed an ELK Stack container within our new Elk-Server VM. Because this new VM was located in a different Azure region, we had to configure it with its own Virtual Network and Security Group. First, we had to create a peering connection between our RedTeam virtual Network and our Elk-Stack virtual network to allow traffic to flow between the two. Secondly, we configured the Elk-server security group to allow SSH connections, and also TCP traffic over Port 5601, which was necessary to access the Kibana application from our home IP.

Overall, I believe that our ultimate Azure virtual network was extremely secure, once again, given the architecture of the Jump Box acting as a gateway, and with the stringent inbound traffic security rules that we have set for our security group firewalls. However, there are still disadvantages to this type of setup. For instance, this particular solution may not be scalable in an environment where more people require access to the network. For this to happen, more people would need access to the public key string for the jump-box (or alternatively we would need to move to a password system, which diminishes security), neither of which are ideal. Secondly, we would have to configure our security rules to allow every single individual’s home workstation’s private IP to be allowed access into the network, which would be tedious to configure, as more people’s IPs would need to be manually added, not to mention that dynamic IPs mean that people’s IPs can change.

An alternative to this solution would be to allow a VPN-style connection, wherein users must go through a local network and a VPN gateway first, before connecting to the Azure Cloud network. This solution would be much more scalable if the number of users or machines needing to access the cloud network remotely were to drastically increase. However, the downside of this type of solution is that it would be more costly and resource-intensive to operate, so any organization considering using a VPN Network would have to do a thorough evaluation of the costs versus the benefits.