**Activity File: Interview Questions**

**Domain: Cloud Security**

**Question 3: Containers**

When is it appropriate to use containers in cloud deployments, and what are the security benefits of doing so?

1. Restate the problem:

It is a good idea to use containers when you are working on a larger or more complex project. For example, when you are trying to reduce the load of responsibility and resources on a machine, assigning each docker a specific task or application can solve this problem. Containers work by isolating the software from its environment and ensure standardization by separating application dependencies from infrastructure. Containers are designed to do one thing very well, and allow elasticity and scalability. “Rules” (or playbooks) run on the containers to do these specific things. Some benefits to this are that you are not paying for resources when they are not being used, and you can dynamically build resources as well. Containers also offer speed of use, as they spin up faster than servers (seconds compared to minutes), all while adding layers of security and accessibility controls. Docker containers are a good idea for use as version control on your application’s entire operating system. They are also ideal for team distribution and collaboration efforts, or when your app needs to go through multiple phases of development, for example.

Since applications run independently in separate containers, each will have their own level of security. This feature of app isolation helps share additional features without any risk factor. In line with the Principle of Least Privilege, access to containers can be provided to internal or external users without granting access to any more than is necessary.

1. Provide a Concrete Example Scenario:

In Project 1, I created an ELK stack that allows us to automate the process of monitoring the performance of multiple virtual machines in one database. Ansible was used to create one container on the Jump Box Provisioner. Three Ansible playbooks were created within that container. The first playbook created and mapped a new ELK container to the ELK server. The next two playbooks installed FileBeat and MetricBeat, which help collect file and system data on the web server VMs, which are sent to the ELK server. Based on the playbook instructions, as well as the NSG security rules, the ports and allowed IP is such that I can access the log data using Kibana through my internet browser.

1. Explain the Solution Requirements:

This virtual network setup, including containers, was an appropriate use of containers in this instance because it is being used to monitor a network with multiple web servers that has multiple layers of security. With regard to security in the Azure portal, there were NSG security rules set up to allow only certain traffic/ports. The machines were all set up to use public SSH Keys instead of passwords, removing the ability to brute force attack the passwords. The use of containers means app isolation, so potential bad actors do not have access to an entire machine(s).

1. Explain the Solution Details:

Getting containers set up within the Jump Box. Docker.io was installed, and Ansible was pulled in. The first container is then created on the Jump Box. The container's SSH Public Key was then added to VM’s within the Azure portal. Next, the container’s hosts file was updated to reflect the webservers IPs, and the config file was updated to include the webservers username remote user).

The initial container had a randomly generated name and was set up within the Jump Box, however the second container was specific to the ELK server. A playbook was created to configure the ELK server. This playbook had similarities to the previous playbook, however one section of YAML code controlled the container source, name, and version (sebp/elk:761). To confirm that the correct container was installed, the playbook was run and the Play Recap showed no errors. Another way to check if things were working correctly would be if the commands to “start” and “attach” the container would change the command line to end up in the root command for that container.

1. Identify Advantages/Disadvantages of the Solution:

Without these automated actions, I would have run commands like syslog or auditd, which would have been a manual process that would have collected more data than I would want. Use of Elasticsearch, Logstash, FileBeat, and MetricBeat help streamline that process by collecting specific data and formatting it in a consistent manner. The data would also be collected within each virtual machine (versus being sent to a centralized location), which isn’t ideal because the tasks would have to run in each independent machine. This could also stress the machine, which has limited resources. Containers help carve out specific resources for specific tasks, making the process more efficient. Each machine and container have their own security rules, creating more layers of security if one item were compromised.