**Cloud Infrastructure as Code**

* **What are the security benefits of defining cloud infrastructure as code?**

Defining cloud infrastructure as code has many benefits. As organizations continue to migrate to the cloud and new software is released more frequently, manually creating and managing cloud infrastructures becomes an unsustainable task. Infrastructure as Code (IaC) allows defining cloud infrastructure using a programming language. The configuration of all the VMs, containers, and networks in your deployment are defined in text files, which can be used with provisioners to automatically recreate machines and networks as necessary. These text files allow for anyone to see exactly how the network is configured and duplicated. IaC enables you to apply concepts like Continuous Integration/Continuous Deployment (CI/CD) to automatically update machines on your network by making changes to the IaC files.

For instance, in my Cybersecurity Project 1, I used Docker to install Ansible, a provisioning tool and a perfect example of IaC. This ensures that my provisioning scripts run identically everywhere and that automated configurations will do exactly the same thing every time they run. Once my Jump-Box was running an Ansible Docker container, I was able to configure the jump box to connect to the other servers so it can configure them. Using ansible I was able to easily configure an Elk stack server to monitor my web servers, as well as installing filebeat and metricbeat to collect logs files, data of events, and metrics from my web servers to later be compiled and analyzed using Kibana.

An alternative to IaC is the Standard Configuration Management, which requires that all configurations, installations and commands be run individually in each server. This alternative is not only less productive, but it also drastically increases the chances for errors. Treating infrastructure as code introduces several benefits such as:

-Speed: a code template and command can create, destroy or configure several cloud resources within minutes.

-Security: codes text files can be read and reviewed by anyone resulting in infrastructure with less security issues.

-Consistency: you will always deploy the infrastructure defined in the IaC.

-Automation: you can use CI/CD systems to run tests on the infrastructure during different stages of development.

During project 1, IaC was used to set up the Elk server with Docker, configure hosts, install filebeat and metricbeat. A playbook was also created and ran using the command ansible-playbook, and to finally test the configurations for the Elk server I launched the Kibana page using the Elk Server's public IP address and once the page is launched we know the configurations were successful. To test the filebeat and metricbeat I checked data from the logs and metrics within kibana.

When analyzing the different approaches to managing infrastructure it is clear that IaC is the best option for cloud infrastructures. The traditional approach can be appropriate for a small local network with physical physical devices, but not practical or efficient for a larger cloud network.