***Certificate Authority Deployment***

Yogaraj Govindarajalu Prabagaran

# Table of Contents

[Table of Contents 2](#_Toc158495083)

[Lab 1- Certificate Authority Deployment 3](#_Toc158495084)

[Part 1- Building the initial Topology. 3](#_Toc158495085)

[Description 3](#_Toc158495086)

[Preparation 3](#_Toc158495087)

[Observations 3](#_Toc158495088)

[Screenshots 4](#_Toc158495089)

[Reflection 6](#_Toc158495090)

[Part 2- Build the topology in the vSphere environment. 6](#_Toc158495091)

[Description 6](#_Toc158495092)

[Preparation 6](#_Toc158495093)

[Observations 6](#_Toc158495094)

[Reflection 7](#_Toc158495095)

[Part 3- Deploy your Certificate Authority. 7](#_Toc158495096)

[Description 7](#_Toc158495097)

[Observations 7](#_Toc158495098)

[Screenshots 8](#_Toc158495099)

[Reflection 9](#_Toc158495100)

[Part 4- Sign the Certificate Signing Request. 9](#_Toc158495101)

[Description 9](#_Toc158495102)

[Preparation 9](#_Toc158495103)

[Observations 9](#_Toc158495104)

[Screenshots 10](#_Toc158495105)

[Reflection 10](#_Toc158495106)

[References 11](#_Toc158495107)

# Lab 1- Certificate Authority Deployment

# Part 1- Building the initial Topology.

## Description

In this lab, I will configure network topologies using Palo Alto NG-Firewalls. I'll assign interfaces to designated zones such as Inside, Internet, DMZ, and Guest. Configuration will involve assigning interfaces to the corresponding network topology, defining zones, adding a virtual router for inter-network routing, and segmenting the network based on zones.

## Preparation

Patch Cables connected to various switches, routers, and palo alto in the below configuration:

* Palo alto Port 1-fe0/0(router)
* PA Port 2-Switch 1 2960
* PA port 3-Switch 2 3750(for more ports)
* VM 3 - router fe0/1  
    
  Palo Alto Web Interface is also accessed to configure various interfaces and virtual routers, apply policies, etc.

## Observations

I’ve assigned IP addresses and naming conventions as instructed. This step was crucial for maintaining consistency and clarity throughout the configuration process.

Interface Assignment: Assigning interfaces to the designated zones, including Inside, Internet, DMZ, and Guest, was straightforward. Each interface was configured to correspond with its respective zone.

Zone Configuration: Defining zones and assigning interfaces to specific zones on the Palo Alto NG-Firewalls was completed without complications. This step facilitated the network segmentation, ensuring proper traffic management and security measures.

Virtual Router Addition: Adding a virtual router to enable routing between the different networks was a seamless process. While this configuration allowed for inter-network communication, the segmentation based on zones ensured that traffic exchange was restricted appropriately.

Network Segmentation: The segmentation of the network based on zones was effectively implemented, enhancing security measures and facilitating the enforcement of zone-based security policies. This segmentation ensured that traffic was isolated within its designated zone, minimizing potential security risks.

Overall, the configuration process proceeded smoothly, with each step contributing to establishing a secure and well-organized network infrastructure using Palo Alto NG-Firewalls.

## Screenshots

A screenshot of a computer

Description automatically generated

Figure 1-1   
This Screenshot displays the Interface section in Palo Alto.

A screenshot of a computer

Description automatically generated

Figure 1-2

This Screenshot displays the Virtual Router section in Palo Alto.

A screenshot of a computer

Description automatically generated

Figure 1-3  
This Screenshot displays the Custom policies in Palo Alto.

## Reflection

In this lab section, I configured network topologies using Palo Alto NG-Firewalls, establishing zones such as Inside, Internet, DMZ, and Guest for specific network functions and security measures. Creating a network diagram illustrated the architecture, emphasizing traffic flow and segmentation. This hands-on experience enhanced my understanding of network security principles and the role of firewalls in enforcing security policies between zones.

Furthermore, creating four distinct zones ensured that interfaces assigned to each zone operated independently, with no communication between them. Policies and routes had to be created to enable communication between zones, highlighting the importance of configuring firewall rules to facilitate inter-zone communication without compromising security measures. This aspect of the lab underscored the necessity of implementing robust security policies to regulate traffic flow and protect network resources effectively.

# Part 2- Build the topology in the vSphere environment.

## Description

In this part 2 of the lab, Rocky Linux virtual machine (VM) will be deployed with additional steps such as updating all packages and installing OpenSSL in the vSphere environment. Additionally, installing OpenSSL will equip the VM with cryptographic functionalities for secure network communication. By executing commands like 'sudo yum update' and 'sudo yum install openssl' in the terminal, we'll leverage Rocky Linux’s package manager to achieve these objectives efficiently. This hands-on approach ensures RockyLinux is properly configured and armed with essential security tools for various computing needs.

## Preparation

* “YG-9578” -LinuxVM will be the hostname for the RockyLinux Virtual Machine.
* “10.172.94.5”- The IP address of the corresponding VM.

## Observations

* RockyLinux VM Creation: I deployed the virtual machine with the help of a template in the vSphere environment.
* Commands:  
  “sudo yum update”- Updates all packages.  
  “yum install -y openssl” installs openssl package.

## Reflection

The task was executed smoothly without any significant difficulties. The Virtual machine is now ready for Task 3- Deploying Certificate Authority.

# Part 3- Deploy your Certificate Authority.

## Description

In this task, I will set up OpenSSL on my Rocky Linux virtual machine (VM) to function as a Certificate Authority (CA). Initially, I'll generate a private key to enable the creation of the CA certificate. I'll generate a Certificate Authority (CA) certificate using this key. Once this is done, my Certificate Authority server will be fully configured and prepared to sign client certificates, facilitating secure communication and authentication within my network environment. This deployment strengthens the security framework of my system by ensuring the integrity and legitimacy of digital certificates employed for various purposes like secure connections, encryption, and identity validation. By adhering to these guidelines, I'll establish a robust Certificate Authority server on my Rocky Linux VM, enabling the implementation of secure communication protocols and enhancing the overall security stance of my network infrastructure.

## Observations

Various commands and details were used to achieve the objective in this lab part.

* “openssl genrsa -aes128 -out yg9578.key 2048”- Generation of Private Key: Creating a private key to facilitate the CA certificate was a pivotal stage in setting up the Certificate Authority. This phase proceeded smoothly, providing the groundwork for the subsequent formation of the CA certificate.
* “openssl req -new -x509 -days 1825 -key /etc/pki/CA/private/yg9578.key -out /etc/pki/CA/certs/yg9578.crt”- Formation of certificate authority: I effectively crafted the Certificate Authority (CA) certificate by employing the previously generated private key. This certificate holds significant importance in verifying and authorizing potential client certificates.
* The following Details were filled out as well:  
  Country name: CA  
  Province: Ontario  
  Locality: Waterloo  
  Org Name: Conestoga  
  Org Unit: ITNS  
  Common name: ITNSYG9578.lab.ca  
  Email: [ygovindarajalaup9578@conestogac.on.ca](mailto:ygovindarajalaup9578@conestogac.on.ca)

## Screenshots

A screenshot of a computer screen

Description automatically generated

Figure 3-1   
This Screenshot illustrates the creation of a private key.

A computer screen shot of a computer code

Description automatically generated

Figure 3-2   
This Screenshot illustrates the generation of certificate authority utilizing the private key.

## Reflection

In this lab section, I successfully deployed a Certificate Authority (CA) on my Rocky Linux virtual machine (VM) using OpenSSL, generating a private key crucial for establishing the CA certificate.

Understanding the components and architecture of a CA, particularly its hierarchical structure with root, intermediate, and end-entity certificates, proved essential in appreciating its role in network security. Deploying a self-signed root CA in this lab provided valuable insights into certificate management and cryptographic protocols, highlighting the importance of trust establishment and secure communication within network environments.

The Root Certificate is a vital component of the Certificate Authority (CA) architecture since it is signed by itself and represents the most significant degree of confidence within the CA hierarchy. At the top of the hierarchy, it means the most significant degree of assurance in the Public Key Infrastructure (PKI) system. Intermediate CA certificates, pivotal in issuing end-entity certificates, are authenticated using the root certificate's private key, thereby inheriting its reliability. Additionally, an indispensable component of CA content is the Certificate Revocation Lists (CRLs), routinely updated by the CA to include information about invalidated certificates. These lists serve as a tool for recipients to confirm the status of certificates, ensuring the integrity of the PKI system by flagging revoked certificates.

# Part 4- Sign the Certificate Signing Request.

## Description

This lab section aims to sign a Certificate Signing Request (CSR) using a Certificate Authority (CA). The following steps involve transferring the CSR key to my CA server for signing. Firstly, I'll copy the CSR key. Subsequently, I'll move the copied CSR key to my CA server to begin the signing process. Once the CA signs the certificate, I will receive a certificate key. My final task is to submit this signed certificate key and my portfolio submission. By following these instructions, I will gain practical experience in the certificate signing process and deepen my understanding of cryptographic operations involving Certificate Authorities.

## Preparation

Additional software known as “WinSCP” was required to copy the CSR file to the Rocky Linux Virtual machine.  
The CSR file was found in the Econestoga Course shell in the Portfolio 1 Submission section.

## Observations

* The WinSCP application was used to copy the CSR file into the Rocky VM.
* The command- “openssl x509 -req -in /etc/pki/CA/private/cert\_ITNSFW1.csr -CA /etc/pki/CA/cert/yg9578.crt CAkey /etc/pki/CA/private/yg9578.key -CAcreateserial -out /etc/pki/CA/private/yg9578\_cert\_ITNSFW1.crt -days 365” depicts that the private key was used to sign the certificate.
* Following the signing process, the certificate was copied back to the user system using WinSCP.

## Screenshots

A screenshot of a computer

Description automatically generated

Figure 4-1   
This screenshot displays the creation of a signed CSR by utilizing the certificate authority.

## Reflection

This segment of the lab provided valuable insights into the crucial roles of the certificate signing request (CSR) and certificate key in cryptographic processes and secure communication. The CSR is a formal request submitted by an entity, such as a user or server, to a Certificate Authority (CA) to obtain a digital certificate. It includes vital information, like the entity's public key and identifying details, necessary for the CA to generate a trusted certificate. This step is pivotal in establishing trust between the entity and the CA, ensuring the integrity of the certificate issuance process.

Additionally, the certificate key, also called the private key, plays a significant role in securing digital certificates and cryptographic operations. The entity keeps the private key confidential when signing the CSR and decrypting data encrypted with the corresponding public key. It is a critical component in authenticating the entity's identity and maintaining the integrity of transmitted data.

In contrast, the public key, derived from the private key, is openly shared with counterparties for encryption purposes and verifying digital signatures. It facilitates secure communication channels and assists in validating the authenticity of the entity's digital certificates.

Understanding the distinction between private and public keys is essential in comprehending asymmetric cryptography concepts, where pairs of keys are utilized for encryption and decryption tasks. While the private key remains confidential and under the entity's control, the public key is distributed for encryption and authentication processes. These cryptographic components collectively form the foundation of secure communication protocols, protecting sensitive information in digital environments.

# References

1. J. D. (2023, February 10). *Configuring Base Parameters on Palo Alto Firewall*. YouTube. <https://www.youtube.com/watch?v=DBqW3MP8jtY>
2. J. D. (2023, July 30). *Initializing Palo Alto Firewall*. YouTube. <https://www.youtube.com/watch?v=51ppiWIr_y4>
3. J. D. (2023, February 10). *Configuring Palo Alto Security Policies*. YouTube. <https://www.youtube.com/watch?v=u4oA1cOMiDo>
4. Sanders, R. (2023, September 12). *What is a Certificate Signing Request (CSR)?* Keyfactor. <https://www.keyfactor.com/blog/what-is-a-certificate-signing-request-csr/#:~:text=A%20certificate%20signing%20request%20(CSR)%20is%20an%20encoded%20file%20containing,encrypt%20traffic%20to%20your%20site>.
5. Kili, A., & Kili, A. (2019, March 12). *How to Generate a CSR (Certificate Signing Request) in Linux*. How to Generate a CSR (Certificate Signing Request) in Linux. <https://www.tecmint.com/generate-csr-certificate-signing-request-in-linux/>

# 