Managing Certificates with Open SSL

(CompTIA Security + SY – 601)

Objectives:

- To work with OpenSSL to manage certificates
- To generate certificates
- To generate a certificate signing request
- To convert certificate formats

Resources:

- Basic OpenSSL commands
- Kali Virtual Machine

Instructions:

Use basic OpenSSL commands

- Sign-in to PT1-Kali VM
- Open the **Terminal** from the menu at the top of the Desktop
- To check the Open SSL version, type the following command: **openssl version**, and then press Enter

```
root@KALI: ~
root@KALI:~# openssl version
OpenSSL 1.1.1d  10 Sep 2019
```

- Run the following command: **mkdir keys**, to change the keys directory

```
root@KALI:~# mkdir keys
```

- Use the **cd** command to change the directory to **keys**
- Generate an asymmetric encryption RSA key pair and extract the public portion to prepare to create a certificate signing request (which occurs below)

```
root@KALI:~/keys# openssl genrsa -out corp.515support.com.key 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.....................+++++
.................................+++++
e is 65537 (0×010001)
root@KALI:~/keys# cat corp.515support.com.key
```

- Run the following command: **cat corp.515support.com.key**, to display the private key

```
root@KALI:~/keys# cat corp.515support.com.key
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAs4XXK9JnH1e8P7CcxJ7nYnCB+dqGr4aOayu9oEH8lGakLwDe
dHn1a+8vOO7jYgG6BXmgriwy2RircJbHd95dlfjPDC1g6lk38wqg+eBKUklq/TBH
5Io9z4AiX/L8Q8NwBcHr7Vd8i8Z3B9yMQ6m8f0F+tixJQqJLaZRQqE1XW52jNdZ+
xhMgrQvykw2XvQATsI1QnNANdUWxxxHPAke/UMlgw5Tqdnl8Ddj5JReNMWdwglFe
V9h0NMKQo7Sp3czAU2KosNzRjLVueBLDK/JUPCu0+oWqXU4goJyBM3kFVsZkxiNw
```

- Extract the public key file for export with a **CSR**
- Use the **ls command** to display the two key files that you have create so far

```
root@KALI:~/keys# ls
corp.515support.com.key  corp.515support.com_public.key
```

- Display the public key files

```
root@KALI:~/keys# cat corp.515support.com_public.key
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAs4XXK9JnH1e8P7CcxJ7n
YnCB+dqGr4aOayu9oEH8lGakLwDeHn1a+8vOO7jYgG6BXmgriwy2RircJbHd95d
lfjPDC1g6lk38wqg+eBKUklq/TBH5Io9z4AiX/L8Q8NwBcHr7Vd8i8Z3B9yMQ6m8
f0F+tixJQqJLaZRQqE1XW52jNdZ+xhMgrQvykw2XvQATsI1QnNANdUWxxxHPAke/
```

Generate a Certificate Signing Request

- Generate a certificate signing request. Type the following command and then **Press Enter**
- Provide the following answers to the prompt

```
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:515 Support
Organizational Unit Name (eg, section) []:webServices
Common Name (e.g. server FQDN or YOUR name) []:webserver.corp.515support.com
Email Address []:admin@515support.com
```

- When prompted to enter a **challenge password** and an **optional company name**, press **Enter**. This OpenSSL command generates a certificate signing request on behalf of the Apache web server service for a website
- Run the following command to display the **.csr** file

```
root@KALI:~/keys# ls
corp.515support.com.csr  corp.515support.com.key  corp.515support.com_public.key
```

- Verify the certificate request. Type the following command, then press **Enter**
- The certificate signing request must be sent to the certificate authority using the **PEM** file in this format: **cat corp.515support.com.csr**

```
root@KALI:~/keys# cat corp.515support.com.csr
-----BEGIN CERTIFICATE REQUEST-----
MIIC4TCCAckCAQAwgZsxCzAJBgNVBAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRl
MRQwEgYDVQQKDAs1MTUgU3VwcG9ydDEUMBIGA1UECwwLd2ViU2VydmljZXMxJjAk
BgNVBAMMHXdlYnNlcnZlci5jb3JwLjUxNXN1cHBvcnQuY29tMSMwIQYJKoZIhvcN
```

Convert Certificate Format

- Run the **ls** command and observe that there are three files in the directory

```
root@KALI:~/keys# ls
corp.515support.com.csr  corp.515support.com.key  corp.515support.com_public.key
root@KALI:~/keys# openssl req -newkey rsa:2048 -nodes -keyout corp.515support.com.key -x509 -days 365 -out
corp.515support.com.crt
Generating a RSA private key
.....................................................+++++
```

- Generate a self-signed certificate

- Provide the following answers to the prompts

```
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:515 Support
Organizational Unit Name (eg, section) []:webServices
Common Name (e.g. server FQDN or YOUR name) []:webserver.corp.515support.com
Email Address []:admin@515support.com
```

- Run the **ls** command again and observe that there are now four files in the directory. A new **.crt** file has been created. Merge the **.key** and **.crt** files (the non-Window PEM formats) into a **.pfx** file (the Windows PKCS # 12 format)

Merge the .key and .crt files (the non-Window PEM format) into a .pfx file (the Windows PKCS# 12 format)

- Type the following command to convert the files and press **Enter**

```
root@KALI:~/keys# openssl pkcs12 -export -name "corp.515support.com" -out corp.515su
pport.com.pfx -inkey corp.515support.com.key -in corp.515support.com.crt
Enter Export Password:
Verifying - Enter Export Password:
```

- When prompted, select **Enter** to skip defining an **Export Password**
- Run the ls command and observe that there are now five files in the directory. A new **.pfx** file has been created

```
root@KALI:~/keys# ls
corp.515support.com.crt  corp.515support.com.key  corp.515support.com_public.key
corp.515support.com.csr  corp.515support.com.pfx
```

Observations:

- **Basic OpenSSL Commands**: The instructions guide the use of basic OpenSSL commands for managing certificates.
- **Environment Setup**: The tasks are performed in a Kali Linux virtual machine.
- **Directory Management**: Users create and navigate directories (mkdir keys and cd keys).
- **Key Generation**: Users generate an RSA key pair and display the private key.
- **Public Key Extraction**: Extraction of the public key is done to prepare for a Certificate Signing Request (CSR).
- **CSR Generation**: A CSR is generated and verified.
- **Certificate Conversion**: Converting certificate formats from PEM to PKCS#12.
- **Self-Signed Certificate**: A self-signed certificate is generated.

Results:

- **Successful Key and Certificate Creation**: RSA key pair and corresponding CSR were created and verified successfully.
- **File Management**: The directory contained the expected files after each command execution (`.key`, `.csr`, `.crt`, and `.pfx` files).

- **Certificate Conversion**: The conversion from PEM format to PKCS#12 format was successful, and the final directory listing confirmed the presence of the `.pfx` file.

Conclusion:

The objectives of the lab were met successfully. The tasks demonstrated the process of managing certificates using OpenSSL, from key generation to creating and verifying a CSR, generating a self-signed certificate, and converting certificate formats. The step-by-step instructions and commands provided a comprehensive guide to handling certificate management tasks.

Future Work:

Future work could focus on:

- **Automating Certificate Management**: Developing scripts to automate the process of certificate creation, signing requests, and format conversion.
- **Integrating with Certificate Authorities (CAs)**: Expanding the lab to include interactions with public and private CAs for certificate signing.
- **Advanced Security Features**: Exploring advanced OpenSSL features such as certificate revocation lists (CRLs) and implementing stronger encryption algorithms.