TNE30024

Deploying Secure Engineering Applications Online

Portfolio Task – Lab 3 Verifying a Chain of Certificates

Pass Task

1 Introduction

In this lab you will learn how to use Openssl to construct and verify a chain of certificates.

2 Purpose

To gain and/or enhance the following practical skills:

- Familiarity with an open source cryptographic system based on X.509
- Creating a chain of certificates.
- Validating certificates that are part of a chain.

3 Preparation

You can prepare for this lab by reading some of the OpenSSL documentation available at http://www.openssl.org/.

You should use this site to preview:

- Genpkey,
- X509,
- Req,
- Verify

4 Methodology

There are two parts to this lab. You will be stepped through the first part.

In the first part of this lab you will construct a root certificate and a chain of two intermediate certificates and a final certificate.

In the second part of the lab you will construct another final certificate signed by the first intermediate certificate..

4.1 Software

This lab will rely upon the OpenSSL suite of programs.

OpenSSL should be installed in your RULE host. Check that it is installed using:

```
which openssl
```

The response should be:

```
/usr/bin/openssl
```

OpenSSL commands are of the form:

```
openssl <command> <options>
```

In this lab we will use the following **OpenSSL** commands:

Command	Description
genpkey	Generate a private key
req	Generate a request for the public key derived from the private key be signed by a certificate authority
x509	Sign a digital certificate
verify	Verify the validity of a certificate chain

4.2 Construct a chain of certificates

4.2.1 Preliminaries

At the end of this section you should have constructed a chain of certificates with the following structure:

```
RootCert
   |
IntCertA1
   |
IntCertA2
   |
FinalCertA
```

1. Begin by creating a new subdirectory for this lab and moving to it:

```
cd
rm -r lab3
mkdir lab3
cd lab3
```

2. Within this subdirectory create certificate and key subdirectories:

```
mkdir certs keys
```

3. Create a certificate extensions file named **v3.ext**

```
authorityKeyIdentifier=keyid,issuer

basicConstraints=CA:TRUE

keyUsage = digitalSignature, nonRepudiation,
keyEncipherment, dataEncipherment
```

4. Generate a private key for the root certificate:

```
openssl genpkey -algorithm RSA -out
keys/root.key
```

5. Generate a self-signed (root) certificate called **root.crt** from the private key. This will ask for a number of fields. Just accept the defaults

```
openssl req -x509 -sha256 -new -key
keys/root.key -days 365 -extensions v3_ca -
out certs/root.crt
```

6. You have now created a root certificate that can be used to sign other certificates. You now have the basic capabilities of a Certificate Authority.

4.2.2 Create the first intermediate certificate (IntCertA) signed using the Root Certificate

1. Create another key that will be signed using the root key. Call it **IntCertA1.key**.

```
openssl genpkey -algorithm RSA -out
keys/IntCertA1.key
```

2. Generate a signing request for this certificate. You will be asked for a challenge text. Use the text "challenge". For the company name use **IntCertA1**. You will need to set the Basic Attribute field CA to true so that it can be used to verify certificates it signs. For all other fields use the defaults:

```
openssl req -new -key keys/IntCertA1.key -
extensions v3_ca -out req.csr
```

3. Sign the request using your root certificate to generate a new certificate called **IntCertA1.crt**:

```
openssl x509 -req -in req.csr -set_serial 10
-CA certs/root.crt -CAkey keys/root.key -
```

```
passin pass:challenge -days 365 -extfile
v3.ext -out certs/IntCertA1.crt
```

4. Finally, verify the certificate:

```
openssl verify -verbose -CAfile
certs/root.crt certs/IntCertA1.crt
```

4.2.3 Create a second intermediate certificate signed using IntCertA1

1. Create another key that will be signed using the **IntCertA1.key**. Call it **IntCertA2.key**.

```
openssl genpkey -algorithm RSA -out
keys/IntCertA2.key
```

2. Generate a signing request for this new certificate. You will be asked for a challenge text. Use the text "challenge". For the company name use **IntCertA2.** For all other fields use the defaults:

```
openssl req -new -key keys/IntCertA2.key -
extensions v3_ca -out req.csr
```

3. Sign the request using **IntCertA1** to generate a new certificate called **IntCertA2.crt**:

```
openssl x509 -req -in req.csr -set_serial 20
-CA certs/IntCertA1.crt -CAkey
keys/IntCertA1.key -passin pass:challenge -
extfile v3.ext -days 365 -out
certs/IntCertA2.crt
```

4. To verify this certificate we need to concatenate it with the intermediate certificate in a single file and then use the root certificate to verify it.
   a. Concatenate the two intermediate certificates:

```
cat certs/IntCertA1.crt certs/IntCertA2.crt >
certs/combinedCerts.crt
```

   b. Verify the certificates:

```
openssl verify -verbose -CAfile
certs/root.crt certs/combinedCerts.crt
```

5. The verification should show the certificates are ok.

4.2.4 Create a final certificate signed using IntCertA2

1. Create another key that will be signed using the **`IntCertA2.key`**. Call it **`FinalCertA.key`**.

   ```
   openssl genpkey -algorithm RSA -out
   keys/FinalCertA.key
   ```

2. Generate a signing request for this new certificate. You will be asked for a challenge text. Use the text "challenge". For the company name use **`FinalCertA.`** For all other fields use the defaults:

   ```
   openssl req -new -key keys/FinalCertA.key -
   out req.csr
   ```

3. Sign the request using **`IntCertA2`** to generate a new certificate called **`FinalCertA.crt`**:

   ```
   openssl x509 -req -in req.csr -set_serial 20
   -CA certs/IntCertA2.crt -CAkey
   keys/IntCertA2.key -passin pass:challenge -
   days 365 -out certs/FinalCertA.crt
   ```

4. Finally, verify the certificate. You will need to add it to the combined certificates file.

   a. Adding the certificate to the combined certificates file:

   ```
   cat certs/FinalCertA.crt >>
   certs/combinedCerts.crt
   ```

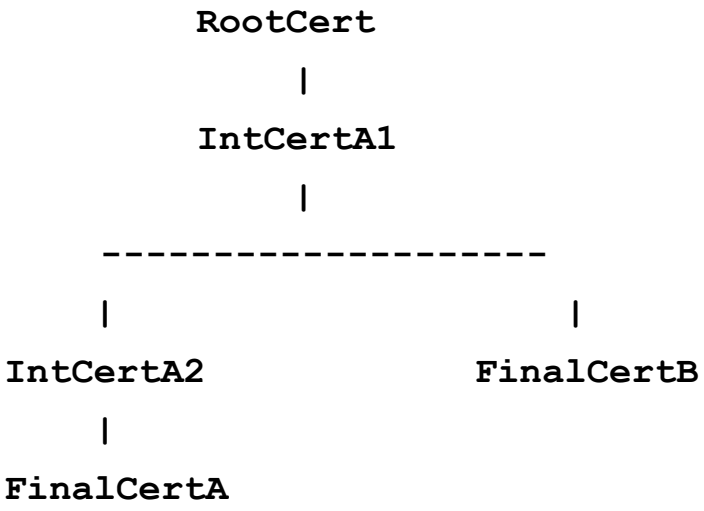
   b. Verify the certificate after moving into **certs**:

   ```
   openssl verify -verbose -CAfile root.crt
   IntCertA1.crt IntCertA2.crt FinalCertA.crt
   ```

5. What do you see? Keep a copy of the output to show the lab supervisor

4.3 Add another final certificate

In this section you are to construct an additional final certificate signed by **`IntCertA1`**. By the end of this section you should have the following structure:



1. Create another key that will be signed using the **`IntCertA1.key`**. Call it **`FinalCertB.key`**.
2. Generate a signing request for this new certificate. You will be asked for a challenge text. Use the text "challenge". For the company name use **`FinalCertB`**. For all other fields use the defaults:
3. Sign the request using **`IntCertA1`** to generate a new certificate called **`FinalCertB.crt`**:
4. Finally, verify the certificate. You will need to add it to the combined certificates file.
5. What do you see? Keep a copy of the output to show the lab supervisor

This concludes the lab task.

5 Assessment

The due date for completion of this lab is via a demonstration to your lab supervisor before the end of the lab class. You will need to show the supervisor the files you have generated in this lab including the text from verifying the certificates. Upon demonstration, your supervisor will discuss your work with you to assess your understanding.