Information Technology

OPENSSL: HOW TO EXTRACT ROOT AND INTERMEDIATE CERTIFICATES FROM CLIENT CERTIFICATE

This is a sample procedure to extract and rebuild required certificates of a Renewed SSL Cert due to either cert expiration or other situations such as additional SAN hosts were added to the cluster cert. In most cases only client certificates were re-issued (private key, public cert) and the need to get the Root Cert and Full Chain Cert need to be manually extracted/rebuilt. This situation is mostly applicable to infrastructure that uses OpenSSL or similar SSL/TLS toolkit used internally in organizations or personal systems. But the methods below can also be used on client certs issued by a Certificate Authority.

EXTRACT CLIENT CERTIFICATE

The following extracts only the client certificate and omitting the inclusion of private key (-nokeys) which supposedly not to be shared to the client users.

Syntax:

```
openssl pkcs12 -in  myCertificates.pfx -out myClientCert.crt -clcerts -nokeys
```

Example:

```
[chad@lxnode15]$ openssl pkcs12 -in lxnode15.vlabs.net.pfx -out lxnode15_client.crt -clcerts -nokeys
Enter Import Password:
MAC verified OK
[chad@lxnode15]$
```

READING THE CERTIFICATE

The goal is to determine the signing authority hosts and grab the Root certificate and Intermmediate Certificate.

Syntax:

```
openssl x509 -in myClientCert.crt -text -noout
```

NOTE: The above command will fail if the cert file is in DER format (binary)
TO READ CERTIFICATES IN DER FORMAT:

```
openssl x509 -in myClientCert.crt -inform DER -text
```

CHECK IF THE CLIENT CERT BELONGS TO THE CORRECT HOST

Proceed to read the certifiacte and look for the values indicated by the Subject CN and Alternative Name if they match the hostnames that this client cert is supposed to be installed.

CHECK CLIENT CERT EXPIRATION

Read the certficate and look for Validity section that describes the "Before" and "After" duration of the certificate.

```
[chad@lxnode15]$ openssl x509 -in lxnode15_client.crt -text -noout | grep -i not
            Not Before: Nov 16 22:32:13 2018 GMT
            Not After : Nov 15 22:32:13 2020 GMT
[chad@lxnode15]$
```

GET THE CA ISSUERS

From the client certificate, we'll grab all issuer certificates (intermmediate and root).
First, we need to get the certificate that signed the client cert (which is either an intermmediate cert or the root cert itself).

Syntax:

```
openssl x509 -in myClientCert.crt -text -noout | grep -i "issuer"
```

Example:

```
[chad@lxnode15]$ openssl x509 -in lxnode15_client.crt -text -noout | grep -i "issuer"
        Issuer: C=US, O=Chads Technoworks, CN=Chads Technoworks IC
                CA Issuers - URI:http://cert1.vlabs.net/pki/Chads%20Technoworks%20IC.crt
                CA Issuers - URI:http://cert2.vlabs.net/pki/Chads%20Technoworks%20IC.crt
[chad@lxnode15]$
```

The example above provides the URI of the signing certificate hosted by multiple servers.
Proceed to download this cert.

```
curl -O http://{cadomain.com}/{pkipath}/{name.cert}
```

or you may rename the download file with something meaningful:

```
curl -o {caSigningCert.crt} http://{cadomain.com}/{pkipath}/{name.cert}
```

sample:

```
[chad@lxnode15]$ curl -o caSigningCert.crt http://cert1.vlabs.net/pki/Chads%20Technoworks%20IC.crt
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
101  2128  101  2128    0     0  35135      0 --:--:-- --:--:-- --:--:-- 76000
[chad@lxnode15]$
```

Now that we have the signing cert, let's examine if this is the root cert by checking the common names (CN) used by the Issuer and the Subject CN. If the CN doesn't match, this means that this is an intermmediate certificate. You might then proceed to grab the signing cert of this intermmediary certificate which most likely is the root.

example:

```
[chad@lxnode15]$ openssl x509 -in caSigningCert.crt -text -noout | grep -i CN=
        Issuer: CN=Chads Technoworks Root
        Subject: C=US, O=Chads Technoworks, CN=Chads Technoworks IC
[chad@lxnode15]$
```

Note that the above CN didn't match, therefore this is an intermmediate cert. Let's go ahead and grab the signing certificate of this which most likely is the root cert.

```
[chad@lxnode15]$ openssl x509 -in caSigningCert.crt -text -noout | grep -i issuer
        Issuer: CN=Chads Technoworks Root
                CA Issuers - URI:http://cert1.vlabs.net/pki/Chads%20Technoworks%20Root.crt
                CA Issuers - URI:http://cert2.vlabs.net/pki/Chads%20Technoworks%20Root.crt
[chad@lxnode15]$

[chad@lxnode15]$ curl -o caRoot.crt http://cert1.vlabs.net/pki/Chads%20Technoworks%20Root.crt
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
101  1317  101  1317    0     0  19549      0 --:--:-- --:--:-- --:--:-- 42483
[chad@lxnode15]$
```

Let's read the cert to verify if this is the root cert:

```
[chad@lxnode15]$ openssl x509 -in caRoot.crt -text -noout | grep -i CN=
unable to load certificate
140639392556872:error:0906D06C:PEM routines:PEM_read_bio:no start line:pem_lib.c:703:Expecting: TRUSTED CERTIFICATE
[chad@lxnode15]$
```

Error above indicates that this is in DER format, thus let's read it correctly using DER extraction.

```
[chad@lxnode15]$ openssl x509 -in caRoot.crt -inform DER -text -noout | grep -i CN=
        Issuer: CN=Chads Technoworks Root
        Subject: CN=Chads Technoworks Root
[chad@lxnode15]$
```

The CN above matches both the Issuer and the CN of this certificate itself, which proves this is the root cert.

You may optionally convert the root cert into PEM format which can be helpful in building the chain cert.

Example:
```
openssl x509 -inform DER -in caRoot.crt -outform PEM -out caRoot.pem
```

CREATE A FULL CHAIN CERTIFICATE
A full chain certificate is a client certificate that has additional information of the lineage of the signing hosts tracing it back to the root.
Now that we have completed the extraction of the client cert (node cert), the intermmediate cert and root cert, we can now proceed to build the chain certificate with the following content sequence:

NODE CERT -> INTERMMEDIATE CERT -> ROOT CERT

You may use a text editor to append these certifiactes in sequence to a file.

```
cat lxnode15_client.crt caSigningCert.crt caRoot.pem > chainCert.pem
```

Note that you need to examine the chain file and remove unneccessary bag attributes and ensuring only the contents starting with "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" exists.

Also remove any ^M characters if you find any.

```
sed -e "s/^M//" file.txt > newFile.txt
```

Use [ctrl] + [v] + [m] keys to generate the special char ^M.