Main Site > Articles

Knowledge article

Creating encrypted image and decrypting that on embedded target

Created 1 year, 2 months ago Active 4 months ago Last edited 1 year, 2 months ago Viewed 63 times

1 min read

linux firmware-upgrade

Share article ... Edit article

8

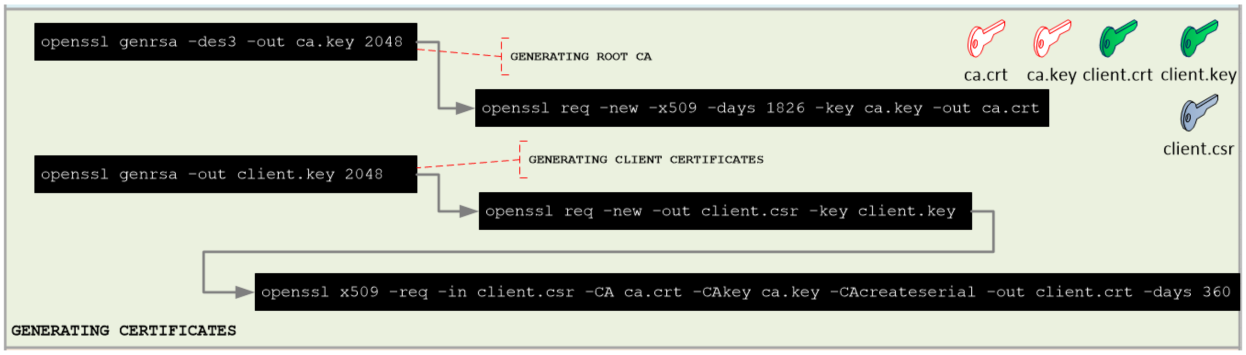
Below are the detailed steps on how to securely upgrading LINUX image onto embedded target

The steps here are generics and is one of the way that describes steps on upgrading the embedded linux target with new image. Although the steps here talks about upgrade using usb same could be applied for remote-upgrade or over the air upgrade There are two important stages involved in upgrade

- **Stage 1**: This is all about how we generate an encrypted image that could be copied to the USB or downloaded to target using remote mechanism
- **Stage 2**: Decrypting this encrypted image, verifying its authenticity before proceeding for upgrading the system

Stage1: Encryption

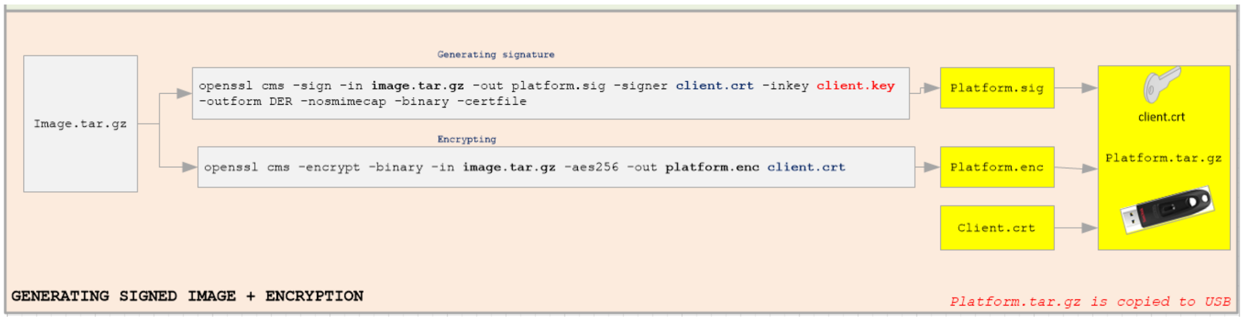
1 To encrypt an image one needs to certificates
2 For this demonstration we would be using self signed certificates
3 The very first step is to create a rootCA certificate and then client certificate that is signed with rootCA along with client csr(certificate change request)
4 The block diagram below shows the way to create rootCA, ca key, client key & csr & certificate



```
openssl genrsa -des3 -out ca.key 2048
openssl req -new -x509 -days 1826 -key ca.key -out ca.crt
openssl genrsa -out client.key 2048
openssl req -new -out client.csr -key client.key
openssl x509 -req -in client.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out client.crt -days 360
```

5 From the above outcome
- ***ca.key*** - This is something that needs to be carefuly stored and would be needed during client certificate revokation
- ***ca.crt*** - This would be never shared and this will be stored in truststore of the target
- ***client.crt*** - this certificate is signed by ca.crt and would be shared along with the tar file that will be copied to usb
- ***client.key*** - This will be stored inside truststore of target and should not be shared

6 Steps to sign and encrypt image, for this demonstration, lets consider image.tar.gz being the linux image that has Boot + RFS + Kernel that can be used for upgrading the system



```
openssl cms -sign -in image.tar.gz -out platform.sig -signer client.crt -inkey client.key
-outform DER -nosmimecap -binary -certfile
openssl cms -encrypt -binary -in image.tar.gz -aes256 -out platform.enc client.crt
```

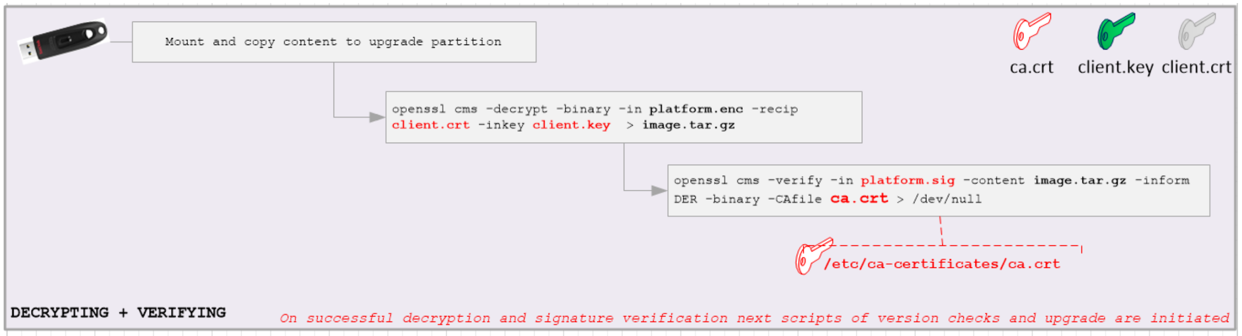
7 How are we encrypting :

```
openssl cms -sign -in image.tar.gz -out platform.sig -signer client.crt -inkey client.key
-outform DER -nosmimecap -binary -certfile ca.crt
openssl cms -verify -in platform.sig -content image.tar.gz -inform DER -binary -CAfile
ca.crt > /dev/null
openssl cms -encrypt -binary -in image.tar.gz -aes256 -out platform.enc client.crt
```

8 Create a tar file called as platform.tar.gz that contains platform.enc, platform.sig & client.crt

Stage2: Decryption

1 Remember the current image on target needs to have ca.crt and client.key stored in trust stored
2 Once USB is connected on target, there needs to be some mechanism (UDEV-rule) that detects this usb and start process of decrypting


Mount and copy content to upgrade partition
ca.crt
client.key
client.crt
/etc/ca-certificates/ca.crt
DECRYPTING + VERIFYING

```
openssl cms -decrypt -binary -in platform.enc -recip
client.crt -inkey client.key  > image.tar.gz
```

```
openssl cms -verify -in platform.sig -content image.tar.gz -inform
DER -binary -CAfile ca.crt > /dev/null
```

On successful decryption and signature verification next scripts of version checks and upgrade are initiated

3 How are we decrypting

- Untar the platyform.tar.gz and obtain
  - platform.enc
  - platform.sig
  - Client.crt

```
openssl cms -decrypt -binary -in platform.enc -recip client.crt -inkey client.key  >
image.tar.gz
openssl cms -verify -in platform.sig -content image.tar.gz -inform DER -binary -CAfile
ca.crt > /dev/null
```

Revoking expired client certificate

1 One one need the client.csr and client.key to revoke the expired certificate
2 Once you have that simply execute following

```
openssl x509 -req -in client.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out client.crt
-days 360
```

> **Note** : Remember the openssl on target and on the host machine that generates encrypted image needs to be same or of higher version else you could see decrypting & verification failures

edited Nov 27, 2021 at 15:20

created Nov 27, 2021 at 15:08
mahesh-gaikwad
753 ● 1 ● 11