CERTIFICATE PINNING WITH SWISSSIGN

1. Certificate-Pinning (Public Key Pinning Extension for http - RFC 7469)

By use of certificate pinning server operators can limit the acceptance of certificates for their web sites. The following limitations are possible:

- Single certificate
- Single certificate requests
- Issuing-CA
- Root-CA

Certificate pinning can avoid man-in-the-middle (MITM) attacks. As soon as a client has visited a page protected by certificate pinning this client will accept for a certain time frame only pinned certificates for this web site (at least one in the certificate chain).

The operator should carefully plan the variant or combination of variants of certificate pinning he wants to use. There is no common rule. The manual here describes how you could pin the SwissSign EV Gold G22 Issuing-CA or a certificate request (CSR).

Please use always 2 different intermediate CAs in your configuration. In the case that one CA is no longer valid or must be revoked you are always able to use the other configured CA.

2. Create a CSR

The created CSR is a backup if you later want to switch from SwissSign EV Gold G22 CA to another issuing CA. You will generate a private/public key pair and a CSR which will be stored safely.

First you start by generating a private/public key pair with a password protected private key. The password can be found in the file “passphrase.txt”. Key length is 4096 bit. You can also use other key length but it should never be less than 2048 bit. The concrete content of the CSR does not matter since pinning is only done on the public key.

```
$>openssl genrsa -aes256 -passout file:passphrase.txt -out backup_csr.privatekey 4096
```

Check of the key (you will be prompted for the password):

```
$>openssl rsa -in backup_csr.privatekey –check
```

Create certificate request (our example):

```
$>openssl req -new -key backup_csr.privatekey -passin file:passphrase.txt -utf8 -out backup_csr.csr -subj "/C=CH/ST=Zürich/L=Glattbrugg/O=SwissSign AG/CN=swisssign.com"
```

3. Calculate pins

3.1 Calculate the pin of a request

```
openssl req -in backup_csr.csr -pubkey -noout | openssl rsa -pubin -outform der | openssl dgst -sha256 -binary | openssl enc -base64
```

the result value will be: NDirQI6weuLiefh9EFjP0Rg8F7iLvBQE7fdD2e+j5r8=

3.2 Calculate the pin of a certificate

In case you want to calculate the pin of an already existing certificate the call should be openssl instead of req, followed by x509:

```
$>openssl x509 -in swisssign.com.pem -pubkey -noout | openssl rsa -pubin -outform der | openssl dgst -sha256 -binary | openssl enc -base64
```

3.3 Calculate the pin of a issuing CA:

Calculate the pin of the issuing CA. (you could use this pin directly if you use this issuing CA too) First you need to download the certificate:

http://swisssign.net/cgi-bin/authority/download/EEFD46CAF7275E91BC5AB6E787CD0AFA550A2642

EV_Gold_G22_2014.der

The certificate is in DER format and should be transformed into PEM format:

```
$>openssl x509 -in EV_Gold_G22_2014.der -inform DER -out EV_Gold_G22_2014.pem -outform PEM
```

In order to calculate the pin you have to extract the public key. The public key must be transformed into the DER format. Based on the public key the SHA256 hash will be calculated and base64 encoded:

```
$>openssl x509 -in EV_Gold_G22_2014.pem -pubkey -noout | openssl rsa -pubin -outform der | openssl dgst -sha256 -binary | openssl enc -base64
```

The result is:

```
mDKR5ptpp7PqVUefxx2Ftq5ymsEuzCEg+EVrLOrQFB8=
```

List of pins of the different SwissSign issuing CA's:

EV_Gold_G22_2014.pem

mDKR5ptpp7PqVUefxx2Ftq5ymsEuzCEg+EVrLOrQFB8=

Gold_G2_2006.pem

QPz8KIddzL/ry99s10MzEtpjxO/PO9extQXCICCuAnQ=

Server_Gold_G22_2014.pem

skyozdmp140ljrHvjRijq3v2/yQ1nyfFyBiA9uOKuw8=

Server_Silver_G22_2014.pem

mJwcSA1WE5bfCsQ5o79wGCvasvwdVsznZlqR1H3YPdI=

Silver_G2_2006.pem

kxgib4yDr+R/X0fCT1nOEtuoxzsYG+5rLqH0Cga8GGk=

The pins can now be used in different server configurations. Please pay attention to the fact that malformed configurations could effect that your site is no longer reachable.

4. Verification

By use of the curl command you can simply verify if the configuration is correct:

```
$>  curl -kIL https://testsite.domain

....

Public-Key-Pins:                    pin-sha256="1DIRtAGU4OG9/VFiB7K/Zx+MKYqctn8UliRGRYeX0Ko=";
pin-sha256="mJwcSA1WE5bfCsQ5o79wGCvasvwdVsznZlqR1H3YPdI=";
pin-sha256="NDirQI6weuLiefh9EFjP0Rg8F7iLvBQE7fdD2e+j5r8=";
max-age=2592000; includeSubDomains

.......
```