Generazione di coppia di chiavi pubblica/privata mediante modulo hardware e generazione del file LDIFF

Antonio De Maglio

21 novembre 2014

1 Introduzione

L'Istituto di Scienza e Tecnologie dell'Informazione(ISTI) ha un contratto con Agenzia per l´Italia Digitale (AgID) per effettuare i test di interoperabilità della Posta Elettronica Certificata(PEC).
In figura 1 possiamo vedere uno schema sul funzionamento della pec, ulteriori informazioni sulla PEC possono essere trovati nell'rfc6109 [IET11]. Questi test di interoperabilità hanno lo scopo di verificare che ogni gestore PEC sia pienamente conforme a quanto stabilito dalle "Regole tecniche del servizio di trasmissione di documenti informatici mediante posta elettronica certificata". Maggiori informazioni sui test possono essere trovate in: [MBV12].

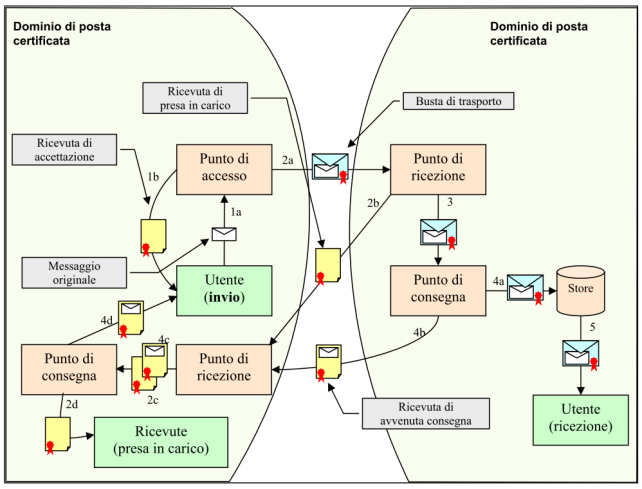

Figura 1: funzionamento PEC

L'ISTI quindi è dotato sia di un proprio gestore PEC, che da una piattaforma per l'esecuzione di una batteria di test sul gestore da verificare.
Le regole tecniche prevedono che il punto di accesso di un gestore firmi le mail con il proprio certificato X.509.
In questa guida saranno descritti tutti passi che permettono di generare la coppia di chiavi pubblica e privata sia mediante un modulo hardware HSM, che mediante chiamate openssl. Successivamente saranno illustrati tutti i passi che ci permetteranno di usare il certificato emesso da AGID per firmare le mail mediante il modulo hardware HSM. Infine descriverò i passi per poter generare e firmare il file "'The LDAP Data Interchange Format"' LDIFF) [**?**] del nostro gestore PEC che invieremo ad AGID.

2 Generazione coppie di chiavi pubbliche e private anche mediante modulo hardware

La piattaforma PEC ISTI può essere configurata sia per utilizzare un modulo hardware (HSM) per firmare i messaggi PEC, che per utilizzare invece chiamate standard openssl. Vedremo come generare le chiavi sia mediante l'HSM, che mediante chiamate a openssl.
Per il nostro gestore dovremo generare tre coppie di chiavi pubblica/privata.

- una coppia di chiavi servirà per firmare i messaggi mediante l'HSM (chiave da 2028 bit)
- un'altra coppia servirà per firmare i messaggi mediante openssl (chiave da 2048 bit)
- un'altra coppia ci permetterà di poter autenticarci sulla piattaforma IG-PEC che permette di scaricare la nuova copia del file LDIF (chiave da 1024bit).

L'HSM presente all'ISTI prodotto da nCypher, divenuta Thales-Security, permette sia di generare la chiave privata sul proprio PC e successivamente importarla nel modulo, ma anche di generare la chiave privata direttamente all'interno del modulo. In questo secondo caso, preferibile dal punto di vista della sicurezza, la chiave privata verrà conservata solo ed esclusivamente all'interno dell'HSM. L'HSM è corredato da un utility che permette di generare interattivamente la chiave privata. Per generare la chiave suddetta utilizzare i seguenti passi:

- connettersi alla macchina linux a cui è connesso il cifratore
- cd /opt/nfast/bin

Digitare il seguente comando:

```
./generatekey hwckk
```

Verrano richieste nella modalità interattiva le varie opzioni:

- protect: come verrà protetta la chiave privata, selezioneremo module
- type: indica il tipo di crittografia della chiave (RSA, DSA, Diffie Hellman) selezioneremo RSA
- size: indica la lunghezza della chiave
- ident: indica una label della chiave. Sia x il nome della chiave, ci riferiremo alla chiave come rsa-x

Una volta creata la chiave privata si può generare la request da inviare ad AGID, con il seguente comando:

```
openssl req -sha1 -config  openssl_pec_cnipa.cnf -new -out req-RV1SIGN.pem
-days 365 engine chil -keyform engine -key rsa-x
```

In particolare dobbiamo notare che "'engine"' indica che la chiave privata è contenuta nel modulo hardware, "'chil"' indica il modulo hardware nChipher. Per generare invece la chiave, che utilizzeremo se non vogliamo utilizzare il modulo hardware,e la chiave per l'autenticazione alla piattaforma igpec si userà il comando:

```
openssl genrsa -out 'key' 1024/2048
```

a seconda della lunghezza della chiave.
Ora possiamo generare le due request mediante i comandi:

```
openssl req -sha1 -new -key KEY-SRV2SIGN.PEM; -config OPENSSL_PEC_CNIPA.cnf
-out req-RV2SIGN.pem
openssl req -sha1 -new -key KEY-PECauthAGID.PEM -config openssl_auth_cnipa.cnf
-out req-PECauthAGID.pem
```

Quindi invieremo ad AGID le tre request:

- req-RV1SIGN.pem
- req-RV2SIGN.pem
- req-PECauthAGID.pem

Agid ci emetterà i certificati in formato p7b, per ottenere il certificato in formato PEM useremo il comando:

```
openssl pkcs7 -inform DER -print_certs -in certificate.p7b -out certificate.pem
```

Per utilizzare il nuovo certificato per firmare le mail mediante il modulo hardware modificheremo nel seguente modo il file che si trova in:

```
/var/pec/signfile
```

```
CERT_CRT=cnipa/srv1-firma.crt #indica il certificato emesso da AGID
CA_CERT=cnipa/pec_agid.crt #indica la catena di CA
KEY=rsa-x #indica la chiave privata memorizzata nel cifratore
```

si ricorda che durante la fase di creazione della chiave avevamo inserito come label x. A questo punto, si può provare ad inviare una mail da un utente per esempio "'user1"'. La mail verrà messa in una busta di anomalia, poichè il nuovo certificato non è ancora presente nel file LDIF, illustrato nel prossimo paragrafo.

Per quanto riguarda invece il file che ci permetterà di autenticarci sulla piattaforma IGPEC una volta che AGID avrà emesso il certificato, dopo averlo convertito in formato pem useremo il seguente comando per generare il file con estensione .p12:

```
openssl pkcs12 -export -inkey key-pecagid.pem  -in pecagid.pem -out pecagid.p12
```

il file con estensione p12, contiene sia la chiave privata, che il certificato.

3 Generazione file LDIFF

Come primo passo si deve convertire il certificato in formato DER con il comando:

```
openssl x509 -inform PEM -in cert.pem -outform DER -out cert.der
```

e successivamente convertirlo in formato base64 con il comando:

```
openssl base64 -in cert.der -out cert.b64
```

quindi possiamo generare l'hash del certificato in formato DER con il comando:

```
openssl sha1 cer.der
```

ora si può usare la procedura presente su pec per formattare il file in base64 in modo da poter essere messo nell'ldiff. In questa fase, quindi si può generare il file LDIFF del nostro gestore così costituito:

```
dn: providerName=DigitPA,o=postacert
objectClass: top
objectClass: provider
providerName: DigitPA
```

a cui appenderemo i due certificati in formato base64, successivamente si aggiungerà:

```
providerCertificateHash: hash primo certificato
providerCertificateHash: hash secondo certificato
mailReceipt: test_interop@test.cert.cnipa.it
LDIFLocationURL: https://cnipapecldif.cnipa.it/pec/cnipa.ldif.p7m
description: Servizio per la verifica dell'interoperabilita' dei Gestori PEC (
 DigitPA)
managedDomains: test.cert.cnipa.it
```

Per verificare la correttezza della procedurà si può prendere un vecchio file LDIFF, sostituire la parte del nostro server PEC con quella appena generata e caricarlo sul nostro gestore PEC. Cosi fatto si può provare ad inviare una mail da un nostro utente, per esempio "'user1"' ad un altro nostro utente, per esempio "'user1"'. Se il messaggio viene messo in una busta di anomalia vuol dire che c'è un errore, al contrario se il nostro gestore riceve la mail correttamente vuol dire che la procedura è stata eseguita tutto correttamente. Bisogna, infine firmare digitalmente il nostro file LDIFF con una delle due chiavi private dei nostri certificati mediante il comando:

```
openssl smime -sign -in "CNIPA.LDIF" -out "CNIPA.LDIF.p7m" -outform DER -inkey
```

Infine invieremo ad AGID il file CNIPA.LDIF.p7m

Riferimenti bibliografici

[IET11] IETF. Rfc6109 - la posta elettronica certificata - italian certified electronic mail. http://tools.ietf.org/html/rfc6109, 2011.

[MBV12] C. Petrucci M. Buzzi, F. Gennai and A. Vinciarelli. E-government services: Quality assurance of the italian certified electronic mail. In *E-government services: Quality Assurance of the Italian Certified Electronic Mail, HCITOCH 2012,Venezia*, 2012.

[pro] Openssl project. Openssl documents. https://www.openssl.org/docs/apps/openssl.html.