Guida all'uso di OpenPGP Card

L'**OpenPGP Card** è una smartcard sulla quale si possono salvare in modo sicuro ed usare le sottochiavi segrete usate da **GnuPG** per crittografia, firma ed anche autenticazione; ne esistono versioni tradizionali, oppure implementate in un dispositivo USB. In ogni caso si possono caricare nuove chiavi sulla scheda, ma non si possono estrarre (per via software) quelle salvate ed è la scheda stessa ad effettuare le operazioni crittografiche.

L'**obiettivo** che si vuole ottenere è di **non avere materiale crittografico segreto sui computer**, in modo che un attaccante remoto non possa impossessarsene, mantenendo comunque la comodità di poter usare le chiavi sulla stessa macchina di uso comune. Per farlo si dovrà avere una chiave principale di lunga durata conservata offline ed usata solo saltuariamente per operazioni di gestione della chiave stessa (creazione di nuove sottochiavi, modifica delle scadenze, firma di altre chiavi per la Web Of Trust) e delle sottochiavi da usare quotidianamente, conservate sulla card.

L'**OpenPGP Card** permette di caricare **tre sottochiavi con finalità di crittografia, firma ed autenticazione**: le prime due si usano normalmente tramite GnuPG, la terza può essere usata per l'autenticazione ssh tramite gpg-agent.

Prerequisiti

Tutte le versioni di GnuPG supportano l'uso di base delle OpenPGP Card, ma per meglio sfruttarne le capacità è meglio usare la versione 2.1 (modern), specialmente in fase di creazione della chiave e delle sue sottochiavi.

In questa guida si da per scontato l'uso di GnuPG 2.1 installato come eseguibile gpg, come ad esempio su Debian da stretch in poi; versioni precedenti o altre distribuzioni potrebbero usare un eseguibile gpg2 o altre varianti.

Nel caso si voglia usare poi la Card per autenticazione ssh con versioni precedenti di GnuPG è necessario installare anche gpg-agent (in 2.1 usato di default).

Per il supporto della Card è poi necessario installare PC/SC Lite.

Sotto distribuzioni Debian o derivate (che già preinstallano gnupg e openssh-client) i pacchetti da installare sono quindi:

```
apt install pcscd gnupg-agent scdaemon
```

Si tenga presente che in alcuni casi l'accesso alla Card può richiedere versioni sufficientemente aggiornate della libereria, ad esempio con il lettore O2 Micro OZ776 è necessario usare una versione di libccid sufficientemente recente (la versione 1.4.15 di Ubuntu Trusty/Mint 17 non funziona, la versione 1.4.22 di Xenial/Mint 18 si).

Si consiglia di svolgere le operazioni di creazione di chiavi e tutto ciò che coinvolga la chiave principale su un sistema pulito, ad esempio una distribuzione live, meglio se scollegato da internet, e di configurare poi le macchine di uso quotidiano secondo quanto descritto nella sezione configurazione su una nuova macchina.

Creazione della chiave

Innanzitutto bisogna creare una chiave principale che verrà associata alla nostra identità; questa dovrà essere da almeno 4096 bit per garantirle sufficiente longevità, dato che la sua sostituzione comporta un nuovo ingresso nella Web of Trust.

Ovviamente, se si ha già una chiave gpg da 4096 bit si può saltare questo passaggio ed usare quella.

Innanzitutto, configurare gpg per l'uso di SHA2 aggiungendo le seguenti righe a ~/.gnupg/gpg.conf:

```
      personal-digest-preferences SHA256
      cert-digest-algo SHA256
      default-preference-list SHA512 SHA384 SHA256 SHA224 AES256 AES192 AES
CAST5 ZLIB BZIP2 ZIP Uncompressed
```

Quindi generare una nuova chiave: notare l'uso di gpg versione 2.1 e la dimensione a 4096 bit (per il resto si sono accettate le scelte di default oppure inseriti i propri dati).

```
      $ gpg --full-gen-key
      gpg (GnuPG) 2.1.15; Copyright (C) 2016 Free Software Foundation, Inc.
      This is free software: you are free to change and redistribute it.
      There is NO WARRANTY, to the extent permitted by law.

      Please select what kind of key you want:
         (1) RSA and RSA (default)
         (2) DSA and Elgamal
         (3) DSA (sign only)
         (4) RSA (sign only)
      Your selection? 1
      RSA keys may be between 1024 and 4096 bits long.
      What keysize do you want? (2048) 4096
      Requested keysize is 4096 bits
      Please specify how long the key should be valid.
               0 = key does not expire
            <n>  = key expires in n days
            <n>w = key expires in n weeks
            <n>m = key expires in n months
            <n>y = key expires in n years
      Key is valid for? (0)
      Key does not expire at all
      Is this correct? (y/N) y

      GnuPG needs to construct a user ID to identify your key.

      Real name: Random User
```

```
Email address: random@example.org
Comment:
You selected this USER-ID:
    "Random User <random@example.org>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: key 3E0EE5BAC50DF7C1 marked as ultimately trusted
gpg: directory '/home/random/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/valhalla/.gnupg/openpgp-
revocs.d/19DB75A75734ABBEDB1124163E0EE5BAC50DF7C1.rev'
public and secret key created and signed.

pub   rsa4096 2016-10-19 [SC]
      19DB75A75734ABBEDB1124163E0EE5BAC50DF7C1
      19DB75A75734ABBEDB1124163E0EE5BAC50DF7C1
uid                      Random User <random@example.org>
sub   rsa4096 2016-10-19 [E]
```

Creazione delle sottochiavi

Questo è il punto in cui è particolarmente importante l'uso di GnuPG 2.1, che rende molto più semplice la creazione di una chiave per l'autenticazione.

Molte delle OpenPGP Card disponibili sul mercato supportano chiavi al massimo a 2048 bit: questo non è un grosso problema dato che al momento sono considerate ancora ragionevolmente sicure, ed una loro eventuale sostituzione in futuro non è particolarmente onerosa (oltre ovviamente alla necessità di usare una Card o un dispositivo diverso).

Per generare le sottochiavi necessarie, si entra in modalità di modifica con l'opzione --expert che abilita la creazione di chiavi di autenticazione.

```
$ gpg --expert --edit-key 3E0EE5BAC50DF7C1
gpg (GnuPG) 2.1.15; Copyright (C) 2016 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Secret key is available.

sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa4096/F303978FEBB6E995
     created: 2016-10-19  expires: never       usage: E
[ultimate] (1). Random User <random@example.org>
```

Quindi si crea prima una normale sottochiave di firma (opzione 4), alla selezione verrà chiesta (due volte) una passphrase per la nuova sottochiave ed anche la passphrase per la chiave principale:

```
gpg> addkey
Please select what kind of key you want:
   (3) DSA (sign only)
   (4) RSA (sign only)
   (5) Elgamal (encrypt only)
   (6) RSA (encrypt only)
   (7) DSA (set your own capabilities)
   (8) RSA (set your own capabilities)
  (10) ECC (sign only)
  (11) ECC (set your own capabilities)
  (12) ECC (encrypt only)
  (13) Existing key
Your selection? 4
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 1y
Key expires at Thu 19 Oct 2017 12:15:49 CEST
Is this correct? (y/N) y
Really create? (y/N) y
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.

sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa4096/F303978FEBB6E995
     created: 2016-10-19  expires: never       usage: E
ssb  rsa2048/40A2ADF0FE9E4620
     created: 2016-10-19  expires: 2017-10-19  usage: S
[ultimate] (1). Random User <random@example.org>
```

Poi occorre creare una normale sottochiave per la crittografia (opzione 6), anche in questo caso verrà chiesta due volte la passphrase per la nuova chiave, ma avendola già data prima non sarà richiesta quella della chiave principale:

```
gpg> addkey
Please select what kind of key you want:
   (3) DSA (sign only)
   (4) RSA (sign only)
   (5) Elgamal (encrypt only)
   (6) RSA (encrypt only)
   (7) DSA (set your own capabilities)
   (8) RSA (set your own capabilities)
  (10) ECC (sign only)
  (11) ECC (set your own capabilities)
  (12) ECC (encrypt only)
  (13) Existing key
Your selection? 6
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 1y
Key expires at Thu 19 Oct 2017 12:18:16 CEST
Is this correct? (y/N) y
Really create? (y/N) y
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.

sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa4096/F303978FEBB6E995
     created: 2016-10-19  expires: never       usage: E
ssb  rsa2048/40A2ADF0FE9E4620
     created: 2016-10-19  expires: 2017-10-19  usage: S
ssb  rsa2048/ED9BBD7B3BFF900F
     created: 2016-10-19  expires: 2017-10-19  usage: E
[ultimate] (1). Random User <random@example.org>
```

Ed infine una sottochiave per l'autenticazione, selezionando l'opzione 8, togliendo le capacità di firma e crittografia e mettendo quella di autenticazione (anche qui occorrerà specificare una passphrase).

```
gpg> addkey
Please select what kind of key you want:
   (3) DSA (sign only)
   (4) RSA (sign only)
   (5) Elgamal (encrypt only)
   (6) RSA (encrypt only)
   (7) DSA (set your own capabilities)
   (8) RSA (set your own capabilities)
  (10) ECC (sign only)
  (11) ECC (set your own capabilities)
  (12) ECC (encrypt only)
  (13) Existing key
Your selection? 8

Possible actions for a RSA key: Sign Encrypt Authenticate
Current allowed actions: Sign Encrypt

   (S) Toggle the sign capability
   (E) Toggle the encrypt capability
   (A) Toggle the authenticate capability
   (Q) Finished

Your selection? s

Possible actions for a RSA key: Sign Encrypt Authenticate
Current allowed actions: Encrypt

   (S) Toggle the sign capability
   (E) Toggle the encrypt capability
   (A) Toggle the authenticate capability
   (Q) Finished

Your selection? e

Possible actions for a RSA key: Sign Encrypt Authenticate
Current allowed actions:

   (S) Toggle the sign capability
   (E) Toggle the encrypt capability
   (A) Toggle the authenticate capability
   (Q) Finished
```

```
Your selection? a

Possible actions for a RSA key: Sign Encrypt Authenticate
Current allowed actions: Authenticate

   (S) Toggle the sign capability
   (E) Toggle the encrypt capability
   (A) Toggle the authenticate capability
   (Q) Finished

Your selection? q
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 1y
Key expires at Thu 19 Oct 2017 12:24:56 CEST
Is this correct? (y/N) y
Really create? (y/N) y
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.

sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa4096/F303978FEBB6E995
     created: 2016-10-19  expires: never       usage: E
ssb  rsa2048/40A2ADF0FE9E4620
     created: 2016-10-19  expires: 2017-10-19  usage: S
ssb  rsa2048/ED9BBD7B3BFF900F
     created: 2016-10-19  expires: 2017-10-19  usage: E
ssb  rsa2048/5F89B668295AE797
     created: 2016-10-19  expires: 2017-10-19  usage: A
[ultimate] (1). Random User <random@example.org>
```

Si può quindi uscire dalla modalità di modifica chiave, salvando le modifiche.

```
gpg> save
Salvataggio su supporto esterno
```

A questo punto la chiave è ancora gestita nel modo tradizionale, su hard disk. Questo è il momento di salvarne un backup e di farne una copia su chiavetta USB da usare in futuro per le modifiche della chiave.

Per il backup ci sono vari metodi, da una semplice e poco efficiente copia di tutta la ~/.gnupg su un ulteriore supporto all'uso di paperkey per salvare una copia su carta; l'importante è che venga fatto e custodito in un luogo sicuro, anche se magari remoto (cassetta di sicurezza).

Per avere una copia della chiave principale da usare sporadicamente, invece, è necessario copiare l'intera ~/.gnupg su una chiavetta USB o simili; una volta fatta la copia verificare che contenga quanto dovuto con:

```
$ gpg --homedir indirizzo/della/chiavetta/gnupg --list-secret-keys
indirizzo/della/chiavetta/gnupg
pubring.gpg
--------------------------------
sec   rsa4096 2016-10-19 [SC]
      19DB75A75734ABBEDB1124163E0EE5BAC50DF7C1
uid           [ultimate] Random User <random@example.org>
ssb   rsa4096 2016-10-19 [E]
ssb   rsa2048 2016-10-19 [S] [expires: 2017-10-19]
ssb   rsa2048 2016-10-19 [E] [expires: 2017-10-19]
ssb   rsa2048 2016-10-19 [A] [expires: 2017-10-19]
```

smontare la chiavetta e riporla in un luogo sicuro; la si userà nuovamente quando sarà necessario cambiare le date di scadenza delle sottochiavi, oppure firmare chiavi di altre persone, per la Web of Trust.

Spostamento delle chiavi sulla Card

Per il prossimo passaggio è estremamente importante assicurarsi di aver fatto le copie delle chiavi e sottochiavi private: in questo passaggio verranno trasferite sulla Card, dalla quale non possono più essere estratte, e cancellate dal disco.

Collegare il lettore al computer, inserire la Card e controllare che venga correttamente riconosciuta:

```
$ gpg --card-status
Reader ...........: ####
Application ID ...: ####
Version ..........: 2.0
Manufacturer .....: ####
Serial number ....: ####
```

```
Name of cardholder: [not set]
Language prefs ...: [not set]
Sex ..............: unspecified
URL of public key : [not set]
Login data .......: [not set]
Signature PIN ....: forced
Key attributes ...: rsa2048 rsa2048 rsa2048
Max. PIN lengths .: 127 127 127
PIN retry counter : 3 3 3
Signature counter : 0
Signature key ....: [none]
Encryption key....: [none]
Authentication key: [none]
General key info..: [none]
```

Se la Card è nuova occorre cambiare poi i PIN della chiave rispetto ai default:

```
$ gpg --card-edit
Reader ...........: ####
Application ID ...: ####
Version ..........: 2.0
Manufacturer .....: ####
Serial number ....: ####
Name of cardholder: [not set]
Language prefs ...: [not set]
Sex ..............: unspecified
URL of public key : [not set]
Login data .......: [not set]
Signature PIN ....: forced
Key attributes ...: rsa2048 rsa2048 rsa2048
Max. PIN lengths .: 127 127 127
PIN retry counter : 3 3 3
Signature counter : 0
Signature key ....: [none]
Encryption key....: [none]
Authentication key: [none]
General key info..: [none]

gpg/card> admin
Admin commands are allowed

gpg/card> passwd
gpg: OpenPGP card no. #### detected
```

```
1 - change PIN
2 - unblock PIN
3 - change Admin PIN
4 - set the Reset Code
Q - quit

Your selection? 1
```

a questo punto verrà chiesto il PIN Standard corrente (il default della carta è 123456) e poi due volte quello nuovo (si faccia attenzione a mettere il PIN corretto, si hanno un massimo di tre errori possibili, dopo verrà bloccato, lo si può sbloccare solo con il PIN amministrativo). Quest'ultimo si cambia dal default selezionando ulteriormente:

```
PIN changed.

1 - change PIN
2 - unblock PIN
3 - change Admin PIN
4 - set the Reset Code
Q - quit

Selezione? 3
PIN changed.
```

dando quello vecchio e ripetendo due volte quello nuovo; qui però tre errori nel dare quello corrente comporteranno l'impossibilità di modificarlo, rendendo di fatto non più recuperabile la Card senza un reset alle condizioni di fabbrica.

A questo menù si può accedere direttamente anche con il comando gpg --change-pin, e ad esempio per recuperare un PIN dimenticato si può selezionare l'opzione 2, Unblock PIN nel qual caso verrà chiesto il PIN amministrativo, e si potrà impostare un nuovo PIN.

Si può quindi passare a spostare le varie sottochiavi sulla Card, aprendo la modalità di editing della chiave:

```
$ gpg --edit-key 3E0EE5BAC50DF7C1
gpg (GnuPG) 2.1.15; Copyright (C) 2016 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Secret key is available.

sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa4096/F303978FEBB6E995
```

```
     created: 2016-10-19  expires: never       usage: E
ssb  rsa2048/40A2ADF0FE9E4620
     created: 2016-10-19  expires: 2017-10-19  usage: S
ssb  rsa2048/ED9BBD7B3BFF900F
     created: 2016-10-19  expires: 2017-10-19  usage: E
ssb  rsa2048/5F89B668295AE797
     created: 2016-10-19  expires: 2017-10-19  usage: A
[ultimate] (1). Random User <random@example.org>
```

passando alla modalità di modifica delle chiavi private, con il comando toggle che consente di selezionare le chiavi:

```
gpg> toggle

sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa4096/F303978FEBB6E995
     created: 2016-10-19  expires: never       usage: E
ssb  rsa2048/40A2ADF0FE9E4620
     created: 2016-10-19  expires: 2017-10-19  usage: S
ssb  rsa2048/ED9BBD7B3BFF900F
     created: 2016-10-19  expires: 2017-10-19  usage: E
ssb  rsa2048/5F89B668295AE797
     created: 2016-10-19  expires: 2017-10-19  usage: A
[ultimate] (1). Random User <random@example.org>
```

per trasferire una ad una le sottochiavi sulla Card andranno selezionate ed inserite, una volta usato toggle il comando key permette di selezionare/deselezionare le chiavi da trasferire con il comando keytocard; pertanto per trasferire la chiave di firma useremo i comandi:

```
gpg> key 2

sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa4096/F303978FEBB6E995
     created: 2016-10-19  expires: never       usage: E
ssb* rsa2048/40A2ADF0FE9E4620
     created: 2016-10-19  expires: 2017-10-19  usage: S
ssb  rsa2048/ED9BBD7B3BFF900F
     created: 2016-10-19  expires: 2017-10-19  usage: E
ssb  rsa2048/5F89B668295AE797
     created: 2016-10-19  expires: 2017-10-19  usage: A
[ultimate] (1). Random User <random@example.org>
```

```
gpg> keytocard
Please select where to store the key:
   (1) Signature key
   (3) Authentication key
Your selection? 1
```

qui occorre selezionare il tipo di chiave e poi verrà chiesta la passphrase della chiave e poi proseguirà con:

```
sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa4096/F303978FEBB6E995
     created: 2016-10-19  expires: never       usage: E
ssb* rsa2048/40A2ADF0FE9E4620
     created: 2016-10-19  expires: 2017-10-19  usage: S
ssb  rsa2048/ED9BBD7B3BFF900F
     created: 2016-10-19  expires: 2017-10-19  usage: E
ssb  rsa2048/5F89B668295AE797
     created: 2016-10-19  expires: 2017-10-19  usage: A
[ultimate] (1). Random User <random@example.org>
```

per trasferire quella di crittografia occorrerà deselezionare la chiave di firma e selezionare quella di autenticazione, e ripetere il comando keytocard:

```
gpg> key 2

sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa4096/F303978FEBB6E995
     created: 2016-10-19  expires: never       usage: E
ssb  rsa2048/40A2ADF0FE9E4620
     created: 2016-10-19  expires: 2017-10-19  usage: S
ssb  rsa2048/ED9BBD7B3BFF900F
     created: 2016-10-19  expires: 2017-10-19  usage: E
ssb  rsa2048/5F89B668295AE797
     created: 2016-10-19  expires: 2017-10-19  usage: A
[ultimate] (1). Random User <random@example.org>

gpg> key 3

sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
```

```
      trust: ultimate      validity: ultimate
ssb   rsa4096/F303978FEBB6E995
      created: 2016-10-19  expires: never       usage: E
ssb   rsa2048/40A2ADF0FE9E4620
      created: 2016-10-19  expires: 2017-10-19  usage: S
ssb*  rsa2048/ED9BBD7B3BFF900F
      created: 2016-10-19  expires: 2017-10-19  usage: E
ssb   rsa2048/5F89B668295AE797
      created: 2016-10-19  expires: 2017-10-19  usage: A
[ultimate] (1). Random User <random@example.org>

gpg> keytocard
Please select where to store the key:
   (2) Encryption key
Your selection? 2
```

di nuovo si sceglie (stavolta solo chiave di cifratura) e verrà chiesta la passphrase della chiave scelta:

```
sec   rsa4096/3E0EE5BAC50DF7C1
      created: 2016-10-19  expires: never       usage: SC
      trust: ultimate      validity: ultimate
ssb   rsa4096/F303978FEBB6E995
      created: 2016-10-19  expires: never       usage: E
ssb   rsa2048/40A2ADF0FE9E4620
      created: 2016-10-19  expires: 2017-10-19  usage: S
ssb*  rsa2048/ED9BBD7B3BFF900F
      created: 2016-10-19  expires: 2017-10-19  usage: E
ssb   rsa2048/5F89B668295AE797
      created: 2016-10-19  expires: 2017-10-19  usage: A
[ultimate] (1). Random User <random@example.org>
infine si ripeterà la procedura con la chiave di autenticazione:

gpg> key 3

sec   rsa4096/3E0EE5BAC50DF7C1
      created: 2016-10-19  expires: never       usage: SC
      trust: ultimate      validity: ultimate
ssb   rsa4096/F303978FEBB6E995
      created: 2016-10-19  expires: never       usage: E
ssb   rsa2048/40A2ADF0FE9E4620
      created: 2016-10-19  expires: 2017-10-19  usage: S
ssb   rsa2048/ED9BBD7B3BFF900F
      created: 2016-10-19  expires: 2017-10-19  usage: E
ssb   rsa2048/5F89B668295AE797
```

```
     created: 2016-10-19  expires: 2017-10-19  usage: A
[ultimate] (1). Random User <random@example.org>

gpg> key 4

sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa4096/F303978FEBB6E995
     created: 2016-10-19  expires: never       usage: E
ssb  rsa2048/40A2ADF0FE9E4620
     created: 2016-10-19  expires: 2017-10-19  usage: S
ssb  rsa2048/ED9BBD7B3BFF900F
     created: 2016-10-19  expires: 2017-10-19  usage: E
ssb* rsa2048/5F89B668295AE797
     created: 2016-10-19  expires: 2017-10-19  usage: A
[ultimate] (1). Random User <random@example.org>

gpg> keytocard
Please select where to store the key:
   (3) Authentication key
Your selection? 3

sec  rsa4096/3E0EE5BAC50DF7C1
     created: 2016-10-19  expires: never       usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa4096/F303978FEBB6E995
     created: 2016-10-19  expires: never       usage: E
ssb  rsa2048/40A2ADF0FE9E4620
     created: 2016-10-19  expires: 2017-10-19  usage: S
ssb  rsa2048/ED9BBD7B3BFF900F
     created: 2016-10-19  expires: 2017-10-19  usage: E
ssb* rsa2048/5F89B668295AE797
     created: 2016-10-19  expires: 2017-10-19  usage: A
[ultimate] (1). Random User <random@example.org>
Uscire salvando:

gpg> save
```

e a questo punto si può controllare che le sottochiavi sono sulla Card:

```
$ gpg --card-status

Reader ...........: ####
```

```
      Application ID ...: ####
      Version ..........: 2.0
      Manufacturer .....: ####
      Serial number ....: ####
      Name of cardholder: [not set]
      Language prefs ...: [not set]
      Sex ..............: unspecified
      URL of public key : [not set]
      Login data .......: [not set]
      Signature PIN ....: forced
      Key attributes ...: rsa2048 rsa2048 rsa2048
      Max. PIN lengths .: 127 127 127
      PIN retry counter : 3 3 3
      Signature counter : 0
      Signature key ....: 2128 412C BC75 34F5 6305  9447 40A2 ADF0 FE9E 4620
            created ....: 2016-10-19 10:15:26
      Encryption key....: 0B24 9C62 4A7D 19FE 7AB9  DCB3 ED9B BD7B 3BFF 900F
            created ....: 2016-10-19 10:17:57
      Authentication key: 40E1 AE4A 7C13 FD8C C6EA  2C56 5F89 B668 295A E797
            created ....: 2016-10-19 10:24:14
      General key info..: sub  rsa2048/40A2ADF0FE9E4620 2016-10-19 Random
User <random@example.org>
      sec   rsa4096/3E0EE5BAC50DF7C1 created: 2016-10-19  expires: never
      ssb   rsa4096/F303978FEBB6E995 created: 2016-10-19  expires: never
      ssb>  rsa2048/40A2ADF0FE9E4620 created: 2016-10-19  expires: 2017-10-19
                                     card-no: ####
      ssb>  rsa2048/ED9BBD7B3BFF900F created: 2016-10-19  expires: 2017-10-19
                                     card-no: ####
      ssb>  rsa2048/5F89B668295AE797 created: 2016-10-19  expires: 2017-10-19
                                     card-no: ####
      e non sono più presenti sul disco

      $ gpg --list-secret-keys
      /home/random/.gnupg/pubring.gpg
      --------------------------------
      sec   rsa4096 2016-10-19 [SC]
            19DB75A75734ABBEDB1124163E0EE5BAC50DF7C1
      uid           [ultimate] Random User <random@example.org>
      ssb   rsa4096 2016-10-19 [E]
      ssb>  rsa2048 2016-10-19 [S] [expires: 2017-10-19]
      ssb>  rsa2048 2016-10-19 [E] [expires: 2017-10-19]
      ssb>  rsa2048 2016-10-19 [A] [expires: 2017-10-19]
```

è però ancora presente la chiave principale.

Rimozione della chiave principale

Usando gpg 2.1 si possono rimuovere sottochiavi semplicemente cancellando file in ~/.gnupg/private-keys-v1.d/; per scoprire quale file cancellare bisogna scoprire il keygrip della chiave in questione:

```
$ gpg --with-keygrip --list-key 3E0EE5BAC50DF7C1
pub   rsa4096 2016-10-19 [SC]
      19DB75A75734ABBEDB1124163E0EE5BAC50DF7C1
      Keygrip = 3BC042D5749A498AA2F904CF548C6B5DEDDF0600
uid           [ultimate] Random User <random@example.org>
sub   rsa4096 2016-10-19 [E]
      Keygrip = 0B2A484CBE52875A07F377A1A87F229C60277642
sub   rsa2048 2016-10-19 [S] [expires: 2017-10-19]
      Keygrip = AD155C5349ECE4D3725AD77269EC9E696A8191E9
sub   rsa2048 2016-10-19 [E] [expires: 2017-10-19]
      Keygrip = BFADD9EEDA2E38A77DCF925E58FFD4FFA2F50B7C
sub   rsa2048 2016-10-19 [A] [expires: 2017-10-19]
      Keygrip = D9F3249B5F583B7B4099B80DCAB343A9BA440BA8
```

A questo punto è sufficiente rimuovere il file corrispondente alla chiave principale:

```
$ rm ~/.gnupg/private-keys-
v1.d/3BC042D5749A498AA2F904CF548C6B5DEDDF0600.key
```

E si può verificare che la chiave sia stata effettivamente rimossa:

```
$ gpg --list-secret-keys 3E0EE5BAC50DF7C1
sec#  rsa4096 2016-10-19 [SC]
      19DB75A75734ABBEDB1124163E0EE5BAC50DF7C1
uid           [ultimate] Random User <random@example.org>
ssb   rsa4096 2016-10-19 [E]
ssb>  rsa2048 2016-10-19 [S] [expires: 2017-10-19]
ssb>  rsa2048 2016-10-19 [E] [expires: 2017-10-19]
ssb>  rsa2048 2016-10-19 [A] [expires: 2017-10-19]
```

dove # dopo sec indica che la chiave segreta non è presente sulla macchina in questione.

Configurazione di ssh

Per abilitare l'autenticazione ssh tramite chiave, aggiungere la seguente riga al file ~/.gnupg/gpg-agent.conf:

```
enable-ssh-support
```

Con una versione sufficientemente recente di gpg questo è sufficiente per far sì che le connessioni SSH usino la chiave gpg per l'autenticazione.

Truelite Srl
Via Monferrato, 6
50142 Firenze
055.7879597
www.truelite.it - info@truelite.it
P. Iva 05316110484 - Capitale sociale 15.000 euro i.v.
Registro imprese Firenze n. 05316110484 - R.E.A. Firenze n. 537498

Alla prima connessione ssh verso un server sul quale possiamo identificarci tramite chiave verrà lanciato il gpg-agent che a sua volta lancerà un pinentry per chiedere il pin che abbiamo impostato sulla card.

Da quel momento in poi il pin verrà tenuto in memoria in modo sicuro, e sarà possibile collegarsi ai server senza digitare nuovamente password.

Per far dimenticare il pin è sufficiente rimuovere la card dal lettore.

Agent forwarding

Con versioni sufficientemente recenti di gnupg e openssh, l'agent forwarding si configura come se si stesse usando l'agent ssh, aggiungendo a ~/.ssh/config delle righe tipo:

```
Host example.org
    ForwardAgent yes
```

per gli host per i quali si desidera abilitare il forwarding.

Troubleshooting

Il funzionamento delle PGPCard per autenticazione è stato reso completamente automatico solo nelle versioni più recenti di gnupg e delle distribuzioni: su versioni precedenti (e.g. Debian jessie) potrebbe essere necessario intervenire manualmente.

Il caso di non funzionamento più comune è che il gpg-agent non stia effettivamente girando: dato che generalmente è lanciato alla bisogna da GnuPG, è sufficiente usare un comando gpg, ad esempio gpg --card-status e da quel momento in poi l'agent sarà disponibile per ssh.

Un secondo caso di non funzionamento potrebbe essere dovuto alla presenza di un altro agent per ssh: da Debian stretch in poi questo non dovrebbe essere presente, ma vecchie versioni o altre distribuzioni potrebbero lanciarlo ad esempio tramite degli script in /etc/X11/Xsession.d (in Debian 90x11-common_ssh-agent, da disattivare.

Un'ulteriore causa di problemi possono essere le vecchie versioni di gnome-keyring-daemon che cercano di sostituirsi all'agent, senza però implementarne tutte le funzionalità: in questo caso per permettere a gpg-agent di fare il suo lavoro è necessario disabilitarlo ad esempio modificando /etc/xdg/autostart/gnome-keyring-ssh.desktop e /etc/xdg/autostart/gnome-keyring-gpg.desktop e sostituendo una riga tipo:

```
OnlyShowIn=GNOME;Unity;MATE;
```

con

```
OnlyShowIn=
```


Truelite Srl
Via Monferrato, 6
50142 Firenze
055.7879597
www.truelite.it - info@truelite.it

Esportazione della chiave pubblica

Adesso che tutto è funzionante, si può inviare la propria chiave pubblica sui keyserver:

```
$ gpg --send-keys 3E0EE5BAC50DF7C1
gpg: sending key 3E0EE5BAC50DF7C1 to [server]
```

Nel caso non abbiate configurato un server in ~/.gnupg/dirmngr.conf (o in ~/.gnupg/gpg.conf) lo si può specificare sulla riga di comando:

```
$ gpg --keyserver keys.gnupg.net --send-keys 3E0EE5BAC50DF7C1
gpg: sending key 3E0EE5BAC50DF7C1 to hkp://keys.gnupg.net
```

Se la chiave non deve essere pubblica, è comunque necessario estrarne la parte pubblica da importare su altre macchine dove si vuole usare la Card:

```
gpg --armour --export 3E0EE5BAC50DF7C1 > 3E0EE5BAC50DF7C1.asc
```

Configurazione su una nuova macchina

Per poter usare la Card si una macchina diversa è necessario far conoscere a GnuPG la chiave: innanzitutto importando la chiave pubblica:

```
gpg --recv-key 3E0EE5BAC50DF7C1
```

oppure:

```
gpg --import 3E0EE5BAC50DF7C1.asc
```

e quindi far ricreare gli stub di chiavi dalle chiavi presenti sulla card, semplicemente con:

```
$ gpg --card-status

Reader ...........: ####
Application ID ...: ####
Version ..........: 2.0
Manufacturer .....: ####
Serial number ....: ####
Name of cardholder: [not set]
Language prefs ...: [not set]
Sex ..............: unspecified
URL of public key : [not set]
Login data .......: [not set]
Signature PIN ....: forced
Key attributes ...: rsa2048 rsa2048 rsa2048
```

```
      Max. PIN lengths .: 127 127 127
      PIN retry counter : 3 3 3
      Signature counter : 0
      Signature key ....: 2128 412C BC75 34F5 6305  9447 40A2 ADF0 FE9E 4620
            created ....: 2016-10-19 10:15:26
      Encryption key....: 0B24 9C62 4A7D 19FE 7AB9  DCB3 ED9B BD7B 3BFF 900F
            created ....: 2016-10-19 10:17:57
      Authentication key: 40E1 AE4A 7C13 FD8C C6EA  2C56 5F89 B668 295A E797
            created ....: 2016-10-19 10:24:14
      General key info..: sub  rsa2048/40A2ADF0FE9E4620 2016-10-19 Random
User <random@example.org>
      sec   rsa4096/3E0EE5BAC50DF7C1  created: 2016-10-19 expires: never
      ssb   rsa4096/F303978FEBB6E995  created: 2016-10-19 expires: never
      ssb>  rsa2048/40A2ADF0FE9E4620  created: 2016-10-19 expires: 2017-10-19
                                      card-no: ####
      ssb>  rsa2048/ED9BBD7B3BFF900F  created: 2016-10-19 expires: 2017-10-19
                                      card-no: ####
      ssb>  rsa2048/5F89B668295AE797  created: 2016-10-19 expires: 2017-10-19
                                      card-no: ####
```