

LINUX@EVENING

SEMINARI di
INFORMATICA LIBERA

solira 

Introduzione alla Crittografia e a GnuPG

mario di raimondo



Sommario



- La Privacy
- Introduzione alla Crittografia
- Da PGP a GnuPG
- Imparare ad usare GnuPG:
 - creazione di chiavi
 - firmare e cifrare
 - gestione delle chiavi

- Le moderni reti di computer ci hanno semplificato la vita e ci hanno messo a disposizione **nuovi strumenti**:

- posta elettronica;
- il web;



- attraverso loro inviamo giornalmente **informazioni confidenziali**:

- corrispondenza personale;
- numeri di carta di credito;
- informazioni sensibili.



Corriamo dei rischi?

- La natura di questi strumenti introduce alcuni problemi:
 - **intercettazioni** (o *sniffing*);
 - vecchio problema;
 - reti “*switchate*”;
 - reti *wireless*!
 - **falsificazioni di identità**;



La Crittografia



- strumento molto importante per preservare la Privacy e la cura della Sicurezza;
- problematiche:
 - segretezza;
 - autenticazione;
- strumenti:
 - sistemi di cifratura;
 - firma digitale.

Crittografia a chiave segreta (o a chiave simmetrica)



- Alice e Bob condividono una **chiave segreta comune**;
- Alice usa la chiave per **cifrare** il messaggio;
- Bob riceve il messaggio e lo **decifra** usando la sua copia della chiave (la stessa);
- Lontane origini:
 - Scitale + fettuccia di cuoio (Spartani, V secolo AC);
 - Cesare;
 - Enigma (Tedeschi, II guerra mondiale).

Cifrature simmetriche odierne



- **DES** (1977-1998): chiave a 64 bit
- **Triple-DES**: chiavi a 192 bit
- **IDEA** (1991, brevettato): chiavi a 128 bit
- **Blowfish** (1993): chiavi da 32 a 448 bit
- **AES** [Rijndael] (2001): chiavi a 128, 192 o 256 bit
- e tanti altri: Serpent, Twofish, CAST5, RC4, RC5, ...

Distribuzione delle chiavi



- Il problema principale dei cifrari simmetrici è la necessità di scambiarsi in modo sicuro (**canale sicuro**) la chiave di cifratura;
- Se ci sono n persone, sono necessarie $n(n-1)/2$ chiavi (scambiate in modo sicuro);
- Questo può essere un **grosso problema**:
 - gruppo molto grande;
 - persone distanti (quindi mancanza di canali sicuri);
 - grande numero di chiavi.

Crittografia a chiave pubblica

- Nel 1976 **Whitfield Diffie** and **Martin Hellman** introdussero il concetto di Crittografia a chiave pubblica (o asimmetrici);



- Alice ha due chiavi:

◆ **chiave pubblica** (comunicata a Bob);

🔑 **chiave segreta** (mantenuta confidenziale);



- Bob usa la chiave pubblica di Alice per cifrare il messaggio;
- Alice usa la sua chiave segreta per decifrarlo;
- analogia: “cassaforte aperta”

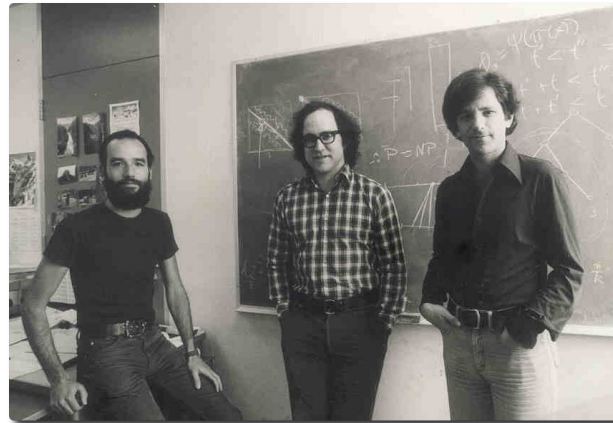
La coppia di chiavi



- La relazione tra le due chiavi deriva da:
 - particolari **proprietà matematiche**;
 - l'assunzione che certi problemi matematici siano difficili da risolvere in tempi pratici:
 - fattorizzazione di grandi numeri;
 - calcolo del logaritmo discreto su certi gruppi algebrici.
- **Proprietà di sicurezza:**
 - la chiave pubblica non da alcuna informazione sulla chiave segreta;
 - il messaggio cifrato non rivela nessuna informazione sul suo contenuto (pur conoscendo la chiave pubblica).
- Risolve il problema di distribuzione delle chiavi!

Esempi di cifrari asimmetrici: RSA

- Nel 1977 **Rivest, Shamir e Adleman** (MIT) pubblicarono il primo sistema di cifratura asimmetrico: il sistema **RSA**



- la sicurezza si basa sulla difficoltà della fattorizzazione di grandi numeri;
- il brevetto è scaduto nel 2000.

Esempi di cifrari asimmetrici: ElGamal

- un sistema alternativo è il sistema **ElGamal** di **Taher ElGamal**, pubblicato del 1985;



- la sicurezza si basa sulla difficoltà nel calcolo del logaritmo discreto su alcuni gruppi algebrici.

Sistemi di Firma digitale

- L'introduzione del concetto di schemi a chiave pubblica ha creato una nuova possibilità: l'uso della **Firma digitale**;
- l'analogo della firma calligrafica;
- garantisce:
 - l'autenticità del documento;
 - l'integrità del testo.



Sistemi di Firma digitale



- Il sistema **RSA** è ambivalente: può essere usato anche come sistema di firma, usando le medesime chiavi;
- esiste anche un sistema di firma digitale che si basa su **ElGamal** ma sono richieste alcune piccole modifiche alle chiavi;
- il governo americano ha adottato nel 1991 come standard una variante rafforzata di ElGamal: il sistema **DSA** (Digital Signature Algorithm).

Simmetrici vs. Asimmetrici



- Simmetrici:
 - molto veloci nelle operazioni di cifratura/decifratura;
 - problemi di distribuzione delle chiavi;
- Asimmetrici:
 - più lenti (richiedono più capacità computazionali);
 - risolve alcuni problemi legati alla distribuzione delle chiavi (ma non tutti!).
- Per cercare di ottenere il meglio da entrambi i sistemi si impiegano **sistemi ibridi**.

Sistemi ibridi di cifratura



- per **cifrare**:
 - viene scelta una chiave di sessione casuale per un sistema di cifratura simmetrica;
 - viene cifrata la chiave di sessione con la chiave pubblica del destinatario;
 - vengono spediti entrambi al destinatario;
- per **decifrare**:
 - il destinatario ottiene la chiave di sessione usando la propria chiave segreta;
 - usa la chiave di sessione per decifrare il messaggio principale.

Sistemi ibridi di firma



- per **firmare**:
 - utilizzando particolari **funzioni hash** (per esempio SHA1) viene creato un “**sunto**” del messaggio (160 bit);
 - viene firmato il “sunto” usando la chiave segreta;
- per **verificare la firma**:
 - viene ricalcolato il “sunto” del messaggio;
 - la verifica avviene sul “sunto” ed utilizzando la chiave pubblica del firmatario.

In origine fu... PGP

- Nel 1991 **Phil Zimmermann** scrisse **PGP** (*Pretty Good Privacy*);
- strumento semplice per cifrare e firmare file e messaggi di posta elettronica;
- il nome deriva da una drogheria di **Lake Wobegon** chiamata "**Ralph's Pretty Good Grocery**" il cui slogan era "se non lo puoi trovare da Ralph, probabilmente puoi anche farne a meno";
- faceva uso di **RSA** (allora brevettato) e di un suo cifrario chiamato **Bass-o-Matic** subito rimpiazzato con **IDEA** (brevettato); usa **crittografia ibrida**;
- **multi-piattaforma**.



La saga di PGP



- PGP fu presto esportato in tutto il mondo;
- dal 1993 Zimmermann era **indagato** dal governa US per “**esportazione non autorizzata di armi**”;
- allora crittosistemi con chiavi superiori ai 40 bit erano considerate “**armi militari**”!
- per ovviare al problema nacque il progetto **PGPi** che portava fuori dagli USA il codice di PGP stampandolo e passandolo ad OCR!
- dal 1999 le regole di esportazione sono state “ammorbidite” e PGP non è più una “arma non esportabile”;
- ci furono pure problemi con il brevetto su RSA...

La saga continua...



- Visti i problemi con i brevetti si pensò di aggiungere altri crittosistemi: CAST5, ElGamal e DSA;
- **avvicendamenti societari:** ViaCrypt, PGP Inc. e... Network Associates Inc. (NAI);
- nel 1997 fu sottoposto e accettato dall'IETF come standard: **OpenPGP** (RFC 2440 e 3156);
- la NAI dopo il 2000 non pubblicò più i sorgenti (non più richiesto dalle leggi di esportazione):
 - molte **polemiche** e timori di possibili “**backdoor**”;
 - Zimmermann lasciò la NAI.

E nacque GnuPG (Gnu Privacy Guard)

- PGP era diventato troppo importante per tutti;
- la **Free Software Foundation (FSF)** creò **GnuPG** (o **GPG**) impiegando lo standard OpenPGP;
- inizialmente sviluppato dal **Werner Koch**;
- la 1.0.0 fu rilasciata il 7 Settembre 1999;
- codice sorgente disponibile sotto **licenza GPL**;
- sviluppato da una **comunità aperta** e possibilità di esaminare il codice alla ricerca di bug e...;
- dal 2000 è **sponsorizzato dal governo tedesco** (documentazione e porting su Windows);
- **solo cifrari non brevettati**: DSA, ElGamal e recentemente RSA;
- cifrari rimossi: IDEA;
- **“quasi compatibile”** con PGP e **multi-piattaforma**.



Il codice aperto è importante!



- Il fatto che il codice di GnuPG sia aperto è **importante** per uno strumento di questo tipo:
 - **piena fiducia** in ciò che fa il programma;
 - semplifica la ricerca di bug o problemi di sicurezza;
 - permette a tutti di contribuire al progetto.
- Nel 2004 un ricercatore francese ha trovato un **serio bug** in una implementazione del sistema di firma ElGamal in GnuPG:
 - data una firma, in poco tempo si poteva trovare la chiave segreta!
 - per fortuna:
 - il sistema di firma di ElGamal non era mai stato il default;
 - poche chiavi compromesse (che sono state revocate).

Impariamo ad usare GnuPG



- GnuPG è un tool da **linea di comando**, non possiede una interfaccia grafica propria;
- esistono svariate **interfacce grafiche e plugin** per interfacciarlo con i programmi di posta elettronica;
- è importante saperlo usare da linea di comando:
 - approccio **multi-piattaforma**;
 - **<paranoia> maggiore sicurezza </paranoia>**
- noi vedremo tutti i comandi:
 - da riga di comando;
 - dall'interfaccia grafica Enigmail.

Interfacce grafiche (o “*frontend*”)



- sito di riferimento per i “*frontend*”:
`http://www.gnupg.org/related_software/frontends.html`
- molti client di posta sono già compatibili:
 - Evolution, Kmail, Sylpheed (linux), ...
- per gli altri esistono dei plugin:
 - Mozilla Thunderbird (multi-piattaforma): **Enigmail**;
 - Outlook, Eudora, The Bat! (Windows);
 - Mail di Apple: GPGMail;
 - Mutt, Pine, ...
- oppure interfacce di gestione:
 - Seahorse, GPA, KGPG (linux);
 - WinPT (Windows).

Procuriamoci GnuPG



- Attraverso il sito di riferimento

`http://www.gnupg.org`

- si possono reperire copie di GnuPG compilate per molte piattaforme:
 - Linux;
 - vari Unix;
 - MacOS X;
 - Microsoft Windows.

Creiamo le nostre chiavi



- per creare una nuova coppia di chiavi usiamo il comando:

```
gpg --gen-key
```

- ci verranno richiesti:

- **tipo di chiavi:**

- DSA e ElGamal (default);
- DSA (solo firma);
- RSA (solo firma);

- **la dimensione delle chiavi:**

- DSA: fissa a 1024 bit;
- ElGamal: 1024 predefinito, 2048 bit massimo consigliato;
- RSA: 1024 predefinito;

- una eventuale **scadenza** della chiave;

- nome, cognome ed email;

- una **passphrase** che proteggerà la nostra chiave segreta.

Creiamo le nostre chiavi (Enigmail)



- Enigmail -> Gestione delle Chiavi OpenPGP
 - Genera -> Nuova coppia di chiavi

Enigmail - Genera chiave OpenPGP

Account / ID utente

Usa la chiave generata per l'identità selezionata

Nessuna passphrase

Passphrase Passphrase (ripeti)

Commento

La chiave scadrà tra La chiave non scadrà

Dimensione della chiave

Keygen Console

NOTA: La creazione della chiave può richiedere anche parecchi minuti. Non uscire dall'applicazione prima del termine dell'operazione. Navigare su internet o svolgere attività che sfruttino intensamente il disco durante la creazione della chiave potrà facilitare la generazione dei numeri casuali e accelerare il processo stesso. Sarai avvertito quando la creazione della chiave sarà completata.

Struttura delle chiavi

- Una chiave è formata da:
 - una **chiave di firma principale**;
 - altre **sottochiavi** (di firma o cifratura) opzionali;
 - un ID numerico della chiave, presumibilmente unico, chiamato “***fingerprint***” della chiave;
 - uno o più identificativi dell'utente.
- per vedere la *fingerprint* di una chiave si può usare il comando: **gpg --fingerprint <id>**



```
pub 1024D/DC4FA677 2005-01-26 Mario Rossi <mariorossi@email.it>
```

```
Impronta digitale = 76CD 9F16 2883 B5AB 4575 D4B2 1D8F B8E3 DC4F A677
```

```
sub 2048g/878A81E2 2005-01-26
```

Struttura delle chiavi (Enigmail)



Gestione chiavi OpenPGP

File Modifica Visualizza Keyserver Genera

Filtra in base a ID utente o ID chiave contenenti:

Account / ID utente	ID chiave	Tipo	Fiduci...	Fiduci...	Scad...	
Mario Rossi <mariorossi@email.it>	DC4FA677	pub/sec	definitiva	definitiva		

Proprietà della chiave

ID utente primario: Mario Rossi <mariorossi@email.it>

ID chiave: 0xDC4FA677

Tipo: coppia di chiavi

Fiducia calcolata: definitiva

Fiducia personale: definitiva

Fingerprint: 76CD 9F16 2883 B5AB 4575 D4B2 1D8F B8E3 DC4F A677

Sottochiave	ID	Algoritmi	Dimensione	Creata	Scadenza
chiave pubblica	0xDC4FA677	DSA	1024	2005-01-26	mai
sottochiave	0x878A81E2	ELG	2048	2005-01-26	mai

I portachiavi

- Ogni utente ha due **portachiavi** (o *keyring*):
 - uno **pubblico**: le chiavi pubbliche dei nostri corrispondenti;
 - uno **privato**: le nostre chiavi segrete;
 - cifrato usando la *passphrase*;
 - **anello più debole** di tutto il sistema;
 - **proteggiamo** il nostro *keyring* segreto!
 - per vedere le chiavi nei nostri *keyring*:


```
gpg --list-keys [id]
```

```
gpg --list-secret-keys [id]
```
 - con Enigmail basta usare il comando:
 - **Enigmail -> Gestione chiavi OpenPGP**



Importare ed esportare le chiavi



- Se vogliamo dare a qualcuno la nostra chiave pubblica la dobbiamo prima **esportare**:

```
gpg --armor --export <id> > chiave.asc
```

- l'opzione “**--armor**” permette di fare il taglia&incolla della chiave.
- Per **importare** una chiave nel nostro portachiavi:

```
gpg --import <nome_del_file>
```
- Ci sono vari modi per diffondere la propria chiave:
 - sito web;
 - allegare alle email;
 - attraverso i **keyserver** (vedremo dopo).

Esempio di chiave esportata in ascii



```
-----BEGIN PGP PUBLIC KEY BLOCK-----  
Version: GnuPG v1.2.4 (GNU/Linux)
```

```
mQGIBEH3gwkRBAD3zCpvQ5+i6AwHoacdAM0X72zn1IqCgHKX3x6DEhzXVNVeEKBu  
tLuHJrvDkrnh5Kja4b67BqqqlOTQBLn+jY0aaT+mFL9wpkJJroDDmE9311uuL640  
ZlbMy+MLqm38iq05n5hTg588EejaA8Gh/9DJm1kCKuVpPe2KN5Y+sR7t7EwCgovCc  
hVIicUA2YglmmKJ/FcmaC1kEAKXZZ2ZwgnlXp4LVD4hRpGgdUCffh9CVGwaIlqq/  
DeGm2rRiUwJrSuNQYFAyqLfOqpd+d2PY62P7jfsUez/dcyaALasMseIq7s6727Zo  
RV1sVO/WJkYZyZoeONVlnI6W/iyw8uTGt3LlQ/8rs18mskmWQxMiG11TQsF+Gh0B  
vqEuBACHff0aDtOUoBSgUyHPuKp3FoHFFhp2/VluGazktNeW5/di4v3k79PO5ML  
XKWZPO0ccMdWN3Eaf24ER3+PtG8ljRsOv+N9vfmNk3Cc4+UQ+zIJ1iUNe9mVgTa/  
GAnclsLNR9IDpgCtQyPJ06LUtTDQWQfSRqGlnfesVzW0tgjynT7YAAAAhTWFyaW8g  
Um9zc2kgPG1hcmlvcm9zc2lAZWlhaWwuaXQ+iFsEEExECABsFAkH3gwkGCwkIBwMC  
AxUCAwMWAgeCHgECF4AACgkQHfY+449xPpnc0BwCbBtJ/8/IRzi7IX9OSB1m7qk3o  
298AnRS67O2xt3YI6BFximEQkXcvbyGluQINBEH3gywQCADmbxnLIOF/rkgIr+sW  
R4VcRY30JP/flwyHe6d4RzrRe6l653AOoALcASVFWmf+C/zMcWw+YZaNmTf3Z739  
lgEyCN9aScTGEM3UblphLdeazrtXRc5RyRWqYzkim1Lpr9aqGgR6EnTElhUnH6uV  
74fhW1xYuiwDgyGsPFFVcw334wC1MVxr/zj0IzocvFu7CNkskcYSncC8MIWmY5nG  
8T9RkviOCTJarSiS90j3FhnsfYsZdKexSLzaLQxKRSOSbk09SOdu14hJ0Y0GtGff  
fnugfia4MJlnJRD5KDNso33GHcvg4oSCo6mUZ14uJch04StamLzX17CBeQvT11zn  
y8qbAAMFB/w0jVch3gTXcAFTPh7SNIUuXJucrX1MaGwYmBD0aRlt46nNelzblzAa  
ZCNELayMi6QALdsWUmRUFjE21RAcfA90EqdnINcu/7rUr7wUrjtp8ds7XSKfduJ5  
BAUHLNQgzpboxlbeN/eEfOwkZa49RNjw7yjkreX9HiqSyPBM9PWahAPtbk2JMzNm9  
rOQgtPDBwE1NMs7FoHqPyuo8bP8/cSte+jl/+mRfMSzeIKTt3viVI9huqIA4OPAY  
fOnvrSwjRYtlZgFzL50OvrtKkobmnqjiEr41QoPEOuoL0VjvFtbyM/tR0cilsrGn  
7k9fABQ+GyuDSte0efJ9UfOb4C1VhwCjiEYEGBECAAYFAkH3gywACgkQHfY+449xP  
pndpvQCfXy5us9Ogd6mrA+ZjacwEbasrHSEAnRI6rphzclc2ty4+ujPC6BpkYdqB  
=UcQT
```

```
-----END PGP PUBLIC KEY BLOCK-----
```


Importare ed esportare le chiavi (Enigmail)

- Con Enigmail abbiamo varie opzioni:
 - **File -> Importa chiavi da un file**
 - **File -> Esporta chiavi in un file**
 - **Modifica -> Importa chiavi dagli Appunti**
 - **Modifica -> Copia chiavi pubbliche negli Appunti**

Cifrare un file

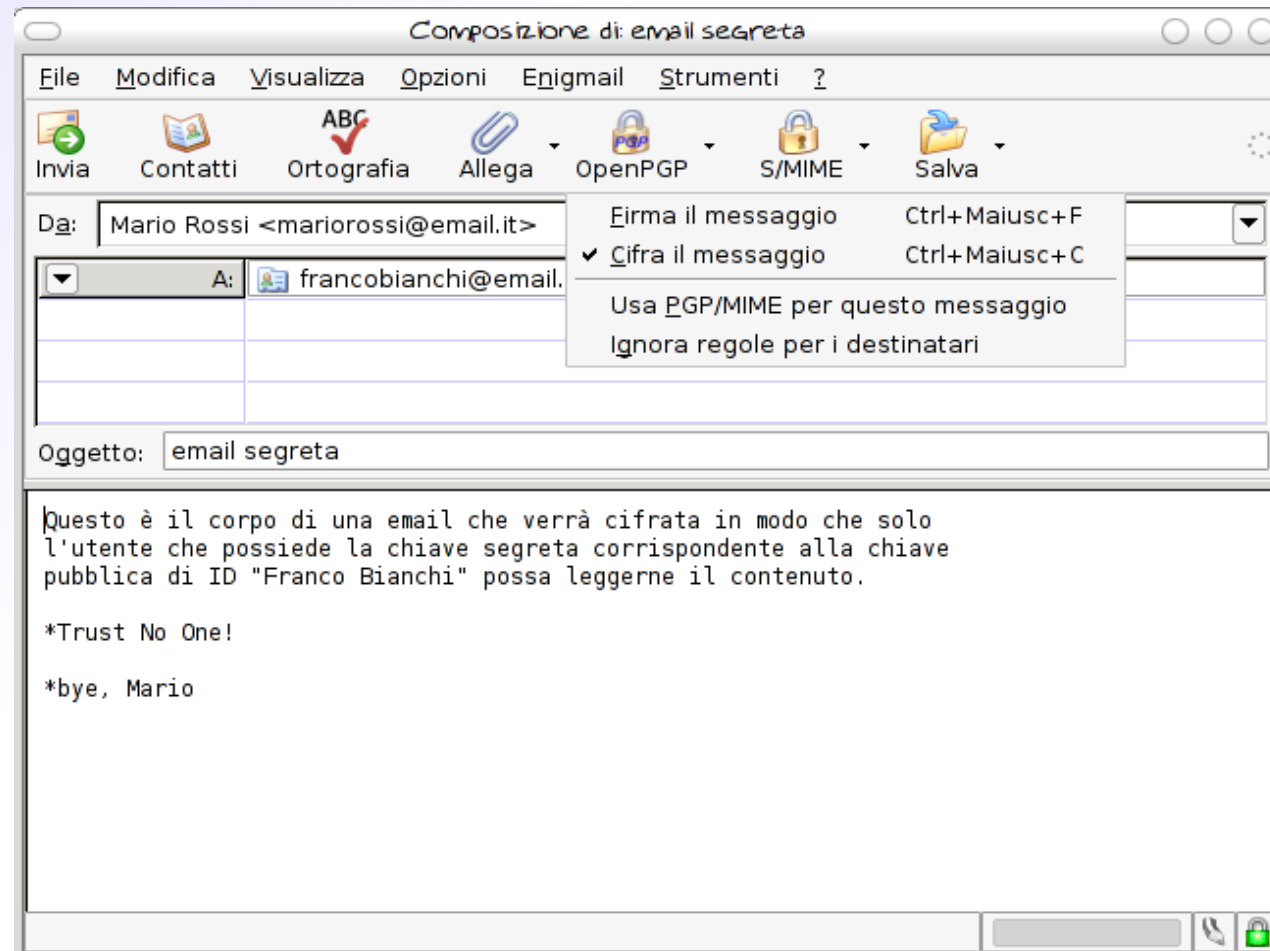
- Per cifrare un file con una chiave pubblica contenuta nel nostro portachiavi pubblico:

```
gpg --recipient <id dest> --encrypt <file>
```



- crea un file con estensione ".gpg";
- se vogliamo un formato compatibile con il **taglia&incolla** dobbiamo aggiungere l'opzione "**--armor**";
- si può cifrare per **più destinatari** (ripetendo l'opzione);
- se vogliamo poter leggere noi stessi il file dobbiamo includerci tra i destinatari o usare l'opzione "**encrypt-to <nostro id>**" nel file di configurazione;
- il file in realtà viene anche compresso.

Cifrare una email (Enigmail)



- la chiave pubblica da usare viene **associata automaticamente** se possibile, altrimenti viene chiesto quale usare.

- Alcune note marginali:
 - Mozilla Thunderbird supporta nativamente un altro sistema di cifratura/firma: **S/MIME**
 - sistema antagonista;
 - con strutture di certificazione centralizzate;
 - interfacciano con smartcard e lettori.

Decifrare un file



- Per **decifrare** un file che è stato cifrato utilizzando la nostra chiave pubblica:

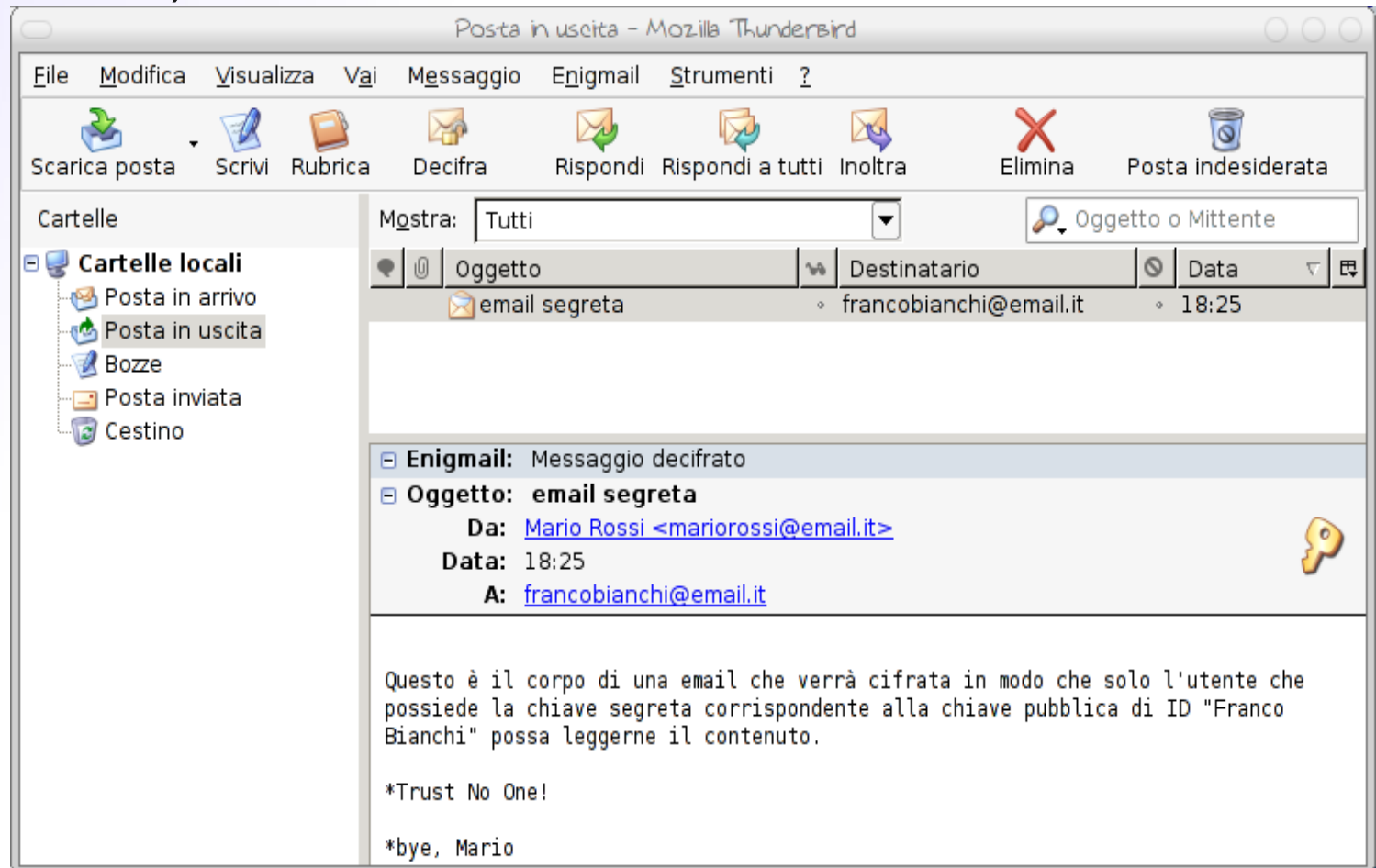
```
gpg <file.gpg>
```

- viene impiegata la propria chiave segreta quindi viene chiesta la *passphrase* per sbloccarla;
- il file decifrato viene salvato nella cartella corrente con il nome privo del suffisso “.gpg”.

Decifrare una email (Enigmail)



- in genere viene decifrata in automatico chiedendo la *passphrase* per sbloccare la chiave segreta (memorizzata per un tempo prestabilito).



Cifrature simmetriche

- Con GnuPG è possibile fare anche cifrature di **tipo simmetrico** (a chiave segreta):
 - utile tra persone di un piccolo gruppo (in famiglia) o per “cifrature veloci”;
 - per **cifrare** un file:


```
gpg --symmetric <file>
```

 - viene richiesta una *passphrase* per generare una chiave di sessione;
 - si può scegliere l'algoritmo simmetrico da usare con l'opzione “`--cypher-algo`” (default CAST5) e visualizzare quelli disponibili con “`gpg --version`”;
 - per **decifrare** si usa la stessa sintassi precedente:

```
gpg <file.gpg>
```

Firmare un file

- Per **firmare** un file con la propria chiave segreta:

```
gpg --sign <file>
```



- crea un file binario con estensione “.gpg” che contiene il messaggio e la firma;
- chiede la *passphrase* per sbloccare la chiave segreta;
- si può usare l'opzione “**--armor**”;
- se si possiede più di una chiave segreta si può specificare il “firmatario” con l'opzione “**--local-user**”;
- per creare un file unico con messaggio leggibile:


```
gpg --clearsign <file>
```

 - crea un file “.asc” con messaggio (leggibile) e firma;
 - ideale per fare il taglia&incolla sul client di posta;
- con “**--detach-sig**” la firma va in file separato.

Esempio di firma in chiaro



```
-----BEGIN PGP SIGNED MESSAGE-----
```

```
Hash: SHA1
```

Questo è il testo dell'email che vogliamo firmare in modalità "clear signature".

Il testo dell'email continua ad essere leggibile e la firma viene messa alla fine.

Anche l'header è importante. Se si modifica qualunque cosa tra l'header ed il trailer la firma viene invalidata (la verifica fallisce).

Saluti

Mario

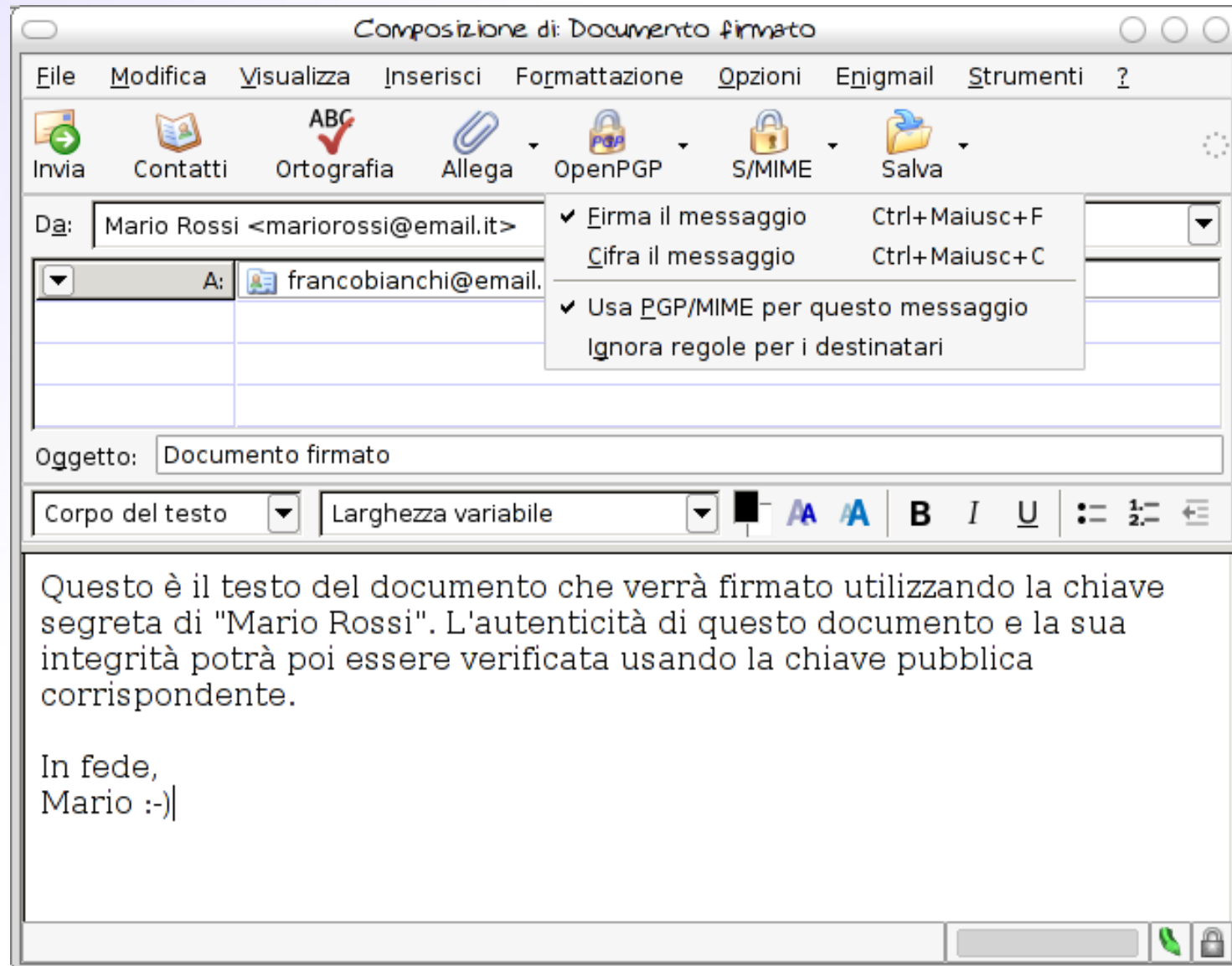
```
-----BEGIN PGP SIGNATURE-----
```

```
Version: GnuPG v1.2.4 (GNU/Linux)
```

```
iD8DBQFB+14BHY+449xPpncRAippAKCMOr45i2+Oa74/XGS85P3p7fgOhQCgmBW6  
7jwYJlUZNL/+nFhLg8SoNb0=  
=Jkrq
```

```
-----END PGP SIGNATURE-----
```

Firmare una email (Enigmail)



Verificare una firma su un file



- Per **verificare** (e recuperare) un file firmato:

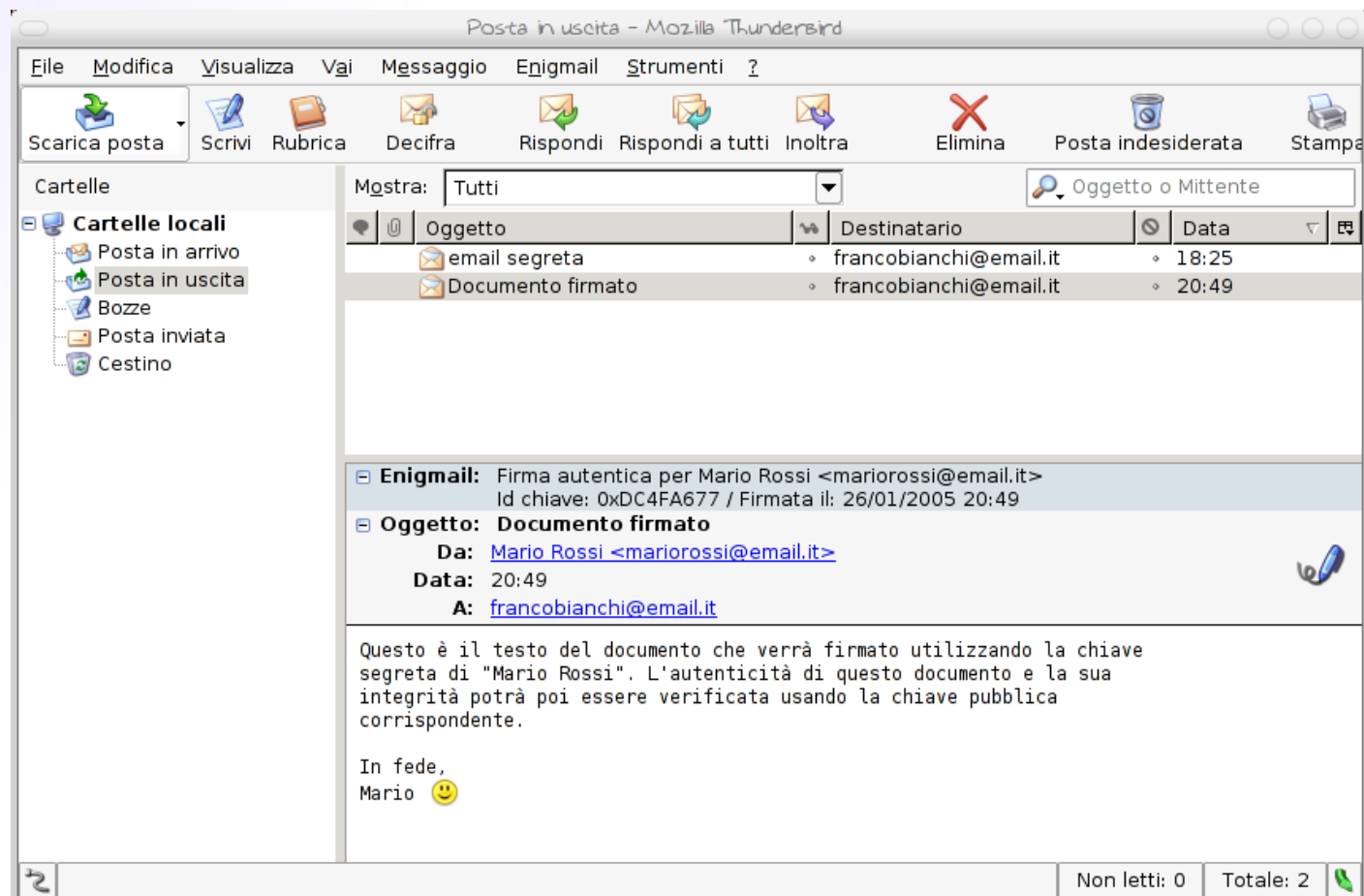
```
gpg <file firmato o file di firma>
```

- il file firmato viene estratto e salvato nella cartella corrente;
- viene segnalata la correttezza della firma;
 - id firmatario;
 - data della firma;
 - e altre informazioni...

Verifica delle firme sulle email (Enigmail)



- In genere le email firmate vengono **automaticamente verificate** (se la chiave pubblica corrispondente è disponibile).



Cifrare e firmare

- In genere se si cifra qualcosa è anche buona norma firmarla:



- da linea di comando basta specificare sia l'opzione “**--encrypt**” che quella di firma;
- da Enigmail basta spuntare entrambe le opzioni nel menu OpenPGP;

- quando ricevuto, GnuPG provvederà automaticamente a decifrare il contenuto e a verificare la firma apposta.



Usare i *keyserver*

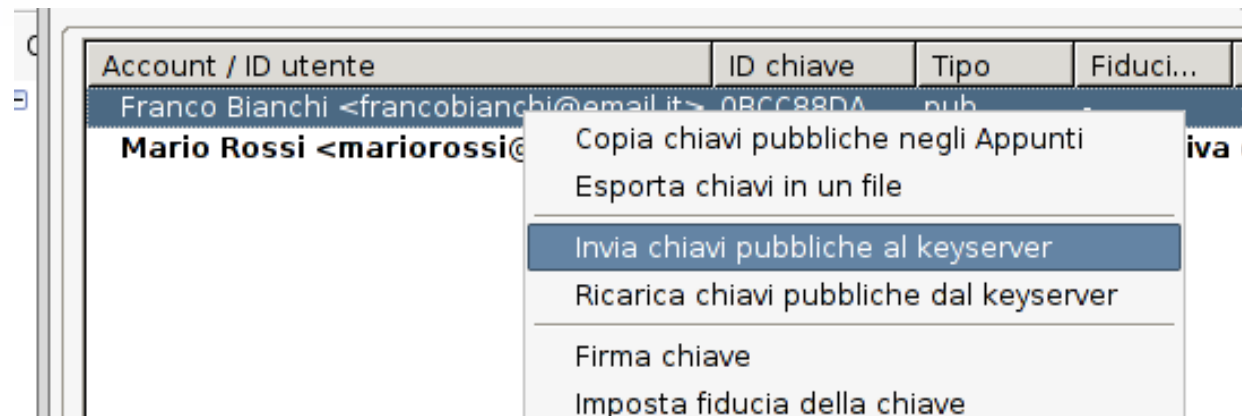
- Una serie di server distribuiti nel mondo:
 - facilitano lo **scambio delle chiavi**;
 - aumentano la **reperibilità** delle stesse;
- attraverso loro si può:
 - **inviare** la propria chiave pubblica;
 - **ricercare** le chiavi pubbliche altrui;
 - i server si **sincronizzano** tra loro;
- sceglietene uno:
 - attraverso l'opzione "**--keyserver**";
 - attraverso le preferenze di Enigmail;
- **ATTENZIONE**: le chiavi pubblicate non si possono cancellare, ma solo revocare! Pubblicate solo quando sicuri.



Inviare una chiave al server



- Per inviare una o più chiavi ai *keyserver*:
`gpg --send-keys <id>`
- con Enigmail basta usare il comando dal menù contestuale:
 - **Invia chiavi pubbliche al keyserver**



Cercare una chiave sui *keyserver*

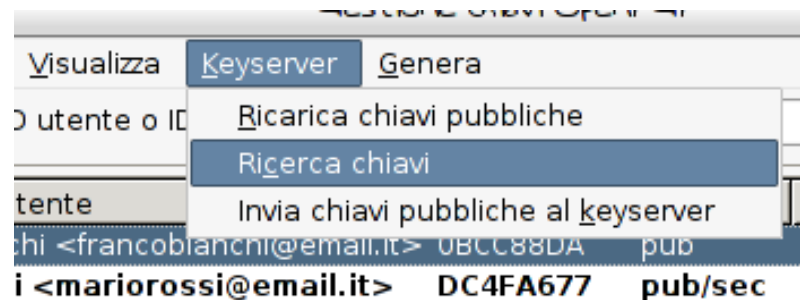


- Per **cercare** una chiave sui *keyserver*:

```
gpg --search-keys <id o email>
```

- se ci sono più occorrenze, queste vengono visualizzate (con relativo identificativo) e viene chiesto di scegliere quale scaricare;

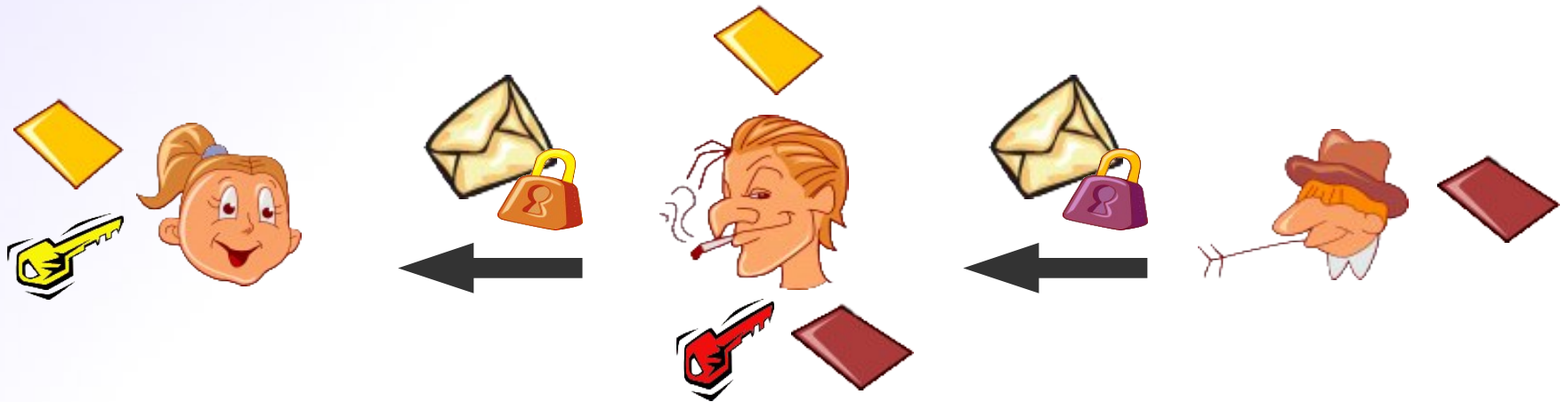
- con Enigmail:



- possono essere **scaricate automaticamente**, ma state attenti...

Attacco “*man in the middle*”

- Purtroppo i crittosistemi a chiave pubblica sono vulnerabili ad un **attacco** sulla rete:



- l'avversario adesso conosce il contenuto dei messaggi in transito senza che sia scoperto;
- esiste un attacco simile per le firme digitali.

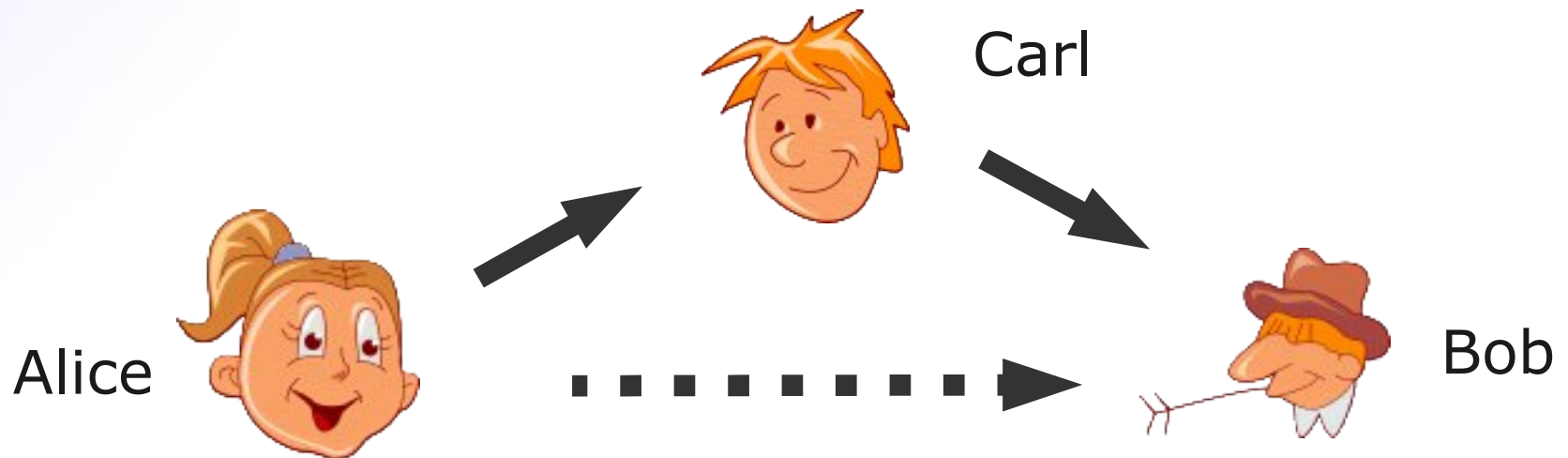
Soluzione: *Web of Trust*



- Dove sta il problema?
 - la chiave pubblica usata per cifrare non corrisponde effettivamente all'identità di chi l'ha creata (e che possiede la chiave segreta);
- in S/MIME la bontà delle chiavi pubbliche sono garantite da **authority certificanti centralizzate**;
- in OpenPGP il compito di certificare le chiavi pubbliche viene **delegato agli utenti stessi**:
 - un utente può **firmare le chiavi** pubbliche degli altri;
 - in un certo senso, l'utente fa da **“garante”**.

Web of Trust: il principio

- Alice si fida di Carl e ha una “copia sicura” della sua chiave pubblica;
- Carl ha firmato la chiave pubblica di Bob e Alice verifica la correttezza di tale firma;
- **Conseguenza:** Alice si può fidare che la chiave di Bob sia autentica (pur non avendo mai avuto a che fare con Bob).



GnuPG ci avverte



- Se non ci sono indizi che la chiave pubblica che stiamo usando sia sicura, GnuPG avvisa in fase di verifica delle firme (anche se corrette):

gpg: Firma fatta mer 26 gen 2005 23:37:07 CET usando DSA con ID 0BCC88DA

gpg: Firma valida da "Franco Bianchi <francobianchi@email.it>"

gpg: ATTENZIONE: questa chiave non è certificata con una firma fidata!

gpg: Non ci sono indicazioni che la firma appartenga al proprietario.

Impronta digitale della chiave primaria:

```
[-] Enigmail: SENZA FIDUCIA Firma autentica per Franco Bianchi <francobianchi@email.it>  
          Id chiave: 0x0BCC88DA / Firmata il: 26/01/2005 23:40  
[-] Oggetto: Email firmata  
          Da: Franco Bianchi <francobianchi@email.it>  
          Data: 23:42  
          A: mariorossi@email.it
```



Firmare una chiave pubblica

- **Importante:** prima di firmare una chiave altrui bisogna **verificare** che essa corrisponda all'identità dichiarata:
 - di **persona** o per **telefono** facendosi dettare la **fingerprint**;
 - ai **key signing party**.



- Per firmare la chiave una volta verificata:

```
gpg --sign-key <id>
```

- si assegna un **grado di fiducia**:
 - non dichiarato;
 - non ho controllato per nulla l'identità;
 - l'ho controllata superficialmente;
 - l'ho controllata molto attentamente;
- viene chiesta la *passphrase* per sbloccare la nostra chiave segreta per creare la firma sulla chiave.



Firmare una chiave pubblica (Enigmail)



- Dal menù contestuale: **Firma chiave**

A screenshot of a dialog box titled "Enigmail - Firma chiave". The dialog contains the following information:

Chiave da firmare: Franco Bianchi <francobianchi@email.it> - 0x0BCC88DA
Fingerprint: C5F3 40CE C436 4D27 77A8 55E9 393F 212D 0BCC 88DA

Firma con la chiave: **Mario Rossi <mariorossi@email.it> - 0xDC4FA677**

Quanto accuratamente hai verificato che la chiave che stai per firmare appartenga effettivamente alla/e persona/e scritta/e qui sopra?

- Non rispondo
- Non ho controllato affatto
- Ho controllato superficialmente
- Ho controllato con grande accuratezza

Firma localmente (non può essere esportata)

Buttons: OK, Annulla

- Una volta firmata bisogna mandare la chiave firmata al *keyserver*, così ognuno potrà scaricare anche la nostra firma al prossimo scaricamento o aggiornamento della stessa.

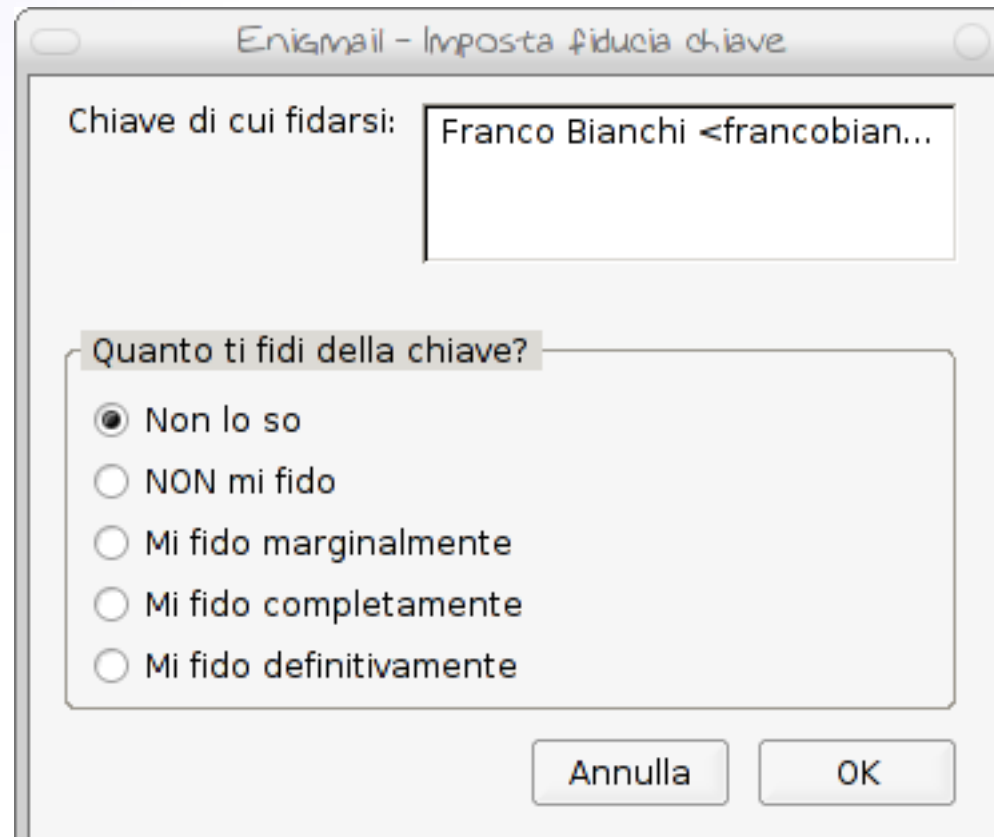
“Il tuo amico è pignolo?”



- Il **Web of Trust** prevede la possibilità per ogni utente di assegnare un “**grado di fiducia**” che nutre nei confronti delle persone che conosce; indica la capacità di verificare le chiavi altrui:
 - sconosciuto (valore iniziale);
 - nessuna;
 - marginale;
 - totale;
 - definitiva (“come se l'avessi firmata io!”).
- Per assegnare il “grado di fiducia”:
`gpg --edit-key <id>` e poi il comando “`trust`”
- sono **valori confidenziali** (evitate cattive figure).

Assegnare il grado di fiducia (Enigmail)

- Dal menu contestuale si usa il comando:
 - **Imposta fiducia della chiave**



Come funziona il *Web of Trust*



- Una chiave è ritenuta **valida** se entrambi le seguenti condizioni sono valide:
 - è firmata da abbastanza firme valide, ovvero:
 - l'hai firmata personalmente,
 - l'ha firmata un utente con fiducia piena, o
 - l'hanno firmata almeno 3 utenti con fiducia marginale;
e
 - il percorso delle firme sulle chiavi che va da questa chiave alla tua propria chiave è lungo al più 5 passi;
- questo è il modo in cui in genere GnuPG funziona, ma i parametri possono anche essere cambiati.

Certificati di revoca

- Se si **dimentica** la *passphrase* o
- si sospetta che:
 - la chiave segreta è stata **compromessa**;
 - si è **perso** la chiave segreta;
- bisogna **revocare** la propria chiave pubblica!
- Per fare ciò bisogna generare un “**certificato di revoca**”:

```
gpg --output revoke.asc --gen-revoke <mio id>
```

 - bisogna farlo **subito dopo** la generazione;
 - stampare e masterizzare il certificato;
 - **<paranoia>** conservarlo al sicuro! **</paranoia>**



Altre cose che si possono fare



- Aggiungere altre **sotto-chiavi** di firma o per cifratura;
- aggiungere una **foto** alla propria chiave pubblica;
- cambiare la/le **date di scadenza**;
- aggiungere **id** (se si usano più email ufficiali);
- **revocare** singole parti (chiavi o id);
- **disabilitare** temporaneamente alcune chiavi nel portachiavi.