Guida PGP
per gli scambi peer-to-peer

Bob Simpson
@bobsimps
14EF F4D1 E421 A50E

last update: 13/02/25

1. Introduzione

Negli scambi p2p il limite più grande è la necessaria fiducia che gli utenti devono avere reciprocamente.
Quando uno scambio avviene tra utenti sconosciuti che non conoscono la reciproca affidabilità, il problema si può mitigare utilizzando un escrow, cioè un intermediario di cui entrambi si fidano (e che non è sempre facile da trovare). Molti servizi, come Bisq o HodlHodl, facilitano questo compito offrendosi come intermediari/arbitri degli scambi che avvengono sulle loro piattaforme.
Se invece consideriamo un gruppo di utenti che si ritengono tra loro affidabili, è evidente che la figura dell'escrow risulti superflua.
Ma come si diventa affidabili? L'affidabilità di un utente generalmente è determinata dalle sue transazioni passate, più un utente ha completato positivamente numerosi scambi e più egli è affidabile.

Col tempo sono nati spontaneamente gruppi di scambi p2p che utilizzano approcci diversi per la gestione dell'affidabilità dei propri membri.
Alcuni gruppi hanno preferito limitare gli accessi dei nuovi utenti ammettendo solo quelli presentati/raccomandati preventivamente da un membro fidato del gruppo, che ne fa da garante fin quando la reputazione del nuovo membro non si assesta.
Altri gruppi hanno preferito svilupparsi mediante il classico metodo dei feedback, cioè brevi valutazioni che gli utenti si scambiano reciprocamente alla fine di ogni scambio.
A prescindere dal tipo di gruppo, dopo che un utente col tempo ha eseguito positivamente numerosi scambi, gli si riconosce un alto grado di affidabilità che viene associata generalmente al suo username.

Considerando le piattaforme di messaggistica (e.g. Telegram), l'username è l'unico modo per identificare univocamente un utente e ciò rappresenta un punto di vulnerabilità. Ad esempio un utente potrebbe essere impossibilitato ad accedere al proprio username, in maniera temporanea o definitiva (ban di Telegram, sim scadute, cellulare perso o rotto, ecc...), di conseguenza non potrebbe più godere dell'affidabilità che col tempo ha guadagnato. Oppure un utente malevolo potrebbe appropriarsi di un username affidabile o crearne uno simile, inducendo gli altri membri in tranelli al fine di truffarli.

Una possibile soluzione è quella di associare l'affidabilità di un utente, non solo al suo username, ma anche alla sua chiave pubblica PGP.
Tramite la crittografia PGP, l'utente che non riesce ad accedere al suo vecchio username e che quindi accede con uno nuovo, può dimostrare inconfutabilmente di essere la stessa persona. In questo modo l'affidabilità dell'utente può migrare dal vecchio al nuovo username tramite il consenso di tutti.
Oppure l'utente malevolo, che si spaccia per un utente noto e affidabile, non potrà mai firmare correttamente un messaggio su richiesta.

Questa soluzione non è mia, non è nuova e non è sconosciuta ai più edotti, ma ha sempre provocato attrito tra gli utenti meno avvezzi alla tecnologia. Scopo di questa guida è di introdurli, nella maniera più semplice che mi è possibile, all'utilizzo di questo potente strumento.

2. Concetti di base

La crittografia PGP permette a qualsiasi utente di generare autonomamente una coppia di chiavi: la *chiave privata* e la relativa *chiave pubblica*.

Non è necessario conoscere i dettagli, ma è importante comprendere che dalla chiave privata è possibile ricavare la chiave pubblica, ma non il viceversa. Da questa particolare proprietà deduciamo, o almeno intuiamo, che tra le due chiavi, la privata è quella più preziosa e che, quindi, merita di essere custodita gelosamente, mentre la chiave pubblica, invece, è quella che può essere condivisa tranquillamente con chiunque in quanto, a partire da questa chiave, non è possibile ricavare quella privata. Come sono fatte queste chiavi? Entrambe sono formate da un testo incomprensibile molto lungo. Generalmente la chiave privata è più lunga della chiave pubblica, ma in entrambi i casi sono stringhe molto lunghe racchiuse tra due tag, uno che identifica l'inizio della chiave (BEGIN) e uno che identifica la fine (END). Per maggiore chiarezza, mostro di seguito come è fatta una chiave privata (a sinistra) e una chiave pubblica (a destra). A causa delle loro eccessive lunghezze, le ho troncate, ma in realtà sappiate che sono molto più lunghe.

```
-----BEGIN PGP PRIVATE KEY BLOCK-----
xhVWOpEcUsDZCHGCDZ4yeDhyeUQFH4r1Gkd+G
i0U1BgBm7zKMb1nQwqykqmb2spMaEppFH+Hr9
csTPjqlrBd+3XmJBtVLVAbJ73BPkn2o4gadxL
r5JCDplurYEOWxVwz8urX6wWI7o9qxuvmvrW1
[……]
CboKgYe52p07eFnONcT5tMGrzMzWOqPbqZKZX
robMHMrJdkBk5wRWIftRvsCr3ks3cQYB6Br76
s11dgsGGUiBpG5yyxjXoz4D6vYk2pFhV2mGzy
DCITyhUZ1u+yjtBHkCRzyaJ
-----END PGP PRIVATE KEY BLOCK-----
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
cjfd3qmsIHkCGmF0BRTRjEt+RNZHJlCwntwGf
Ad53Dw4fMNnjCxV/2lE91qV4w2yDbe/
6j2Nc58E2eGFQsxEVk+Ptkxvbo4x+zCfMHJw
[……]
mfnQeWjjdVOoAHSYy710aM87+R0lOiS3eCZH+
QVwV5fg1KyI4oNOj4JraC9esLNVcPJMS0eTJn
wdlligRMiLNpg===iMp/
-----END PGP PUBLIC KEY BLOCK----
```

Attenzione, anche i tag fanno parte delle chiavi, quindi quando condividerete la chiave pubblica con un utente, non dovrete passargli solo il testo incomprensibile, ma anche i tag di BEGIN e END, inclusi i trattini.

```
cjfd3qmsIHkCGmF0BRTRjEt+RNZHJlCwntwGf
Ad53Dw4fMNnjCxV/2lE91qV4w2yDbe/
6j2Nc58E2eGFQsxEVk+Ptkxvbo4x+zCfMHJw
[……]
mfnQeWjjdVOoAHSYy710aM87+R0lOiS3eCZH+
QVwV5fg1KyI4oNOj4JraC9esLNVcPJMS0eTJn
wdlligRMiLNpg===iMp/
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
cjfd3qmsIHkCGmF0BRTRjEt+RNZHJlCwntwGf
Ad53Dw4fMNnjCxV/2lE91qV4w2yDbe/
6j2Nc58E2eGFQsxEVk+Ptkxvbo4x+zCfMHJw
[……]
mfnQeWjjdVOoAHSYy710aM87+R0lOiS3eCZH+
QVwV5fg1KyI4oNOj4JraC9esLNVcPJMS0eTJn
wdlligRMiLNpg===iMp/
-----END PGP PUBLIC KEY BLOCK----
```

Per comodità si preferisce non identificare un utente mediante la sua chiave pubblica (che essendo molto lunga, è un po' scomoda) ma bensì mediante il *fingerprint* (impronta digitale), ovvero un codice alfanumerico di 40 caratteri che viene ricavato dalla chiave pubblica.

CHIAVE PRIVATA (testo lunghissimo incomprensibile) → **CHIAVE PUBBLICA (testo lungo incomprensibile)** → **FINGERPRINT: D34C EAC3 FBF9 5201 E656 C886 734B 912C D44B BDE7**

Nonostante l'enorme riduzione, anche un identificativo di 40 caratteri potrebbero essere scomodo da utilizzare. In tal caso è possibile identificare l'utente anche mediante i soli ultimi 8 caratteri, chiamato short-id (D44B BDE7), oppure più comunemente mediante gli ultimi 16 caratteri, chiamato long-id (734B 912C D44B BDE7).

3. Cosa possiamo fare con PGP

La crittografia PGP assolve a quattro importanti funzioni:
- criptare un messaggio oppure decriptarlo (crypt/decrypt)
- firmare un messaggio oppure verificarne la firma (sign/verify)

Supponiamo che Alice e Bob abbiano generato le coppie di chiavi col loro software PGP, che poi entrambi abbiano condiviso reciprocamente le chiavi pubbliche e che infine le abbiano importate nei rispettivi software PGP.
Quindi:
- nel software di Alice è presente la sua chiavi privata, la sua chiave pubblica e la chiave pubblica di Bob (che ha appena importato);
- nel software di Bob è presente la sua chiavi privata, la sua chiave pubblica e la chiave pubblica di Alice (che ha appena importato).

Quando Alice vorrà inviare un messaggio **criptato** a Bob, il suo software utilizzerà la chiave pubblica di Bob per criptarlo, dopodiché il messaggio criptato potrà essere inviato a Bob.
Questo messaggio criptato potrà essere **decriptato** solo da Bob in quanto il suo software utilizzerà la sua chiave privata. Il messaggio criptato potrebbe essere raggiunto anche da altri utenti, o addirittura potrebbe essere reso pubblico, ma solo Bob potrà decriptarlo e leggerlo.

Quando Alice vorrà **firmare** un messaggio, il suo software utilizzerà la sua chiave privata per aggiungere la firma in coda al messaggio in chiaro. Quindi il software fornirà il messaggio leggibile da tutti con in coda la firma creata dal software.
Grazie all'aggiunta della firma, chiunque potrà **verificare** che il messaggio sia stato firmato proprio da Alice in quanto i software PGP utilizzeranno la firma e la chiave pubblica di Alice (preventivamente importata) a tal scopo.

Ovviamente è possibile combinare entrambe le funzionalità, cioè Alice potrebbe decidere sia di firmare che di criptare un messaggio. In questo caso il suo software utilizzerà la chiave privata di Alice per firmare il messaggio poi la chiave pubblica di Bob per criptare il messaggio e la firma. In maniera speculare Bob sarà l'unico che potrà decriptare (e quindi leggere) il messaggio, tramite la sua chiave privata, e verificarne l'autenticità tramite la firma e la chiave pubblica di Alice.

---

4. Quali software utilizzare

Benché il pieno potenziale di PGP si esprime tramite un computer utilizzando software come Kleopatra o, meglio ancora, da Terminale, il modo più semplice e immediato per avvicinare un neofita alla crittografia PGP è tramite le app per smartphone.
Su Android l'app gratuita che spesso viene consigliata è "openkeychain". Invece su iOS purtroppo le app gratuite hanno dei limiti o funzionalità mancanti; un'alternativa completa ma a pagamento è "iPGMail".
Sebbene queste siano le app consigliate, potrebbero esisterne di migliori. Sentitevi sempre liberi di cercare alternative valide e, qualora ne trovaste, scrivetemi cosicché possa inserirle nei prossimi aggiornamenti di questa guida. Ad ogni modo le modalità di utilizzo dovrebbero essere simili, indipendentemente dall'app installata.

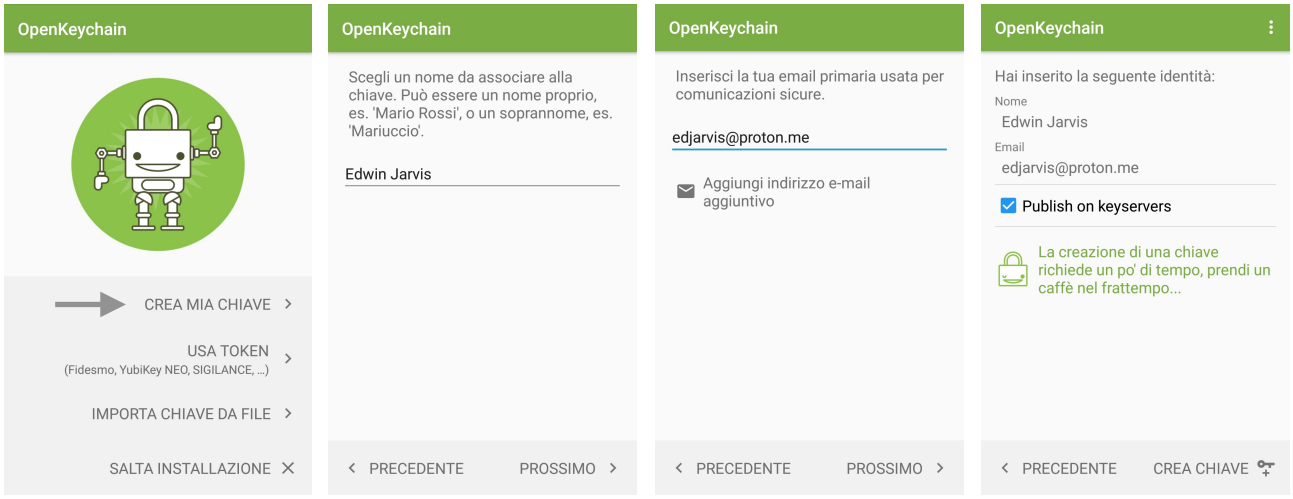
---

5. Generiamo le nostre chiavi PGP

Per generare la vostra coppia di chiavi (cioè quella privata e quella pubblica) l'app vi chiederà tre cose:
- NOME: Potete inserire qualsiasi nome, siete liberi di scegliere se proteggere la privacy inserendo un nickname o se inserire nome e cognome reali.

- EMAIL: Anche qui potete inserire qualsiasi email, anche fasulla o inventata, ma si consiglia di inserirne una a cui avete accesso. Inoltre, per proteggere la privacy potreste anche crearne una dedicata a tale scopo, ad esempio su protonmail o simili.
- PASSWORD: A volte è chiamata anche passphrase proprio per indicare che non dovrebbe essere una singola parola corta, ma dovrebbe invece essere una password robusta, magari lunga con lettere maiuscole, minuscole, numeri e simboli,... volendo anche più parole. Siete liberi di scegliere il compromesso più adatto per voi.

Una volta inseriti questi dati il software genera la vostra coppia di chiavi, dopodiché potrete visualizzare i dettagli, come il fingerprint. Come esempio, vediamo di seguito gli screenshot dell'app openkeychain:



Osservando l'ultima schermata notiamo che l'app permette anche di pubblicare la chiave pubblica sui keyserver dopo averla generata. Approfondiamo quest'aspetto nella sezione successiva.

---

6. Condividiamo e pubblichiamo la nostra chiave pubblica

Una volta creata la coppia di chiavi sarebbe opportuno, innanzitutto, condividere il fingerprint col mondo. Ad esempio potreste inserirlo nella Bio di Telegram, nel footer del vostro sito web, nelle slide dei vostri speech, nei biglietti da visita, ecc... In questo modo le persone avrebbero più fonti dove reperire il vostro fingerprint.
Inoltre è di fondamentale importanza che condividiate la vostra chiave pubblica con i vostri contatti in modo che essi possano utilizzarla per inviarvi un messaggio criptato o per verificare i messaggi che firmerete. Tramite l'app pgp che utilizzate è possibile condividere la chiave pubblica con un amico, che potrà a sua volta importarla nella propria app pgp. Sarà sua cura poi confrontare il fingerprint della chiave che ha appena importato col fingerprint che avete inserito nella Bio di Telegram, in modo da constatare che la chiave importata è quella giusta.

La chiave pubblica, oltre ad essere condivisa tramite lo scambio tra singoli utenti, può essere anche pubblicata su un vostro sito, sulla vostra pagina GitHub, su un canale Telegram, ecc... In questo modo gli utenti saprebbero dove reperirla senza doverla richiedere in privato.
Proprio a tale scopo sono nati i keyserver, cioè dei server dedicati all'archiviazione delle chiavi pubbliche e che permettono a qualsiasi utente sia di pubblicare la propria chiave che di cercare quelle degli altri utenti. Ne esistono diversi, ad esempio:

https://keys.openpgp.org/ *https://keyserver.ubuntu.com/*

Generalmente le app pgp offrono la possibilità di pubblicare le chiavi sui keyserver (come abbiamo visto nell'ultima schermata).
Sui keyserver, una volta caricata la vostra chiave pubblica, chiunque può trovarla cercando il vostro fingerprint o il long-id. Spesso, per preservare la privacy, di default non è permessa la ricerca tramite l'indirizzo email, ma volendo potreste abilitarla verificando la vostra email sul keyserver. Ad ogni modo l'utilizzo di un keyserver, sebbene consigliato, non è un passaggio necessario.

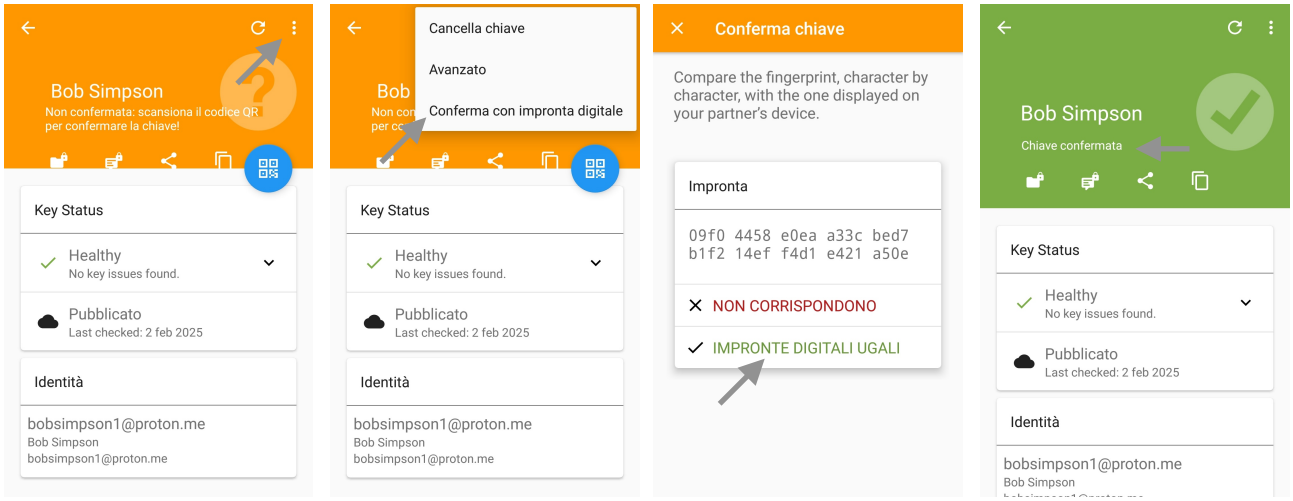
---

7. Importiamo le chiavi pubbliche degli altri

Ora che abbiamo creato la nostra coppia di chiavi e pubblicato/condiviso la nostra chiave pubblica, siamo pronti per firmare un qualsiasi messaggio. Prima però dedichiamo questa sezione all'importazione delle chiavi dei nostri amici. Possedere le loro chiavi pubbliche è necessario per poter verificare i loro eventuali messaggi firmati o, nel caso ne aveste bisogno, per criptare un messaggio che potranno decriptare e leggere solo loro.
Il vostro amico potrebbe inviarvi la sua chiave pubblica tramite un messaggio, una email, oppure potrebbe indicarvi un sito dove l'ha pubblicata, oppure potreste cercarla su un keyserver.
Ad esempio potete trovate la mia su https://keys.openpgp.org/ cercando il mio fingerprint:

09F0 4458 E0EA A33C BED7 B1F2 14EF F4D1 E421 A50E

Volendo è possibile cercare la chiave anche con il long-id, cioè gli ultimi 16 caratteri (14EF F4D1 E421 A50E) oppure mediante l'email. Ma su alcuni keyserver un utente potrebbe avere più chiavi con la stessa email e potreste non sapere quale importare. Il fingerprint (o il long-id) invece è univoco, quindi cercando quello sicuramente non potrete sbagliare. Openkeychain permette la ricerca su openpgp.org direttamente dall'app.
Una volta importata la chiave, è importante verificate che il suo fingerprint corrisponda con quello indicato dal vostro amico. Ad esempio il mio fingerprint, oltre ad essere stato appena indicato, lo trovate anche nella mia Bio di Telegram e su GitHub. Alcune app, come openkeychain, richiedono anche una conferma di questa verifica:



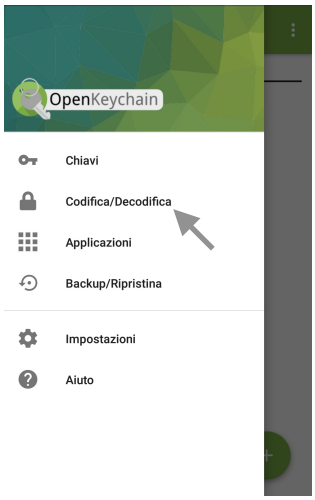
Nella prima immagine, appena sotto al nome, è specificato che la chiave non è confermata in quanto non avete controllato il suo fingerprint.

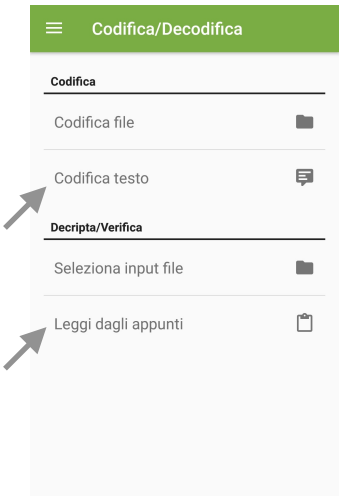
Cliccando i puntini in alto a destra e poi su "Conferma con impronta digitale", vi si aprirà la terza schermata dove verrà mostrato il fingerprint della chiave importata. Confrontatela con quella che ho indicato in questa guida oppure con quella che ho in Bio e, se è la stessa, confermate cliccando su "impronte digitali uguali". Una volta confermato, vi comparirà questa informazione sotto il nome.

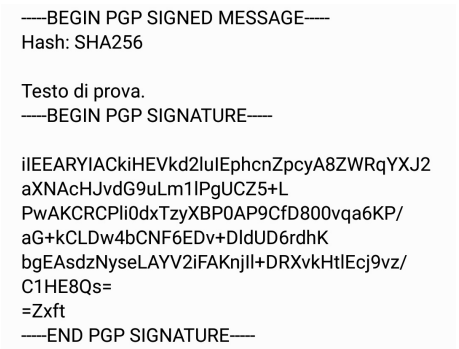
---

7. Firmiamo un messaggio

Supponiamo di voler firmare un messaggio con scritto "Testo di prova."







Dalla prima schermata possiamo notare che sono presenti le funzionalità di *Codifica* e *Decodifica* e sembrano mancare quelle di *Firma* e *Verifica*. In realtà openkeychain raggruppa le quattro funzionalità chiamandole "Codifica/Decodifica". Proseguiamo e passiamo alla seconda schermata.

Cliccando su "Codifica testo" ci si presenterà una schermata dove potremo scrivere il messaggio e scegliere se firmarlo solo, oppure se criptarlo anche (indicando la chiave pubblica del destinatario). In questa guida ci limitiamo solo a firmare il messaggio senza criptarlo, in questo modo tutti possono leggerlo. Il risultato è riportato nella terza immagine.

La funzionalità "Codifica file" permette, in maniera analoga, di firmare e/o criptare un file (ma non è argomento di questa guida).

---

8. Verifichiamo un messaggio

Per verificare un messaggio firmato occorre semplicemente copiarlo, dopodiché utilizzando la funzionalità "Leggi dagli appunti" della sezione Decripta/Verifica, il programma analizzerà il messaggio firmato che avete copiato e ne verificherà la validità tramite le chiavi che possiede.

---

9. Esempio n°1 - Accesso perso all'username

Supponiamo che improvvisamente Telegram mi banni senza motivo e che quindi non abbia più accesso al mio solito username. Non avendo altra scelta, mi registro nuovamente scegliendo un nuovo username e raggiungo il mio gruppo scrivendo:

Ciao ragazzi, sono Bob Simpson, telegram mi ha bannato senza motivo, quindi non riesco ad accedere col mio solito username. Il mio nuovo username è //////

Se però mi mettessi nei panni di un qualsiasi membro del gruppo, mi domanderei: "Come posso essere sicuro che il messaggio sia stato scritto da Bob? E se invece fosse stato scritto da una persona malevola che vuole spacciarsi per lui allo scopo di truffare me e gli altri?"
È evidente che, nonostante le mie buone intenzioni, questo messaggio da solo non basta, richiederebbe un'eccessiva fiducia che, come sappiamo, deve essere sempre ridotta al minimo.
Per fortuna, tramite la crittografia PGP, posso firmare il mio messaggio in modo tale che i membri del gruppo (che posseggono già o possono reperire facilmente la mia chiave pubblica) possono verificarlo tramite la loro app. Quindi apro la mia app e cerco la funzione che mi permette di firmare un messaggio. Dopo aver inserito il messaggio da firmare, l'app mi fornirà il seguente messaggio firmato:

-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA256

Ciao ragazzi, sono Bob Simpson, telegram mi ha bannato senza motivo, quindi non riesco ad accedere col mio solito username. Il mio nuovo username è ////// e per dimostrarvi che sono io firmo questo mex.

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

iQIzBAEBCAAdFiEECfBEWODqozy+17HyFO/00eQhpQ4FAmeejRsACgkQFO/00eQh
pQ6Mog//UVXD4oD3JuAO0uYRBouKTDk0RoKPhcnqPbV/9InX+1SgUm5KVNoV4DmU
2MBxwm++Qvfsv8NMmDjfDjPANThpHYFs2xEGMxTCc2GIFUbnTKTdQPAz6mphwIK+
VqsyqFZjHChfKBp5XC21rMZLqPEsQVXjprbWV1q1RtDS7e9BHaT+OaYRqREEGirF
la6eyIdDBw+1suu7mlye0xTBZHuae+P99aJeEpAgu0GOewp4wo98xrqC+UUrAN4a
C7Auf2AOrmtbIOp6BH8kzrEb2sILAFOurf/qAsoiPol97Ck0/nNMiRrTSS/5Ibhc
Q9qWrMYSH67qBJoAWIIJSud8rbR4VbceIqM+RSYPMJQS3HIbFZzFLt07vr08ErAR
NAUPa8be44dmISBaqFBsPNFc9Ld+ZRMLplZx4Rd/dg/UTD3OPBjxBiApDuSI53PF
IYz06dcfi9T54dCITOewijK8eerTYmaevaXP5mKkIorD/WmMVzDdHr/hK3D2Kv3S
CO++cMN485t6VdsKQUHDM/rMlzS3OI4Htk0u9dONyGIe2oIDpBP7M5fix0w70FYy
G54zN++nYxtNQcKNZKAj1iEDMGaivDFhx/TmLgVsA/JGrBMa2hlQoC9ALgoX/Uid
wfKno7pmhKLa4FbtrMcFAVGmeGOg5zkTKjO9jFC60DXXuZLtO9M=
=gKH9
-----END PGP SIGNATURE-----

A questo punto, se avete importato la mia chiave pubblica, potrete verificare autonomamente questo messaggio. Dai provate. Utilizzate la funzione di verifica della firma della vostra app, copiate ed incollate tutto il messaggio in grassetto, inclusi i tag BEGIN ed END, quindi dal primo trattino fino all'ultimo. L'app vi confermerà che il messaggio è stato firmato correttamente tramite la mia chiave.
La scritta "Non Codificato" si riferisce al fatto che il messaggio incollato non è criptato, ma è bensì leggibile da tutti.

---

10. Esempio n°2 - Lo scammer camuffato

Nei gruppi di scambio p2p, gli utenti sconosciuto spesso sono considerati quasi sempre degli scammer fino a prova contraria.
Se questi utenti hanno buone intenzioni allora comprenderanno la necessaria iniziale mancanza di fiducia da parte degli altri membri e cercheranno di migliorare la loro affidabilità guadagnandosi pian piano la fiducia degli altri.
Se invece questi utente sono scammer, allora non vorranno perdere tempo, cercheranno di truffare i membri inesperti tramite promesse vantaggiose,

ma più frequentemente cercheranno di spacciarsi per membri affidabili che godono di una buona reputazione. Ad esempio su Telegram uno scammer potrebbe creare un account con un username molto simile a quello di un utente affidabile, ad esempio sostituendo una "O" (vocale) con uno "0" (numero), copiando anche l'immagine e la Bio in modo da creare un clone quasi identico del profilo dell'utente affidabile. In questo caso, se non si è attenti, è facile convincersi che l'interlocutore sia l'utente affidabile, mentre invece lo scammer sarà pronto a tendervi un tranello.

Una possibile soluzione che eviterebbe quasi sempre di cascare in queste trappole è sicuramente utilizzare le firme PGP.
Una volta che due utenti hanno stabilito cosa scambiare e in che modalità, prima di procedere allo scambio dovrebbero sempre richiedersi a vicenda di firmare un messaggio (è consigliabile che il messaggio contenga anche la data e l'ora, in questo modo non si corre il rischio che il messaggio firmato venga riutilizzato).
Uno scammer, anche se avesse copiato alla perfezione tutto il profilo di un utente affidabile (incluso il fingerprint nella Bio), non potrà mai firmare un messaggio se non possiede la chiave privata.
La firma reciproca tra gli utenti, poco prima degli scambi, potrebbe essere una buona prassi per ridurre di molto le truffe di questo tipo.

C'è da precisare però che l'uso delle firme non è la soluzione definitiva e che la sicurezza totale di uno scambio non si potrà mai raggiungere. Ad esempio un utente affidabile potrebbe sempre improvvisamente diventare malevolo, oppure uno scammer potrebbe sempre operare inizialmente in maniera onesta per poi truffare gli utenti una volta raggiunto un buon grado di affidabilità. Queste falle sono dovute alla natura umana che, negli scambi p2p tra esseri umani, non è possibile eliminare del tutto.

---

11. Conclusioni

Tramite la verifica delle firme PGP, che consta di un semplice copia e incolla, è possibile avere la certezza matematica/crittografica che un messaggio sia stato firmato dall'utente in possesso della proprio chiave privata. Ciò svincola gli utenti dall'essere identificati esclusivamente tramite l'username di una specifica applicazione, permettendo quindi di potersi sempre identificare ovunque. Con questa guida non si vuole invitare gli utenti ad abbandonare i comodi username che usiamo tutti, è irrealistico, ma bensì si vuole consigliare di associarvi anche un fingerprint (oppure long-id o short-id). Ciò permetterebbe solo di aggiungerebbe i benefici mostrati in questa guida.