GPG: a legjobb szabad titkosító program (2. rész)

Mick onnan folytatja a GnuPG-ről szóló cikkét, ahol abbahagyta – és ha lehet, még jobban erőt vesz rajta az üldözési rögeszme.

A múlt hónapban bemutattam a ***GNU Privacy Guard***-ot, az ***OpenPGP*** szabvány szabad, de kevéssé használt megvalósítását. A ***GnuPG***, ahogyan már biztosan tudod, nagyon hasznos lehet fájlok titkosításához a megfejtéséhez – ez főleg a levelezésnél előnyös –, valamint digitális aláírások létrehozásához és a hitelességük ellenőrzéséhez.
Múlt alkalommal helyszűke miatt sajnos épphogy csak el tudtam magyarázni a nyilvános kulcsú titkosítás alapjait, a bizalomhálót, az egymás kulcsai aláírásának és az ismeretlen kulcsok ellenőrzésének a fontosságát. Ezután már csak néhány telepítéssel kapcsolatos gyakorlati tanácsra és a digitálisan aláírt fájlok hitelességellenőrzési módjának rövid taglalására maradt lehetőség.
Ebben a hónapban azoknak is a kedvére teszek, akiket a téma mélyrehatóan érdekel. Folytassuk onnan, ahol abbahagytuk.

A kulcspár létrehozása

A fájlok titkosításához és a titkosított üzenetek megfejtéséhez, illetve a digitális aláírások készítéséhez saját kulcspár szükséges: egy nyilvános és egy titkos kulcs. Hozzunk létre GnuPG kulcspárt! A `gpg` ezt a feladatot párbeszédes formában oldja meg: a kulcsok sikeres legyártása érdekében a parancssoron az egyszerű `gpg --gen-key` parancsot kell begépelni, aminek hatására a program kérdéseket tesz fel, amelyeket meg kell válaszolnunk. Az 1. listán (23. CD Magazin/GPG könyvtár) láthatunk példát a kulcskészítéshez szükséges párbeszédre (a felhasználó által beírt részt dőlten szedtük). Észreveheted, hogy a folyamat során számos döntést kell meghoznod: meg kell adnod a kulcs típusát, a hosszát, az élettartamát és a levélcímet, amelyet a kulcshoz tervezel társítani.
Általános kulcspár létrehozásához válaszd a ***DSA/ElGamal***-lehetőséget (ez az első). Ez két kulcspárt készít: a DSA-kulcspárt, ez használható az aláírások készítésére és ellenőrzésére, valamint az ElGamal-kulcspárt, amit a `gpg` titkosításra és visszafejtésre használ. Igaz, így rögtön két kulcspárod lesz, de ne aggódj, nem fogja megnehezíteni az életedet. A DSA- és az ElGamal-kulcsok – a két titkos és két nyilvános kulcs egyaránt – egy-egy fájlban találhatók.
Ha csupán az aláíráshoz van szükséged a kulcsokra, választhatod a DSA-lehetőséget (ez a második), ha pedig csak titkosításra szeretnéd használni őket, az ElGamal-lehetőséggel teheted meg. Nem ajánlom, hogy az ElGamal-kulcspárt használd mind titkosításra, mind aláírásra (bár ez a negyedik lehetőség), ugyanis ***Bruce Schneider*** „Applied Cryptography" (Alkalmazott kriptográfia) című könyvében ismertetett egy egyszerű módszert az ilyen módon titkosított üzenetek megfejtésére. Ez az eljárás („chosen plaintext") csak akkor alkalmazható sikerrel, ha a titkosításhoz és az aláíráshoz ugyanazt a kulcsot használjuk, ezért ezt a módszert az elkerülendők iskolapéldájaként említhetjük.
A kulcsméret különös fontossággal bír. A GnuPG által támogatott legkisebb kulcs 768 bites, azonban az 1024 bites kulcs használata ajánlott, ugyanis ez az arany középút a biztonság és a használhatóság között. A hosszabb kulcs biztonságosabb, de a titkosítás tovább tart vele. A rövidebb kulcsot gyorsabban létrehozza a gép, és a mindennapi használatban kevesebbet várakoztatja az embert, viszont nem annyira biztonságos. Megjegyezném, ha DSA-, illetve ElGamal-kulcspárt hozol létre, a DSA-kulcs hossza mindenképpen 1024 bit lesz, a feltett kérdés csak az ElGamal-kulcs hosszára vonatkozik.
Fontos azt is eldönteni, hogy a kulcs mennyi ideig maradjon forgalomban. Ha úgy állítod be őket, hogy visszavonásig érvényesek legyenek, egyfelől megmenekülsz az új kulcsok létrehozásának kényelmetlenségétől, viszont kellemetlen helyzetbe kerülhetsz, ha esetleg elfelejted a jelmondatodat, és előzőleg nem készítettél visszavonási tanúsítványt (erről később bővebben írok) – ilyen esetben ugyanis elég körülményes a nyilvános kulcs kivonása a forgalomból, azaz a kulcskiszolgálókról való törlése.
Másrészről ha a kulcsod lejárati ideje adott, nem kell aggódnod, hogy a régen elfeledett és használaton kívüli nyilvános kulcsod az idők végezetéig tárolódik egy kulcskiszolgálón: ha a levélcímed megváltozik, vagy úgy döntesz, hogy a régi kulcsod már

Páncél (Armor) és más beállítások

A `gpg --export` és `--gen-revoke` kapcsolói valójában parancsok, ezek határozzák meg a `gpg` teendőit. Ha azt is meg akarod mondani a `gpg`-nek, hogy hogyan tegye, amit tesz, a parancsok előtt különböző beállításokat alkalmazhatsz. Ilyen beállítás lehet például a `--armor` (ASCII-páncél alkalmazása) és a `--output` (kimenet a megadott fájlba). A `--output` vár még egy argumentumot (meg kell adni a fájl nevét), a `--armor` kapcsolónak nincs argumentuma.

A páncélozott ASCII (kiterjesztése .ASC) hordozhatóbb adatformátum, mint a `gpg` alapértelmezett bináris formátuma (.GPG kiterjesztés). A bináris adattal ellentétben a páncélozott ASCII-t ki lehet vágni, és be lehet illeszteni mondjuk egy levelezőprogramba. Ha lemezre mented, a páncélozott ASCII-fájl teljesen szokványos szöveges állomány. Ebből kifolyólag a kulcsok terjesztésére, mentésére és továbbítására ASCII-páncélt érdemes használni. Azért említem ezt itt meg, mert a beállításokat más úton is meg lehet adni: az options fájlban, ami a *~/.gnupg* könyvtárban található. Ha a `gpg` alapértelmezett fájlformátumát páncélozott ASCII-ra akarod változtatni, az options fájlhoz add hozzá a következő sort:

```
armor
```

Általában mindent, amit a parancssorban meg lehet adni, az *options* fájlból is be lehet állítani, csak a kezdő `"--"`-t kell elhagyni. A parancssorban megadott beállítások felülbírálják az options fájlban lévőket. További részleteket a gpg(1) súgóoldalán találsz.

2. lista Egy nyilvános kulcs

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.0.6 (GNU/Linux)
Comment: For info see http://www.gnupg.org

mQGiBDug+KURBACtQFiSy057YI5Q7I5EDQjWxn/laQ
KUiXsbpt3ar5bZ7ObjGxpcdWiwNqYqWMcSFwEDEKHd
Cr098CGK0B247I/Y9xg1WDOpDQolbgq4Z94uEWENXm
lpftNxQjbz9Dil3VtoRHh325hFb9DzGBx5JB8mb4kp
up9uc+c11UDXuf3hXwCguIezZMF4q5VCv3xIDDlTk7
W6WvMD/3gc3ob2gix68F7JvYduAGrXjg8ExTx7x8MP
0PnmcHBkLefzBYWrQGIdaqaTOSBGVpPxRJAfpdSieG
HKLCyFcV7h1FyIVYBqUmw81cq0Ap62nYVJxCucRWIv
UObW98q/8dxx8K5HVTJ8ItUMh47C/GkkNtlYKYcKKr
aBb/9oulzHA/kBGNM7F1zlVHbAUEgJ+vJ1rbnXnY4E
luAG/I28rE36aVfwiiGzXOR/KUtBl7ln9f7b6z4EfB
FqQuauY9T8f2kEEb9gRtEKlOVDr1///adSrCVv/C9w
Hk2xAyuAhh1NLFb37tTCrm687twHtcyTHgwffnnFSs
Pp79OrdvOzI/5iLbQdRW1lc2UgS292YWNzIDxlbWVz
ZUBnbm9tZS5odT6IVwQTEQIAFwUCO6D4pQULBwoDBA
MVAwIDFgIBAheAAAoJEGQlJf7FBw/7J6kAoI/D5puP
O9v0QOxPYeAs1+Dxf0jaAJ9BdBXHM6vJsnJrMLORKj
eaxPj3johGBBARAgAGBQI7sczTAAoJECXTh7xXlOhV
ZIMAoJnigXUHM6CVvYlF4avmUPV/3/RsAJ4zf6O1UT
D282uY4q8upywdtUyworkBDQQ7oPi3EAQA3mCnLcPC
Le0zbq8WQuAdGvd9UwYg7/sDZs3CqLQEXHvV8XbFKm
BuAuBKklu2XB8dUFyVVKz12Aa0Hlce93Ty1INsL7Ns
bHwouLK8Z176N4gFQqEpfurf1iQylkilifbrj1QvwX
gLthQd/vxDlVsebpEw9UqdDJgTLgtFeqxRlgsAAwYE
ANuB189hTgJBUb58NNdD9ZOzM8HQXV37gVyvghMbRE
r/V5I7G4Jq72dr3rCei7X/zGrtpJaCB8zJLXqM8GZZ
176IHI2TE8rQxGJ+gSmTbWmlXktazTpiajDochlKqP
1z9GkYq6Wrt+fWatNLrDGNAd+9YJJe51M0Z7NTxJY1
3/z+iEYEGBECAAYFAjug+LcACgkQZCUl/sUHD/v2LQ
CeIFJn6B5/UN4zcHwktPYOqgaVRgIAoKH6Xhimw7vo
nZgBdBJTD97/1nFw=Oy2y
-----END PGP PUBLIC KEY BLOCK-----
```

nem elég biztonságos, esetleg a tudomásodra jut, hogy valaki lemásolta a titkos kulcsodat és valamiért nem tudod visszavonni – semmi gond, a kulcs előbb-utóbb elöregszik. A rövid élettartam legfőbb hátránya, hogy adott időközönként új kulcsokat kell létrehozni, az új nyilvános kulcsot terjeszteni kell, és talán a legnehezebb feladatként rá kell bírnunk a többieket, hogy ezt használják.

Régebben csak határozatlan élettartamú kulcsokat használtam, de mára meggyőződésemmé vált, hogy a meghatározott lejárati idő előnyei felülmúlják a hátrányait. Ezért azt a tanácsolom, hogy a kulcs érvényességét 18 és 24 hónap közötti időtartamra állítsd be. Nekem az egy év túl rövid (tempis fugit!), de nem hiszem, hogy egy másfél évnél hosszabb életű kulcs ellenállhatna a számítástechnika fejlődése következtében elérhetővé váló számítási teljesítménynek és az új kulcstörési módszereknek.

Ezt követően meg kell adnod egy nevet, levélcímet és esetleg megjegyzést is. Felhívnám a figyelmedet, hogy a későbbiek folyamán a kulcshoz további levélcímeket adhatsz. Ezt a `gpg --edit-key` kapcsoló után megadott `adduid`, illetve `addkey` paranccsal teheted meg.

A kulcsok létrehozásához utoljára egy jó jelmondat szükséges. Amikor „jelmondatot” írok, arra is nagy gondot fordítok, hogy szóközöket is tartalmazzon, ugyanis minél hosszabb, annál biztonságosabb. A jelmondatban jó, ha van kis- és nagybetű, számok és írásjelek (például bOTTLE rockeT!). Az utóbbi időben a jelmondatok létrehozásához dobókockát és szólistát használok. Ha kedvet kaptál a kipróbálásához, az eljárás pontos leírását megtalálod a ➲ http://www.diceware.org oldalon. Semmiképp ne válassz rövid, könnyen kitalálható jelmondatot. Nem kell, hogy a jelmondat így nézzen ki: „B1&SSja-sd0c as-d$%@KFSAAs-,ssd w0a-00sdp23m”, de ez sem megfelelő: „Az én béna jelmondatom”. Teljesen rendjén való, ha egy-egy bonyolultabb jelmondatot felírsz egy kis kártyára, és a levéltárcádban magadnál tartod (csak arra vigyázz, hogy használat után mindig tedd el, ne maradjon szem előtt!).

Visszavonási tanúsítvány készítése

Miután létrehoztad a kulcspárodat, érdemes azonnal visszavonási tanúsítványt készítened. Abban az esetben, ha a kulcsodat vissza akarnád vonni, ezt a karakterláncot kell a kulcskiszolgálónak elküldeni.

Természetesen bármikor létrehozhatsz visszavonási tanúsítványt, ezt azonban érdemes rögtön a kulcsok létrehozása után megtenni, mert még a legkörültekintőbb felhasználókkal is előfordulhat, hogy elfelejtik a jelmondatukat. A jelmondatra szükség van a visszavonási tanúsítvány létrehozásakor, a későbbiekben viszont, a felhasználás folyamán már szükségtelen.

Ezért jó, ha azonnal elkészítjük a visszavonási tanúsítványt és eltesszük egy biztonságos helyre (akár ki is lehet nyomtatni és a „metatérben” tárolni, ugyanis a tanúsítványok nem túl hosszúak). Mindössze arról bizonyosodj meg, hogy a fájl jogosultsága megegyezik a titkos kulcsodéval (például csoport és a világ által nem írható vagy olvasható). Nem annyira rettenetes, ha valaki a visszavonási tanúsítványt megszerzi és érvényteleníti vele a kulcsodat, mintha a titkos kulcsodat használná, bár a lejárati ideje előtt forgalomból kivont kulcs kellemetlenségeket okozhat.

A visszavonási tanúsítvány létrehozásához add ki a következő parancsot:

```
gpg --output vissz_tan_fÆjlnØv.asc
➥--gen-revoke kulcsnØv
```

ahol ***vissz_tan_fájlnév.asc*** a fájl neve, amelyikbe a `gpg` elmenti a tanúsítványt (csak arra figyelj, hogy .asc legyen a kiterjesztése) és a kulcsnév a szóban forgó kulcs azonosítója (például 0586AF78) vagy a nevednek egy része („Smooth JoJo” pont elég lenne a példakulcsunkhoz).

Nyilvános kulcsod terjesztése

A GnuPG a fájljait a felhasználói könyvtáradban található ***.gnupg*** könyvtárban tárolja. Minden titkos kulcsot a ***secring.gpg*** fájlban tárol, minden nyilvános kulcsot a ***pubring.gpg***-ben. Alapértelmezetten a secring.gpg csak a tulajdonos által olvasható fájl, ezt hagyd is így: nagyon fontos, hogy ezt a fájlt megvédd. Mindenképpen mentsd hajlékony lemezre vagy CD-re és tárold biztonságos helyen. Ha valaki megszerzi a titkos kulcsod másolatát, kitalálhatják vagy megfejthetik a jelmondatodat és ellophatják a személyazonosságodat (de legalábbis megfejthetik a neked szánt titkos üzeneteket).

A ***pubring.gpg*** és a ***secring.gpg***} egyaránt bináris adatfájl. Ahhoz, hogy a kulcsokhoz további kulcsokat tudjál hozzáadni, módosítani vagy a készletből törölni, a `gpg` parancsot kell

3. lista Kulcs szerkesztése (hitelesítése)

```
jojo@linux:~ > gpg --edit-key dan
gpg (GnuPG) 1.0.4; Copyright (C) 2000 Free
Software Foundation, Inc.
This program comes with ABSOLUTELY NO
WARRANTY.
This is free software, and you are welcome
to redistribute it under certain conditions.
See the file COPYING for details.


pub  1024D/C9F34866  created: 2001-07-27
     expires: 2001-08-10      trust: -/q
sub  2048g/C5569A5B  created: 2001-07-27
     expires: 2001-08-10
(1)  Dan Sparty (Party on!)
<dan@boogiemeister.com>

Command> sign

pub  1024D/C9F34866  created: 2001-07-27
     expires: 2001-08-10      trust: -/q
             Fingerprint: FD084 F92C EC62
             8955 98E2 58FB 178A 2673 D1F3
             6866

     Dan Sparty (Party on!)
<dan@boogiemeister.com>

Are you really sure that you want to sign
this key with your key: "John J. Figplucker
(Smooth JoJo) <jojo@figpluckers-supreme.to>"

Really sign? y

You need a passphrase to unlock the secret
key for user:
       "John J. Figplucker (Smooth JoJo)
       <jojo@figpluckers-supreme.to>"
1024-bit DSA key, ID C1C34866, created 2001-
07-27

Command> save
jojo@linux:~ >
```

használnod a megfelelő kapcsolókkal.
Tegyük fel, hogy nyilvános kulcsodat terjeszteni szeretnéd a barátaid között. Ehhez a kulcsot ki kell emelnünk a nyilvános kulcscsomódból, és el kell helyeznünk egy szövegfájlban (lásd a „ASCII páncél vagy bináris GPG fájlok” széljegyzetet). A nyilvános kulcs képernyőre való kiíratásához csak ennyit kell megadnod:

```
gpg --armor --export
```

A kimenet a *2. listában* találhatóhoz hasonlít. Ezt a kulcsot igény szerint bárhova lehet másolni, illetve beilleszteni.
A fenti példa egy kicsit egyszerű volt: ha nem adsz meg felhasználói azonosítót, a `gpg` alapértelmezett kulcspárod nyilvános részét írja ki. Ha csak egy titkos kulcsod van, akkor az ahhoz tartozó kulcspár az alapértelmezett, így a hozzá tartozó nyilvános kulcsot kapjuk meg.
Ha más nyilvános kulcsot szeretnél terjeszteni, egy felhasználói azonosítót (levélcímet) kell megadni.
Példánkat folytatva: ha Mr. Figplucker kulcscsomójáról akarjuk JoJo nyilvános kulcsát terjeszteni, a következőt kell megadunk:

```
gpg --armor --export jojo
```

Ez azt is mutatja, hogy a `gpg` okosan megpróbál rájönni, hogy melyik kulccsal szeretnél dolgozni. Valójában a `grep` parancshoz hasonlóan működik: megadod a cím egy részletét vagy valami más jellegzetes, a kulcsot azonosító szövegrészt, és a `gpg` az első mintára illeszkedő kulcsot használja fel. A saját kulcscsomóim kezelése során a legegyszerűbbnek az bizonyult, ha a teljes levélcímet megadom, mert több titkos kulcsom is van, amelyeken a felhasználói név azonos.
Például: `gpg --armor --export mick@visi.com`
Ha már itt tartunk, a kulcsot nem kötelező a képernyőre kiíratni, a `--output` kapcsolóval azonnal fájlba is irányítható. JoJo nyilvános kulcsával ez a következőképpen nézne ki:

```
gpg --armor --output jojo_pub.asc --export jojo
```

Ekkor a *jojo_pub.asc* tartalmazza a kulcsot.
Lementetted már az új kulcspárodat? Nyilvános és titkos kulcscsomódat egyaránt terjesztheted, ezt azonban nem ajánlom. Sokkal egyszerűbb, ha a *pubring.gpg* és *secring.gpg* kulcscsomófájlokat egy az egyben biztonságos helyre mented a *~/.gnupg* könyvtárból. Ha valamilyen okból kifolyólag mégis ragaszkodsz a terjesztéshez, a fentiekhez hasonló módon teheted meg, a `--export` helyett a `--export-secret-keys` kapcsoló használatával.

Azt hittem, a 128 bites kulcs már hosszúnak számít

Lehet, hogy a Blowfish, a Triple DES (3DES) és a hasonló kulcsok hosszához vagy szokva, amelyeknél a 128 bites titkosítás már erősnek minősül. Ez a szimmetrikus titkosító algoritmusok esetén igaz, amikor ugyanazzal a kulccsal titkosítunk és fejtjük meg a titkosított adatfolyamot. A nyilvános kulcsú titkosítás világában a kulcsok durván tízszer hosszabbak, mert ez teljesen más matematikai alapokon nyugszik, mint a szimmetrikus eljárások. Olyan, mintha az almát hasonlítanád a körtéhez.

Más kulcsának beolvasása, ellenőrzése és aláírása

Barátod, ***Dan Sparty*** éppen most küldött neked egy levelet, amelyhez a *danskey.asc* fájlban az új nyilvános kulcsát mellékelte. Az új kulcsot nyilvános kulcscsomódra a következőképpen fűzheted fel:

```
gpg --import ./danskey.asc
```

Álljunk csak meg egy pillanatra! Az internetes levelezés köztudottan nem a biztonságos adatátviteli módok közé tartozik. Honnan tudhatod, hogy Dan kulcsát nem cserélték-e ki útközben?
Egyszerű: a kulcs ujjlenyomatát kell ellenőrizned. Minden gpg-kulcsnak van egy ellenőrző karakterlánca, ez az ujjlenyomat.

Ez minden kulcs(pár)ra nézve egyedi, de elég rövid ahhoz, hogy fel lehessen olvasni telefonon vagy fel lehessen írni egy képeslapra. Hívd fel Dant telefonon és kérd meg, hogy olvassa fel a kulcsa ujjlenyomatát; aminek meg kell egyeznie a következő parancs kimenetével (ezt a parancsot az új kulcs beolvasása után kell kiadni):

```
gpg --fingerprint dan
```

Megjegyezném, hogy itt is, miként a `--export` parancsnál, elegendő, ha a kulcs kiválasztásához az őt egyértelműen azonosító névrészletet adod meg. A kimenet ehhez fog hasonlítani:

```
pub  1024D/C9F34866 2001-07-27 Dan Sparty
     (Party Down!) <dan@boogiemeister.org>
      Key fingerprint = D084 F92C EC62 8955
      98E2  58FB 178A 2673 D1F3 6866
sub  1024g/C5569A5B 2001-07-27 [expires:
     2001-08-10]
```

Egy másik megoldás (tegyük fel, még csak délelőtt van, és nem akarod Dant felébreszteni), a nyilvános levelezőlistákon és Usenet-hírekben utánanézni annak, hogy Dan hogyan írta alá a leveleit. Ez kihangsúlyozza az ujjlenyomatok fontos tulajdonságát: minél több helyen jelenik meg a nyilvános kulcsod, illetve az ujjlenyomata, annál nehezebben tudják meghamisítani a személyazonosságodat.
Most, hogy már biztosan tudod, hogy a kulcs valóban Dané és nem hamisítvány, Dan kedvéért megteheted, hogy az aláírásoddal hitelesítheted a kulcsát, azaz aláírhatod a titkos kulcsoddal. Hogy ezt megtedd, a `gpg`-t a `--edit-key` paranccsal kell futtatnod. Ennek eredménye a `--gen-key` esetében már megfigyelt üzemmód lesz. A *3. listán* látható a programmal folytatott párbeszéd, amelynek során a felhasználó alapértelmezett kulcsával ír alá egy nyilvános kulcsot.
Észrevetted a `save` parancsot a végén? Ez menti a kulcsokon véghezvitt változásokat (ebben az esetben az aláírást) és kilép a párbeszédüzemmódból. Ha a kulcsot most a `gpg --list-sigs dan` paranccsal nézed meg, a következőt látod:

```
jojo@linux:~ > gpg --list-sigs dan
pub  1024D/B9E0868B 2001-07-27 Dan Sparty
(Party On!)
     <dan@boogiemeister.org>
sig       B9E0868B 2001-07-27  Dan Sparty
(Party On!)
        <dan@boogiemeister.org>
sig       C1C34866 2001-07-27  John J.
Figplucker
         (Smooth JoJo) <jojo@figpluckers-
supreme.to>
sub  1024g/A0B78448 2001-07-27 [expires: 2001-
08-26]
sig       B9E0868B 2001-07-27  Dan Sparty
(Party On!)
           <dan@boogiemeister.org>
```

Dan saját aláírása mellett (a gpg a kulcs létrehozásakor azt titkos párjával magától aláírja) most már JoJoé is ott díszeleg. Ezek után JoJonak terjesztenie kell Dan nyilvános kulcsának új, aláírt változatát:

```
gpg -- output dan_jojosig.asc --export dan
```

JoJonak ezt a fájlt el kell juttatnia Danhez, például levélben – hiszen nyilvános kulcsról van szó, így a biztonság nem létfontosságú kérdés. Dannek az aláírt kulcsot fel kell fűznie a kulcscsomójára, ahol az új változat helyettesíti a régit:

```
gpg --import ./dan_jojosig.asc
```

Lehet, hogy a „terjesztés” nem tűnik ésszerűnek, hiszen már létező kulcsról van szó, tulajdonképpen Dan frissíti a nyilvános kulcsát, és nem új kulcsot fűz fel. De bízz bennem, ezt kell tennie ahhoz, hogy csatlakozzon azon büszke `gpg`-felhasználókhoz, akik vették a fáradtságot és a kulcsukat aláíratták egy ismerősükkel.
Most, hogy Dannek megvan a csúcsszuper hitelesített kulcsa, készen áll, hogy elküldje egy kulcskiszolgálóra, és mások titkosított üzeneteket küldhessenek neki, és még többen aláírhassák a kulcsát. Ehhez a következő parancsot használja:

```
gpg --keyserver pgp.mit.edu --send-keys
➥dan@boogiemeister.org
```

A `--keyserver` kapcsolóval lehet megadni a PGP/GPG kulcskiszolgáló nevét vagy IP-címét. Másik megoldásként bele lehet írni a

```
keyserver pgp.mit.edu
```

sort a *~/.gnupg/options* fájlba. (Ugyanebben a fájlban érdemes megadni a ***charset*** `iso-8859-2` sort is, így a magyar ékezeteket is helyesen értelmezi a progam – a főszerk.)
Ha ez utóbbi eljárást alkalmazzuk, tudnunk kell, hogy ennek hatására a `gpg` elkezdi magától letölteni az aláírások ellenőrzéséhez szükséges nyilvános kulcsokat a kiszolgálóról. Emlékszel, amikor a múlt hónapban ellenőriztem a programcsomag aláírását? Az első próbálkozásnál még nem volt a kulcscsomómon az aláírást létrehozó kulcs nyilvános párja, ezért hibaüzenetet kaptam. Meg kellett keresnem és le kellett töltenem az ellenőrzéshez szükséges kulcsot, amit a `gpg` a fenti sor alkalmazása esetén saját magától megtesz. Neked kell eldöntened, hogy melyik viselkedés áll a legközelebb az elvárásaidhoz: van, aki szereti használni ezt a lehetőséget, van, akit idegesít. (A parancssorban megadott `--no-auto-key-retrieve` kapcsolóval felülbírálhatod az önműködő letöltést.)

Titkosítás és titkosított üzenetek megfejtése GnuPG-vel

Végre elérkezett a pillanat, amikor JoJo elkezdhet titkosítani mindent, amit csak ér. Tételezzük fel, hogy JoJo titkosított üzenetet szeretne küldeni Dannek. Ennek legegyszerűbb módja, ha a levelet megírja a kedvenc szövegszerkesztőjében és a fájlt menti. JoJo tehát ír egy levelet a `vi` segítségével, és ***dan0729.txt*** néven menti. A fájlt a következő paranccsal titkosítja:

```
gpg --output dan0729.txt.asc --encrypt
➥--recipient dan@boogiemeister.org dan0729.txt
```

JoJo ekkor elküldi a ***dan0729.txt.asc*** fájlt, akár mellékletként, akár a levél törzsében (JoJonak be van állítva az `armor` sor az ***options*** fájljában).
Megjegyezném, ha JoJo nem adja meg a `--armor` kapcsolót a parancssoron és az `armor` sor nincs benne az ***options*** fájlban, a kimeneti fájlnak a ***dan0729.txt.gpg*** nevet illik adni, mert ilyenkor a kódolt fájl a `gpg` bináris alakjában jön létre. Ilyen esetben

csak mellékletként lehet elküldeni. Ne felejtsd el: az ASCII páncéllal felvértezett fájl sokoldalúbb felhasználást tesz lehetővé, ezzel szemben az eredeti gpg-alak kisebb fájlméretet eredményez, ezért olyan esetben használandó, amikor a fájl mérete fontos tényező.
Amikor Dan megkapja a fájlt, mentenie kell és a következő paranccsal meg kell fejtenie a rejtjelezést:

```
gpg --output dan0729.txt
➥--decrypt dan0729.txt.asc
```

A titkosítással ellentétben ebben az esetben nem kell megadni a megfejtéshez használandó kulcsot, a gpg magától kitalálja, hogy melyiket kell alkalmaznia. Hasonlóképpen lényegtelen, hogy Dan milyen formátumban kapja meg a fájlt (szöveges vagy bináris), a gpg ezt is önműködően kezeli. Az üzenet megfejtéséhez Dannek meg kell adnia a titkos kulcsához tartozó jelmondatot. Ha nem tudja beírni a helyes jelmondatot, nem képes visszafejteni a fájlt.

Hitelesítés és ellenőrzés GnuPG-vel

A hitelesítés és az ellenőrzés sok tekintetben hasonlít a titkosításhoz és az üzenet megfejtéséhez. Tegyük fel, JoJo ír egy fontos, de nem bizalmas levelet Dannek. Ilyenkor szükséges, hogy Dan ellenőrizni tudja a levél hitelességét, de nem kell titkosítania. A ***beercontract.txt*** fájl aláírásához JoJonak a következő parancsot kell kiadnia:

```
gpg --output beercontract_signed.txt
➥--clearsign beercontract.txt
```

Ez a parancs fejlécet és láblécet ad a fájlhoz és ***beercontract_signed.txt*** néven menti. Fontos, hogy a kimeneti fájlnév és az eredeti fájl neve ne ugyanaz legyen, ebben az esetben ugyanis az eredeti fájl helyén egy a gpg-aláírás fej- és láblécével ellátott üres fájlt találnánk. A szöveges módot akkor érdemes használni, ha a levelet a levelezőprogramba be akarjuk illeszteni, vagy ki akarjuk onnan vágni.
A másik lehetőség, igen, kitaláltad, a bináris aláírás. Ebből kétféle létezik: ha tömörített bináris fájlt szeretnél, amely a levelet és az aláírást egyaránt tartalmazza, a `--clearsign` helyett használd a `--sign` parancsot. Egy sokkal kisebb bináris fájl létrehozásához, amely csak az aláírást tartalmazza, használd a `--detach-sig` parancsot. Ebben az esetben két fájlod lesz: a bináris aláírás és az eredeti levél. Mind a `--detach-sig`, mind a `--sign` előtt használhatod a `--output` kapcsolót.
Amikor Dan megkapja JoJo sörszerződését, ellenőrizni tudja az aláírást. Ehhez a fájlt a merevlemezre kell mentenie, mondjuk ***bcs.asc*** néven és a következő parancsot kell kiadnia:

```
gpg --verify bcs.asc
```

Ne felejtsük el, ha Dannek nincs meg JoJo nyilvános kulcsa, a gpg hibával tér vissza. Ha Dan kulcscsomóján megtalálható JoJo nyilvános kulcsa, és az aláírás hitelesnek bizonyul, a program valami ehhez hasonlót fog kiírni:

```
gpg: Signature made Fri 27 Jul 2001 04:46:46
PM CDT
      using DSA key ID C1C34866
gpg: Good signature from "John J. Figplucker
      (Smooth JoJo)
       <jojo@figpluckers-supreme.to>"
```

Dan ekkor és csakis ekkor lehet biztos abban, hogy azt a szerződést, amit az imént kapott, JoJo titkos kulcsával hitelesítették. Lehet, hogy JoJo tarkójához pisztolyt fogtak? Nem tudjuk. Lehet, hogy JoJo a jelmondatát felírta a billentyűzete aljára és most a munkatársai űznek gúnyt belőlünk? Megint azt kell mondanom: nem tudhatjuk biztosan. De ha hiszünk abban, hogy JoJo felelősségteljesen bánik a kulcsaival és gondosan őrzi őket, minden okunk megvan feltételezni, hogy az ő érvényes aláírását ellenőriztük.

A GnuPG barátságosabb felületei

Remélem, nem ijesztettelek meg ezzel a sok kapcsolóval és paranccsal (Isten hozott a Unix világában!). Ez inkább csak áttekintés volt, sok mindenről még nem eshetett szó e cikksorozat keretében. Meggyőződésem, hogy a gpg fontos és hasznos segédeszköz. Ez olyannyira igaz, hogy sokan dolgoznak azon, hogy barátságosabbá tegyék. A hivatalos GnuPG grafikus felületet neve GNU Privacy Assistant (GPA). A program még fejlesztés alatt áll, de nagyon ígéretes.
A Gimp Toolkitre épül, ezért csöppet sem meglepő módon kellemes látványt nyújt.
Természetesen más grafikus felületek is léteznek a gpg-hez: ilyen a Seahorse és a GnomePGP, a KDE Geheimnis nevű alkalmazása és a TkPGP (ezt Tk-val írták, ezért viszonylag ablakkezelő-független). Ezek mellett léteznek még modulok és bővítmények az elterjedtebb levelezőprogramokhoz is. További tájékoztatásért látogass el a GnuPG weblapjára és vess egy pillantást a ***Frontends*** részre.

Összefoglalás

Véget ért a kétrészes bevezetés a GnuPG alapvető használatába. A gpg olyan program, amelyet sokkal többeknek kellene használniuk, ezért kérlek, ne habozz: titkosíts. Pontosabban olyan kulcsokkal titkosíts, amelyeket ismerőseid aláírtak és ellenőriztek.

Mick Bauer (mick@visi.com) hálózati biztonsággal foglalkozó szaktanácsadó. 1995 óta a Linux elkötelezett híve, 1997 óta pedig OpenBSD prófétaként tevékenykedik. Mick minden kérdést és megjegyzést szívesen fogad.

Kapcsolódó címek

Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition, Bruce Schneier. John Wiley & Sons, 1995.
Dice-Based Password Generation Method (Jelszavak létrehozása dobókockával) ➲ http://www.diceware.com
The GNU Privacy Handbook
➲ http://www.gnupg.org/gph/en/manual.html
Kevésbé bőbeszédű, mint a gpg(1) súgóoldal, de részletesebb, mint ez a cikk.
A hivatalos GnuPG honlap ➲ http://www.gnupg.org. Erről az oldalról tölthető le a GnuPG legfrissebb változata és a GPA, valamint sok hasznos tudnivaló is található itt.
➲ http://www.gnupg.org/download.html/