PGP

Nachtrag zum letzten Jahr

Matthias Brüstle

Nürnberg 2015-11-22

Hash-Algorithmen

- Hash-Algorithmen erzeugen von Daten einen kurzen (16 – 64 Bytes), representativen Wert, der dann signiert wird.
- Daten sind bei GnuPG Dateien (Dokumente) und öffentliche Schlüssel mit dazu gehörigen Informationen
- bekannteste Hash-Algorithmen: MD5, SHA-1, SHA-224, SHA-256, SHA-284, SHA-512

Hash-Algorithmen: Status

- MD5: Vernichtend geschlagen!
  Erzeugung von falschen SSL-Zertifikaten (2008)
  „*Creating a rogue CA certificate*“
  `http://www.win.tue.nl/hashclash/rogue-ca/`
- SHA-1: Größere Lücken in der Verteidigung.
  Erzeugung von zufälligen Kollisionen (2015)
  „*The SHAppening: freestart collisions for SHA-1*“
  `https://sites.google.com/site/itstheshappening/`
  Mehr Informationen von 2014:
  `https://konklone.com/post/why-google-is-hurrying-the-web-to-kill-sha-1`

Hash-Algorithmen in GnuPG

- `gpg –export KEYID | gpg –list-packets`
  `:signature packet: algo 1, keyid ...`
  `version 4, created 1447619937, md5len ...`
  `digest algo 10, begin of digest 39 c5`

  MD5: 1, SHA-1: 2 (Standard), SHA-512: 10 (RFC4880)

- `gpg –export KEYID | pgpdump`
  `Old: Signature Packet(tag 2)(187 bytes)`
  `Ver 4 – new`
  `Sig type - Positive certification ...`
  `Pub alg - RSA Encrypt or Sign(pub 1)`
  `Hash alg - SHA512(hash 10)`

Aus der gpg.conf vom letzten mal

```
...
# Bevorzugte Algorithmen
personal-digest-preferences SHA512 SHA384 SHA256 SHA224
default-preference-list SHA512 SHA384 SHA256 SHA224 AES256 AES192 AES BZIP2
ZLIB ZIP Uncompressed # Für neu erzeugte Schlüssel
cert-digest-algo SHA512
...

$ gpg –export KEYID | pgpdump
...
    Hashed Sub: preferred symmetric algorithms(sub 11)(3 bytes)
        Sym alg - AES with 256-bit key(sym 9)
        Sym alg - AES with 192-bit key(sym 8)
        Sym alg - AES with 128-bit key(sym 7) (Ja, ich hab eine andere conf)
    Hashed Sub: preferred hash algorithms(sub 21)(4 bytes)
        Hash alg - SHA512(hash 10)
        Hash alg - SHA384(hash 9)
        Hash alg - SHA256(hash 8)
        Hash alg - SHA224(hash 11)
    Hashed Sub: preferred compression algorithms(sub 22)(3 bytes)
        Comp alg - ZLIB <RFC1950>(comp 2)
        Comp alg - ZIP <RFC1951>(comp 1)
        Comp alg - Uncompressed(comp 0)
...
```

Was passiert im Problemfall? (gpg1)

Fall 1: Schlüsselerzeugung ohne gpg.conf

```
Hashed Sub: preferred symmetric algorithms(sub 11)(5 bytes)
Sym alg - AES with 256-bit key(sym 9)
Sym alg - AES with 192-bit key(sym 8)
Sym alg - AES with 128-bit key(sym 7)
Sym alg - CAST5(sym 3)
Sym alg - Triple-DES(sym 2)
Hashed Sub: preferred hash algorithms(sub 21)(5 bytes)
Hash alg - SHA256(hash 8)
Hash alg - SHA1(hash 2)
Hash alg - SHA384(hash 9)
Hash alg - SHA512(hash 10)
Hash alg - SHA224(hash 11)
Hashed Sub: preferred compression algorithms(sub 22)(3 bytes)
Comp alg - ZLIB <RFC1950>(comp 2)
Comp alg - BZip2(comp 3)
Comp alg - ZIP <RFC1951>(comp 1)
```

Was passiert im Problemfall? (gpg1)

Fall 2: Signieren und verschlüsseln wobei:
- eigener Schlüssel gute Präferenzen hat und
- fremder Schlüssel **SHA-1** (oder nichts) präferiert

```
gpg: Good signature from "XXX" [ultimate]
Primary key fingerprint: ...
gpg: binary signature, digest algorithm SHA1
```

Oh!

Was passiert im Problemfall? (gpg1)

Fall 2: Signieren und verschlüsseln wobei:
- eigener Schlüssel gute Präferenzen hat und
- fremder Schlüssel **3DES** (oder nichts) präferiert

```
An sich selbst verschlüsselt:

gpg: public key encrypted data: good DEK

:encrypted data packet:

      ...

gpg: 3DES encrypted data
```

Oh!

Was tun?

- Eigenen Schlüssel prüfen (--edit-key: showpref)
  - Präferenzen ändern: gpg –edit-key: setpref
  - SHA-1 in Selbstsignaturen nur über wirres Anlegen und Löschen von User-IDs oder eigentlich besser neuen Schlüssel
- Auch den fremden Schlüssel prüfen?
- Auf jeden Fall jeden Algorithmus, der besser ist als SHA-1 und 3DES in die Präferenzen aufnehmen. (Alle SHA-2! RIPE-MD/160? Alle AES! Blow-/Twofisch? CAST5 eher nicht.)

Problem vom letzten mal: Schlüsselumzug

- Nichts am alten Schlüssel tun: Enigmail verschlüsselt eMails einfach an alle passenden Empfängerschlüssel.
- Schlüssel widerrufen: Alle Signaturen ungültig.
- Schlüssel ablaufen lassen: Alle Signaturen ungültig.
- Nur Verschlüsselungsschlüssel ablaufen lassen: Signaturen bleiben gültig, kann aber nicht mehr zum Verschlüssel verwendet werden (gpg –edit-key -> key *n* -> expire)

Saenks!

Matthias Brüstle

Nürnberg 2015-11-22