IETF-113 OpenPGP WG meeting

draft-ietf-openpgp-crypto-refresh

openpgp@ietf.org

2022-03-21

https://datatracker.ietf.org/doc/draft-ietf-openpgp-crypto-refresh/05/

IETF-113 OpenPGP WG Meeting Logistics

- When: Monday March 21 0900-1100 UTC (1000-1200 meeting local time)
- Notes: https://notes.ietf.org/notes-ietf-113-openpgp
- Jabber: openpgp@jabber.ietf.org
- Remote access: https://meetings.conf.meetecho.com/ietf113/?group=openpgp&short=&item=1
- On-site tool: https://meetings.conf.meetecho.com/onsite113/?group=openpgp&short=&item=1
- WG page: https://datatracker.ietf.org/wg/openpgp/documents/

IETF 113 Meeting Tips

- In-person participants:
  - Make sure to sign into the session using the Meetecho (usually the “onsite tool” client) from the Datatracker agenda
  - Use Meetecho to join the mic queue
  - Keep audio and video off if not using the onsite version
- Remote participants
  - Make sure your audio and video are off unless you are chairing or presenting during a session
  - Use of a headset is strongly recommended
- Everyone:
  - Be patient:-)
  - This is the 1st "hybrid" IETF meeting, and we're in the first session slot, so let's be ok with learning as we go

Note Well

This is a reminder of IETF policies in effect on various topics such as patents or code of conduct. It is only meant to point you in the right direction. Exceptions may apply. The IETF's patent policy and the definition of an IETF "contribution" and "participation" are set forth in BCP 79; please read it carefully.

As a reminder:

- By participating in the IETF, you agree to follow IETF processes and policies.
- If you are aware that any IETF contribution is covered by patents or patent applications that are owned or controlled by you or your sponsor, you must disclose that fact, or not participate in the discussion.
- As a participant in or attendee to any IETF activity you acknowledge that written, audio, video, and photographic records of meetings may be made public.
- Personal information that you provide to IETF will be handled in accordance with the IETF Privacy Statement.
- As a participant or attendee, you agree to work respectfully with other participants; please contact the ombudsteam (https://www.ietf.org/contact/ombudsteam/) if you have questions or concerns about this.

Definitive information is in the documents listed below and other IETF BCPs. For advice, please talk to WG chairs or ADs:

- [BCP 9](https://www.rfc-editor.org/info/bcp9) (Internet Standards Process)
- [BCP 25](https://www.rfc-editor.org/info/bcp25) (Working Group processes)
- [BCP 25](https://www.rfc-editor.org/info/bcp25) (Anti-Harassment Procedures)
- [BCP 54](https://www.rfc-editor.org/info/bcp54) (Code of Conduct)
- [BCP 78](https://www.rfc-editor.org/info/bcp78) (Copyright)
- [BCP 79](https://www.rfc-editor.org/info/bcp79) (Patents, Participation)
- https://www.ietf.org/privacy-policy/ (Privacy Policy)

Agenda

- Administrivia & Agenda bash (chairs, 5)
- Chartered work:
  - Design team modus operandi (chairs, 5)
  - Latest I-D (Justus Winter, 20)
    - https://datatracker.ietf.org/doc/draft-ietf-openpgp-crypto-refresh/
  - Issues/Merge-requests for discussion (various DT members, 45)
    - https://gitlab.com/openpgp-wg/rfc4880bis
  - Next steps (chairs, 5)
- Other presentations (time may shift if above takes longer):
  - A post-quantum approach for openpgp (Falko Strenzke, 20)
  - End-to-end encryption definition (Mallory Knodel, 20)
    - https://datatracker.ietf.org/doc/draft-knodel-e2ee-definition/
- "if time allows" (late request)
  - Key transparency (Aron Wussler)

Design Team

- List + Public Archive: https://mailarchive.ietf.org/arch/browse/openpgp-dt/
- Members: Daniel Huigens, Daniel Kahn Gillmor, Jeffrey Lau, Justus Winter, Niibe Yukata, Paul Wouters, Stephen Farrell, Werner Koch (for earlier meetings)
- Met most weeks, ~29 times
- Worked well: productive processing merge requests
- Goal: declare victory soon, once we think ready for WGLC
- Chairs likely to re-use mechanism (if people willing) if/when WG re-chartered for more work

Changes in draft-ietf-openpgp-crypto-refresh-05

Who's read the draft?

We'd like to get a sense as to how many people have read -05?
Are there specific problems/issues with that you'd like to mention now (we can talk about 'em in a bit)

5.14.2. v2 SEIP Data Packet Format

- AEAD
- SEIPDv2 better than AEDv2
  - Tag 20: Reserved (formerly AEAD Encrypted Data Packet)
- Fields!
  - cipher
  - AEAD mode
  - salt
- $\mathrm{HKDF}_{\mathrm{SHA256}}$
  - authenticates context → key separation
  - per-message key → robust encrypted reply

Message Key Derivation

- SK from v5 PKESK or v5 SKESK
- HKDF$_{\text{SHA256}}$
  - salt from v2 SEIPD (32 octets)
  - info:
    - packet tag
    - packet version
    - cipher
    - AEAD mode
    - chunk size
  - out:
    - message key
    - left-most parts of nonce

v2 SEIPD Chunked AEAD

- cipher setup
  - message key from HKDF
  - nonce:
    - (NLEN - 8) octets from HKDF
    - 8 octet BE counter from 0
  - AAD:
    - packet tag
    - packet version
    - cipher
    - AEAD mode
    - chunk size
    - (final chunk only) plaintext length
- chunking
  - chunk size: power of two, 64 bytes to 4 megabytes
  - one or more chunks
    - last one may be shorter
  - followed by tag
  - final zero-sized chunk

Example v4 and v5 messages

```
v4 SKESK Packet
    S2K
    AES-128
    ESK (plain: cipher+sk)

v1 SEIPD Packet
    AES-128, CFB mode
    (cipher from ESK)

  | Literal Data Packet
  | MDC Packet
```

```
v5 SKESK Packet
    S2K
    AES-128, OCB mode
    IV
    ESK

v2 SEIPD Packet
    AES-128, OCB mode
    Chunk size
    Salt

  | Literal Data Packet
  | Padding Packet
```

Status

GnuPG

- NIIBE has an interoperable implementation

Sequoia

- Justus has an interoperable implementation
- EAX good, GCM not great, OCB problematic
- used to generate test vectors

Ecosystem

- need reply-to-message API

v5 sigs and v5 keys; v5 ESKs and v2 SEIP

Signature matches Pubkey version

- v5 keys make v5 sigs/certifications
- v4 keys make v4 sigs/certifications

Encrypted Session Keys match SEIP version

- v2 SEIP uses v5 PKESK and v5 SKESK
- v1 SEIP uses v3 PKESK and v4 SKESK

Which Pubkeys take which SEIP version?

- Advertised in Features subpacket, as before

5.3 v5 SKESK

- AEAD
- key separation (also HKDF$_{\text{SHA256}}$)
- robust parsing

Sequoia

- branch exists
- Much nicer SKESK5 API

3.7.2.1. Secret-Key Encryption

- key separation (also $\mathrm{HKDF}_{\mathrm{SHA256}}$)
- robust parsing

5.5. v5 Key Material Packets

- public parameter octet count

OpenPGP.js

- Daniel has an interoperable implementation
  - GopenPGP patched to consume artifacts
- used to generate test vectors

5.1. v5 PKESK

- Recipient fingerprint

3.7.1.4. Argon2

- Argon2id
- RFC9106
  - parameter recommendation: $t=1$, $p=4$, $m=2^{21}$
  - m in kibibytes (KiB)

```
Octet  0:     0x04 (the S2K tag)
Octets 1-16: 16-octet salt value
Octet  17:    one-octet number of passes t
Octet  18:    one-octet degree of parallelism p
Octet  19:    one-octet exponent indicating the memory size m
```

GnuPG

- NIIBE has an interoperable implementation

Sequoia

- Justus has an interoperable implementation
- used to generate test vectors

5.2.3. v5 Signature Packet Format

- prefix salt, defends against
  - attacks on hash function collision resistance
  - the evil web-app attack
  - EdDSA fault injection attack
- related: 5.4. v5 One-Pass Signature Packet
  - issuer fingerprint (v3 had Key ID)
  - includes salt

OpenPGP.js

- Daniel has an interoperable implementation
  - GopenPGP patched to consume artifacts
- used to generate test vectors

5.15. Padding Packet (Tag 21)

- like the Marker packet, but free form
- can appear anywhere
  - guidance for backwards compatibility
- body SHOULD be random

Sequoia

- branch exists

General improvements

- 3.2.1. Using MPIs to encode other data
- 4.2.2. Legacy Format Packet Lengths
- 5.2.3.33. Intended Recipient Fingerprint
- 11.1. Transferable Public Keys
  - valid w/o User ID
  - ongoing discussion wrt mandatory self-sig
- newer toolchain
  - markdown
  - one sentence per line
  - Gitlab-supported workflow
- tables
- structure

Algorithms

- public key algorithms
  - MTI: EdDSA and ECDH
- curves
  - MTI: Ed25519 and "ECDH using Curve25519"
  - SHOULD: Ed448 and X448
- ciphers
  - MTI: AES-128
  - in: Camellia-*
  - out (archive exception): IDEA, TripleDES, or CAST5
- AEAD modes
  - MTI: OCB
  - in: GCM (FIPS approved)
- hash algorithms
  - MTI: SHA2-256
  - in: SHA{2,3}-*
  - out: MD5, SHA-1, and RIPE-MD/160

12.2. Key IDs and Fingerprints

- v5 fingerprints are 32 octets
- v5 Key IDs are the left-most 8 octets

OpenPGP.js

- Daniel has an interoperable implementation
    - GopenPGP patched to consume artifacts
- used to generate test vectors

Issues where DT would like to see broader discussion

- Certificate structure (Daniel Huigens)
- KeyIDs and fingerprints (Paul Wouters/Stephen Farrell)
- SHA-1/sha1collisiondetect (Daniel Kahn Gillmor)

Certificate structure (Daniel Huigens)

- Some implementations today require a self-signed User ID
- For some use cases it's useful not to publish a User ID
  - Publishing keys where the User ID has not been verified
  - Publishing keys without revealing personal details in the User ID
  - Making it easier to hide whether an email address exists or not

Certificate structure (II)

- RFC4880 required a User ID, but not a self-signature
- Crypto refresh requires neither a User ID nor a self-signature

Certificate structure (III)

- Do we need a User ID?
  - When encrypting to a key, you need to verify ownership some other way anyway
  - When verifying using a key, you need to verify that the signer is the sender
    - Sign email headers?
    - "Signer's User ID" subpacket?
  - Catch mistakes?

Certificate structure (IV)

- Do we need a self-signature?
    - A place to store expiration, preferences, flags, features, etc.
    - Can't be removed (unless you have another signature without them)

Certificate structure (V)

- Tentative proposal:
  - Require a direct-key signature
  - Only look at key properties in the direct-key signature
  - User IDs are optional
  - If sender identity is important, sign it as part of the message (e.g. email headers)

Certificate structure (VI)

- Questions to the WG:
  - Is this even in charter?
    - Some of this is not very crypto-refresh-related
    - Algorithm preferences are crypto-adjacent
    - V5 keys are a nice opportunity to fix this
  - Do we care about User ID-specific preferences?
    - If yes → still look at key properties in User ID self-signatures?
  - Do we care about PGP/Inline, and clearsigned messages, where the sender is not signed?
    - If yes → use "Signer's User ID" subpacket?

KeyIDs and fingerprints (Paul Wouters/Stephen Farrell)

- Fingerprint is full-sized hash output (256 bits)
- KeyIDs have historically been shorter in a (possibly unwise) attempt to make them human-consumable
- DT didn't reach consensus on a sufficiently good change to this situation
    - Perhaps closest we came was roughly: "don't recommend any specific form of KeyID but do document the security considerations of some of the possible approaches to KeyIDs

SHA-1/sha1collisiondetect (Daniel Kahn Gillmor)

https://gitlab.com/openpgp-wg/rfc4880bis/-/issues/99

https://marc-stevens.nl/research/papers/C13-S.pdf

SHA-1 is trouble

- Known mechanism for collisions
    - (but probably not a problem for primary key fingerprints)
- Still necessary in OpenPGP as long as v4 keys circulate

sha1collisiondetect is SHA-1 except during known collsions

- Noisy breakage better than silent breakage
- Not a formally-specified standard

Driving out SHA-1

- Encourage sha1cd (or similar) where implementations use SHA-1?

Next Steps

- DT members would like to declare victory - meaning offering a draft that could be input for WGLC
    - At that point DT will dissolve
- We're not there yet but closing in on that
- Once we get there it'll be time to consider re-chartering if there's an appetite for that.
    - If we get there chairs might constitute another DT (current one having worked well)