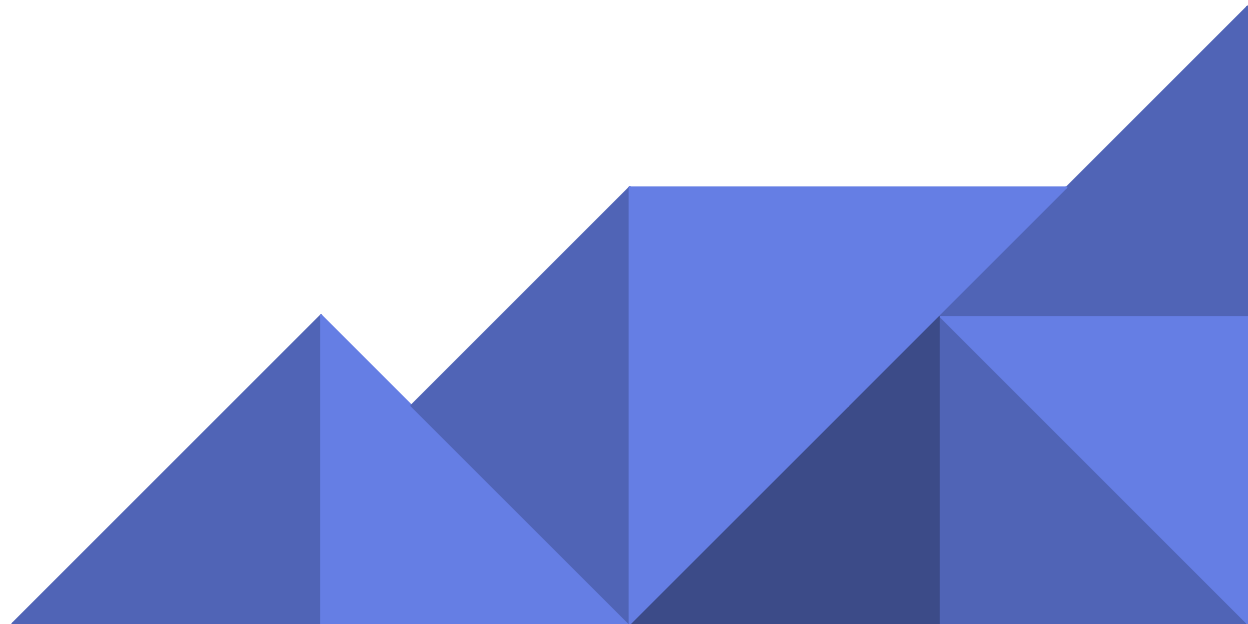
IETF 120

Persistent Symmetric Keys

Daniel Huigens



2024-07-22

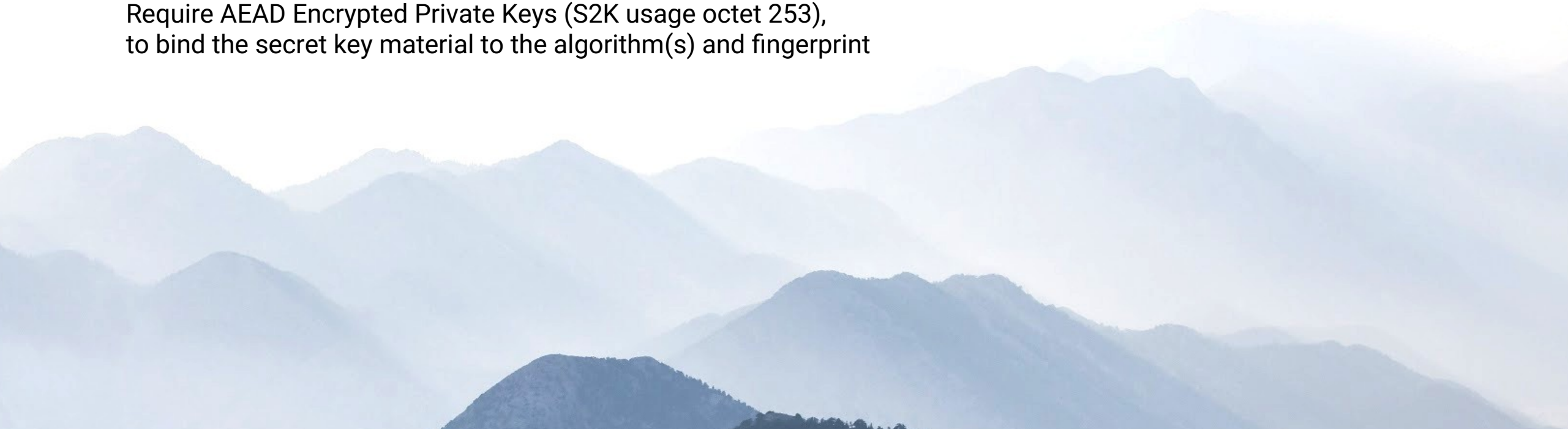
Current Status

- Adopted by the WG: draft-ietf-openpgp-persistent-symmetric-keys (diff with draft-huigens)
- Experimental implementations in forks/branches of OpenPGP.js and go-crypto
  (but latest changes haven't been incorporated yet)

New ~~Public~~ Persistent Key Algorithms

ID	Alg.	Public Key	Secret Key	Signature	PKESK
128	AEAD	sym. algo, AEAD algo, fprt seed	key material	N/A	IV, ciphertext
129	HMAC	hash algo, fingerprint seed	key material	authentication tag	N/A

Require AEAD Encrypted Private Keys (S2K usage octet 253),
to bind the secret key material to the algorithm(s) and fingerprint



Guidance for usage

- Symmetric (re-)encryption
- Symmetric attestations
- Attesting to signature verifications?

Spec out symmetric persistent algorithm space

- Declare that persistent algorithm IDs 128-255 are symmetric
- Define private/experimental persistent symmetric algorithm space (200-210, or 228-238?)

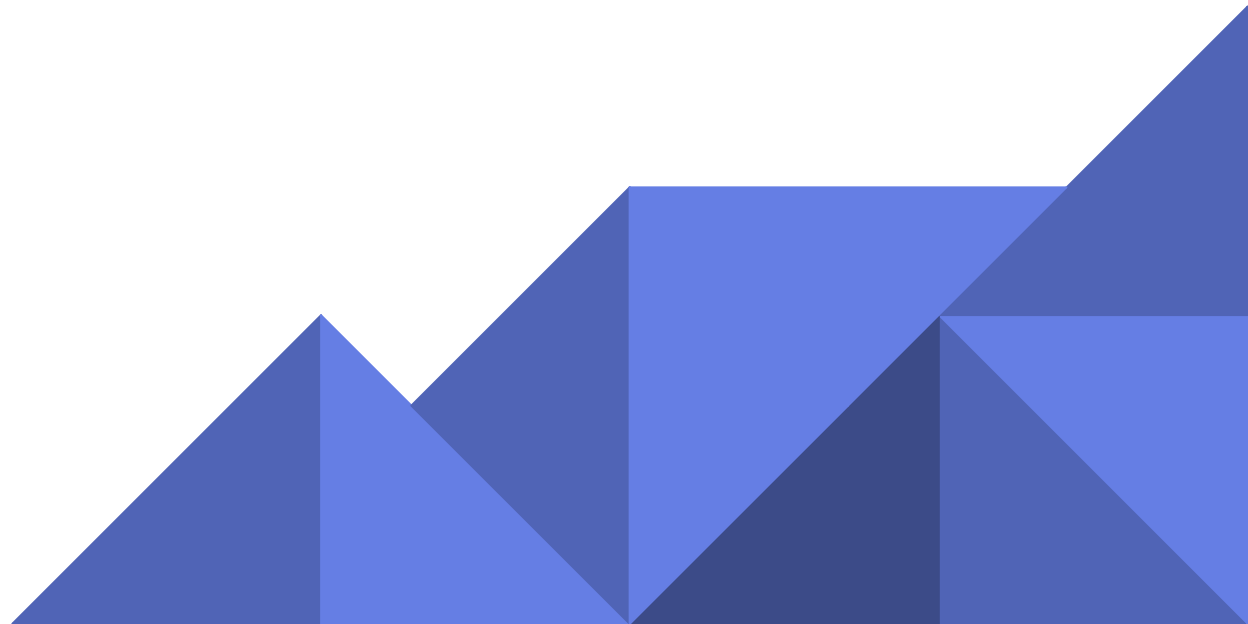
Questions for the WG

- Anything (else) we should add?
- Any other feedback/thoughts?

IETF 120

Signature Salt Notation

Daniel Huigens



2024-07-22

Salting v4 signatures

- Create a signature notation subpacket with a random salt
- Length dependent on the hash algorithm (same as for v6 signatures)

Security advantage (compared to doing nothing)

- Protects against fault attacks in deterministic signing algorithms (e.g. EdDSA)

Security disadvantage (compared to v6 salting)

- Doesn’t necessarily protect against common prefix attacks

Current Status

- Sequoia: `salt@notations.sequoia-pgp.org` (32 octets)
- OpenPGP.js v6 and GopenPGP v3: `salt@notations.openpgpjs.org` (dependent on hash algorithm)
- Draft: `salt` (dependent on hash algorithm)

Questions for the WG

- Any feedback/thoughts?
- Adopt?