Persistent Symmetric Keys

Recap: what are we doing here?

- Currently, OpenPGP keys can only store asymmetric key material
- This is overkill when encrypting or authenticating data for yourself
  - Like drafts, sent messages, archival, backup, etc.
- So, let's introduce symmetric algorithms (AEAD and HMAC)
- And, rename the Public Key Algorithms Registry to the Persistent Key Algorithms Registry

Proposed algorithms

ID	Algorithm	Public Key Format	Secret Key Format	Signature Format	PKESK Format
128	AEAD	sym. algo, AEAD algo, fingerprint seed [Section 5.1]	key material	N/A	IV, ciphertext [Section 5.3]
129	HMAC [RFC2104]	hash algo, fingerprint seed [Section 5.2]	key material	authentication tag [Section 5.4]	N/A
130 to 139	Reserved for Future Persistent Symmetric Key Algorithms				
140 to 143	Private or Experimental Persistent Symmetric Key Algorithms				

Table 1: Persistent Symmetric Key Algorithm registrations

Key flags

- We currently have: `sign data`, `encrypt communications`, `encrypt storage`
- The latter two have seemed useless until now
- Might be slightly useful now
- `sign communications` and `sign storage` would be very useful!
- So: rename `sign data` to `sign communications` and add `sign storage`?

Key flags (cont'd)

- Say that symmetric keys MUST (or SHOULD?) have the `storage` key flag
- Asymmetric keys MUST (or SHOULD?) have the `communications` key flag
  if there is also a `storage` key in the same TSK/TPK (bundle?),
  otherwise they have both flags (same as today)

Key flags (cont'd)

- When encrypting and/or signing a message, indicate the target usage
- `communication`, `storage`, or `mixed` (default)

Message	Encryption keys	Signing keys
For communication	Use communication keys (asymmetric)	Use communication keys (asymmetric)
For storage	Use storage keys (symmetric if available)	Use storage keys (symmetric if available?)
Mixed	Use any keys (prefer storage, i.e. symmetric if available?)	Use communication keys (asymmetric)

Key flags (cont'd)

- Is it useful to disallow encryption with a symmetric key?
  - If you're sending a message to yourself that might then be forwarded using proxy re-encryption
  - But could also just pass the public key
  - But² we might anyway want a signal that private keys are allowed to be used
    - Then, default to `communication` instead of `mixed`

Suggested key structure (1)

Key packet	Key flag
Primary key (v6, HMAC)	Sign storage
Subkey (v6, AEAD)	Encrypt storage
Primary key (v6, ML-DSA)	Sign communications
Subkey (v6, ML-KEM)	Encrypt communications
Primary key (v4, EdDSA)	Sign communications
Subkey (v4, ECDH)	Encrypt communications

Suggested key structure (2)

Key packet	**Key flag**
Primary key (v6, ML-DSA)	Sign communications
Subkey (v6, ML-KEM)	Encrypt communications
Subkey (v6, AEAD)	Encrypt storage
Primary key (v4, EdDSA)	Sign communications
Subkey (v4, ECDH)	Encrypt communications

API considerations

- Allow generating multiple or mixed TSKs
- Allow using a private key in place of a public key
- When encrypting and/or signing a message, allow indicating the target usage and select the appropriate key(s) to be used

SOP proposal

- Allow generating multiple keys in `sop generate-key[s]` (!114)
- `sop encrypt` and `[inline-]verify` can take `KEYS` or `CERTS` (!110)
- `sop encrypt --one-recipient-per-certificate-bundle` (!124)
- `sop sign --one-{sender,signer}-per-key-bundle` (!125)
- `sop {encrypt,sign,inline-sign}`
  `--target={communication,storage,mixed}` (!38)

Questions for the WG

- Does the proposal on key flags make sense?
  - Do people agree we need a key flag for signing storage?
- Do we need guidance on key structure?
- Let’s discuss (sub)key selection? :)

Thoughts? Questions?