openpgp
Internet-Draft
Updates: 4880 (if approved)
Intended status: Standards Track
Expires: 23 January 2025

D. Shaw
Jabberwocky Tech
A. Gallagher, Ed.
PGPKeys.EU

22 July 2024

OpenPGP Replacement Key Signalling Mechanism
draft-ietf-openpgp-replacementkey-00

Abstract

This document specifies a method in OpenPGP to suggest a replacement for an expired, revoked, or deprecated primary key.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://andrewgdotcom.gitlab.io/draft-gallagher-openpgp-replacementkey. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-openpgp-replacementkey/.

Discussion of this document takes place on the OpenPGP Working Group mailing list (mailto:openpgp@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/openpgp/. Subscribe at https://www.ietf.org/mailman/listinfo/openpgp/.

Source for this draft and an issue tracker can be found at https://gitlab.com/andrewgdotcom/draft-gallagher-openpgp-replacementkey.

Status of This Memo

This Internet-Draft will expire on 23 January 2025.

Table of Contents

1. Introduction

The OpenPGP message format [I-D.ietf-openpgp-crypto-refresh] defines two ways to invalidate a primary key. One way is that the primary key may be explicitly revoked via a key revocation signature. OpenPGP also supports the concept of key expiration, a date after which the key should not be used. When a primary key is revoked or expires, very often there is another key that is intended to replace it.

A key owner may also create a new primary key that is intended to deprecate and replace their existing primary key, but without revoking or expiring that key. This is useful during the rollout of new key versions and algorithms which may not (yet) enjoy universal support. In such cases, a key owner may prefer that their correspondents use their new primary key, but if this is not possible for technical reasons they may continue to use the non-preferred key, which remains valid.

In the past some key owners have created key transition documents, which are signed, human-readable statements stating that a newer primary key should be preferred by their correspondents. It is desirable that this process be automated through a standardised machine-readable mechanism.

This document is to specify the format of a Signature Subpacket to be optionally included in a revocation signature or self-signature on a primary key. This subpacket contains a pointer to a suggested replacement for the primary key that is signed over, or a primary key for which the current key is the suggested replacement. The replacement key may then be automatically retrieved and (if supported and validated) used instead of the original key.

2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.1. Terminology

In OpenPGP, the term "key" has historically been used loosely to refer to several distinct concepts. Care is therefore required when talking about "keys" in a non-specific sense. In this document, we use the following convention:

* "replacement key" and "original key" always refer to a public key and, unless otherwise qualified, to a full Transferable Public Key (TPK).

* "target key" refers to either a replacement key or an original key that is referred to by a Replacement Key Subpacket.

* "current key" refers to the primary public key belonging to the self-signature currently under discussion.

3. The Replacement Key Subpacket

The Replacement Key Subpacket is a Signature Subpacket ([I-D.ietf-openpgp-crypto-refresh] section 5.2.3.7), and all general Signature Subpacket considerations from there apply here as well. The value of the Signature Subpacket type octet for the Replacement Key Subpacket is (insert this later).

A Preferred Key Server subpacket ([I-D.ietf-openpgp-crypto-refresh] section 5.2.3.26) MAY be included in the revocation or direct key signature to recommend a location and method to fetch the replacement key. Note however that since this subpacket automatically also applies to the current key, it cannot be used to set the replacement key's preferred keyserver to a different value than that of the current key.

The absence of a Replacement Key Subpacket SHOULD NOT be interpreted as meaning that there is no replacement (or original) for the current key. The "no replacement" bit SHOULD be used instead (see below).

The Replacement Key Subpacket MUST only be used in the hashed subpackets area of a primary key revocation or direct key signature.

4. Format of the Replacement Key Subpacket

The format of the Replacement Key Subpacket is:

Octets	Field	Notes
1	Subpacket Version	MUST be 0x01
1	Class	
1	Target Key Version (1)	optional
N1	Target Key Fingerprint (1)	optional
M	Target Key Imprint (1)	optional
1	Target Key Version (2)	optional
N2	Target Key Fingerprint (2)	optional
M	Target Key Imprint (2)	optional
...	...	...

Table 1: Replacement Key Subpacket Fields

The subpacket version octet MUST be set to 0x01 to indicate the version of the Replacement Key Subpacket as specified in this document. An implementation that encounters a subpacket version octet that is different than the version(s) it is capable of understanding MUST disregard that Replacement Key Subpacket.

Note that if the critical bit on the Replacement Key Subpacket is set, a receiving application could consider the whole self-signature to be in error ([I-D.ietf-openpgp-crypto-refresh] section 5.2.3.7). The critical bit therefore SHOULD NOT be set on the Replacement Key Subpacket.

The class octet contains flags that indicate the form and semantics of the subpacket:

Flag bit	Flag name	Form of remainder of packet
0x80	No replacement	No optional fields
0x40	Inverse relationship	Multiple targets may be given

Table 2: Replacement Key Subpacket Flags

The 0x80 bit of the class octet is the "no replacement" bit. When set, this explicitly specifies there is no replacement (or original) for the current key.

The 0x40 bit of the class octet is the "inverse relationship" bit. When set, this means that the target key(s) identified by the packet are the primary keys for which the current key is the replacement key. If both the 0x80 and 0x40 bits are set, it means that the current key is not a replacement for any other key.

All other bits of the class octet are currently undefined and MUST be set to zero.

If the class octet does not have the 0x80 bit set to indicate there is no replacement, the Replacement Key Subpacket MUST also contain 1 octet for the version of the target key, N octets for the fingerprint of the target primary key, and M octets for an imprint of the target primary key (see below). If the class octet has the 0x40 bit set, the subpacket MAY repeat the three optional fields one or more times, to refer to multiple target keys that the current key is a replacement for.

If present, the length of the Target Key Fingerprint field (N) MUST equal the fingerprint length corresponding to the immediately preceding Target Key Version field, e.g. 20 octets for version 4, or 32 octets for version 6. If present, the length of the Target Key Imprint field (M) MUST equal the length of the output of the digest algorithm used by the enclosing signature, e.g. 32 octets for SHA2-256.

If the intent is to state that the replacement (or original) key is unknown, then no Replacement Key Subpacket should be included in the revocation signature.

4.1. Key Imprints

An imprint of a public key packet is a generalisation of a fingerprint. It is calculated in the same way as the fingerprint, except that it MAY use a digest algorithm other than the one specified for the fingerprint. Conversely, the fingerprint of a public key packet can be considered a special case of an imprint. A public key packet has only one fingerprint, but may have any number of imprints, each using a different digest algorithm.

When used in a Replacement Key Subpacket, an imprint MUST use the same digest algorithm as the enclosing signature. This guards against key-substitution attacks when referring to keys that use weaker digest algorithms in their fingerprints. If the signature’s

digest algorithm is the same as that used by the fingerprint, then the imprint and the fingerprint will be identical. In such a case, the imprint MUST still be included for parsing reasons.

4.2. Graph Topology

A given signature MUST contain at most one Replacement Key Subpacket. If a signature contains more than one such subpacket, a receiving application MUST disregard them all. This imposes a simple graph topology on replacement key relationships:

* An original key MUST NOT claim to have more than one replacement key.

* An original key that claims to have a replacement key MUST NOT claim to be the replacement key for any other key(s).

In addition, the order of the original keys specified in an inverse-relationship Replacement Key Subpacket is meaningful. If a replacement key is supported by a receiving application, but is not usable for the desired purpose (for example, it may not have an encryption-capable subkey), the application MAY use the ordering of the original keys in its inverse Replacement Key Subpacket (if one exists) to indicate which original key is preferred as a fallback. The original keys SHOULD therefore be listed in order of decreasing preference.

5. Trust and Validation of the Replacement Key Subpacket

5.1. Key Equivalence Binding

The existence of a matching pair of forward and inverse Replacement Key Subpackets on the most recent direct self-signatures (or key revocations) over two primary keys, with each referring to the other primary key, forms a Key Equivalence Binding. If one primary key is validated for use in a particular context, then a bound-equivalent primary key and its subkeys are also valid, regardless of any User ID certifications over the second primary key (or lack thereof).

The equivalence binding is invalidated under the following circumstances:

* if either primary key is hard-revoked.

* if either primary key overrides the equivalence binding with a new direct self-signature that a) does not contain a Replacement Key Subpacket, or b) contains a Replacement Key Subpacket that does not refer to the other key.

* if either signature that forms the equivalence binding has expired.

Note however:

* If either primary key is soft-revoked or expired, the equivalence binding is unaffected.

* If either primary key is hard-revoked, then the equivalence binding is invalidated and the other key is unaffected.

* Other properties (such as expiry dates, usage preferences, custom notations) SHOULD NOT be applied across the equivalence binding.

* Key Equivalence is transitive; if A is equivalent to B and B is equivalent to C, then A is equivalent to C.

If two or more primary keys are bound-equivalent, they MUST be treated as a single key for the purposes of the Web of Trust, particularly when calculating partial trust values.

5.2. Without a Key Equivalence Binding

The Replacement Key Subpacket MUST NOT be treated as a Web of Trust certification over either the current or replacement key. In the absence of a Key Equivalence Binding, a receiving implementation SHOULD validate the replacement key as they would any other TPK. If the replacement key is supported, and validates successfully, it SHOULD be preferred over the current key when determining which TPK to use for correspondence.

It is also suggested that the key owner asks the third parties who certified their original key to certify the replacement key. Distribution of the replacement key over a trusted mechanism (such as WKD) MAY also be used to confer legitimacy.

6. Placement of the Replacement Key Subpacket

The Replacement Key Subpacket is only meaningful on a primary key revocation or direct key signature, and MUST NOT appear elsewhere. A replacement subkey can be directly added by the key owner with no need for the indirection provided by this subpacket. The Replacement Key Subpacket MUST be placed in the hashed subpackets area of the signature to prevent a possible key substitution attack. If the Replacement Key Subpacket was allowed in the unhashed subpackets area, an attacker could add a bogus Replacement Key Subpacket to an existing signature.

7. Security Considerations

A Key Equivalence Binding requires the active consent of both primary key owners. This is to prevent one key owner from unilaterally claiming signatures made by the other key owner, using the same argument that motivates the embedded Primary Key Binding signature in a signing-capable subkey's binding signature.

The Target Key Imprint is included to mitigate against weaknesses in the fingerprint digest algorithm used by older key versions. By including a digest over the target primary public key packet, using the same digest algorithm as the enclosing signature, we ensure that the indirect cryptographic binding between the equivalent keys is of the same overall strength as a signature made directly over the target primary public key (as in a certification signature or subkey binding signature). We intentionally chose not to use embedded back-signatures or third-party certifications, both to keep the design simple and to limit the size of the subpacket(s) required.

In the absence of a complete Key Equivalence Binding, the Replacement Key Subpacket MUST be treated as merely advisory. In this scenario, it provides information for the purposes of key discovery and order of preference only, without any trust statement regarding the replacement key. Implementations SHOULD NOT infer any trust value from a single Replacement Key Subpacket, and SHOULD validate the replacement key as they would any other key.

In addition, as this document is an update of [I-D.ietf-openpgp-crypto-refresh], the security considerations there should be carefully reviewed.

8. IANA Considerations

This document requests that the following entry be added to the OpenPGP Signature Subpacket registry:

Type	Name	Specification
TBC	Replacement Key	This document

Table 3: Signature Subpacket Registry

9. Normative References

[I-D.ietf-openpgp-crypto-refresh]
Wouters, P., Huigens, D., Winter, J., and N. Yutaka, "OpenPGP", Work in Progress, Internet-Draft, draft-ietf-openpgp-crypto-refresh-13, 4 January 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-openpgp-crypto-refresh-13>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

Appendix A. Example Workflows

A.1. Alice Revokes her Primary Key

* Bob wants to send Alice a message; Bob has Alice’s original key but they have not corresponded for some time.

* Bob’s client refreshes Aliceâ\200\231s original key from a keyserver (by fingerprint); it contains a revocation signature with a Replacement Key Subpacket.

* Bob’s client looks up Aliceâ\200\231s replacement key from a keyserver (by fingerprint); it is certified by the same people that certified her original key (some of whom Bob may trust) and/or Aliceâ\200\231s original key itself (which Bob’s policy may consider sufficient).

* Bob’s client uses Aliceâ\200\231s replacement key instead of the original key.

There are other means to achieve a similar result, such as WKD or Autocrypt, but they may not be available. For example, Aliceâ\200\231s service provider may not support WKD, and Alice may not have sent Bob an autocrypt message since revoking her original primary key.

A.2. Alice Creates a V6 Primary Key

* Bob wants to send Alice a message and has Alice’s v4 original key.

* Either Bob’s copy of Alice’s original key already has the Replacement Key Subpacket pointing to a v6 primary key, or Bob refreshes Alice’s original key from a keyserver and sees a new Replacement Key Subpacket.

* If Bob has a v6 implementation, it can proceed with fetching Alice's v6 replacement key, validating it, etc, and then use it to send his message to Alice.

* If Bob doesn't have a v6 implementation, it can continue to use Alice's v4 original key.

WKD does not currently allow more than one valid TPK to be returned for a query, therefore it cannot easily support this use case.

Appendix B. Acknowledgments

The authors would like to thank Bart Butler, Kai Engert, Daniel Kahn Gillmor, Daniel Huigens, Simon Josefsson, Heiko SchÃ¤fer, Falko Strenzke, Justus Winter and Aron Wussler for suggestions and discussions.

Appendix C. Document History

Note to RFC Editor: this section should be removed before publication.

C.1. Changes Between draft-gallagher-openpgp-replacementkey-02 and draft-ietf-openpgp-replacementkey-00

* Standardised capitalisation and terminology

C.2. Changes Between -01 and -02

* Specified Public Key Imprints.

* Specified inverse relationship flag and packet format.

* Restricted graph topology.

* Specified Key Equivalence Binding.

* Guidance re subpacket placement escalated from SHOULD to MUST, and critical bit to SHOULD NOT.

C.3. Changes Between -00 and -01

* Added example workflows.

* Specifically describe "deprecation without expiry or revocation" use case.

* Add note about weakness of signatures over fingerprints.

* Miscellaneous clarifications.

C.4. Changes Between draft-shaw-openpgp-replacementkey-00 and draft-gallagher-openpgp-replacementkey-00

* Changed algid octet to key version octet.

* Changed initial subpacket version number to 1.

* Clarified semantics of some edge cases.

Authors' Addresses

Daphne Shaw
Jabberwocky Tech
Email: dshaw@jabberwocky.com

Andrew Gallagher (editor)
PGPKeys.EU
Email: andrewg@andrewg.com