openpgp
Internet-Draft
Intended status: Informational
Expires: October 6, 2019

D. Gillmor
ACLU

April 04, 2019

Abuse-Resistant OpenPGP Keystores
draft-dkg-openpgp-abuse-resistant-keystore-00

Abstract

OpenPGP transferable public keys are composite certificates, made up of primary keys, user IDs, identity certifications ("signature packets"), subkeys, and so on. They are often assembled by merging multiple certificates that all share the same primary key, and distributed in public keystores.

Unfortunately, since any third-party can add certifications with any content to any OpenPGP certificate, the assembled/merged form of a certificate can become unwieldy or undistributable.

This draft documents techniques that an archive of OpenPGP certificates can use to mitigate the impact of these third-party certificate flooding attacks.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 6, 2019.

Table of Contents

1. Introduction

1.1. Requirements Language

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.

1.2. Terminology

- "OpenPGP certificate" (or just "certificate") is used interchangeably with [RFC4880]'s "Transferable Public Key". The term "certificate" refers unambiguously to the entire composite object, unlike "key", which might also be used to refer to a primary key or subkey.

- An "identity certification" (or just "certification") is an [RFC4880] signature packet that covers OpenPGP identity information - that is, any signature packet of type 0x10, 0x11, 0x12, or 0x13. Certifications are said to (try to) "bind" a primary key to a User ID.

- The primary key that makes the certification is known as the "issuer". The primary key over which the certification is made is known as the "subject".

- A "first-party certification" is issued by the primary key of a certificate, and binds itself to a user ID in the certificate. That is, the issuer is the same as the subject. This is sometimes referred to as a "self-sig".

- A "third-party certification" is a made over a primary key and user ID by some other certification-capable primary key. That is, the issuer is different than the subject. (The elusive "second-party" is presumed to be the verifier who is trying to interpret the certificate)

- A "keystore" is any collection of OpenPGP certificates. Keystores typically receive mergeable updates over the course of their lifetime which might add to the set of OpenPGP certificates they hold, or update the certificates.

- "Certificate discovery" is the process whereby a user retrieves an OpenPGP certificate based on user ID. A user attempting to discover a certificate from a keystore will search for a substring of the known user IDs, most typically an e-mail address if the

user ID is an [RFC5322] name-addr or addr-spec. Some certificate discovery mechanisms look for an exact match on the known user IDs. [I-D.koch-openpgp-webkey-service] and [I-D.shaw-openpgp-hkp] are both certificate discovery mechanisms.

- "Certificate validation" is the process whereby a user decides whether a given user ID in an OpenPGP certificate is acceptable. For example, if the certificate has a user ID of "Alice alice@example.org [1]" and the user wants to send an e-mail to alice@example.org, the mail user agent might want to ensure that the certificate is valid for this e-mail address before encrypting to it. This process can take different forms, and can consider many different factors, some of which are not directly contained in the certificate itself. For example, certificate validation might consider whether the certificate was fetched via DANE ([RFC7929]) or WKD ([I-D.koch-openpgp-webkey-service]); or whether it has seen e-mails from that address signed by the certificate in the past; or how long it has known about certificate.

- "Certificate update" is the process whereby a user fetches new information about a certificate, potentially merging those OpnePGP packets to change the status of the certificate. Updates might include adding or revoking user IDs or subkeys, updating expiration dates, or even revoking the entire certificate by revoking the primary key directly. A user attempting to update a certificate typically queries a keystore based on the certificate's fingerprint.

- A "keyserver" is a particular kind of keystore, typically means of publicly distributing OpenPGP certificates or updates to them. Examples of keyserver software include [SKS] and [MAILVELOPE-KEYSERVER]. One common HTTP interface for keyservers is [I-D.shaw-openpgp-hkp].

- A "synchronizing keyserver" is a keyserver which gossips with other peers, and typically acts as an append-only log. Such a keyserver is typically useful for certificate discovery, certificate updates, and revocation information. They are typically _not_ useful for certificate validation, since they make no assertions about whether the identities in the certificates they server are accurate. As of the writing of this document, [SKS] is the canonical synchronizing keyserver implementation, though other implementations exist.

- An "e-mail-validating keyserver" is a keyserver which attempts to verify the identity in an OpenPGP certificate's user ID by confirming access to the e-mail account, and possibly by confirming access to the secret key. Some implementations permit

removal of a certificate by anyone who can prove access to the e-mail address in question. They are useful for certificate discovery based on e-mail address and certificate validation (by users who trust the operator), but some may not be useful for certificate update or revocation, since a certificate could be simply replaced by an adversary who also has access to the e-mail address in question. [MAILVELOPE-KEYSERVER] is an example of such a keyserver.

- "Cryptographic validity" refers to mathematical evidence that a signature came from the secret key associated with the public key it claims to come from. Note that a certification may be cryptographically valid without the signed data being true (for example, a given certificate with the user ID "Alice alice@example.org [2]" might not belong to the person who controls the e-mail address "alice@example.org" even though the self-sig is cryptographically valid). In particular, cryptographic validity for user ID in a certificate is typically insufficient evidence for certificate validation. Also note that knowledge of the public key of the issuer is necessary to determine whether any given signature is cryptographically valid. Some keyservers perform cryptographic validation in some contexts. Other keyservers (like [SKS]) perform no cryptographic validation whatsoever.

2. Problem Statement

Many public keystores (including both the [SKS] keyserver network and [MAILVELOPE-KEYSERVER]) allow anyone to attach arbitrary data (in the form of third-party certifications) to any certificate, bloating that certificate to the point of being impossible to effectively retrieve. For example, some OpenPGP implementations simply refuse to process certificates larger than a certain size.

This kind of Denial-of-Service attack makes it possible to make someone else's certificate unretrievable from the keystore, preventing certificate discovery. It also makes it possible to swamp a certificate that has been revoked, preventing certificate update, potentially leaving the client of the keystore with the compromised certificate in an unrevoked state locally.

Additionally, even without malice, OpenPGP certificates can potentially grow without bound.

The rest of this document describes some mitigations that can be used by keystores that are concerned about these problems but want to continue to offer some level of service for certificate discovery, certificate update, or certificate validation.

3. Simple Mitigations

These steps can be taken by any keystore that wants to avoid obviously malicious abuse. They can be implemented on receipt of any new packet, and are based strictly on the structure of the packet itself.

3.1. Limited Packet Sizes

While [RFC4880] permits OpenPGP packet sizes of arbitrary length, OpenPGP certificates rarely need to be so large. An abuse-resistant keystore SHOULD reject any OpenPGP packet larger than 8383 octets. (This cutoff is chosen because it guarantees that the packet size can be represented as a one- or two-octet [RFC4880] "New Format Packet Length", but it could be reduced further)

This may cause problems for user attribute packets that contain large images, but it's not clear that these images are concretely useful in any context. Some keystores MAY extend this limit for user attribute packets specifically, but SHOULD NOT allow even user attributes packets larger than 65536 octets.

3.2. Strict User IDs

[RFC4880] indicates that User IDs are expected to be UTF-8 strings. An abuse-resistant keystore MUST reject any user ID that is not valid UTF-8.

Some abuse-resistant keystores MAY only accept User IDs that meet even stricter conventions, such as an [RFC5322] name-addr or addr-spec, or a URL like "ssh://host.example.org".

As simple text strings, User IDs don't need to be nearly as long as any other packets. An abuse-resistant keystore SHOULD reject any user ID packet larger than 1024 octets.

3.3. Drop or Standardize Unhashed Subpackets

[RFC4880] signature packets contain an "unhashed" block of subpackets. These subpackets are not covered by any cryptographic signature, so they are ripe for abuse.

An abuse-resistant keysetore SHOULD strip out all unhashed subpackets.

Note that some certifications only identify the issuer of the certification by an unhashed Issuer ID subpacket. If a certification's hashed subpacket section has no Issuer ID or Issuer

Fingerprint (see [I-D.ietf-openpgp-rfc4880bis]) subpacket, then an abuse-resistant keystore that has cryptographically validated the certification SHOULD make the unhashed subpackets contain only a single subpacket. That subpacket should be of type Issuer Fingerprint, and should contain the fingerprint of the issuer.

A special exception may be made for unhashed subpackets in a third-party certification that contain attestations from the certificate's primary key as described in Section 6.

3.4. Drop User Attributes

Due to size concerns, some abuse-resistant keystores MAY choose to ignore user attribute packets entirely, as well as any certifications that cover them.

3.5. Drop Non-exportable Certifications

An abuse-resistant keystore MUST NOT accept any certification that has the "Exportable Certification" subpacket present and set to 0. While most keystore clients will not upload these "local" certifications anyway, a reasonable public keystore that wants to minimize data has no business storing or distributing these certifications.

3.6. Accept Only Cryptographically-verifiable Certifications

An abuse-resistant keystore that is capable of doing cryptographic validation MAY decide to reject certifications that it cannot cryptographically validate.

This may mean that the keystore rejects some packets while it is unaware of the public key of the issuer of the packet.

3.7. Accept Only Profiled Certifications

An aggressively abuse-resistant keystore MAY decide to only accept certifications that meet a specific profile. For example, it MAY reject certifications with unknown subpacket types, unknown notations, or certain combinations of subpackets. This can help to minimize the amount of room for garbage data uploads.

Any abuse-resistant keystore that adopts such a strict posture should clearly document what its expected certificate profile is, and should have a plan for how to extend the profile if new types of certification appear that it wants to be able to distribute.

4. Contextual Mitigations

The following mitigations may cause some packets to be dropped after the keystore receives new information, or as time passes. This is entirely reasonable for some keystores, but it may be surprising for any keystore that expects to be append-only (for example, some keyserver synchronization techniques may expect this property to hold).

Note also that many of these mitigations depend on cryptographic validation.

A keystore that needs to be append-only, or which cannot perform cryptographic validation MAY omit these mitigations.

Note that [GnuPG] anticipates some of these suggestions with its "clean" subcommand, which is documented as:

> Compact (by removing all signatures except the selfsig) any user ID that is no longer usable (e.g. revoked, or expired). Then, remove any signatures that are not usable by the trust calculations. Specifically, this removes any signature that does not validate, any signature that is superseded by a later signature, revoked signatures, and signatures issued by keys that are not present on the keyring.

4.1. Drop Superseded Signatures

An abuse-resistant keystore SHOULD drop all signature packets that are explicitly superseded. For example, there's no reason to retain or distribute a self-sig by key K over User ID U from 2017 if the keystore have a cryptographically-valid self-sig over <K,U> from 2019.

Note that this covers both certifications and signatures over subkeys, as both of these kinds of signature packets may be superseded.

Getting this right requires a nuanced understanding of subtleties in [RFC4880] related to timing and revocation.

4.2. Drop Expired Signatures

If a signature packet is known to only be valid in the past, there is no reason to distribute it further. An abuse-resistant keystore with access to a functionally real-time clock SHOULD drop all

certifications and subkey signature packets with an expiration date in the past.

Note that this assumes that the keystore and its clients all have roughly-synchronized clocks. If that is not the case, then there will be many other problems!

4.3. Drop Dangling User IDs, User Attributes, and Subkeys

If enough signature packets are dropped, it's possible that some of the things that those signature packets cover are no longer valid.

An abuse-resistant keystore which has dropped all certifications that cover a User ID SHOULD also drop the User ID packet.

Note that a User ID that becomes invalid due to revocation MUST NOT be dropped, because the User ID's revocation signature itself remains valid, and needs to be distributed.

A primary key with no User IDs and no subkeys and no revocations MAY itself also be removed from distribution, though note that the removal of a primary key may make it impossible to cryptographically validate other certifications held by the keystore.

4.4. Drop All Other Elements of a Directly-Revoked Certificate

If the primary key of a certiifcate is revoked via a direct key signature, an abuse-resistant keystore SHOULD drop all the rest of the associated data (user IDs, user attributes, and subkeys, and all attendant certifications and subkey signatures). This defends against an adversary who compromises a primary key and tries to flood the certificate to hide the revocation.

Note that the direct key revocation signature MUST NOT be dropped.

In the event that an abuse-resistant keystore is flooded with direct key revocation signatures, it should retain the strongest, earliest revocation.

In particular, if any of the revocation signatures has a "Reason for Revocation" of "Key material has been compromised", the keystore SHOULD retain the earliest such revocation signature (by signature creation date).

If none have "Key material has been compromised", but some have "No reason specified", or lack a "Reason for Revocation" entirely, then the keystore SHOULD retain the earliest such revocation signature.

Otherwise, the abuse-resistant keystore SHOULD retain the earliest direct key revocation signature it has seen.

If any of the date comparisons results in a tie between two revocation signatures of the same severity, an abuse-resistant keystore SHOULD retain the signature that sorts earliest based on a binary string comparison of the signature packet itself.

4.5. Implicit Expiration Date

A particularly aggressive abuse-resistant keystore MAY choose an implicit expiration date for all signature packets. For example, a signature packet that claims no expiration could be treated by the keystore as expiring 3 years after issuance.

FIXME: it's not clear what should happen with signature packets marked with an explicit expiration that is longer than implicit maximum. Should it be capped to the implicit date, or accepted?

Warning: This idea is pretty radical, and it's not clear what it would do to an ecosystem that depends on such a keystore. It probably needs more thinking.

5. First-party-only Keystores

In addition to all of the mitigations above, some keystores may resist abuse by declining to carry third-party certifications entirely.

A first-party-only keystore _only_ accepts and distributes cryptographically-valid first-party certifications. Given a primary key that the keystore understands, it will only attach user IDs that have a valid self-sig, and will only accept and re-distribute subkeys that are also cryptographically valid (including requiring cross-sigs for signing-capable subkeys as recommended in [RFC4880]).

This effectively solves the problem of abusive bloating attacks on any certificate, because the only party who can make a certificate overly large is the holder of the secret corresponding to the primary key itself.

However, first-party-only keystores also introduce new problems, for those people who rely on the keystore for discovery of third-party certifications. Section 6 attempts to address this lack.

6. First-party-attested Third-party Certifications

We can augment a first-party-only keystore to allow it to distribute third-party certifications as long as the first-party has signed off on the specific third-party certification.

An abuse-resistant keystore SHOULD only accept a third-party certification if it meets the following criteria:

- The third-party certification MUST be cryptographically valid. Note that this means that the keystore needs to know the primary key for the issuer of the third-party certification.

- The third-party certification MUST have an unhashed subpacket of type Embedded Signature, the contents of which we'll call the "attestation". This attestation is from the certificate's primary key over the third-party certification itself, as detailed in the steps below:

- The attestation MUST be an OpenPGP signature packet of type 0x50 (Third-Party Confirmation signature)

- The attestation MUST contain a notation subpacket

- The attestation MUST contain a hashed "Issuer Fingerprint" subpacket with the fingerprint of the primary key of the certificate in question.

- The attestation MUST NOT be marked as non-exportable.

- The attestation MUST contain a hashed Notation subpacket with the name "ksok", and an empty (0-octet) value.

- The attestation MUST contain a hashed "Signature Target" subpacket with "public-key algorithm" that matches the public-key algorithm of the third-party certification.

- The attestation's hashed "Signature Target" subpacket MUST use a reasonably strong hash algorithm (as of this writing, any [RFC4880] hash algorithm except MD5, SHA1, or RIPEMD160), and MUST have a hash value equal to the hash over the third-party certification with all unhashed subpackets removed.

- The attestation MUST be cryptographically valid, verifiable by the primary key of the certificate in question.

What this means is that a third-party certificate will only be accepted/distributed by the keystore if:

- the keystore knows about both the first- and third-parties.

- the third-party has made the identity assertion

- the first-party has confirmed that they're OK with the third-party certification being distributed by any keystore.

FIXME: it's not clear whether the "ksok" notification is necessary - it's in place to avoid some accidental confusion with any other use of the Third-Party Confirmation signature packet type, but the author does not know of any such use that might collide.

6.1. Key Server Preferences "No-modify"

[RFC4880] section 5.2.3.17 ("Key Server Preferences") defines a "No-modify" bit. That bit has never been respected by any keyserver implementation that the author is aware of. This section effectively asks an abuse-resistant keystore to treat that bit as always set, whether it is present in the certificate or not.

6.2. Client Interactions

The multi-stage layer of creating such an attestation (certificate creation by the first-party, certification by the third-party, attestation by the first-party, then handoff to the keystore) may represent a usability obstacle to a user who needs a third-party-certified OpenPGP certificate.

No current OpenPGP client can easily create the attestions described in this section. More implementation work needs to be done to make it easy (and understandable) for a user to perform this kind of attestation.

7. Side Effects and Ecosystem Impacts

7.1. Designated Revoker

A first-party-only keystore might decline to distribute revocations made by the designated revoker. This is a risk to certificate-holder who depend on this mechanism. Perhaps this document should be amended to include these

7.2. Certification-capable Subkeys

Much of this discussion assumes that primary keys are the only certification-capable keys in the OpenPGP ecosystem. Some proposals have been put forward that assume that subkeys can be marked as certification-capable. If subkeys are certification-capable, then

much of the reasoning in this draft becomes much more complex, as subkeys themselves can be revoked by their primary key without invalidating the key material itself. That is, a subkey can be both valid (in one context) and invalid (in another context) at the same time. So questions about what data can be dropped are much fuzzier.

The author of this draft recommends _not_ considering any subkeys to be certification-capable to avoid this headache.

8. Security Considerations

These mitigations defend individual OpenPGP certificates against bloating attacks. They collectively reduce the amount of data that such a keystore needs to track over time, but given the near-infinite space of possible OpenPGP keys that can be generated, the keystore in aggregate can still be made to grow without bound. This document proposes no clear measures to defend against such a denial of service attack against the keystore itself.

Section 7.1 describes a potentially

TODO (more security considerations)

9. Privacy Considerations

Public OpenPGP keystores often distribute names or e-mail addresses of people. Some people do not want their names or e-mail addresses distributed in a public keystore, or may change their minds about it at some point. Append-only keystores are particularly problematic in that regard. The mitigation in Section 4.4 can help such users strip their details from keys that they control. However, if an OpenPGP certificate with their details is uploaded to a keystore, but is not under their control, it's unclear what mechanisms can be used to remove the certificate that couldn't also be exploited to take down an otherwise valid certificate.

Third-party certifications effectively map out some sort of social graph. While the certifications basically only assert a binding between user IDs, the parties those user IDs represent in the real world, and cryptographic key material, those connections may be potentially sensitive, and users may not want to see these maps built.

TODO (more privacy considerations)

10. User Considerations

Section 6.2 describes some outstanding work that needs to be done to help users understand how to produce and distribute a third-party-certified OpenPGP certificate to an abuse-resistant keystore.

11. IANA Considerations

This document asks IANA to register the "ksok" notation name in the OpenPGP Notation IETF namespace, with a reference to this document, as defined in Section 6.

12. Document Considerations

[RFC Editor: please remove this section before publication]

This document is currently edited as markdown. Minor editorial changes can be suggested via merge requests at https://gitlab.com/dkg/draft-openpgp-abuse-resistant-keystore or by e-mail to the author. Please direct all significant commentary to the public IETF OpenPGP mailing list: openpgp@ietf.org

13. References

13.1. Normative References

[I-D.ietf-openpgp-rfc4880bis]
Koch, W., carlson, b., Tse, R., and D. Atkins, "OpenPGP Message Format", draft-ietf-openpgp-rfc4880bis-06 (work in progress), November 2018.

[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/info/rfc2119>.

[RFC4880] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", RFC 4880, DOI 10.17487/RFC4880, November 2007, <https://www.rfc-editor.org/info/rfc4880>.

[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/info/rfc8174>.

13.2. Informative References

[GnuPG] Koch, W., "Using the GNU Privacy Guard", n.d., <https://www.gnupg.org/documentation/manuals/gnupg.pdf>.

[I-D.koch-openpgp-webkey-service] Koch, W., "OpenPGP Web Key Directory", draft-koch-openpgp-webkey-service-07 (work in progress), November 2018.

[I-D.shaw-openpgp-hkp] Shaw, D., "The OpenPGP HTTP Keyserver Protocol (HKP)", draft-shaw-openpgp-hkp-00 (work in progress), March 2003.

[MAILVELOPE-KEYSERVER] Oberndoerfer, T., "Mailvelope Keyserver", n.d., <https://github.com/mailvelope/keyserver/>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/info/rfc5322>.

[RFC7929] Wouters, P., "DNS-Based Authentication of Named Entities (DANE) Bindings for OpenPGP", RFC 7929, DOI 10.17487/RFC7929, August 2016, <https://www.rfc-editor.org/info/rfc7929>.

[SKS] Pennock, P., "SKS Keyserver Documentation", March 2018, <https://bitbucket.org/skskeyserver/sks-keyserver/wiki/Home>.

13.3. URIs

[1] mailto:alice@example.org

[2] mailto:alice@example.org

Author's Address

Daniel Kahn Gillmor
American Civil Liberties Union
125 Broad St.
New York, NY 10004
USA

Email: dkg@fifthhorseman.net