        Distributing OpenPGP Keys with Signed Keylist Subscriptions
                         draft-mccain-keylist-01

Abstract

   This document specifies a system by which an OpenPGP client may
   subscribe to an organization's keylist to keep its internal keystore
   up-to-date. Ensuring that all members of an organization have their
   colleagues' most recent PGP public keys is critical to maintaining
   operational security. Without the most recent keys and a source of
   trust for those keys (as this document specifies), users must
   manually update and sign each others keys -- a system that is
   untenable in larger organizations. This document proposes a
   experimental format for the keylist file as well as requirements for
   clients who wish to implement this experimental keylist subscription
   functionality.

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 February 22, 2019.

Copyright Notice

Table of Contents

[1](#). Introduction

   This document specifies a system by which clients may subscribe to
   cryptographically signed keylists. This system allows for seamless
   key rotation across entire organizations and enhances operational
   security. To enable cross-client compatibility, this document
   provides a experimental format for the keylist, its cryptographic
   verification, and the method by which it is retreived by the client.
   The user interface by which a client provides this functionality to
   the user is out of scope, as is the process by which the client
   retrieves public keys. Other non-security-related implementation
   details are also out of scope.

1.1. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in [RFC2119] .

1.2. Terminology

This document uses the terms "OpenPGP", "public key", "private key",
"signature", and "fingerprint" as defined by OpenPGP Message Format
[RFC4880] .

The term "keylist" is defined as a list of OpenPGP public keys
identified by their fingerprints and accessible via a URI. The exact
format of this data is specified in Section 3 .

An "authority key" is defined as the OpenPGP secret key used to sign
a particular keylist. Every keylist has a corresponding authority
key, and every authority key has at least one corresponding keylist.
A single authority key SHOULD NOT be used to sign multiple keylists.

To be "subscribed" to a keylist means that a program will retreive
that keylist on a regular interval. After retrieval, that program
will perform an update to an internal OpenPGP keystore.

A "client" is a program that allows the user to subscribe to
keylists. A client may be an OpenPGP client itself or a separate
program that interfaces with an OpenPGP client to update its
keystore.

1.3. Note to Readers

RFC Editor: please remove this section prior to publication.

Development of this Internet draft takes place on GitHub at Keylist-
RFC [1].

A mailing list is available for discussion at Keylists mailing list
[2].

2. Functions and Procedures

As new keys are created and other keys are revoked, it is critical
that all members of an organization have the most recent set of keys
available on their computers. Keylists enable organizations to
publish a directory of OpenPGP keys that clients can use to keep
their internal keystores up-to-date.

2.1. Subscribing to Keylists

A single client may subscribe to any number of keylists. When a client first subscribes to a keylist, it SHOULD update or import every key present in the keylist into its local keystore. Keylist subscriptions SHOULD be persistent --that is, they should be permanently stored by the client to enable future automatic updates.

To subscribe to a keylist, the client must be aware of the keylist URI (defined in [RFC3986]), the keylist's signature URI, and the fingerprint of the authority key used to sign the keylist. The protocol used to retrieve thbe keylist and its signature SHOULD be HTTPS (see [RFC2818]), however other implementation are possible. A client implementing keylist functionality MUST support the retrieval of keylists and signatures over HTTPS. All other protocols are OPTIONAL.

A client MUST NOT employ a trust-on-first-use model for determining the fingerprint of the authority key; it must be explicitly provided by the user.

The process by which the client stores its keylist subscriptions is out of scope, as is the means by which subscription functionality is exposed to the end-user.

2.2. Periodic Updates

The primary purpose of keylists is to enable periodic updates of OpenPGP clients' internal keystores. We RECOMMEND that clients provide a default refresh interval of less than one day, however we also RECOMMEND that clients allow the user to select this interval. The exact time at which updates are performed is not critical.

To perform an update, the client MUST perform the following steps on each keylist to which it is subscribed. The steps SHOULD be performed in the given order.

1. Obtain a current copy of the keylist from its URI.

2. Obtain a current copy of the keylist's signature data from its URI.

3. Using the keylist and the keylist's signature, cryptographically verify that the keylist was signed using the authority key. If the signature does not verify, the client MUST abort the update of this keylist and SHOULD alert the user. The client SHOULD NOT abort the update of other keylists to which it is subscribed, unless they too fail signature verification.

4. Validate the format of the keylist according to [Section 3](#) . If
    the keylist is in an invalid format, the client MUST abort the
    update this keylist and SHOULD alert the user.

5. For each fingerprint listed in the keyfile, if a copy of the
    associated public key is not present in the client's local
    keystore, retrieve it from a keyserver. If it is already present
    and not revoked, refresh it from a keyserver. If it is present
    and revoked, ignore it. The method by which keys are retrieved
    and updated is out of scope.

[2.3](#). Cryptographic Verification of Keylists

To ensure authenticity of a keylist during an update, the client MUST
verify that the keylist's data matches its cryptographic signature,
and that the public key used to verify the signature matches the
authority key fingerprint given by the user.

For enhanced security, it is RECOMMENDED that keylist operators sign
each public key listed in their keylist with the authority private
key. This way, an organization can have an internal trust
relationship without requiring members of the organization to certify
each other's public keys.

[3](#). Data Element Formats

The following are definitions of the data types we will be creating
to support this new feature set.

[3.1](#). Keylist

The keylist MUST be encoded in UTF-8 [[RFC3629](#)] . Each line MUST begin
either with a comment, a public key fingerprint, or whitespace. A
comment is defined as a string of characters between a hash symbol
(#, U+0023) and a newline or an end of file (EOF). The keylist
SHOULD end with a newline. The fingerprint MUST be the full
40-character hexadecimal public key fingerprint, as defined in
OpenPGP Message Format [[RFC4880](#)] . Space characters (' ', U+0020) MAY
be included anywhere in the fingerprint. Lines SHOULD NOT exceed 128
characters in length.

It is RECOMMENDED that keylist maintainers describe each key using a
comment, for example:

1326 CB16 ... DDBF 52A1 # Miles' Key

To extract the public key fingerprints from a keylist, a client
SHOULD perform the following steps, in order:

   1. Strip the keylist of all comments, as defined above, including
       the preceding hash symbol but excluding the trailing newline.

   2. Strip the keylist of all non-breaking whitespace.

   Performing these steps will result in one public key fingerprint per
   line.

3.2. Signature

   The signature file MUST be an ASCII-armored 'detached signature' of
   the keylist file, as defined in OpenPGP Message Format [RFC4880] .

4. In Practice

   GPG Sync, an open source program created by one of the authors,
   implements this experimental standard. GPG Sync is used by First
   Look Media and the Freedom of the Press Foundation to keep OpenPGP
   keys in sync across their organizations, as well as to publish their
   employee's OpenPGP keys to the world. These organizations
   collectively employ more than 200 people and have used the system
   described in this document successfully for multiple years.

   GPG Sync's existing code can be found at
   <https://github.com/firstlookmedia/gpgsync>

   First Look Media's keylist file can be found at
   <https://github.com/firstlookmedia/gpgsync-firstlook-fingerprints>

5. Security Considerations

5.1. Security Benefits

   The keylist subscription functionality defined in this document
   provide a number of security benefits, including:

   o The ability for new keys to be quickly distributed across an
      organization.

   o It removes the complexity of key distribution from end users,
      allowing them to focus on the content of their communications
      rather than on key management.

   o The ability for an organization to prevent the spread of falsely
      attributed keys by centralizing the public key discovery process
      within their organization.

5.2. Security Drawbacks

   There is a situation in which keylist subscriptions could pose a
   potential security threat. If the both the authority key and the
   keylist distribution system were to be compromised, it would be
   possible for an attacker to distribute false keys. We believe,
   however, that the security benefits of this system strongly outweigh
   the drawbacks.

6. IANA Considerations

   This document has no actions for IANA.

7. References

7.1. Normative References

   [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>.

   [RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818,
              DOI 10.17487/RFC2818, May 2000,
              <https://www.rfc-editor.org/info/rfc2818>.

   [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO
              10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November
              2003, <https://www.rfc-editor.org/info/rfc3629>.

   [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform
              Resource Identifier (URI): Generic Syntax", STD 66,
              RFC 3986, DOI 10.17487/RFC3986, January 2005,
              <https://www.rfc-editor.org/info/rfc3986>.

   [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>.

7.2. URIs

   [1] https://github.com/firstlookmedia/keylist-rfc

   [2] https://www.freelists.org/list/keylists

Authors' Addresses

    R. Miles McCain
    First Look Media

    Email: ietf@sendmiles.email
    URI: https://rmrm.io

    Micah Lee
    The Intercept

    Email: micah.lee@theintercept.com
    URI: https://micahflee.com/

    Nat Welch
    First Look Media

    Email: nat.welch@firstlook.media
    URI: https://natwelch.com