pretty Easy privacy (pEp): Contact and Channel Authentication through Handshake
draft-marques-pep-handshake-05

Abstract

   In interpersonal messaging, end-to-end encryption means for public
   key distribution and verification of its authenticity are needed; the
   latter to prevent man-in-the-middle (MITM) attacks.

   This document proposes a new method to easily verify a public key is
   authentic by a Handshake process that allows users to easily
   authenticate their communication channel. The new method is targeted
   to Opportunistic Security scenarios and is already implemented in
   several applications of pretty Easy privacy (pEp).

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

In interpersonal messaging end-to-end encryption means for public key
distribution and verification of its authenticity are needed.

Examples for key distribution include:

o Exchange public keys out-of-band before encrypted communication

o Use of centralized public key stores (e.g., OpenPGP Key Servers)

o Ship public keys in-band when communicating

To prevent man-in-the-middle (MITM) attacks, additionally the
authenticity of a public key needs to be verified. Methods to
authenticate public keys of peers include, e.g., to verify digital

signatures of public keys (which may be signed in a hierarchical or
flat manner, e.g., by a Web of Trust (WoT)), to compare the public
key's fingerprints via a suitable independent channel, or to scan a
QR mapping of the fingerprint (cf. Section 2).

This document proposes a new method to verify the authenticity of
public keys by a Handshake process that allows users to easily verify
their communication channel. Fingerprints of the involved peers are
combined and mapped to (common) Trustwords [I-D.birk-pep-trustwords].
The successful manual comparison of these Trustwords is used to
consider the communication channel as trusted.

The proposed method is already implemented and used in applications
of pretty Easy privacy (pEp) [I-D.birk-pep]. This document is
targeted to applications based on the pEp framework and Opportunistic
Security [RFC7435]. However, it may be also used in other
applications as suitable.

Note: The pEp framework [I-D.birk-pep] proposes to automatize the use
of end-to-end encryption for Internet users of email and other
messaging applications and introduces methods to easily allow
authentication.

1.1. Requirements Language

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. Terms

The following terms are defined for the scope of this document:

o Trustwords: A scalar-to-word representation of 16-bit numbers (0
   to 65535) to natural language words. When doing a Handshake,
   peers are shown combined Trustwords of both public keys involved
   to ease the comparison. [I-D.birk-pep-trustwords]

o Trust On First Use (TOFU): cf. [RFC7435], which states: "In a
   protocol, TOFU calls for accepting and storing a public key or
   credential associated with an asserted identity, without
   authenticating that assertion. Subsequent communication that is
   authenticated using the cached key or credential is secure against
   an MiTM attack, if such an attack did not succeed during the
   vulnerable initial communication."

o Man-in-the-middle (MITM) attack: cf. [RFC4949], which states: "A
   form of active wiretapping attack in which the attacker intercepts
   and selectively modifies communicated data to masquerade as one or
   more of the entities involved in a communication association."

2. Problem Statement

To secure a communication channel, in public key cryptography each
involved peer needs a key pair. Its public key needs to be shared to
other peers by some means. However, the key obtained by the receiver
may have been substituted or tampered with to allow for re-encryption
attacks. To prevent such man-in-the-middle (MITM) attacks, an
important step is to verify the authenticity of a public key
obtained.

2.1. Use Cases

Such a verification process is useful in at least two scenarios:

o Verify channels to peers, e.g., to make sure opportunistically
   exchanged keys for end-to-end encryption are authentic.

o Verify channels between own devices (in multi-device contexts),
   e.g., for the purpose of importing and synchronizing keys among
   different devices belonging to the same user (cf.
   [I-D.pep-keysync]). This scenario is comparable to Bluetooth
   pairing before starting data transfers.

2.2. Existing Solutions

Current methods to authenticate public keys of peers include:

o Digitally signed public keys are verified by a chain of trust.
   Two trust models are common in today's implementations.

   * Signing is carried out hierarchically, e.g. in a Public Key
      Infrastructure (PKI) [RFC5280], in which case the verification
      is based on a chain of trust with a Trust Anchor (TA) at the
      root.

   * Signing of public keys is done in a flat manner (by a Web of
      Trust), e.g., key signing in OpenPGP [RFC4880], where users
      sign the public keys of other users. Verification may be based
      on transitive trust.

o Peers are expected to directly compare the public key's
   fingerprints by any suitable independent channel - e.g, by phone

or with a face-to-face meeting. This method is often used in
OpenPGP [RFC4880] contexts.

o The public keys' fingerprints are mapped into a QR code, which is
   expected to be scanned between the peers when they happen to meet
   face-to-face. This method is, e.g., used in the chat application
   Threema [threema].

o The public keys' fingerprints are mapped into numerical codes
   which decimal digits only (so-called "safety numbers"), which
   makes the strings the humans need to compare easier in respect to
   hexadecimal numbers, but longer and thus nevertheless cumbersome.
   This method is, e.g., used in the chat application Signal
   [signal].

Some of the methods can even be used in conjunction with Trustwords
[I-D.birk-pep-trustwords] or the PGP Word list [PGP.wl].

None of the existing solutions meets all requirements set up by pEp
[I-D.birk-pep], e.g.:

o Easy solution that can be handled easily by ordinary users

o In case privacy and security contradict each other, privacy is
   always preferred over security (e.g., the Web of Trust contradicts
   privacy)

o No central entities

Most of today's systems lack easy ways for users to authenticate
their communication channel. Some methods leak private data (e.g.,
their social graph) or depend on central entities. Thus, none of
today's systems fulfills all of the pEp requirements (cf. above).

3. The pEp Handshake Proposal

In pretty Easy privacy (pEp), the proposed approach for peers to
authenticate each other is to engage in the pEp Handshake.

In current pEp implementations (cf. Section 7.2), the same kinds of
keys as in OpenPGP are used. Such keys include a fingerprint as
cryptographic hash over the public key. This fingerprint is normally
represented in a hexadecimal form, consisting of ten 4-digit
hexadecimal blocks, e.g.:

   8E31 EF52 1D47 5183 3E9D EADC 0FFE E7A5 7E5B AD19

Each block may also be represented in decimal numbers from 0 to 65535 or in other numerical forms, e.g.

o Hexadecimal: 8E31

o Decimal: 36401

o Binary: 1000111000110001

3.1. Verification Process

In the pEp Handshake the fingerprints of any two peers involved are combined and displayed in form of Trustwords [I-D.birk-pep-trustwords] for easy comparison by the involved parties.

The default verification process involves three steps:

1. Combining fingerprints by XOR function

    Any two peers' fingerprints are combined bit-by-bit using the Exclusive-OR (XOR) function resulting in the Combined Fingerprint (CFP).

2. Mapping result to Trustwords:

    The CFP is then mapped to 16-bit Trustwords (i.e., every 4-digit hexadecimal block is mapped to a given Trustword) resulting in the Trustword Mapping (TWM).

3. Verify Trustwords over independent channel

    The resulting Trustwords (TWM) are compared and verified over an independent channel, e.g., a phone line. If this step was successful, the channel can be marked as authenticated.

Note: In prior implementations of pEp the fingerprints of any two peers were concatenated. While this has the advantage that the own identity's Trustwords can be printed on a business card (like with fingerprints) or on contact sites or in signature texts of emails, this at the same time has the drawback that users might not carefully compare the words as they start to remember and recognize their Trustwords in the concatenated mapping. The avoid this undesired training effect, Trustwords for any peer-to-peer combination shall (very likely) differ.

3.1.1. **Short, Long and Full Trustword Mapping**

   The more an ordinary user needs to contribute to a process, the less
   likely a user will carry out all steps carefully. In particular, it
   was observed that a simple (manual) comparison of OpenPGP
   fingerprints is rarely carried out to the full extent, i.e., mostly
   only parts of the fingerprint are compared, if at all.

   For usability reasons and to create incentives for people to actually
   carry out a Handshake (while maintaining an certain level of
   entropy), pEp allows for different entropy levels, i.e.:

   1. Full Trustword Mapping (F-TWM) MUST represent the maximum entropy
       achievable by the mapping. This means all Trustwords of a TWM
       MUST be displayed and compared.

       E.g., the fingerprint

          F482 E952 2F48 618B 01BC 31DC 5428 D7FA ACDC 3F13

       is mapped to

          dog house brother town fat bath school banana kite task

   2. Long Trustword Mapping (L-TWM) requires a number of Trustwords
       that MUST retain at least 128 bits of entropy. Thus, L-TWM
       results into at least eight Trustwords to be compared by the
       user.

       E.g., the fingerprint

          F482 E952 2F48 618B 01BC 31DC 5428 D7FA ACDC [3F13]

       is mapped to

          dog house brother town fat bath school banana kite [remaining
          Trustword(s) omitted]

   3. Short Trustword Mapping (S-TWM) requires a number of Trustwords
       that MUST retain at least 64 bits of entropy. Thus, S-TWM
       results into at least four Trustwords to be compared by the user.

       E.g., the fingerprint

          F482 E952 2F48 618B 01BC [31DC 5428 D7FA ACDC 3F13]

       is mapped to

           dog house brother town fat [remaining Trustwords omitted]

[3.1.2](). Display Modes

The pEp Handshake has three display modes for the verification
process. All of the following modes MUST be implemented:

1. S-TWM mode (default)

    By default the S-TWM SHOULD be displayed to the user for
    comparison and verification. An easy way to switch to L-TWM mode
    MUST be implemented. An easy way to switch to fingerprint mode
    (see below) SHOULD be implemented. An easy way to switch to
    F-TWM mode MAY be implemented

2. L-TWM mode

    There are situations, where S-TWM is not sufficient (e.g.,
    communication parties that are more likely under attack), in
    which the L-TWM MAY be displayed to the user by default. An easy
    way to switch to F-TWM mode MUST be implemented. An easy way to
    switch to fingerprint mode (see below) SHOULD be implemented. An
    easy way to switch to S-TWM mode MAY be implemented

3. F-TWM mode

    The full F-TWM MUST be implemented too, to address high risk
    scenarios. An easy way to switch to fingerprint mode (see below)
    SHOULD be implemented. Easy ways to switch to L-TWM or S-TWM
    mode MAY be implemented.

4. Fingerprint mode (fallback)

    To retain compatibility to existing OpenPGP users (that know
    nothing about Trustwords), the fingerprint mode, a fallback to
    compare the original fingerprints (usually in hexadecimal form)
    MAY be used. An easy way to switch to a least one of the TWM
    modes MUST be implemented.

If the verification process was successful, the user confirms it,
e.g. by setting a check mark. Once the user has confirmed it, the
Privacy Status [I-D.marques-pep-rating] for this channel MUST be
updated accordingly.

4. Security Considerations

   A (global) adversary can pre-generate all Trustwords any two users
   expect to compare and try to engage in MITM attacks which fit - it
   MUST NOT be assumed public keys and thus fingerprints to be something
   to stay secret, especially as in pEp public keys are aggressively
   distributed to all peers. Also similar Trustwords can be generated,
   which spelled on the phone might sound very similar.

5. Privacy Considerations

   [[TODO]]

6. IANA Considerations

   This document has no actions for IANA.

7. Implementation Status

7.1. Introduction

   This section records the status of known implementations of the
   protocol defined by this specification at the time of posting of this
   Internet-Draft, and is based on a proposal described in [RFC7942].
   The description of implementations in this section is intended to
   assist the IETF in its decision processes in progressing drafts to
   RFCs. Please note that the listing of any individual implementation
   here does not imply endorsement by the IETF. Furthermore, no effort
   has been spent to verify the information presented here that was
   supplied by IETF contributors. This is not intended as, and must not
   be construed to be, a catalog of available implementations or their
   features. Readers are advised to note that other implementations may
   exist.

   According to [RFC7942], "[...] this will allow reviewers and working
   groups to assign due consideration to documents that have the benefit
   of running code, which may serve as evidence of valuable
   experimentation and feedback that have made the implemented protocols
   more mature. It is up to the individual working groups to use this
   information as they see fit."

7.2. Current software implementing pEp

   The following software implementing the pEp protocols (to varying
   degrees) already exists:

   o pEp for Outlook as add-on for Microsoft Outlook, release
      [SRC.pepforoutlook]

o pEp for iOS (implemented in a new MUA), release [SRC.pepforios]

o pEp for Android (based on a fork of the K9 MUA), release
   [SRC.pepforandroid]

o pEp for Thunderbird as a new add-on for Thunderbird, beta
   [SRC.pepforthunderbird]

o Enigmail/pEp as add-on for Mozilla Thunderbird, release (will be
   discontinued early 2021) [SRC.enigmailpep]

pEp for Android, iOS, Outlook and Thunderbird are provided by pEp
Security, a commercial entity specializing in end-user software
implementing pEp while Enigmail/pEp is pursued as community project,
supported by the pEp Foundation.

All software is available as Free Software and published also in
source form.

Handshake is already implemented in all platforms listed above.

8. Acknowledgments

Special thanks to Volker Birk and Leon Schumacher who developed the
original concept of the pEp Handshake.

This work was initially created by pEp Foundation, and then reviewed
and extended with funding by the Internet Society's Beyond the Net
Programme on standardizing pEp. [ISOC.bnet]

9. References

9.1. Normative References

[I-D.birk-pep]
           Birk, V., Marques, H., and B. Hoeneisen, "pretty Easy
           privacy (pEp): Privacy by Default", draft-birk-pep-05
           (work in progress), November 2019.

[I-D.birk-pep-trustwords]
           Hoeneisen, B. and H. Marques, "IANA Registration of
           Trustword Lists: Guide, Template and IANA Considerations",
           draft-birk-pep-trustwords-05 (work in progress), January
           2020.

   [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>.

   [RFC4949] Shirey, R., "Internet Security Glossary, Version 2",
              FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007,
              <https://www.rfc-editor.org/info/rfc4949>.

   [RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection
              Most of the Time", RFC 7435, DOI 10.17487/RFC7435,
              December 2014, <https://www.rfc-editor.org/info/rfc7435>.

9.2. Informative References

   [I-D.marques-pep-rating]
              Marques, H. and B. Hoeneisen, "pretty Easy privacy (pEp):
              Mapping of Privacy Rating", draft-marques-pep-rating-03
              (work in progress), January 2020.

   [I-D.pep-keysync]
              Birk, V., Hoeneisen, B., and K. Bristol, "pretty Easy
              privacy (pEp): Key Synchronization Protocol (KeySync)",
              draft-pep-keysync-00 (work in progress), November 2019.

   [ISOC.bnet]
              Simao, I., "Beyond the Net. 12 Innovative Projects
              Selected for Beyond the Net Funding. Implementing Privacy
              via Mass Encryption: Standardizing pretty Easy privacy's
              protocols", June 2017, <https://www.internetsociety.org/
              blog/2017/06/12-innovative-projects-selected-for-beyond-
              the-net-funding/>.

   [PGP.wl]   "PGP word list", November 2017,
              <https://en.wikipedia.org/w/
              index.php?title=PGP_word_list&oldid=749481933>.

   [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>.

   [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S.,
              Housley, R., and W. Polk, "Internet X.509 Public Key
              Infrastructure Certificate and Certificate Revocation List
              (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008,
              <https://www.rfc-editor.org/info/rfc5280>.

   [RFC7942]   Sheffer, Y. and A. Farrel, "Improving Awareness of Running
               Code: The Implementation Status Section", BCP 205,
               RFC 7942, DOI 10.17487/RFC7942, July 2016,
               <https://www.rfc-editor.org/info/rfc7942>.

   [signal]   "Signal", n.d., <https://signal.org/>.

   [SRC.enigmailpep]
               "Source code for Enigmail/pEp", July 2020,
               <https://enigmail.net/index.php/en/download/source-code>.

   [SRC.pepforandroid]
               "Source code for pEp for Android", July 2020,
               <https://pep-security.lu/gitlab/android/pep>.

   [SRC.pepforios]
               "Source code for pEp for iOS", July 2020,
               <https://pep-security.ch/dev/repos/pEp_for_iOS/>.

   [SRC.pepforoutlook]
               "Source code for pEp for Outlook", July 2020,
               <https://pep-security.lu/dev/repos/pEp_for_Outlook/>.

   [SRC.pepforthunderbird]
               "Source code for pEp for Thunderbird", July 2020,
               <https://pep-security.lu/dev/repos/pEp_for_Thunderbird/>.

   [threema]   "Threema - Seriously secure messaging", n.d.,
               <https://threema.ch>.

Appendix A. Document Changelog

   [[RFC Editor: This section is to be removed before publication]]

   o draft-marques-pep-handshake-05:

      * Typos and update references

   o draft-marques-pep-handshake-04:

      * Updated terms and references

   o draft-marques-pep-handshake-03:

      * Updated terms and references

   o draft-marques-pep-handshake-02:

       * Update Sections "Display modes" and "Short, Long and Full
          Trustword Mapping"

       * Add Privacy and IANA Considerations sections

       * Minor editorial changes

    o draft-marques-pep-handshake-01:

       * Fix references

       * Rewrite Sections "Display modes" and "Short, Long and Full
          Trustword Mapping"

       * Add reason why not to concatenate and map fingerprints instead

       * Minor editorial changes

    o draft-marques-pep-handshake-00:

       * Initial version

Appendix B. Open Issues

    [[RFC Editor: This section should be empty and is to be removed
    before publication]]

    o Add description for further processes to change the trust level,
       e.g., to remove trust or even mistrust a peer and alike.

Authors' Addresses

    Hernani Marques
    pEp Foundation
    Oberer Graben 4
    CH-8400 Winterthur
    Switzerland

    Email: hernani.marques@pep.foundation
    URI: https://pep.foundation/

Bernie Hoeneisen
pEp Foundation
Oberer Graben 4
CH-8400 Winterthur
Switzerland

Email: bernie.hoeneisen@pep.foundation
URI: https://pep.foundation/