Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 19 June 2023

H. Marques
B. Hoeneisen
pEp Foundation

16 December 2022

pretty Easy privacy (pEp): Contact and Channel Authentication through Handshake
draft-pep-handshake-00

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

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 19 June 2023.

Copyright Notice

Table of Contents

1. Introduction

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

Examples for key distribution include:

* Exchange public keys out-band before starting encrypted communications

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

* 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.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.pep-general]. 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.pep-general] 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", "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. Terms

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

* Trustwords: A representation of 16-bit natural numbers (0 to 65535) as natural language words: For each natural language a fixed number-to-word map can be defined as convention and registered with IANA. Trustwords are generated from the combined public key fingerprints of a both communication partners. Trustwords are used for verification and establishment of trust (for the respective keys and communication partners). [I-D.pep-trustwords]

* 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."

* 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."

  Note: Historically, MITM has stood for '_Man_-in-the-middle'. However, to indicate that the entity in the middle is not always a human attacker, MITM can also stand for 'Machine-in-the-middle' or 'Meddler-in-the-middle'.

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:

* Verify channels to peers, e.g., to make sure opportunistically (in-band) exchanged keys for end-to-end encryption are authentic.

* 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:

* 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.

* 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.

* 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].

* 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.pep-trustwords] or the PGP Word list [PGP.wl].

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

* Easy solution that can be handled easily by ordinary users, also for users which are physically distant from each other

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

* No central entities to be used

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 6.2), 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.

* Hexadecimal: 8E31

* Decimal: 36401

* 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.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 a 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.pep-rating] for this channel MUST be updated accordingly.

4. Security Considerations

4.1. Pre-Generation of all Trustwords

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.

Using time- or memory-intensive hash algorithms to create Trustwords of any two fingerprints could be used to make extensive pre-generation attacks more expensive.

4.2. Wrong Comparision of Trustwords

It might happen that users comparing Trustwords-similarly as it happens to fingerprint comparisons-only compare the first (or, in any case, to less) Trustwords, thus having way too less entropy in place.

5. IANA Considerations

This document has no actions for IANA.

6. Implementation Status

6.1. 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."

6.2. Current software implementations of pEp

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

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

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

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

* pEp for Thunderbird as an add-on for Thunderbird, release [SRC.pepforthunderbird]

Note: The former community project Enigmail/pEp as add-on for Thunderbird was discontinued and replaced by pEp's own add-on for Thunderbird [SRC.pepforthunderbird] in 2021.

pEp for Android, iOS, Outlook and Thunderbird are provided by pEp Security, a commercial entity specializing in end-user pEp implementations.

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

Handshake is already implemented in all platforms listed above.

7. 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]

8. References

8.1. Normative References

[I-D.pep-general]
Birk, V., Marques, H., and B. Hoeneisen, "pretty Easy privacy (pEp): Privacy by Default", Work in Progress, Internet-Draft, draft-pep-general-01, 21 October 2022, <https://www.ietf.org/archive/id/draft-pep-general-01.txt>.

[I-D.pep-trustwords]
Hoeneisen, B. and H. Marques, "IANA Registration of Trustword Lists: Guide, Template and IANA Considerations", Work in Progress, Internet-Draft, draft-pep-trustwords-00, 16 December 2022, <https://www.ietf.org/archive/id/draft-pep-trustwords-00.txt>.

[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>.

[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>.

8.2. Informative References

[I-D.pep-keysync]
Birk, V., Hoeneisen, B., and K. Bristol, "pretty Easy privacy (pEp): Key Synchronization Protocol (KeySync)", Work in Progress, Internet-Draft, draft-pep-keysync-02, 13 July 2020, <https://www.ietf.org/archive/id/draft-pep-keysync-02.txt>.

[I-D.pep-rating]
Marques, H. and B. Hoeneisen, "pretty Easy privacy (pEp): Mapping of Privacy Rating", Work in Progress, Internet-Draft, draft-pep-rating-00, 16 December 2022, <https://www.ietf.org/archive/id/draft-pep-rating-00.txt>.

[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.pepforandroid]
"Source code for pEp for Android", December 2022, <https://pep-security.lu/gitlab/android/pep>.

[SRC.pepforios]
"Source code for pEp for iOS", December 2022, <https://pep-security.lu/gitlab/iOS/pep4ios>.

[SRC.pepforoutlook]
"Source code for pEp for Outlook", December 2022, <https://pep-security.lu/gitlab/win/pEpForOutlook>.

[SRC.pepforthunderbird]
"Source code for pEp for Thunderbird", December 2022, <https://pep-security.lu/gitlab/thunderbird/pEpForThunderbird>.

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

Appendix A. Document Changelog

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

* draft-pep-handshake-00:
  - Extend Security Considerations
  - Minor changes (mostly editorial)
* draft-marques-pep-handshake-05:
  - Typos and update references
* draft-marques-pep-handshake-04:
  - Updated terms and references
* draft-marques-pep-handshake-03:
  - Updated terms and references
* 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
* 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
* 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]]

* 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/