Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 26, 2018

V. Birk
H. Marques
pEp Foundation
B. Hoeneisen
Ucom.ch
Feb 22, 2018

pretty Easy privacy (pEp): Trustwords concept
draft-birk-pep-trustwords-00

Abstract

In public-key cryptography comparing the public keys' fingerprints of the communication partners involved is vital to ensure that there is no man-in-the-middle (MITM) attack on the communication channel. Fingerprints normally consist of a chain of hexadecimal chars. However, comparing hexadecimal strings is often impractical for regular users and prone to misunderstandings.

To mitigate these challenges, this memo proposes the comparision of trustwords as opposed to hexadecimal strings. Trustwords are common words in a natural language (e.g., English) to which the hexidecimal strings are mapped to. This makes the verification process more natural.

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 August 26, 2018.

Table of Contents

1. Introduction

In public-key cryptography comparing the public keys' fingerprints of the communication partners involved is vital to ensure that there is no man-in-the-middle (MITM) attack on the communication channel. Fingerprints normally consist of a chain of hexadecimal chars. However, comparing hexadecimal strings is often impractical for regular users and prone to misunderstandings.

To mitigate these challenges, this memo proposes the comparision of trustwords as opposed to hexadecimal strings. Trustwords are common words in a natural language (e.g., English) to which the hexidecimal strings are mapped to. This makes the verification process more natural.

Trustwords are used to achieve easy contact verification in pEp's proposition of Privacy by Default [pEp] for end-to-end encryption situations after the peers have exchanged public keys opportunistically.

Trustwords may also be used for purposes other than contact verification.

2. Terms

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].

3. The Concept of Trustword Mapping

4. Example

A fingerprint typically looks like:

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

Its mapping to trustwords looks like:

dog house brother town fat bath school banana kite task

[[Actual mapping for English should be used here and perhaps an example for another language.]]

Instead of the former hexadecimal string, users can compare ten common words of their language.

5. Previous work

The basic concept of trustwork mapping has been already documented in the past, e.g. for use in One-Time Passwords (OTP) [RFC2289] or the PGP Word List ("Pretty Good Privacy word list" [PGPwordlist], also called a biometric word list, to compare fingerprints.

6. Number of Trustwords for a language

Previsous proposals have the shortcoming of a limited number of trustwords and they are usually only available in English. If the number of trustwords is low, a lot of trustworks need to be compared, which make a comparision somewhat cumbersome for users, i.e. leads to degraded usability. To reduce the number of trustwords to compare, 16-bit scalars are mapped to natural language words. Therefore, the size (by number of key--value pairs) of any key--value pair structure MUST be 65536, the keys being the enumeration of the Trustwords (starting at 0) and the values being individual natural language words in the respective language.

However, the number of unique values to be used in a language may be less than 65536. This can be addressed e.g. by using the same value (trustword) for more than one key. However, the entropy of the representation is slightly reduced.

Example. A Trustwords list of just 42000 words still allows for an entropy of log_2(42000) ~= 15.36 bits in 16-bit mappings.

It is for further study, what minimal number of words (or entropy) should be required.

7. The nature of the words

Every Trustwords list SHOULD be cleared from swearwords in order to not offense users. This is a task to be carried out by speakers of the respective natural lnaguage.

8. IANA Considerations

Each natural language requires a different set of trustwords. To allow implementors for identical trustword lists, a IANA registry is to be established. The IANA registration policy according to [RFC8126] will likely be "Expert Review" and "Specification Required".

An IANA registration will contain:

- language code accoring to ISO 639-3
- version number
- list of up to 65536 trustwords

The details of the IANA registry and requirements for the expert to assess the specification are for further study.

9. Security Considerations

There are no special security considerations.

10. Acknowledgements

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. [bnet]

11. References

[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", Jun 2017, <https://www.internetsociety.org/blog/2017/06/12-innovative-projects-selected-for-beyond-the-net-funding/>.

[pEp] pEp Foundation, "pretty Easy privacy (pEp): Privacy by Default [Internet-Draft]", Jan 2018, <https://tools.ietf.org/html/draft-birk-pep-01>.

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

[RFC1760] Haller, N., "The S/KEY One-Time Password System", RFC 1760, DOI 10.17487/RFC1760, February 1995, <https://www.rfc-editor.org/info/rfc1760>.

[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>.

[RFC2289] Haller, N., Metz, C., Nesser, P., and M. Straw, "A One-Time Password System", STD 61, RFC 2289, DOI 10.17487/RFC2289, February 1998, <https://www.rfc-editor.org/info/rfc2289>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

Authors' Addresses

Volker Birk
pEp Foundation

Email: vb@pep-project.org

Hernani Marques
pEp Foundation

Email: hernani.marques@pep.foundation

Bernie Hoeneisen
Ucom Standards Track Solutions GmbH

Email: bernie@ietf.hoeneisen.ch (bernhard.hoeneisen AT ucom.ch)
URI: http://www.ucom.ch/

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 12, 2020

B. Hoeneisen
H. Marques
pEp Foundation

January 09, 2020

IANA Registration of Trustword Lists: Guide, Template and IANA Considerations
draft-birk-pep-trustwords-05

Abstract

This document specifies the IANA Registration Guidelines for Trustwords, describes corresponding registration procedures, and provides a guideline for creating Trustword list specifications.

Trustwords are common words in a natural language (e.g., English), which hexadecimal strings are mapped to. Such a mapping makes verification processes like fingerprint comparisons more practical, and less prone to misunderstandings.

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 July 12, 2020.

Table of Contents

1. Introduction

In public-key cryptography, comparing the respective public key fingerprints for each of the communication partners involved is vital to ensure that there is no Man-in-the-Middle (MITM) attack on the communication channel. These fingerprints normally consist of a chain of hexadecimal characters, which are often impractical, cumbersome, and prone to misunderstandings for end-users.

To mitigate these challenges, several systems offer Trustword comparison as an alternative to these hexadecimal strings. Trustwords are common words in a natural language (e.g., English), which these hexadecimal strings are mapped to. Using Trustwords makes verification processes like fingerprint comparisons more natural for users.

For example, in pEp's Privacy by Default proposition [I-D.birk-pep] Trustwords are used to facilitate easy contact verification for end-to-end encryption. Trustword comparison is offered after the peers have opportunistically exchanged public keys. Examples of Trustword lists used by current pEp implementations can be found here in CSV format: https://pep.foundation/dev/repos/pEpEngine/file/tip/db.

In addition to contact verification, Trustwords are also used for other purposes, such as Human-Readable 128-bit Keys [RFC1751], One Time Passwords (OTP) [RFC1760] [RFC2289], SSH host-key verification, VPN server certificate verification, deriving private keys in blockchain applications for cryptocurrencies, and to import or synchronize secret keys across multiple devices owned by a single user [I-D.pep-keysync]. Further ideas include the use of Trustwords for private key recovery in case of loss, contact verification in Extensible Messaging and Presence Protocol (XMPP) [RFC6120], or for X.509 certificate verification in browsers [RFC3647].

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:

- pEp Handshake: The process of one user contacting another over an independent channel in order to verify Trustwords (or fingerprints as a fallback). This can be done in-person or through established verbal communication channels, like a phone call. [I-D.marques-pep-handshake]

- 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. The Concept of Trustword Mapping

2.1. Example

As already discussed, fingerprints normally consist of a string of hexadecimal characters. A typical fingerprint looks like this:

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

Instead of the hexadecimal string, Trustwords allow users to compare ten common words of a language of their choosing. For example, the above fingerprint, mapped to English Trustwords, might appear as:

dog house brother town fat bath school banana kite task

The same fingerprint might appear in German Trustwords as:

klima gelb lappen weg trinken alles kaputt rasen rucksack durch

Note: These examples are for illustration purposes only, and are not derived from any published Trustword list.

2.2. Previous work

The basic concept of Trustword mapping - also known as a biometric word list - for fingerprint comparison is well-documented. Examples of this concept are used with One-Time Passwords (OTP) [RFC1751] [RFC1760] [RFC2289], as well as the PGP Word List ("Pretty Good Privacy word list" [PGP.wl]. Furthermore, cryptocurrencies use a similar concept for deriving private keys [bitcoin.wl].

[[TODO: Explain each previous usage a bit further and synchronize with section Section 1.]]

Regarding today's needs, previous proposals have the following shortcomings:

- Small/limited word lists, which generally result in more words to compare
- Existing word lists are usually only available in English, which limits their usefulness for non-English speakers

Furthermore, there are differences in the basic concept:

- The Trustword concept suggested herein intends to improve usability and security for all users, instead of only the technically-savvy.

- In many use cases, Trustwords are only read (aloud) during the comparison process, rather than being written or typed. For example, two users might compare their respective Trustwords during a phone call. Verbal comparison reduces the need to keep the actual Trustwords short. The use of longer Trustwords increases the entropy within the system, as it allows for a larger dictionary, and thus reduces the likelihood of phonetic collisions.

2.3. Number of Trustwords for a language

If the number of Trustwords in a dictionary is low, shorter parts of the original string (e.g., fingerprint) can be mapped to a single Trustword. Thus, many Trustwords will need to be compared, which results in a potentially cumbersome process for users, and lead to reduced usability.

To reduce the number of Trustwords that need to be compared, pEp's Privacy by Default proposition [I-D.birk-pep] calls for 16-bit scalars to be mapped to natural language words. Therefore, the size (by number of key-value pairs) of any key-value pair structure is 65536. However, the number of unique values to be used in a language may be smaller than this number. This discrepancy can be addressed by using the same value, or Trustword, for more than one key. In such cases, the entropy of the representation is slightly reduced. For example, a Trustword list of 42000 words still allows for an entropy of log_2(42000), which is roughly 15.36 bits in 16-bit mappings. As a consequence such Trustword lists are not bijective.

On the other hand, small Trustword lists allow for Trustwords consisting of words with shorter strings (number of short words per natural language is normally limited), which are easier to use in implementations where Trustwords have to be typed or written, such as in OTP applications.

Note: This specification allows for registration of variable numbers of Trustwords per dictionary.

2.4. Language

Although English is used around the world, the vast majority of the global population is not English-speaking. For an application to be useful to as wide of a user base as possible, localization is essential. Therefore, this specification allows for registration of Trustword lists in different languages.

In applications where two humans are attempting to establish secure communications, it is likely that they share a common language. At

this time, no real-world use cases for Trustword list translation capability have been identified. Because the translation process inherently - and drastically - increases complexity from an IANA registration standpoint, the topic of Trustword translation is beyond the scope of this document.

2.5. The nature of the words

Every Trustword list SHOULD be clear of offensive language (i.e., swear/curse words, slurs, derogatory language, etc.). This process SHOULD be performed by native speakers of each respective language.

3. Security Considerations

There are no specific security considerations.

4. Privacy Considerations

[[TODO]]

5. IANA Considerations

Each natural language requires a different set of Trustwords. To allow implementers for identical Trustword lists, a IANA registry is to be established. The IANA registration policy according to [RFC8126] is "Expert Review" and "Specification Required".

[[Note: Further details of the IANA registry and requirements for the expert to assess the specification are for further study. A similar approach as used in [RFC6117] is likely followed.]]

5.1. Registration Template (XML chunk)

```
<record>
  <languagecode>
    <!--  ISO 639-3 (e.g. eng, deu, ...) -->
  </languagecode>
  <bitsize>
    <!-- How many bits can be mapped with this list
         (e.g. 8, 16, ...) -->
  </bitsize>
  <numberofuniquewords>
    <!-- number of unique words registered
         (e.g. 256, 65536, ...) -->
  </numberofuniquewords>
  <bijective>
    <!-- whether or not the list allows for a two-way-mapping
         (e.g. yes, no) -->
```

```
      </bijective>
      <version>
        <!-- version number within language
             (e.g. b.1.2, n.0.1, ...)  -->
      </version>
      <registrationdocs>
        <!-- Change accordingly -->
        <xref type="rfc" data="rfc2551"/>
      </registrationdocs>
      <requesters>
        <!-- Change accordingly -->
        <xref type="person" data="John_Doe"/>
        <xref type="person" data="Jane_Dale"/>
      </requesters>
      <additionalinfo>
        <paragraph>
          <!-- Text with additional information about
               the Wordlist to be registered -->
        </paragraph>
        <artwork>
          <!-- There can be artwork sections, too -->
        </artwork>
      </additionalinfo>
      <wordlist>
        <!-- Change accordingly -->
        <w0>first</w0>
        <w1>second</w1>
        [...]
        <w65535>last<w65535>
      </wordlist>
    </record>

    <people>
      <person id="John_Doe">
        <name> <!-- Firstname Lastname --> </name>
        <org> <!-- Organization Name --> </org>
        <uri> <!-- mailto: or http: URI --> </uri>
        <updated> <!-- date format YYYY-MM-DD --> </updated>
      </person>
      <!-- repeat person section for each person -->
    </people>
```

Authors of a Wordlist are encouraged to use these XML chunks as a template to create the IANA Registration Template.

5.2. IANA Registration

An IANA registration will contain the fallowing elements:

5.2.1. Language Code (<languagecode>)

The language code follows the ISO 639-3 specification [ISO639], e.g., eng, deu.

[[Note: It is for further study, which of the ISO 639 Specifications is most suitable to address the Trustwords' challenge.]]

Example usage for German:

e.g. <languagecode>deu</languagecode>

5.2.2. Bit Size (<bitsize>)

The bit size is the number of bits that can be mapped with the Wordlist. The number of registered words in a word list MUST be 2 ^ "(<bitsize>)".

Example usage for 16-bit Wordlist:

e.g. <bitsize>16</bitsize>

5.2.3. Number Of Unique Words (<numberofuniquewords>)

The number of unique words that are registered.

e.g. <numberofuniquewords>65536</numberofuniquewords>

5.2.4. Bijectivity (<bijective>)

Whether the registered Wordlist has a one-to-one mapping, meaning the number of unique words registered equals 2 ^ "(<bitsize>)".

Valid content: (yes | no)

e.g. <bijective>yes</bijective>

5.2.5. Version (<version>)

The version of the Wordlist MUST be unique within a language code.

[[Note: Requirements to a "smart" composition of the version number are for further study]]

e.g. <version>b.1.2</version>

5.2.6. Registration Document(s) (<registrationdocs>)

Reference(s) to the Document(s) containing the Wordlist

```
e.g.  <registrationdocs>
        <xref type="rfc" data="rfc4979"/>
      </registrationdocs>

e.g.  <registrationdocs>
        <xref type="rfc" data="rfc8888"/> (obsoleted by RFC 9999)
        <xref type="rfc" data="rfc9999"/>
      </registrationdocs>

e.g.  <registrationdocs>
        [International Telecommunications Union,
        "Wordlist for Foobar application",
        ITU-F Recommendation B.193, Release 73, Mar 2009.]
      </registrationdocs>
```

5.2.7. Requesters (<requesters>)

The persons requesting the registration of the Wordlist. Usually these are the authors of the Wordlist.

```
e.g.  <requesters>
        <xref type="person" data="John_Doe"/>
      </requesters>

      <people>
        <person id="John_Doe">
          <name>John Doe</name>
          <org>Example Inc.</org>
          <uri>mailto:john.doe@example.com</uri>
          <updated>2018-06-20</updated>
        </person>
      </people>
```

Note: If there is more than one requester, there must be one <xref> element per requester in the <requesters> element, and one <person> chunk per requester in the <people> element.

5.2.8. Further Information (<additionalinfo>)

Any other information the authors deem interesting.

```
e.g.  <additionalinfo>
        <paragraph>more info goes here</paragraph>
      </additionalinfo>
```

Note: If there is no such additional information, then the <additionalinfo> element is omitted.

5.2.9. Wordlist (<wordlist>)

The full Wordlist to be registered. The number of words MUST be a power of 2 as specified above. The element names serve as key used for enumeration of the Trustwords (starting at 0) and the elements contains the values being individual natural language words in the respective language.

```
e.g.  <wordlist>
        <w0>first</w0>
        <w1>second</w1>
        [...]
        <w65535>last<w65535>
      </wordlist>

] ]>
```

[[Note: The exact representation of the Wordlist is for further study.]]

6. Acknowledgments

The authors would like to thank the following people who have provided feedback or significant contributions to the development of this document: Andrew Sullivan, Claudio Luck, Daniel Kahn Gilmore, Kelly Bristol, Michael Richardson, Rich Salz, Volker Birk, and Yoav Nir.

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]

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>.

[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>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

7.2. Informative References

[bitcoin.wl] "Seed Phrase", June 2019, <https://en.bitcoin.it/w/index.php?title=Seed_phrase&oldid=66492#Word_Lists>.

[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.marques-pep-handshake] Marques, H. and B. Hoeneisen, "pretty Easy privacy (pEp): Contact and Channel Authentication through Handshake", draft-marques-pep-handshake-04 (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.

[ISO639] "Language codes - ISO 639", n.d., <https://www.iso.org/iso-639-language-codes.html>.

[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>.

[RFC1751] McDonald, D., "A Convention for Human-Readable 128-bit Keys", RFC 1751, DOI 10.17487/RFC1751, December 1994, <https://www.rfc-editor.org/info/rfc1751>.

[RFC1760] Haller, N., "The S/KEY One-Time Password System", RFC 1760, DOI 10.17487/RFC1760, February 1995, <https://www.rfc-editor.org/info/rfc1760>.

[RFC2289] Haller, N., Metz, C., Nesser, P., and M. Straw, "A One-Time Password System", STD 61, RFC 2289, DOI 10.17487/RFC2289, February 1998, <https://www.rfc-editor.org/info/rfc2289>.

[RFC3647] Chokhani, S., Ford, W., Sabett, R., Merrill, C., and S. Wu, "Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework", RFC 3647, DOI 10.17487/RFC3647, November 2003, <https://www.rfc-editor.org/info/rfc3647>.

[RFC6117] Hoeneisen, B., Mayrhofer, A., and J. Livingood, "IANA Registration of Enumservices: Guide, Template, and IANA Considerations", RFC 6117, DOI 10.17487/RFC6117, March 2011, <https://www.rfc-editor.org/info/rfc6117>.

[RFC6120] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, DOI 10.17487/RFC6120, March 2011, <https://www.rfc-editor.org/info/rfc6120>.

Appendix A. IANA XML Template Example

This section contains a non-normative example of the IANA Registration Template XML chunk.

```
    <record>
      <languagecode>lat</languagecode>
      <bitsize>16</bitsize>
      <numberofuniquewords>57337</numberofuniquewords>
      <bijective>no</bijective>
      <version>n.0.1</version>
      <registrationdocs>
        <xref type="rfc" data="rfc2551"/>
      </registrationdocs>
      <requesters>
        <xref type="person" data="Julius_Caesar"/>
      </requesters>
      <additionalinfo>
        <paragraph>
          This Wordlist has been optimized for
          the Roman Standards Process.
        </paragraph>
      </additionalinfo>
      <wordlist>
        <w0>errare</w0>
        <w1>humanum</w1>
        [...]
        <w65535>est<w65535>
      </wordlist>
    </record>

    <people>
      <person id="Julius_Caesar">
        <name>Julius Caesar</name>
        <org>Curia Romana</org>
        <uri>mailto:julius.cesar@example.com</uri>
        <updated>1999-12-31</updated>
      </person>
    </people>
```

Appendix B. Document Changelog

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

- draft-birk-pep-trustwords-05:
  - Update terms and references
- draft-birk-pep-trustwords-04:
  - Add Privacy Considerations section
  - Swapped Security and IANA Consideration Sections

    * Corrected typo in ISO references

    * Updated Introduction, Terms and concept Sections

  o draft-birk-pep-trustwords-03:

    * Update references

    * Minor edits

  o draft-birk-pep-trustwords-02:

    * Minor editorial changes and bug fixes

    * Added more items to Open Issues

    * Add usage example

  o draft-birk-pep-trustwords-01:

    * Included feedback from mailing list and IETF-101 SECDISPATCH WG, e.g.

      + Added more explanatory text / less focused on the main use case

      + Bit size as parameter

    * Explicitly stated translations are out-of-scope for this document

    * Added draft IANA XML Registration template, considerations, explanation and examples

    * Added Changelog to Appendix

    * Added Open Issue section to Appendix

Appendix C. Open Issues

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

- Better explain previous work on Trustwords
- More explanatory text for Trustword use cases, properties and requirements

- Further details of the IANA registry and requirements for the expert to assess the specification
- Decide which ISO language code either 639-1 or 639-3 to use, i.e., ISO-639-1 (e.g., ca, de, en, ...) as currently used in pEp implementations (running code) or ISO-639-3 (eng, deu, ita, ...)
- Adjust exact representation of wordlists
  * e.g. XML, CSV, ...
  * Syntax for non-ASCII letters or language symbols (UTF-8) in Wordlists
- Need for optional entropy value assigned to words, to account for similar phonetics among words in the same wordlist?
- Need for an additional field, to define what a wordlist is optimized for, e.g., "entropy", "minimize word lengths", ...?
- Work out (requirements for) "smart" composition of the version number
- Decide whether in non-bijective Wordlists the redundant words need to be repeated in the IANA Registration
- Register only a hash over the wordlist with IANA?
- Does it make sense to open registrations for other patterns than just words, e.g., images?

Authors' Addresses

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

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

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

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

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 13, 2018

E. Foudil

Y. Shafranovich
Nightwatch Cybersecurity
February 09, 2018

A Method for Web Security Policies
draft-foudil-securitytxt-03

Abstract

When security risks in web services are discovered by independent security researchers who understand the severity of the risk, they often lack the channels to disclose them properly. As a result, security issues may be left unreported. security.txt defines a standard to help organizations describe the process for security researchers to disclose security vulnerabilities securely.

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 http://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 August 13, 2018.

Copyright Notice

Table of Contents

1. Introduction

1.1. Motivation

Many security researchers encounter situations where they are unable to responsibly disclose security issues to companies because there is no course of action laid out. security.txt is designed to help assist

in this process by making it easier for companies to designate the preferred steps for researchers to take when trying to reach out.

As per section 4 of [RFC2142], there is an existing convention of using the <SECURITY@domain> email address for communications regarding security issues. That convention provides only a single, email-based channel of communication for security issues per domain, and does not provide a way for domain owners to publish information about their security disclosure policies.

In this document, we propose a richer, machine-parsable and more extensible way for companies to communicate information about their security disclosure policies, which is not limited to email and also allows for additional features such as encryption.

1.2. Terminology

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

2. Note to Readers

> *Note to the RFC Editor:* Please remove this section prior to publication.

Development of this draft takes place on Github at: https://github.com/securitytxt/security-txt

3. The Specification

security.txt is a text file that SHOULD be located under the /.well-known/ path ("/.well-known/security.txt") [RFC5785] for web properties. If it is not possible to place the security.txt file in the /.well-known/ path or setup a redirect, web-based services MAY place the file in the top-level path as a fall back option. For web-based services, the instructions MUST be accessible via the Hypertext Transfer Protocol [RFC1945] as a resource of Internet Media Type "text/plain" with the default charset parameter set to "utf-8" per section 4.1.3 of [RFC2046]. For file systems and version control repositories a .security.txt file SHOULD be placed in the root directory.

This text file contains multiple directives with different values. The "directive" is the first part of a field all the way up to the colon ("Contact:"). Directives are case-insensitive. The "value" comes after the directive ("https://example.com/security"). A "field" always consists of a directive and a value ("Contact:

https://example.com/security"). A security.txt file can have an unlimited number of fields. It is important to note that you need a separate line for every field. One MUST NOT chain multiple values for a single directive. Everything MUST be in a separate field.

A security.txt file MUST only apply to the domain in the URI used to retrieve it, not to any of its subdomains or parent domains.

```
# The following only applies to example.com.
https://example.com/.well-known/security.txt

# This only applies to subdomain.example.com.
https://subdomain.example.com/.well-known/security.txt

# This security.txt file applies to IPv4 address of 192.0.2.0.
http://192.0.2.0/.well-known/security.txt

# This security.txt file applies to IPv6 address of 2001:db8:8:4::2.
http://[2001:db8:8:4::2]/.well-known/security.txt
```

3.1. Comments

Comments can be added using the # symbol:

```
# This is a comment.
```

You MAY use one or more comments as descriptive text immediately before the field. Parsers can then associate the comments with the respective field.

3.2. Separate Fields

A separate line is required for every new value and field. You MUST NOT chain everything into a single field. Every line MUST end either with a carriage return and line feed characters (CRLF / %x0D %x0A) or just a line feed character (LF / %x0A).

3.3. Contact:

Add an address that researchers MAY use for reporting security issues. The value can be an email address, a phone number and/or a contact page with more information. The "Contact:" directive MUST always be present in a security.txt file. URIs SHOULD be loaded over HTTPS. Security email addresses SHOULD use the conventions defined in section 4 of [RFC2142], but there is no requirement for this directive to be an email address.

The value MUST follow the general syntax described in [RFC3986]. This means that "mailto" and "tel" URI schemes MUST be used when specifying email addresses and telephone numbers.

The precedence is in listed order. The first field is the preferred method of contact. In the example below, the e-mail address is the preferred method of contact.

Contact: mailto:security@example.com
Contact: tel:+1-201-555-0123
Contact: https://example.com/security-contact.html

3.4. Encryption:

This directive allows you to point to an encryption key that you want security researchers to use for encrypted communication. You MUST NOT directly add your key to the field, instead the value of this field MUST be a URI pointing to a location where the key can be retrieved from. If the key is being retrieved from a website, then the key MUST be loaded over HTTPS.

When it comes to verifying the authenticity of the key, it is always the security researcher's responsibility to make sure the key being specified is indeed one they trust. Researchers MUST NOT assume that this key is used to generate the signature file referenced in Section 3.5.

Example of a PGP key available from a web server:

Encryption: https://example.com/pgp-key.txt

Example of a PGP key available from an OPENPGPKEY DNS record under "security@example.com" (as per [RFC7553] and [RFC7929]):

Encryption: dns:5d2d3ceb7abe552344276d47d36._openpgpkey.example.com?type=OPENPGPKEY

3.5. Signature:

In order to ensure the authenticity of the security.txt file one SHOULD use the "Signature:" directive, which allows you to link to an external signature by specifying the full URI where the signature is located as per [RFC3986]. External signature files SHOULD be named "security.txt.sig" and also be placed under the /.well-known/ path. External signature files SHOULD be loaded over HTTPS.

When it comes to verifying the authenticity of the file, it is always the security researcher's responsibility to make sure the key being specified is indeed one they trust.

Here is an example of an external signature file.

Signature: https://example.com/.well-known/security.txt.sig

3.6. Policy:

With the Policy directive, you can link to where your security policy and/or disclosure policy is located. This can help security researchers understand what you are looking for and how to report security vulnerabilities.

Policy: https://example.com/security-policy.html

3.7. Acknowledgments:

This directive allows you to link to a page where security researchers are recognized for their reports. The page SHOULD list individuals or companies that disclosed security vulnerabilities and worked with you to remediate the issue.

Acknowledgments: https://example.com/hall-of-fame.html

Example security acknowledgments page:

We would like to thank the following researchers:

(2017-04-15) Frank Denis - Reflected cross-site scripting
(2017-01-02) Alice Quinn - SQL injection
(2016-12-24) John Buchner - Stored cross-site scripting
(2016-06-10) Anna Richmond - A server configuration issue

3.8. Hiring:

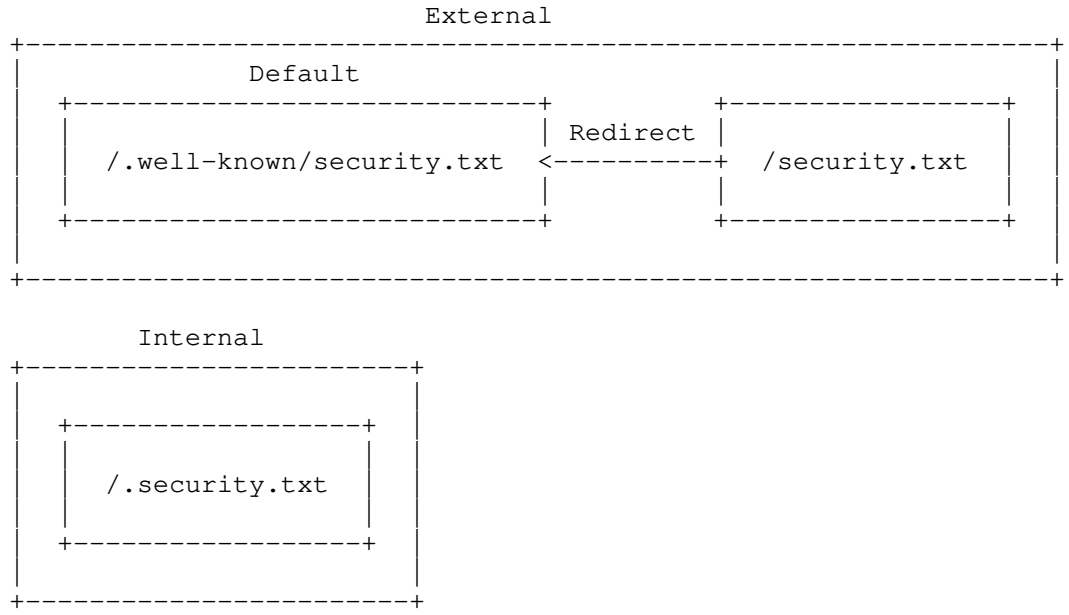
The "Hiring" directive is for linking to the vendor's security-related job positions.

Hiring: https://example.com/jobs.html

3.9. Example

```
# Our security address
Contact: mailto:security@example.com

# Our PGP key
Encryption: https://example.com/pgp-key.txt

# Our security policy
Policy: https://example.com/security-policy.html

# Our security acknowledgments page
Acknowledgments: https://example.com/hall-of-fame.html

# Verify this security.txt file
Signature: https://example.com/.well-known/security.txt.sig
```

4. Location of the security.txt file



4.1. Web-based services

Web-based services SHOULD place the security.txt file under the /.well-known/ path; e.g. https://example.com/.well-known/security.txt. A security.txt file located under the top-level path SHOULD either redirect to the security.txt file under the /.well-known/ path or be used as a fall back.

4.2. Filesystems

File systems SHOULD place the security.txt file under the root directory; e.g., /.security.txt, C:.security.txt.

```
user:/$ l
.security.txt
example-directory-1/
example-directory-2/
example-directory-3/
example-file
```

4.3. Internal hosts

A .security.txt file SHOULD be placed in the root directory of an internal host to trigger incident response.

4.4. Extensibility

Like many other formats and protocols, this format may need to be extended over time to fit the ever-changing landscape of the Internet. Therefore, extensibility is provided via an IANA registry for headers fields as defined in Section 7.2. Any fields registered via that process MUST be considered optional. To encourage extensibility and interoperability, implementors MUST ignore any fields they do not explicitly support.

5. File Format Description

The expected file format of the security.txt file is plain text (MIME type "text/plain") as defined in section 4.1.3 of [RFC2046] and is encoded using UTF-8 [RFC3629] in Net-Unicode form [RFC5198].

The following is an ABNF definition of the security.txt format, using the conventions defined in [RFC5234].

```
body                    = *line (contact-field eol) *line

line                    = *1(field / comment) eol

eol                     = *WSP \[CR\] LF

field                   = contact-field /
                          encryption-field /
                          acknowledgments-field /
                          ext-field

fs                      = ":"

comment                 = "#" *(WSP / VCHAR / %xA0-E007F)

contact-field           = "Contact" fs SP (email / uri / phone)

email                   = <Email address as per {{RFC5322}}>

phone                   = "+" *1(DIGIT / "-" / "(" / ")" / SP)

uri                     = <URI as per {{RFC3986}}>

encryption-field        = "Encryption" fs SP uri

signature-field         = "Signature" fs SP uri

policy-field            = "Policy" fs SP uri

acknowledgments-field   = "Acknowledgments" fs SP uri

hiring-field            = "Hiring" fs SP uri

ext-field               = field-name fs SP unstructured

field-name              = <as per section 3.6.8 of {{RFC5322}}>

unstructured            = <as per section 3.2.5 of {{RFC5322}}>
```

"ext-field" refers to extension fields, which are discussed in Section 4.4

6. Security considerations

Organizations creating security.txt files will need to consider several security-related issues. These include exposure to sensitive information and attacks where limited access to a server could grant the ability to modify the contents of the security.txt file or affect

how it is served. Organizations SHOULD also monitor their security.txt files regularly to detect tampering.

To ensure the authenticity of the security.txt file, organizations SHOULD sign the file and include the signature using the "Signature:" directive.

As stated in Section 3.4 and Section 3.5, both encryption keys and external signature files SHOULD be loaded over HTTPS.

Websites MUST reserve the security.txt namespace to ensure no third-party can create a page with the "security.txt" name.

7. IANA Considerations

example.com is used in this document following the uses indicated in [RFC2606].

192.0.2.0 and 2001:db8:8:4::2 are used in this document following the uses indicated in [RFC6890].

7.1. Well-Known URIs registry

The "Well-Known URIs" registry should be updated with the following additional values (using the template from [RFC5785]):

URI suffix: security.txt

URI suffix: security.txt.sig

Change controller: IETF

Specification document(s): this document

7.2. Registry for security.txt Header Fields

IANA is requested to create the "security.txt Header Fields" registry in accordance with [RFC8126]. This registry will contain header fields for use in security.txt files, defined by this specification.

New registrations or updates MUST be published in accordance with the "Specification Required" guidelines as described in section 4.6 of [RFC8126]. Any new field thus registered is considered optional by this specification unless a new version of this specification is published.

New registrations and updates MUST contain the following information:

1. Name of the field being registered or updated

2. Short description of the field

3. Whether the field can appear more than once

4. The document in which the specification of the field is published

5. New or updated status, which MUST be one of: current: The field is in current use deprecated: The field is in current use, but its use is discouraged historic: The field is no longer in current use

An update may make a notation on an existing registration indicating that a registered field is historical or deprecated if appropriate.

The initial registry contains these values:

Field Name: Acknowledgment
Description: link to page where security researchers are recognized
Multiple Appearances: Yes
Published in: this document
Status: current

Field Name: Contact
Description: contact information to use for reporting security issues
Multiple Appearances: Yes
Published in: this document
Status: current

Field Name: Encryption
Description: link to a key to be used for encrypted communication
Multiple Appearances: Yes
Published in: this document
Status: current

Field Name: Signature
Description: signature used to verify the authenticity of the file
Multiple Appearances: No
Published in: this document
Status: current

Field Name: Policy
Description: link to security policy page
Multiple Appearances: No
Published in: this document
Status: current

8. Contributors

The editors would like to acknowledge the help provided during the development of this document by Tom Hudson, Joel Margolis, Jobert Abma, Gerben Janssen van Doorn, Austin Heap, Justin Calmus, and Casey Ellis.

9. References

9.1. Normative References

[RFC1945] Berners-Lee, T., Fielding, R., and H. Frystyk, "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, DOI 10.17487/RFC1945, May 1996, <https://www.rfc-editor.org/info/rfc1945>.

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, DOI 10.17487/RFC2046, November 1996, <https://www.rfc-editor.org/info/rfc2046>.

[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>.

[RFC2142] Crocker, D., "Mailbox Names for Common Services, Roles and Functions", RFC 2142, DOI 10.17487/RFC2142, May 1997, <https://www.rfc-editor.org/info/rfc2142>.

[RFC2606] Eastlake 3rd, D. and A. Panitz, "Reserved Top Level DNS Names", BCP 32, RFC 2606, DOI 10.17487/RFC2606, June 1999, <https://www.rfc-editor.org/info/rfc2606>.

[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>.

[RFC5198] Klensin, J. and M. Padlipsky, "Unicode Format for Network Interchange", RFC 5198, DOI 10.17487/RFC5198, March 2008, <https://www.rfc-editor.org/info/rfc5198>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

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

[RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known Uniform Resource Identifiers (URIs)", RFC 5785, DOI 10.17487/RFC5785, April 2010, <https://www.rfc-editor.org/info/rfc5785>.

[RFC6890] Cotton, M., Vegoda, L., Bonica, R., Ed., and B. Haberman, "Special-Purpose IP Address Registries", BCP 153, RFC 6890, DOI 10.17487/RFC6890, April 2013, <https://www.rfc-editor.org/info/rfc6890>.

[RFC7553] Faltstrom, P. and O. Kolkman, "The Uniform Resource Identifier (URI) DNS Resource Record", RFC 7553, DOI 10.17487/RFC7553, June 2015, <https://www.rfc-editor.org/info/rfc7553>.

[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>.

9.2. Informative References

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

Appendix A. Note to Readers

> *Note to the RFC Editor:* Please remove this section prior to publication.

Development of this draft takes place on Github at https://github.com/securitytxt/security-txt

Appendix B. Document History

Note to the RFC Editor: Please remove this section prior to publication.

B.1. Since draft-foudil-securitytxt-00

- Moved to use IETF's markdown tools for draft updates
- Added table of contents and a fuller list of references
- Moved file to .well-known URI and added IANA registration (#3)
- Added extensibility with an IANA registry for fields (#34)
- Added text explaining relationship to RFC 2142 / security@ email address (#25)
- Scope expanded to include internal hosts, domains, IP addresses and file systems
- Support for digital signatures added (#19)

The full list of changes can be viewed via the IETF document tracker: https://tools.ietf.org/html/draft-foudil-securitytxt-01

B.2. Since draft-foudil-securitytxt-01

- Added appendix with pointer to Github and document history
- Added external signature file to the well known URI registry (#59)
- Added policy field (#53)
- Added diagram explaining the location of the file on public vs. internal systems
- Added recommendation that external signature files should use HTTPS (#55)
- Added recommendation that organizations should monitor their security.txt files (#14)

The full list of changes can be viewed via the IETF document tracker: https://tools.ietf.org/html/draft-foudil-securitytxt-02

B.3. Since draft-foudil-securitytxt-02

- Use "mailto" and "tel" (#62)
- Fix typo in the "Example" section (#64)
- Clarified that the root directory is a fall back option (#72)
- Defined content-type for the response (#68)
- Clarify the scope of the security.txt file (#69)
- Cleaning up text based on the NITS tools suggestions (#82)
- Added clarification for newline values
- Clarified the encryption field language, added examples of DNS-stored encryption keys (#28 and #94)

Full list of changes can be viewed via the IETF document tracker: https://tools.ietf.org/html/draft-foudil-securitytxt-03

Authors' Addresses

Edwin Foudil

Email: contact@edoverflow.com

Yakov Shafranovich
Nightwatch Cybersecurity

Email: yakov+ietf@nightwatchcybersecurity.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: 24 February 2021

E. Foudil

Y. Shafranovich
Nightwatch Cybersecurity

23 August 2020

A File Format to Aid in Security Vulnerability Disclosure
draft-foudil-securitytxt-10

Abstract

When security vulnerabilities are discovered by researchers, proper reporting channels are often lacking. As a result, vulnerabilities may be left unreported. This document defines a format ("security.txt") to help organizations describe their vulnerability disclosure practices to make it easier for researchers to report vulnerabilities.

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 24 February 2021.

Table of Contents

1. Introduction

1.1. Motivation, Prior Work and Scope

Many security researchers encounter situations where they are unable to report security vulnerabilities to organizations because there are no reporting channels to contact the owner of a particular resource and no information available about the vulnerability disclosure practices of such owner.

As per section 4 of [RFC2142], there is an existing convention of using the <SECURITY@domain> email address for communications regarding security vulnerabilities. That convention provides only a single, email-based channel of communication for security vulnerabilities per domain, and does not provide a way for domain owners to publish information about their security disclosure practices.

There are also contact conventions prescribed for Internet Service Providers (ISPs) in section 2 of [RFC3013], for Computer Security Incident Response Teams (CSIRTs) in section 3.2 of [RFC2350] and for site operators in section 5.2 of [RFC2196]. As per [RFC7485], there is also contact information provided by Regional Internet Registries (RIRs) and domain registries for owners of IP addresses, autonomous system numbers (ASNs), and domain names. However, none of these address the issue of how security researchers can locate contact information and vulnerability disclosure practices for organizations in order to report vulnerabilities.

In this document, we define a richer and more extensible way for organizations to communicate information about their security disclosure practices and ways to contact them. Other details of vulnerability disclosure are outside the scope of this document. Readers are encouraged to consult other documents such as [ISO.29147.2018] or [CERT.CVD].

As per [CERT.CVD], "vulnerability response" refers to reports of product vulnerabilities which is related but distinct from reports of network intrusions and compromised websites ("incident response"). The mechanism defined in this document is intended to be used for the former ("vulnerability response"). If implementors want to utilize this mechanism for incident response, they should be aware of additional security considerations discussed in Section 6.1.

The "security.txt" file is intended to be complementary and not as a substitute or replacement for other public resources maintained by organizations regarding their security disclosure practices.

1.2. Terminology

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. Note to Readers

> *Note to the RFC Editor:* Please remove this section prior to publication.

Development of this draft takes place on Github at: https://github.com/securitytxt/security-txt

3. The Specification

This document defines a text file to be placed in a known location that provides information about the vulnerability disclosure practices of a particular organization. This is intended to help security researchers when disclosing security vulnerabilities.

By convention, the file is named "security.txt".

When made available on HTTP servers, it MUST be placed under the /.well-known/ path (as "/.well-known/security.txt") [RFC8615] of a domain name or IP address. For legacy compatibility, a security.txt file might be placed at the top level path (see Section 4.1). For file systems a "security.txt" file SHOULD be placed in the root directory of the file system.

On HTTP servers, the file MUST be accessed via HTTP 1.0 or a higher version and the "https" scheme (as per [RFC1945] and section 2.7.2 of [RFC7230]). It MUST have a Content-Type of "text/plain" with the default charset parameter set to "utf-8" (as per section 4.1.3 of [RFC2046]).

This text file contains multiple fields with different values. A field contains a "name" which is the first part of a field all the way up to the colon ("Contact:") and follows the syntax defined for "field-name" in section 3.6.8 of [RFC5322]. Field names are case-insensitive (as per section 2.3 of [RFC5234]). The "value" comes after the field name ("https://example.com/security") and follows the syntax defined for "unstructured" in section 3.2.5 of [RFC5322]. The file may also contain blank lines.

A "field" MUST always consist of a name and a value ("Contact: https://example.com/security"). A security.txt file can have an unlimited number of fields. It is important to note that each field MUST appear on its own line. Unless specified otherwise by the field definition, multiple values MUST NOT be chained together for a single field. Unless otherwise indicated in a definition of a particular field, any field MAY appear multiple times.

Implementors should be aware that some of the fields may contain URIs using percent-encoding (as per section 2.1 of [RFC3986]).

3.1. Scope of the File

For HTTP servers, a "security.txt" file MUST only apply to the domain or IP address in the URI used to retrieve it, not to any of its subdomains or parent domains.

A "security.txt" file that is found in a file system MUST only apply to the folder in which it is located and that folder’s subfolders. The file does not apply to any of the folder’s parent or sibling folders.

A "security.txt" file MAY also apply to products and services provided by the organization publishing the file. Implementors SHOULD use the policy directive (as per Section 3.5.7) to provide additional details regarding scope and details of their vulnerability disclosure process.

Some examples appear below:

```
# The following only applies to example.com.
https://example.com/.well-known/security.txt

# This only applies to subdomain.example.com.
https://subdomain.example.com/.well-known/security.txt

# This security.txt file applies to IPv4 address of 192.0.2.0.
https://192.0.2.0/.well-known/security.txt

# This security.txt file applies to IPv6 address of 2001:db8:8:4::2.
https://[2001:db8:8:4::2]/.well-known/security.txt

# This file applies to the /example/folder1 directory and subfolders.
/example/folder1/security.txt
```

3.2. Comments

Any line beginning with the "#" (%x30) symbol MUST be interpreted as a comment. The content of the comment may contain any ASCII or Unicode characters in the %x21-7E and %x80-FFFFF ranges plus the tab (%x09) and space (%x20) characters.

Example:

```
# This is a comment.
```

3.3. Line Separator

Every line MUST end either with a carriage return and line feed characters (CRLF / %x0D %x0A) or just a line feed character (LF / %x0A).

3.4. Digital signature

It is RECOMMENDED that a security.txt file be digitally signed using an OpenPGP cleartext signature as described in section 7 of [RFC4880]. When digital signatures are used, it is also RECOMMENDED that implementors use the "Canonical" field (as per Section 3.5.2), thus allowing the digital signature to authenticate the location of the file.

When it comes to verifying the key used to generate the signature, it is always the security researcher's responsibility to make sure the key being used is indeed one they trust.

3.5. Field Definitions

3.5.1. Acknowledgments

This field indicates a link to a page where security researchers are recognized for their reports. The page being referenced should list individuals or organizations that reported security vulnerabilities and collaborated to remediate them. Organizations should be careful to limit the vulnerability information being published in order to prevent future attacks.

If this field indicates a web URL, then it MUST begin with "https://" (as per section 2.7.2 of [RFC7230]).

Example:

Acknowledgments: https://example.com/hall-of-fame.html

Example security acknowledgments page:

We would like to thank the following researchers:

(2017-04-15) Frank Denis - Reflected cross-site scripting
(2017-01-02) Alice Quinn - SQL injection
(2016-12-24) John Buchner - Stored cross-site scripting
(2016-06-10) Anna Richmond - A server configuration issue

3.5.2. Canonical

This field indicates the canonical URIs where the security.txt file is located, which is usually something like "https://example.com/.well-known/security.txt". If this field indicates a web URL, then it MUST begin with "https://" (as per section 2.7.2 of [RFC7230]). The purpose of this field is to allow a digital signature to be applied to the locations of the "security.txt" file.

Canonical: https://www.example.com/.well-known/security.txt
Canonical: https://someserver.example.com/.well-known/security.txt

3.5.3. Contact

This field indicates an address that researchers should use for reporting security vulnerabilities such as an email address, a phone number and/or a web page with contact information. The "Contact" field MUST always be present in a security.txt file. If this field indicates a web URL, then it MUST begin with "https://" (as per section 2.7.2 of [RFC7230]). Security email addresses should use the conventions defined in section 4 of [RFC2142].

The value MUST follow the URI syntax described in [RFC3986]. This means that "mailto" and "tel" URI schemes must be used when specifying email addresses and telephone numbers, as defined in [RFC6068] and [RFC3966]. When the value of this field is an email address, it is RECOMMENDED that encryption be used (as per Section 3.5.4).

The precedence SHOULD be in listed order. The first field is the preferred method of contact. In the example below, the email address is the preferred method of contact.

Contact: mailto:security@example.com
Contact: mailto:security%2Buri%2Bencoded@example.com
Contact: tel:+1-201-555-0123
Contact: https://example.com/security-contact.html

3.5.4. Encryption

This field indicates an encryption key that security researchers should use for encrypted communication. Keys MUST NOT appear in this field - instead the value of this field MUST be a URI pointing to a location where the key can be retrieved. If this field indicates a web URL, then it MUST begin with "https://" (as per section 2.7.2 of [RFC7230]).

When it comes to verifying the authenticity of the key, it is always the security researcher’s responsibility to make sure the key being specified is indeed one they trust. Researchers must not assume that this key is used to generate the digital signature referenced in Section 3.4.

Example of an OpenPGP key available from a web server:

Encryption: https://example.com/pgp-key.txt

Example of an OpenPGP key available from an OPENPGPKEY DNS record:

Encryption: dns:5d2d37ab76d47d36._openpgpkey.example.com?type=OPENPGPKEY

Example of an OpenPGP key being referenced by its fingerprint:

Encryption: openpgp4fpr:5f2de5521c63a801ab59ccb603d49de44b29100f

3.5.5. Expires

This field indicates the date and time after which the data contained in the "security.txt" file is considered stale and should not be used (as per Section 6.3). The value of this field follows the format defined in section 3.3 of [RFC5322]. It is RECOMMENDED that the value of this field be less than a year into the future to avoid staleness.

This field MUST always be present and MUST NOT appear more than once.

Expires: Thu, 31 Dec 2020 18:37:07 -0800

3.5.6. Hiring

The "Hiring" field is used for linking to the vendor's security-related job positions. If this field indicates a web URL, then it MUST begin with "https://" (as per section 2.7.2 of [RFC7230]).

Hiring: https://example.com/jobs.html

3.5.7. Policy

This field indicates a link to where the vulnerability disclosure policy is located. This can help security researchers understand the organization's vulnerability reporting practices. If this field indicates a web URL, then it MUST begin with "https://" (as per section 2.7.2 of [RFC7230]).

Example:

Policy: https://example.com/disclosure-policy.html

3.5.8. Preferred-Languages

This field can be used to indicate a set of natural languages that are preferred when submitting security reports. This set MAY list multiple values, separated by commas. If this field is included then at least one value MUST be listed. The values within this set are language tags (as defined in [RFC5646]). If this field is absent, security researchers may assume that English is the language to be used (as per section 4.5 of [RFC2277]).

The order in which they appear MUST NOT be interpreted as an indication of priority - rather these MUST be interpreted as all being of equal priority.

This field MUST NOT appear more than once.

Example (English, Spanish and French):

Preferred-Languages: en, es, fr

3.6. Example of an unsigned "security.txt" file

```
# Our security address
Contact: mailto:security@example.com

# Our OpenPGP key
Encryption: https://example.com/pgp-key.txt

# Our security policy
Policy: https://example.com/security-policy.html

# Our security acknowledgments page
Acknowledgments: https://example.com/hall-of-fame.html
```

3.7. Example of a signed "security.txt" file

```
----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA256

# Canonical URL
Canonical: https://example.com/.well-known/security.txt

# Our security address
Contact: mailto:security@example.com

# Our OpenPGP key
Encryption: https://example.com/pgp-key.txt

# Our security policy
Policy: https://example.com/security-policy.html

# Our security acknowledgments page
Acknowledgments: https://example.com/hall-of-fame.html
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v2.2

[signature]
-----END PGP SIGNATURE-----
```

4. Location of the security.txt file

4.1. Web-based services

Web-based services MUST place the security.txt file under the "/.well-known/" path; e.g. https://example.com/.well-known/security.txt as per [RFC8615]. For legacy compatibility, a security.txt file might be placed at the top-level path or redirect (as per section 6.4 of [RFC7231]) to the security.txt file under the "/.well-known/" path. If a "security.txt" file is present in both locations, the one in the "/.well-known/" path MUST be used.

Retrieval of "security.txt" files and resources indicated within such files may result in a redirect (as per section 6.4 of [RFC7231]). Researchers should perform additional triage (as per Section 6.2) to make sure these redirects are not malicious or pointing to resources controlled by an attacker.

4.2. Filesystems

File systems SHOULD place the "security.txt" file under the root directory; e.g., "/security.txt", "C:\security.txt".

Example file system:

```
/example-directory-1/
/example-directory-2/
/example-directory-3/
/example-file
/security.txt
```

4.3. Extensibility

Like many other formats and protocols, this format may need to be extended over time to fit the ever-changing landscape of the Internet. Therefore, extensibility is provided via an IANA registry for fields as defined in Section 7.2. Any fields registered via that process MUST be considered optional. To encourage extensibility and interoperability, implementors MUST ignore any fields they do not explicitly support.

In general, implementors should "be conservative in what you do, be liberal in what you accept from others" (as per [RFC0793]).

5. File Format Description and ABNF Grammar

The expected file format of the security.txt file is plain text (MIME type "text/plain") as defined in section 4.1.3 of [RFC2046] and is encoded using UTF-8 [RFC3629] in Net-Unicode form [RFC5198].

The following is an ABNF definition of the security.txt format, using the conventions defined in [RFC5234].

```
body             =  signed / unsigned

signed           =  sign-header unsigned sign-footer

sign-header      =  < headers and line from section 7 of [RFC4880] >

sign-footer      =  < OpenPGP signature from section 7 of [RFC4880] >

unsigned         =  *line (contact-field eol)
                    *line (expires-field eol)
                    *line [lang-field eol] *line
                    ; order of fields within the file is not important

line             =  [ (field / comment) ] eol

eol              =  *WSP [CR] LF

field            =  ack-field /
                    can-field /
                    contact-field /
                    encryption-field /
                    hiring-field /
                    policy-field /
                    ext-field

fs               =  ":"

comment          =  "#" *(WSP / VCHAR / %x80-FFFFF)

ack-field        =  "Acknowledgments" fs SP uri

can-field        =  "Canonical" fs SP uri

contact-field    =  "Contact" fs SP uri

expires-field    =  "Expires" fs SP date-time

encryption-field =  "Encryption" fs SP uri
```

```
  hiring-field     =  "Hiring" fs SP uri

  lang-field       =  "Preferred-Languages" fs SP lang-values

  policy-field     =  "Policy" fs SP uri

  date-time        =  < imported from section 3.3 of [RFC5322] >

  lang-tag         =  < Language-Tag from section 2.1 of [RFC5646] >

  lang-values      =  lang-tag *(*WSP "," *WSP lang-tag)

  uri              =  < URI as per [RFC3986] >

  ext-field        =  field-name fs SP unstructured

  field-name       =  < imported from section 3.6.8 of [RFC5322] >

  unstructured     =  < imported from section 3.2.5 of [RFC5322] >
```

"ext-field" refers to extension fields, which are discussed in Section 4.3

6. Security Considerations

In addition to the security considerations of [RFC8615], the following considerations apply.

6.1. Compromised Files and Incident Response

An attacker that has compromised a website is able to compromise the "security.txt" file as well or setup a redirect to their own site. This can result in security reports not being received by the organization or sent to the attacker.

To protect against this, organizations should use the "Canonical" field to indicate the locations of the file (as per Section 3.5.2), digitally sign their "security.txt" files (as per Section 3.4), and regularly monitor the file and the referenced resources to detect tampering.

Security researchers should triage the "security.txt" file including verifying the digital signature and checking any available historical records before using the information contained in the file. If the "security.txt" file looks suspicious or compromised, it should not be used.

While it is not recommended, implementors may choose to use the information published within a "security.txt" file for incident response. In such cases, extreme caution should be taken before trusting such information, since it may have been compromised by an attacker. Implementors should use additional methods to verify such data including out of band verification of the PGP signature, DNS-based approaches, etc.

6.2. Redirects

When retrieving the file and any resources referenced in the file, researchers should record any redirects since they can lead to a different domain or IP address controlled by an attacker. Further inspections of such redirects is recommended before using the information contained within the file.

6.3. Incorrect or Stale Information

If information and resources referenced in a "security.txt" file are incorrect or not kept up to date, this can result in security reports not being received by the organization or sent to incorrect contacts, thus exposing possible security issues to third parties. Not having a security.txt file may be preferable to having stale information in this file. Organizations must use the "Expires" field (see Section 3.5.5) to indicate to researchers when the data in the file is no longer valid.

Organizations should ensure that information in this file and any referenced resources such as web pages, email addresses, and telephone numbers are kept current, are accessible, controlled by the organization, and are kept secure.

6.4. Intentionally Malformed Files, Resources and Reports

It is possible for compromised or malicious sites to create files that are extraordinarily large or otherwise malformed in an attempt to discover or exploit weaknesses in parsing code. Implementors should make sure that any such code is robust against large or malformed files and fields and may choose not to parse files larger than 32 KBs, having fields longer than 2,048 characters or containing more than 1,000 lines. The ABNF grammar (as defined in Section 5) can also be used as a way to verify these files.

The same concerns apply to any other resources referenced within security.txt files, as well as any security reports received as a result of publishing this file. Such resources and reports may be hostile, malformed or malicious.

6.5. No Implied Permission for Testing

The presence of a security.txt file might be interpreted by researchers as providing permission to do security testing against that asset. This might result in increased testing against an organization by researchers. On the other hand, a decision not to publish a security.txt file might be interpreted by the organization operating that website to be a way to signal to researchers that permission to test that particular site or project is denied. This might result in pushback against researchers reporting security issues to that organization.

Therefore, implementors shouldn't assume that presence or absence of a "security.txt" file grants or denies permission for security testing. Any such permission may be indicated in the company's vulnerability disclosure policy (as per Section 3.5.7) or a new field (as per Section 4.3).

6.6. Multi-user Environments

In multi-user / multi-tenant environments, it may possible for a user to take over the location of the "security.txt" file. Organizations should reserve the "security.txt" namespace at the root to ensure no third-party can create a page with the "security.txt" AND "/.well-known/security.txt" names.

6.7. Protecting Data in Transit

To protect a "security.txt" file from being tampered with in transit, implementors should use HTTPS (as per [RFC2818]) when serving the file itself and for retrieval of any web URLs referenced in it (except when otherwise noted in this specification). As part of the TLS handshake, implementors should validate the provided X.509 certificate in accordance with [RFC6125] and the following considerations:

* Matching is performed only against the DNS-ID identifiers.

* DNS domain names in server certificates MAY contain the wildcard character '*' as the complete left-most label within the identifier.

The certificate may also be checked for revocation via the Online Certificate Status Protocol (OCSP) [RFC6960], certificate revocation lists (CRLs), or similar mechanisms.

In cases where the "security.txt" file cannot be served via HTTPS (such as a filesystem or localhost) or is being served with an invalid certificate, additional human triage is recommended since the contents may have been modified while in transit.

As an additional layer of protection, it is also recommended that organizations digitally sign their "security.txt" file with OpenPGP (as per Section 3.4). Also, to protect security reports from being tampered with or observed while in transit, organizations should specify encryption keys (as per Section 3.5.4) unless HTTPS is being used for report submission.

However, the determination of validity of such keys is out of scope for this specification. Security researchers need to establish other secure means to verify them.

6.8. Spam and Spurious Reports

Similar to concerns in [RFC2142], denial of service attacks via spam reports would become easier once a "security.txt" file is published by an organization. In addition, there is an increased likelihood of reports being sent in an automated fashion and/or as result of automated scans without human triage. Attackers can also use this file as a way to spam unrelated third parties by listing their resources and/or contact information.

Organizations need to weigh the advantages of publishing this file versus the possible disadvantages and increased resources required to triage security reports.

Security researchers should review all information within the "security.txt" file before submitting reports in an automated fashion or as resulting from automated scans.

7. IANA Considerations

example.com is used in this document following the uses indicated in [RFC2606].

192.0.2.0 and 2001:db8:8:4::2 are used in this document following the uses indicated in [RFC6890].

Implementors should be aware that any resources referenced within a security.txt file MUST NOT point to the Well-Known URIs namespace unless they are registered with IANA (as per [RFC8615]).

7.1. Well-Known URIs registry

The "Well-Known URIs" registry should be updated with the following additional values (using the template from [RFC8615]):

URI suffix: security.txt

Change controller: IETF

Specification document(s): this document

Status: permanent

7.2. Registry for security.txt Fields

IANA is requested to create the "security.txt Fields" registry in accordance with [RFC8126]. This registry will contain fields for use in security.txt files, defined by this specification.

New registrations or updates MUST be published in accordance with the "Expert Review" guidelines as described in sections 4.5 and 5 of [RFC8126]. Any new field thus registered is considered optional by this specification unless a new version of this specification is published.

Designated Experts are expected to check whether a proposed registration or update makes sense in the context of industry accepted vulnerability disclosure processes such as [ISO.29147.2018] and [CERT.CVD], and provides value to organizations and researchers using this format.

New registrations and updates MUST contain the following information:

1. Name of the field being registered or updated

2. Short description of the field

3. Whether the field can appear more than once

4. The document in which the specification of the field is published (if available)

5. New or updated status, which MUST be one of:

    * current: The field is in current use

    * deprecated: The field is in current use, but its use is discouraged

* historic: The field is no longer in current use

6. Change controller

An update may make a notation on an existing registration indicating that a registered field is historical or deprecated if appropriate.

The initial registry contains these values:

Field Name: Acknowledgments
Description: link to page where security researchers are recognized
Multiple Appearances: Yes
Published in: this document
Status: current
Change controller: IESG

Field Name: Canonical
Description: canonical URL for this file
Multiple Appearances: No
Published in: this document
Status: current
Change controller: IESG

Field Name: Contact
Description: contact information to use for reporting vulnerabilities
Multiple Appearances: Yes
Published in: this document
Status: current
Change controller: IESG

Field Name: Expires
Description: date and time after which this file is considered stale
Multiple Appearances: No
Published in: this document
Status: current
Change controller: IESG

Field Name: Encryption
Description: link to a key to be used for encrypted communication
Multiple Appearances: Yes
Published in: this document
Status: current
Change controller: IESG

Field Name: Hiring
Description: link to the vendor’s security-related job positions
Multiple Appearances: Yes
Published in: this document

Status: current
Change controller: IESG

Field Name: Policy
Description: link to security policy page
Multiple Appearances: Yes
Published in: this document
Status: current
Change controller: IESG

Field Name: Preferred-Languages
Description: list of preferred languages for security reports
Multiple Appearances: No
Published in: this document
Status: current
Change controller: IESG

8. Contributors

The authors would like to acknowledge the help provided during the development of this document by Tom Hudson, Jobert Abma, Gerben Janssen van Doorn, Austin Heap, Stephane Bortzmeyer, Max Smith, Eduardo Vela, and Krzysztof Kotowicz.

The authors would also like to acknowledge the feedback provided by multiple members of IETF's LAST CALL, SAAG, and SECDISPATCH lists.

Yakov would like to also thank L.T.S. (for everything).

9. References

9.1. Normative References

[RFC1945] Berners-Lee, T., Fielding, R., and H. Frystyk, "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, DOI 10.17487/RFC1945, May 1996, <https://www.rfc-editor.org/info/rfc1945>.

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, DOI 10.17487/RFC2046, November 1996, <https://www.rfc-editor.org/info/rfc2046>.

[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>.

[RFC2142] Crocker, D., "Mailbox Names for Common Services, Roles and Functions", RFC 2142, DOI 10.17487/RFC2142, May 1997, <https://www.rfc-editor.org/info/rfc2142>.

[RFC2277] Alvestrand, H., "IETF Policy on Character Sets and Languages", BCP 18, RFC 2277, DOI 10.17487/RFC2277, January 1998, <https://www.rfc-editor.org/info/rfc2277>.

[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>.

[RFC3966] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966, DOI 10.17487/RFC3966, December 2004, <https://www.rfc-editor.org/info/rfc3966>.

[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>.

[RFC5198] Klensin, J. and M. Padlipsky, "Unicode Format for Network Interchange", RFC 5198, DOI 10.17487/RFC5198, March 2008, <https://www.rfc-editor.org/info/rfc5198>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

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

[RFC5646] Phillips, A., Ed. and M. Davis, Ed., "Tags for Identifying Languages", BCP 47, RFC 5646, DOI 10.17487/RFC5646, September 2009, <https://www.rfc-editor.org/info/rfc5646>.

[RFC6068] Duerst, M., Masinter, L., and J. Zawinski, "The 'mailto' URI Scheme", RFC 6068, DOI 10.17487/RFC6068, October 2010, <https://www.rfc-editor.org/info/rfc6068>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <https://www.rfc-editor.org/info/rfc6125>.

[RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, DOI 10.17487/RFC6960, June 2013, <https://www.rfc-editor.org/info/rfc6960>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

[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>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

9.2. Informative References

[CERT.CVD] Software Engineering Institute, Carnegie Mellon University, "The CERT Guide to Coordinated Vulnerability Disclosure (CMU/SEI-2017-SR-022)", 2017.

[ISO.29147.2018] International Organization for Standardization (ISO), "ISO/IEC 29147:2018, Information technology -- Security techniques -- Vulnerability disclosure", 2018.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

[RFC2196] Fraser, B., "Site Security Handbook", FYI 8, RFC 2196, DOI 10.17487/RFC2196, September 1997, <https://www.rfc-editor.org/info/rfc2196>.

[RFC2350] Brownlee, N. and E. Guttman, "Expectations for Computer Security Incident Response", BCP 21, RFC 2350, DOI 10.17487/RFC2350, June 1998, <https://www.rfc-editor.org/info/rfc2350>.

[RFC2606] Eastlake 3rd, D. and A. Panitz, "Reserved Top Level DNS Names", BCP 32, RFC 2606, DOI 10.17487/RFC2606, June 1999, <https://www.rfc-editor.org/info/rfc2606>.

[RFC3013] Killalea, T., "Recommended Internet Service Provider Security Services and Procedures", BCP 46, RFC 3013, DOI 10.17487/RFC3013, November 2000, <https://www.rfc-editor.org/info/rfc3013>.

[RFC6890] Cotton, M., Vegoda, L., Bonica, R., Ed., and B. Haberman, "Special-Purpose IP Address Registries", BCP 153, RFC 6890, DOI 10.17487/RFC6890, April 2013, <https://www.rfc-editor.org/info/rfc6890>.

[RFC7485] Zhou, L., Kong, N., Shen, S., Sheng, S., and A. Servin, "Inventory and Analysis of WHOIS Registration Objects", RFC 7485, DOI 10.17487/RFC7485, March 2015, <https://www.rfc-editor.org/info/rfc7485>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

Appendix A. Note to Readers

Note to the RFC Editor: Please remove this section prior to publication.

Development of this draft takes place on Github at https://github.com/securitytxt/security-txt

Appendix B. Document History

Note to the RFC Editor: Please remove this section prior to publication.

B.1. Since draft-foudil-securitytxt-00

* Moved to use IETF's markdown tools for draft updates
* Added table of contents and a fuller list of references
* Moved file to .well-known URI and added IANA registration (#3)
* Added extensibility with an IANA registry for fields (#34)
* Added text explaining relationship to RFC 2142 / security@ email address (#25)
* Scope expanded to include internal hosts, domains, IP addresses and file systems
* Support for digital signatures added (#19)

The full list of changes can be viewed via the IETF document tracker: https://tools.ietf.org/html/draft-foudil-securitytxt-01

B.2. Since draft-foudil-securitytxt-01

* Added appendix with pointer to Github and document history
* Added external signature file to the well known URI registry (#59)
* Added policy field (#53)
* Added diagram explaining the location of the file on public vs. internal systems
* Added recommendation that external signature files should use HTTPS (#55)
* Added recommendation that organizations should monitor their security.txt files (#14)

The full list of changes can be viewed via the IETF document tracker: https://tools.ietf.org/html/draft-foudil-securitytxt-02

B.3. Since draft-foudil-securitytxt-02

* Use "mailto" and "tel" (#62)

* Fix typo in the "Example" section (#64)
* Clarified that the root directory is a fallback option (#72)
* Defined content-type for the response (#68)
* Clarify the scope of the security.txt file (#69)
* Cleaning up text based on the NITS tools suggestions (#82)
* Added clarification for newline values
* Clarified the encryption field language, added examples of DNS-stored encryption keys (#28 and #94)
* Added "Hiring" field

B.4. Since draft-foudil-securitytxt-03

* Added "Hiring" field to the registry section
* Added an encryption example using a PGP fingerprint (#107)
* Added reference to the mailing list (#111)
* Added a section referencing related work (#113)
* Fixes for idnits (#82)
* Changing some references to informative instead of normative
* Adding "Permission" field (#30)
* Fixing remaining ABNF issues (#83)
* Additional editorial changes and edits

B.5. Since draft-foudil-securitytxt-04

* Addressing IETF feedback (#118)
* Case sensitivity clarification (#127)
* Syntax fixes (#133, #135 and #136)
* Removed permission field (#30)

* Removed signature field and switched to inline signatures (#93 and #128)

* Adding canonical field (#100)

* Text and ABNF grammar improvements plus ABNF changes for comments (#123)

* Changed ".security.txt" to "security.txt" to be consistent

B.6. Since draft-foudil-securitytxt-05

* Changing HTTPS to MUST (#55)

* Adding language recommending encryption for email reports (#134)

* Added language handling redirects (#143)

* Expanded security considerations section and fixed typos (#30, #73, #103, #112)

B.7. Since draft-foudil-securitytxt-06

* Fixed ABNF grammar for non-chainable fields (#150)

* Clarified ABNF grammar (#152)

* Clarified redirect logic (#143)

* Clarified comments (#158)

* Updated references and template for well-known URI to RFC 8615

* Fixed nits from the IETF validator

B.8. Since draft-foudil-securitytxt-07

* Addressing AD feedback (#165)

* Fix for ABNF grammar in lang-values (#164)

* Fixing idnits warnings

* Adding guidance for designated experts

B.9. Since draft-foudil-securitytxt-08

* Added language and example regarding URI encoding (#176)
* Add "Expires" field (#181)
* Changed language from "directive" to "field" (#182)
* Addressing last call feedback (#179, #180 and #183)
* Clarifying order of fields (#174)
* Revert comment/field association (#158)

B.10. Since draft-foudil-securitytxt-09

* Adjust ABNF to allow blank lines between directives (#191)
* Make "Expires" field required (#190)
* Adding a warning about the well-known URI namespace (#188)
* Adding scope language around products/services (#185)
* Addressing last call feedback (#189)

Full list of changes can be viewed via the IETF document tracker: https://tools.ietf.org/html/draft-foudil-securitytxt

Authors' Addresses

Edwin Foudil

Email: contact@edoverflow.com

Yakov Shafranovich
Nightwatch Cybersecurity

Email: yakov+ietf@nightwatchcybersecurity.com

INTERNET-DRAFT
Intended Status: Proposed Standard
Expires: 18 June 2018

R. Housley
Vigil Security

18 December 2017

Use of the Hash-based Merkle Tree Signature (MTS) Algorithm in the Cryptographic Message Syntax (CMS) <draft-housley-cms-mts-hash-sig-08>

Abstract

This document specifies the conventions for using the Merkle Tree Signatures (MTS) digital signature algorithm with the Cryptographic Message Syntax (CMS). The MTS algorithm is one form of hash-based digital signature.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

Table of Contents

1. Introduction

This document specifies the conventions for using the Merkle Tree Signatures (MTS) digital signature algorithm with the Cryptographic Message Syntax (CMS) [CMS] signed-data content type. The MTS algorithm is one form of hash-based digital signature that can only be used for a fixed number of signatures. The MTS algorithm is described in [HASHSIG]. The MTS algorithm uses small private and public keys, and it has low computational cost; however, the signatures are quite large.

1.1. ASN.1

CMS values are generated using ASN.1 [ASN1-B], using the Basic Encoding Rules (BER) and the Distinguished Encoding Rules (DER) [ASN1-E].

1.2. Terminology

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 RFC 2119 [KEYWORDS].

2. MTS Digital Signature Algorithm Overview

Merkle Tree Signatures (MTS) are a method for signing a large but fixed number of messages. An MTS system depends on a one-time signature method and a collision-resistant hash function.

This specification makes use of the MTS algorithm specified in [HASHSIG], which is the Leighton and Micali adaptation [LM] of the original Lamport-Diffie-Winternitz-Merkle one-time signature system [M1979][M1987][M1989a][M1989b]. It makes use of the LM-OTS one-time signature scheme and the SHA-256 one-way hash function [SHS].

2.1. Hierarchical Signature System (HSS)

The MTS system specified in [HASHSIG] uses a hierarchy of trees. The Hierarchical N-time Signature System (HSS) allows subordinate trees to be generated when needed by the signer. Otherwise, generation of the entire tree might take weeks or longer.

An HSS signature as specified in specified in [HASHSIG] carries the number of signed public keys (Nspk), followed by that number of signed public keys, followed by the LMS signature as described in Section 2.2. Each signed public key is represented by the hash value at the root of the tree, and the signature over that public key is an LMS signature as described in Section 2.2.

The elements of the HSS signature value for a stand-alone tree can be summarized as:

```
u32str(0) ||
lms_signature_on_message
```

The elements of the HSS signature value for a tree with Nspk levels can be summarized as:

```
u32str(Nspk) ||
lms_signature_on_public_key[0] || public_key[1] ||
lms_signature_on_public_key[1] || public_key[2] ||
   ...
lms_signature_on_public_key[Nspk-2] || public_key[Nspk-1] ||
lms_signature_on_public_key[Nspk-1] || public_key[Nspk] ||
lms_signature_on_message
```

2.2. Leighton-Micali Signature (LMS)

Each tree in the system specified in [HASHSIG] uses the Leighton-Micali Signature (LMS) system. LMS systems have two parameters. The first parameter is the height of the tree, h, which is the number of levels in the tree minus one. The [HASHSIG] specification supports five values for this parameter: h=5; h=10; h=15; h=20; and h=25. Note that there are 2^h leaves in the tree. The second parameter is the number of bytes output by the hash function, m, which the amount of data associated with each node in the tree. The [HASHSIG] specification supports only the SHA-256 hash function [SHS], with m=32.

Five tree sizes are specified in [HASHSIG]:

```
LMS_SHA256_M32_H5;
LMS_SHA256_M32_H10;
LMS_SHA256_M32_H15;
LMS_SHA256_M32_H20; and
LMS_SHA256_M32_H25.
```

An LMS signature consists of four elements: a typecode indicating the particular LMS algorithm, the number of the leaf associated with the LM-OTS signature, an LM-OTS signature as described in Section 2.3, and an array of values that is associated with the path through the tree from the leaf associated with the LM-OTS signature to the root. The array of values contains the siblings of the nodes on the path from the leaf to the root but does not contain the nodes on the path itself. The array for a tree with height h will have h values. The first value is the sibling of the leaf, the next value is the sibling of the parent of the leaf, and so on up the path to the root.

The four elements of the LMS signature value can be summarized as:

```
u32str(q) ||
ots_signature ||
u32str(type) ||
path[0] || path[1] || ... || path[h-1]
```

2.3. Leighton-Micali One-time Signature Algorithm (LM-OTS)

Merkle Tree Signatures (MTS) depend on a one-time signature method. [HASHSIG] specifies the use of the LM-OTS. An LM-OTS has five parameters.

- n - The number of bytes associated with the hash function. [HASHSIG] supports only SHA-256 [SHS], with n=32.

- H - A preimage-resistant hash function that accepts byte strings of any length, and returns an n-byte string.

- w - The width in bits of the Winternitz coefficients. [HASHSIG] supports four values for this parameter: w=1; w=2; w=4; and w=8.

- p - The number of n-byte string elements that make up the LM-OTS signature.

- ls - The number of left-shift bits used in the checksum function, which is defined in Section 4.5 of [HASHSIG].

The values of p and ls are dependent on the choices of the parameters n and w, as described in Appendix A of [HASHSIG].

Four LM-OTS variants are defined in [HASHSIG]:

```
LMOTS_SHA256_N32_W1;
LMOTS_SHA256_N32_W2;
LMOTS_SHA256_N32_W4; and
LMOTS_SHA256_N32_W8.
```

Signing involves the generation of C, an n-byte random value.

The LM-OTS signature value can be summarized as:

```
u32str(type) || C || y[0] || ... || y[p-1]
```

3. Algorithm Identifiers and Parameters

The algorithm identifier for an MTS signature is id-alg-mts-hashsig:

```
id-alg-mts-hashsig  OBJECT IDENTIFIER ::= { iso(1) member-body(2)
      us(840) rsadsi(113549) pkcs(1) pkcs9(9) smime(16) alg(3) 17 }
```

When the id-alg-mts-hashsig algorithm identifier is used for a signature, the AlgorithmIdentifier parameters field MUST be absent (that is, the parameters are not present; the parameters are not set to NULL).

The signature values is a large OCTET STRING. The signature format is designed for easy parsing. Each format includes a counter and type codes that indirectly providing all of the information that is needed to parse the value during signature validation. The first 4 octets of the signature value contains the number of signed public keys (Nspk) in the HSS. The first 4 octets of each LMS signature value contains type code, which tells how to parse the remaining parts of the signature value. The first 4 octets of each LM-OTS signature value contains type code, which tells how to parse the remaining parts of the signature value.

4. Signed-data Conventions

As specified in [CMS], the digital signature is produced from the message digest and the signer's private key. If signed attributes are absent, then the message digest is the hash of the content. If signed attributes are present, then the hash of the content is placed in the message-digest attribute, the set of signed attributes is DER encoded, and the message digest is the hash of the encoded attributes. In summary:

```
IF (signed attributes are absent)
THEN md = Hash(content)
ELSE message-digest attribute = Hash(content);
     md = Hash(DER(SignedAttributes))

Sign(md)
```

When using [HASHSIG], the fields in the SignerInfo are used as follows:

digestAlgorithms SHOULD contain the one-way hash function used to compute the message digest on the eContent value. Since the hash-based signature algorithms all depend on SHA-256, it is strongly RECOMMENDED that SHA-256 also be used to compute the message digest on the content.

Further, the same one-way hash function SHOULD be used to compute the message digest on both the eContent and the signedAttributes value if signedAttributes are present. Again, since the hash-based signature algorithms all depend on SHA-256, it is strongly RECOMMENDED that SHA-256 be used.

signatureAlgorithm MUST contain id-alg-mts-hashsig. The algorithm parameters field MUST be absent.

signature contains the single HSS signature value resulting from the signing operation as specified in [HASHSIG].

5. Security Considerations

5.1. Implementation Security Considerations

Implementations must protect the private keys. Compromise of the private keys may result in the ability to forge signatures. Along with the private key, the implementation must keep track of which leaf nodes in the tree have been used. Loss of integrity of this tracking data can cause an one-time key to be used more than once. As a result, when a private key and the tracking data are stored on non-volatile media or stored in a virtual machine environment, care must be taken to preserve confidentiality and integrity.

An implementation must ensure that a LM-OTS private key is used to generate a signature only one time, and ensure that it cannot be used for any other purpose.

The generation of private keys relies on random numbers. The use of inadequate pseudo-random number generators (PRNGs) to generate these values can result in little or no security. An attacker may find it much easier to reproduce the PRNG environment that produced the keys, searching the resulting small set of possibilities, rather than brute force searching the whole key space. The generation of quality random numbers is difficult. RFC 4086 [RANDOM] offers important guidance in this area.

When computing signatures, the same hash function SHOULD be used for all operations. In this specification, only SHA-256 is used. Using only SHA-256 reduces the number of possible failure points in the signature process.

5.2. Algorithm Security Considerations

At Black Hat USA 2013, some researchers gave a presentation on the current sate of public key cryptography. They said: "Current cryptosystems depend on discrete logarithm and factoring which has

seen some major new developments in the past 6 months" [BH2013]. They encouraged preparation for a day when RSA and DSA cannot be depended upon.

A post-quantum cryptosystem is a system that is secure against quantum computers that have more than a trivial number of quantum bits. It is open to conjecture when it will be feasible to build such a machine. RSA, DSA, and ECDSA are not post-quantum secure.

The LM-OTP one-time signature, LMS, and HSS do not depend on discrete logarithm or factoring, as a result these algorithms are considered to be post-quantum secure.

Today, RSA is often used to digitally sign software updates. This means that the distribution of software updates could be compromised if a significant advance is made in factoring or a quantum computer is invented. The use of MTS signatures to protect software update distribution, perhaps using the format described in [FWPROT], will allow the deployment of software that implements new cryptosystems.

6. IANA Considerations

This document has no actions for IANA.

7. Acknowledgements

Many thanks to Panos Kampanakis, Jim Schaad, and Sean Turner for their careful review and comments.

8. Normative References

[ASN1-B] ITU-T, "Information technology -- Abstract Syntax Notation One (ASN.1): Specification of basic notation", ITU-T Recommendation X.680, 2015.

[ASN1-E] ITU-T, "Information technology -- ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T Recommendation X.690, 2015.

[CMS] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <http://www.rfc-editor.org/info/rfc5652>.

[HASHSIG] McGrew, D., M. Curcio, and S. Fluhrer, "Hash-Based Signatures", Work in progress. <draft-mcgrew-hash-sigs-07>

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

[SHS] National Institute of Standards and Technology (NIST), FIPS Publication 180-3: Secure Hash Standard, October 2008.

9. Informative References

[BH2013] Ptacek, T., T. Ritter, J. Samuel, and A. Stamos, "The Factoring Dead: Preparing for the Cryptopocalypse", August 2013. <https://media.blackhat.com/us-13/us-13-Stamos-The-Factoring-Dead.pdf>

[CMSASN1] Hoffman, P. and J. Schaad, "New ASN.1 Modules for Cryptographic Message Syntax (CMS) and S/MIME", RFC 5911, DOI 10.17487/RFC5911, June 2010, <http://www.rfc-editor.org/info/rfc5911>.

[CMSASN1U] Schaad, J. and S. Turner, "Additional New ASN.1 Modules for the Cryptographic Message Syntax (CMS) and the Public Key Infrastructure Using X.509 (PKIX)", RFC 6268, DOI 10.17487/RFC6268, July 2011, <http://www.rfc-editor.org/info/rfc6268>.

[FWPROT] Housley, R., "Using Cryptographic Message Syntax (CMS) to Protect Firmware Packages", RFC 4108, DOI 10.17487/RFC4108, August 2005, <http://www.rfc-editor.org/info/rfc4108>.

[LM] Leighton, T. and S. Micali, "Large provably fast and secure digital signature schemes from secure hash functions", U.S. Patent 5,432,852, July 1995.

[M1979] Merkle, R., "Secrecy, Authentication, and Public Key Systems", Stanford University Information Systems Laboratory Technical Report 1979-1, 1979.

[M1987] Merkle, R., "A Digital Signature Based on a Conventional Encryption Function", Lecture Notes in Computer Science crypto87, 1988.

[M1989a] Merkle, R., "A Certified Digital Signature", Lecture Notes in Computer Science crypto89, 1990.

[M1989b] Merkle, R., "One Way Hash Functions and DES", Lecture Notes in Computer Science crypto89, 1990.

[PKIXASN1] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, DOI 10.17487/RFC5912, June 2010, <http://www.rfc-editor.org/info/rfc5912>.

[PQC] Bernstein, D., "Introduction to post-quantum cryptography", 2009. <http://www.pqcrypto.org/www.springer.com/cda/content/document/cda_downloaddocument/9783540887010-c1.pdf>

[RANDOM] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <http://www.rfc-editor.org/info/rfc4086>.

Appendix: ASN.1 Module

```
   MTS-HashSig-2013
     { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
       id-smime(16) id-mod(0) id-mod-mts-hashsig-2013(64) }

   DEFINITIONS IMPLICIT TAGS ::= BEGIN

   EXPORTS ALL;

   IMPORTS
     PUBLIC-KEY, SIGNATURE-ALGORITHM, SMIME-CAPS
       FROM AlgorithmInformation-2009  -- RFC 5911 [CMSASN1]
         { iso(1) identified-organization(3) dod(6) internet(1)
           security(5) mechanisms(5) pkix(7) id-mod(0)
           id-mod-algorithmInformation-02(58) }
     mda-sha256
       FROM PKIX1-PSS-OAEP-Algorithms-2009  -- RFC 5912 [PKIXASN1]
         { iso(1) identified-organization(3) dod(6)
           internet(1) security(5) mechanisms(5) pkix(7) id-mod(0)
           id-mod-pkix1-rsa-pkalgs-02(54) } ;

   --
   -- Object Identifiers
   --

   id-alg-mts-hashsig  OBJECT IDENTIFIER ::= { iso(1) member-body(2)
         us(840) rsadsi(113549) pkcs(1) pkcs9(9) smime(16) alg(3) 17 }

   --
   -- Signature Algorithm and Public Key
   --

   sa-MTS-HashSig SIGNATURE-ALGORITHM ::= {
        IDENTIFIER id-alg-mts-hashsig
        PARAMS ARE absent
        HASHES { mda-sha256 }
        PUBLIC-KEYS { pk-MTS-HashSig }
        SMIME-CAPS { IDENTIFIED BY id-alg-mts-hashsig } }

   pk-MTS-HashSig PUBLIC-KEY ::= {
       IDENTIFIER id-alg-mts-hashsig
       KEY MTS-HashSig-PublicKey
       PARAMS ARE absent
       CERT-KEY-USAGE
           { digitalSignature, nonRepudiation, keyCertSign, cRLSign } }

   MTS-HashSig-PublicKey ::= OCTET STRING
```

```
   --
   -- Expand the signature algorithm set used by CMS [CMSASN1U]
   --

   SignatureAlgorithmSet SIGNATURE-ALGORITHM ::=
        { sa-MTS-HashSig, ... }

   --
   -- Expand the S/MIME capabilities set used by CMS [CMSASN1]
   --

   SMimeCaps SMIME-CAPS ::= { sa-MTS-HashSig.&smimeCaps, ... }

   END
```

Author's Address

Russ Housley
Vigil Security, LLC
918 Spring Knoll Drive
Herndon, VA 20170
USA

EMail: housley@vigilsec.com

INTERNET-DRAFT
Intended Status: Proposed Standard
Expires: 1 January 2019

R. Housley
Vigil Security

1 July 2018

Use of the Hash-based Merkle Tree Signature (MTS) Algorithm in the Cryptographic Message Syntax (CMS) <draft-housley-cms-mts-hash-sig-10>

Abstract

This document specifies the conventions for using the Merkle Tree Signatures (MTS) digital signature algorithm with the Cryptographic Message Syntax (CMS). The MTS algorithm is one form of hash-based digital signature.

Status of this Memo

Table of Contents

1. Introduction

This document specifies the conventions for using the Merkle Tree Signatures (MTS) digital signature algorithm with the Cryptographic Message Syntax (CMS) [CMS] signed-data content type. The MTS algorithm is one form of hash-based digital signature that can only be used for a fixed number of signatures. The MTS algorithm is described in [HASHSIG]. The MTS algorithm uses small private and public keys, and it has low computational cost; however, the signatures are quite large.

1.1. ASN.1

CMS values are generated using ASN.1 [ASN1-B], using the Basic Encoding Rules (BER) and the Distinguished Encoding Rules (DER) [ASN1-E].

1.2. Terminology

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. MTS Digital Signature Algorithm Overview

Merkle Tree Signatures (MTS) are a method for signing a large but fixed number of messages. An MTS system depends on a one-time signature method and a collision-resistant hash function.

This specification makes use of the MTS algorithm specified in [HASHSIG], which is the Leighton and Micali adaptation [LM] of the original Lamport-Diffie-Winternitz-Merkle one-time signature system [M1979][M1987][M1989a][M1989b].

As implied by the name, the hash-based signature algorithm depends on a collision-resistant hash function. The hash-based signature algorithm specified in [HASHSIG] currently uses only the SHA-256 one-way hash function [SHS], but it also establishes an IANA registry to permit the registration of additional one-way hash functions in the future.

2.1. Hierarchical Signature System (HSS)

The MTS system specified in [HASHSIG] uses a hierarchy of trees. The Hierarchical N-time Signature System (HSS) allows subordinate trees to be generated when needed by the signer. Otherwise, generation of

the entire tree might take weeks or longer.

An HSS signature as specified in specified in [HASHSIG] carries the number of signed public keys (Nspk), followed by that number of signed public keys, followed by the LMS signature as described in Section 2.2. Each signed public key is represented by the hash value at the root of the tree, and it also contains information about the tree structure. The signature over the public key is an LMS signature as described in Section 2.2.

The elements of the HSS signature value for a stand-alone tree can be summarized as:

```
u32str(0) ||
lms_signature  /* signature of message */
```

The elements of the HSS signature value for a tree with Nspk levels can be summarized as:

```
u32str(Nspk) ||
signed_public_key[1] ||
signed_public_key[2] ||
   ...
sigend_public_key[Nspk-1] ||
signed_public_key[Nspk] ||
lms_signature_on_message
```

where, as defined in Section 7 of [HASHSIG], a signed_public_key is the lms_signature over the public key followed by the public key itself.

2.2. Leighton-Micali Signature (LMS)

Each tree in the system specified in [HASHSIG] uses the Leighton-Micali Signature (LMS) system. LMS systems have two parameters. The first parameter is the height of the tree, h, which is the number of levels in the tree minus one. The [HASHSIG] specification supports five values for this parameter: h=5; h=10; h=15; h=20; and h=25. Note that there are 2^h leaves in the tree. The second parameter is the number of bytes output by the hash function, m, which the amount of data associated with each node in the tree. The [HASHSIG] specification supports only the SHA-256 hash function [SHS], with m=32.

Currently, the hash-based signature algorithm supports five tree sizes:

LMS_SHA256_M32_H5;
LMS_SHA256_M32_H10;
LMS_SHA256_M32_H15;
LMS_SHA256_M32_H20; and
LMS_SHA256_M32_H25.

The [HASHSIG] specification establishes an IANA registry to permit the registration of additional tree sizes in the future.

An LMS signature consists of four elements: the number of the leaf associated with the LM-OTS signature, an LM-OTS signature as described in Section 2.3, a typecode indicating the particular LMS algorithm, and an array of values that is associated with the path through the tree from the leaf associated with the LM-OTS signature to the root. The array of values contains the siblings of the nodes on the path from the leaf to the root but does not contain the nodes on the path itself. The array for a tree with height h will have h values. The first value is the sibling of the leaf, the next value is the sibling of the parent of the leaf, and so on up the path to the root.

The four elements of the LMS signature value can be summarized as:

```
u32str(q) ||
ots_signature ||
u32str(type) ||
path[0] || path[1] || ... || path[h-1]
```

2.3. Leighton-Micali One-time Signature Algorithm (LM-OTS)

Merkle Tree Signatures (MTS) depend on a one-time signature method. [HASHSIG] specifies the use of the LM-OTS. An LM-OTS has five parameters.

n - The number of bytes associated with the hash function. [HASHSIG] supports only SHA-256 [SHS], with n=32.

H - A preimage-resistant hash function that accepts byte strings of any length, and returns an n-byte string.

w - The width in bits of the Winternitz coefficients. [HASHSIG] supports four values for this parameter: w=1; w=2; w=4; and w=8.

p - The number of n-byte string elements that make up the LM-OTS signature.

ls - The number of left-shift bits used in the checksum function, which is defined in Section 4.5 of [HASHSIG].

The values of p and ls are dependent on the choices of the parameters n and w, as described in Appendix A of [HASHSIG].

Currently, the hash-based signature algorithm supports four LM-OTS variants:

LMOTS_SHA256_N32_W1;
LMOTS_SHA256_N32_W2;
LMOTS_SHA256_N32_W4; and
LMOTS_SHA256_N32_W8.

The [HASHSIG] specification establishes an IANA registry to permit the registration of additional variants in the future.

Signing involves the generation of C, an n-byte random value.

The LM-OTS signature value can be summarized as:

u32str(otstype) || C || y[0] || ... || y[p-1]

3. Algorithm Identifiers and Parameters

The algorithm identifier for an MTS signature is id-alg-mts-hashsig:

id-alg-mts-hashsig OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) smime(16) alg(3) 17 }

When the id-alg-mts-hashsig algorithm identifier is used for a signature, the AlgorithmIdentifier parameters field MUST be absent (that is, the parameters are not present; the parameters are not set to NULL).

The signature values is a large OCTET STRING. The signature format is designed for easy parsing. Each format includes a counter and type codes that indirectly providing all of the information that is needed to parse the value during signature validation.

4. Signed-data Conventions

As specified in [CMS], the digital signature is produced from the message digest and the signer’s private key. If signed attributes are absent, then the message digest is the hash of the content. If

signed attributes are present, then the hash of the content is placed in the message-digest attribute, the set of signed attributes is DER encoded, and the message digest is the hash of the encoded attributes. In summary:

```
IF (signed attributes are absent)
THEN md = Hash(content)
ELSE message-digest attribute = Hash(content);
     md = Hash(DER(SignedAttributes))

Sign(md)
```

When using [HASHSIG], the fields in the SignerInfo are used as follows:

digestAlgorithms SHOULD contain the one-way hash function used to compute the message digest on the eContent value. Since the hash-based signature algorithms all depend on SHA-256, it is strongly RECOMMENDED that SHA-256 also be used to compute the message digest on the content.

Further, the same one-way hash function SHOULD be used to compute the message digest on both the eContent and the signedAttributes value if signedAttributes are present. Again, since the hash-based signature algorithms all depend on SHA-256, it is strongly RECOMMENDED that SHA-256 be used.

signatureAlgorithm MUST contain id-alg-mts-hashsig. The algorithm parameters field MUST be absent.

signature contains the single HSS signature value resulting from the signing operation as specified in [HASHSIG].

5. Security Considerations

5.1. Implementation Security Considerations

Implementations must protect the private keys. Compromise of the private keys may result in the ability to forge signatures. Along with the private key, the implementation must keep track of which leaf nodes in the tree have been used. Loss of integrity of this tracking data can cause an one-time key to be used more than once. As a result, when a private key and the tracking data are stored on non-volatile media or stored in a virtual machine environment, care must be taken to preserve confidentiality and integrity.

An implementation must ensure that a LM-OTS private key is used to generate a signature only one time, and ensure that it cannot be used

for any other purpose.

The generation of private keys relies on random numbers. The use of inadequate pseudo-random number generators (PRNGs) to generate these values can result in little or no security. An attacker may find it much easier to reproduce the PRNG environment that produced the keys, searching the resulting small set of possibilities, rather than brute force searching the whole key space. The generation of quality random numbers is difficult. RFC 4086 [RANDOM] offers important guidance in this area.

The generation of hash-based signatures also depends on random numbers. While the consequences of an inadequate pseudo-random number generator (PRNGs) to generate these values is much less severe than the generation of private keys, the guidance in [RFC4086] remains important.

When computing signatures, the same hash function SHOULD be used for all operations. In this specification, only SHA-256 is used. Using only SHA-256 reduces the number of possible failure points in the signature process.

5.2. Algorithm Security Considerations

At Black Hat USA 2013, some researchers gave a presentation on the current sate of public key cryptography. They said: "Current cryptosystems depend on discrete logarithm and factoring which has seen some major new developments in the past 6 months" [BH2013]. They encouraged preparation for a day when RSA and DSA cannot be depended upon.

A post-quantum cryptosystem is a system that is secure against quantum computers that have more than a trivial number of quantum bits. It is open to conjecture when it will be feasible to build such a machine. RSA, DSA, and ECDSA are not post-quantum secure.

The LM-OTP one-time signature, LMS, and HSS do not depend on discrete logarithm or factoring, as a result these algorithms are considered to be post-quantum secure.

Today, RSA is often used to digitally sign software updates. This means that the distribution of software updates could be compromised if a significant advance is made in factoring or a quantum computer is invented. The use of MTS signatures to protect software update distribution, perhaps using the format described in [FWPROT], will allow the deployment of software that implements new cryptosystems.

6. IANA Considerations

This document has no actions for IANA.

7. Acknowledgements

Many thanks to Panos Kampanakis, Jim Schaad, and Sean Turner for their careful review and comments.

8. Normative References

[ASN1-B] ITU-T, "Information technology -- Abstract Syntax Notation One (ASN.1): Specification of basic notation", ITU-T Recommendation X.680, 2015.

[ASN1-E] ITU-T, "Information technology -- ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T Recommendation X.690, 2015.

[CMS] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <http://www.rfc-editor.org/info/rfc5652>.

[HASHSIG] McGrew, D., M. Curcio, and S. Fluhrer, "Hash-Based Signatures", Work in progress. <draft-mcgrew-hash-sigs-11>

[RFC2219] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/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/info/rfc8174>.

[SHS] National Institute of Standards and Technology (NIST), FIPS Publication 180-3: Secure Hash Standard, October 2008.

9. Informative References

[BH2013] Ptacek, T., T. Ritter, J. Samuel, and A. Stamos, "The Factoring Dead: Preparing for the Cryptopocalypse", August 2013. <https://media.blackhat.com/us-13/us-13-Stamos-The-Factoring-Dead.pdf>

[CMSASN1] Hoffman, P. and J. Schaad, "New ASN.1 Modules for Cryptographic Message Syntax (CMS) and S/MIME", RFC 5911, DOI 10.17487/RFC5911, June 2010, <http://www.rfc-editor.org/info/rfc5911>.

[CMSASN1U] Schaad, J. and S. Turner, "Additional New ASN.1 Modules for the Cryptographic Message Syntax (CMS) and the Public Key Infrastructure Using X.509 (PKIX)", RFC 6268, DOI 10.17487/RFC6268, July 2011, <http://www.rfc-editor.org/info/rfc6268>.

[FWPROT] Housley, R., "Using Cryptographic Message Syntax (CMS) to Protect Firmware Packages", RFC 4108, DOI 10.17487/RFC4108, August 2005, <http://www.rfc-editor.org/info/rfc4108>.

[LM] Leighton, T. and S. Micali, "Large provably fast and secure digital signature schemes from secure hash functions", U.S. Patent 5,432,852, July 1995.

[M1979] Merkle, R., "Secrecy, Authentication, and Public Key Systems", Stanford University Information Systems Laboratory Technical Report 1979-1, 1979.

[M1987] Merkle, R., "A Digital Signature Based on a Conventional Encryption Function", Lecture Notes in Computer Science crypto87, 1988.

[M1989a] Merkle, R., "A Certified Digital Signature", Lecture Notes in Computer Science crypto89, 1990.

[M1989b] Merkle, R., "One Way Hash Functions and DES", Lecture Notes in Computer Science crypto89, 1990.

[PKIXASN1] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, DOI 10.17487/RFC5912, June 2010, <http://www.rfc-editor.org/info/rfc5912>.

[PQC] Bernstein, D., "Introduction to post-quantum cryptography", 2009. <http://www.pqcrypto.org/www.springer.com/cda/content/document/cda_downloaddocument/9783540887010-c1.pdf>

[RANDOM] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <http://www.rfc-editor.org/info/rfc4086>.

Appendix: ASN.1 Module

```
   MTS-HashSig-2013
     { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
       id-smime(16) id-mod(0) id-mod-mts-hashsig-2013(64) }

   DEFINITIONS IMPLICIT TAGS ::= BEGIN

   EXPORTS ALL;

   IMPORTS
     PUBLIC-KEY, SIGNATURE-ALGORITHM, SMIME-CAPS
       FROM AlgorithmInformation-2009  -- RFC 5911 [CMSASN1]
         { iso(1) identified-organization(3) dod(6) internet(1)
           security(5) mechanisms(5) pkix(7) id-mod(0)
           id-mod-algorithmInformation-02(58) }
     mda-sha256
       FROM PKIX1-PSS-OAEP-Algorithms-2009  -- RFC 5912 [PKIXASN1]
         { iso(1) identified-organization(3) dod(6)
           internet(1) security(5) mechanisms(5) pkix(7) id-mod(0)
           id-mod-pkix1-rsa-pkalgs-02(54) } ;

   --
   -- Object Identifiers
   --

   id-alg-mts-hashsig  OBJECT IDENTIFIER ::= { iso(1) member-body(2)
         us(840) rsadsi(113549) pkcs(1) pkcs9(9) smime(16) alg(3) 17 }

   --
   -- Signature Algorithm and Public Key
   --

   sa-MTS-HashSig SIGNATURE-ALGORITHM ::= {
        IDENTIFIER id-alg-mts-hashsig
        PARAMS ARE absent
        HASHES { mda-sha256 }
        PUBLIC-KEYS { pk-MTS-HashSig }
        SMIME-CAPS { IDENTIFIED BY id-alg-mts-hashsig } }

   pk-MTS-HashSig PUBLIC-KEY ::= {
       IDENTIFIER id-alg-mts-hashsig
       KEY MTS-HashSig-PublicKey
       PARAMS ARE absent
       CERT-KEY-USAGE
           { digitalSignature, nonRepudiation, keyCertSign, cRLSign } }

   MTS-HashSig-PublicKey ::= OCTET STRING
```

```
   --
   -- Expand the signature algorithm set used by CMS [CMSASN1U]
   --

   SignatureAlgorithmSet SIGNATURE-ALGORITHM ::=
        { sa-MTS-HashSig, ... }

   --
   -- Expand the S/MIME capabilities set used by CMS [CMSASN1]
   --

   SMimeCaps SMIME-CAPS ::= { sa-MTS-HashSig.&smimeCaps, ... }

   END
```

Author's Address

Russ Housley
Vigil Security, LLC
918 Spring Knoll Drive
Herndon, VA 20170
USA

EMail: housley@vigilsec.com

SecDispatch
Internet-Draft
Intended status: Informational
Expires: September 6, 2018

Y. Nir
Dell EMC
T. Fossati
Nokia
Y. Sheffer
Intuit
T. Eckert
Huawei

March 5, 2018

Considerations For Using Short Term Certificates
draft-nir-saag-star-01

Abstract

Recently there has been renewed interest in an old idea: Issue certificates with short validity periods and forego revocation processing, reasoning that expiration is a sufficient replacement for revocation as long as that expiration is not too far off.

This document covers considerations, both security and operational, for using such Short Term Auto Renewed (STAR) certificates for various scenarios where Using a revocation protocol is considered inappropriate.

Table of Contents

1. Introduction

Certificates ([RFC5280]) are used in multiple protocols such as the Internet Key Exchange (IKEv2-[RFC7296]) and the Transport Layer Security protocol (TLS-[RFC5246]). Certificates are used to authenticate communicating parties to each other. Certificates are

issued by Certificate Authorities (CAs) to End Entities (EE) to be used to authenticate them to Relying Parties (RPs) in security protocols. Systems that use secure communications typically include certificate authorities, end entities and relying parties, with some nodes in the network having more than one of these roles.

When deploying a system involving secure communications, one of the challenges is how to deal with compromise of an End Entity's private key. The standard way of dealing with this is adding a protocol layer for revocation such as CRLs ([RFC5280]) or OCSP ([RFC6960]).

Such revocation protocols have drawbacks. Although caching of CRLs and OCSP responses is allowed, each setup of a secure channel may require accessing the CRL distribution point (DP) or the OCSP responder. This is both time consuming and provides the system with a few more modes of failure. Assuring reliability of the revocation service increases the cost, and overcoming the latency issue requires changes to the security protocols. All other things being equal, a system that includes revocation checking is more complex and less reliable than a system that does not include it.

For these reasons it is attractive to forego revocation checking. Some deployed systems do this by either eliminating the CRL DP and OCSP extensions from the certificates, or ignoring network and timeout errors in fetching revocation information. Both practices reduce the security of the system.

An alternative solution to the revocation problem is to issue certificates with a short validity period and forego revocation checking. Normally certificates are issued with a validity period of between a few months and a few years. With a shorter validity period if the private key is compromised the potential for abuse is lower because the certificate and its private key expire within a short period of time - a few hours to a few days.

The rest of this document describes operational and security considerations with using short term certificates.

1.1. Conventions Used in This Document

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].

Throughout this document we will use the term DP to denote a server for revocation information, either a CRL distribution point or an OCSP Responder. For our purposes they are the same.

We use the term longevity for the period of time between certificate issuance and the time of its expiration as indicated in the notAfter field of the certificate. Note that issuance time may be different from the notBefore field in the certificate.

The text describes end entities as renewing their certificates because the usual operational model for certificates is one of "pull": end entities create certificate requests and send them to CAs for signature. Some systems are designed around a "push" operation where either the CA or a management function generates a new certificate and installs it on the end entity. The text in the document uses pull terminology, but is equally relevant for a push design.

2. Short Term Auto Renewed Certificates

Short term certificates are like any other [RFC5280] certificates except that the period of time between their issuance and their notAfter date is relatively short. Whereas normally certificates are issued for a period of time between a few months and a few years, short term certificates usually expire after a few hours, a few days, or at a limit a couple of weeks.

The certificates discussed in this document have neither a CRL DP extension nor an OCSP authorityInformationAccess extension. In other words such certificates cannot be revoked. Instead, they are valid until they expire.

Automatic certificate renewal is getting ever more popular with enrollment protocols such as EST ([RFC7030]) or ACME ([I-D.ietf-acme-acme]). For short term certificates automatic renewal is essential as a human cannot be expected to flawlessly perform a manual renewal every few days or hours. This document does not recommend any particular automatic renewal method, but Section 4.4 recommends that some such method be used. Automatic renewal processing can roll over the keys from one certificate to its successor, or it can generate new keys with each Certificate generation. As revocation may not exist, multiple certificates for the same EE may be valid at any given time.

The solution for revocation in this scheme is to stop the automatic renewal. The existing compromised certificate will remain valid until it expires. See the considerations in Section 5.1 about revocation.

[Topalovic] describes the design of a system involving STAR certificates for the web, and analyzes its security and efficacy. It

concludes that STAR certificates can be as secure as certificates with OCSP revocation.

2.1. Alternative Design: OCSP Stapling

Relying parties can also avoid the need for contacting the DP at connection setup by having the End Entities implement OCSP stapling. This feature has the EEs rather than the RPs retrieve the OCSP response and send it as part of the protocol. OCSP stapling is described for TLS in [RFC6961] and [RFC6066], and for IKE in [RFC4806].

STAR has several advantages over OCSP stapling:

- A CA that only signs certificates is simpler than a CA that both signs certificates and issues OCSP responses. In fact, a CA for STAR does not need to keep any record of issued certificates.

- A system that does not use CRLs or OCSP need not have an always-available DP for delivering those CRLs or OCSP responses. This reduces both complexity and attack surface.

- OCSP stapling in TLS versions prior to 1.3 works only for the server as end entity. There was no provision for sending the OCSP response for a client certificate in the protocol.

2.2. The Case For Foregoing Revocation

When explaining PKI to people, it is hard to justify why the CA or a delegate needs to both sign blob-1 (the certificate) and also sign blob-2 (the CRL or OCSP response) to tell relying parties that blob-1 is still valid. Surely one signed blob should be enough.

The explanation that we come up with is that traditionally issuing a certificate required human intervention, while the revocation checking object could be issued automatically and at great frequency. So blob-1 would have to be valid for long enough to not over-burden the human charged with maintaining them, while blob-2 could be re-issued frequently.

This explanation no longer holds up. While the initial certificate enrollment may need to be initiated by a human, protocols exist today that make certificate renewal just as automated as CRL issuance. Certificates can be just as frequently issued as CRLs were in the past. The added complexity is no longer needed.

In real systems such as the web relying parties or end entities cache revocation objects as long as it’s allowed. If a CRL has a

nextUpdate field that is 4 days in the future, a typical system will not attempt to fetch a new one before those 4 days have elapsed. For this reason, moving to STAR certificates provides a similar level of security to what is generally practiced on the web.

3. Use Cases

This section lists some use cases where STAR certificates seem to be more appropriate than long-lived certificates with revocation checking. The purpose of this section is only motivational. None of the following sections are intended to be a definition of the use case or the standard by which future documents or implementations will be measured for sufficiency.

3.1. Data Center Network Hosts

TBA

3.2. Distributed System Installed in One Or More Data Centers

This is a system installed in multiple hosts in one or more data centers that fulfills some task and requires mutual authentication of its components. An example of such a system is a Storage Area Network (SAN).

3.2.1. Distributed Network Security Functions

This example of a distributed system is multiple network security functions (NSF) [RFC8192] where the SDN controller needs to authenticate the NSFs with which it communicates, and some NSFs need to communicate with each other.

3.3. Certificate Delegation for Content Delivery Networks

TBA

3.4. Autonomic Networking Infrastructure

The Autonomic Network Infrastructure (see [I-D.ietf-anima-reference-model]) is an IETF ANIMA Working Group developed network system architecture to provide the foundation for both future "autonomic networks" (AN), as well as the infrastructure to enable zero-touch secure bootstrapping of domain-wide PKI certificates for network equipment (BRSKI, see [I-D.ietf-anima-bootstrapping-keyinfra]) as well as the set-up of a zero-touch, secure communications fabric for management of existing networks (ACP, see [I-D.ietf-anima-autonomic-control-plane], especially in the context of evolving SDN control & management (see

[I-D.ietf-anima-stable-connectivity]) . These domain certificates are furthermore meant to be reuseable across all network services between network equipment in that domain, therefore allowing to eliminate the need for per-service crypto management (IGP, multicast, BGP, netconf/COPS/radius connections,...).

Overall, the PKI related functions of ANI intend to increase proliferation of PKI security through simplification, achieved through automation and making solutions more resilient by minimizing managed component requirements. CRL or OCSP introduce another set of servers/services that needs to be managed/automated/distributed. The connectivity requirement to such servers and/or the grace periods during which connectivity to them is not required introduce more complex system/security design parameters.

With ANI/ACP/BRSKI, renewal of certificates is fully automated and therefore shorter lifetimes of certificates can easily be used to avoid the additional need for CRL/OCSP. The limitation on reducing certificate lifetimes is only the desired maximum length of time during which connectivity to a CA for renewal may not exist - and the maximum renewal rate of certificates that can be supported by those CA.

Because of the ACP, connectivity to the CA is also more resilient against network/provisioning/configuration problems than network without an ACP. Lastly, the whole ANI is built and maintained autonomously without the need of any configurations except for one or more seed-nodes that perform an expanded version of a PKI Registration Authority.

4. Operational Considerations

The motivations for using short-term certificates are operational. We don't want the latency introduced by fetching the CRL from the DP; we don't want the cost of making the DP 99.999% reliable, and we don't want the cost of making the network paths from all RPs to the DP always available.

Deploying short term certificates comes with its own set of operational considerations, and some of these are enumerated in the following sub-sections.

4.1. Certificate Lifetime and Renewal Schedule

Since we do not assume the CA to be close to 100% available it makes sense for End Entities to renew their certificates well in advance. While the security considerations in Section 5.2 set an upper limit on the longevity of a STAR certificate, operational necessity sets

the frequency of renewal. It is necessary to strike a balance between renewing too often which leads to increased load on the CA and renewing too seldom which increases the risk of having the certificate expire while either the CA or the End Entity are down.

Individual system properties play a significant role here. Systems where both the CA and the EEs are expected to be up all of the time absent a fault may choose to renew a day or even an hour before expiration, while systems with nodes that are only up infrequently and for short periods of time may choose to renew the certificates whenever the EEs happen to be up.

As a general rule of thumb for systems where the CA is mostly available it makes sense for the EE to make the first attempt to renew its certificate about half-way through its lifetime. If that attempt fails because the CA is not available an EE SHOULD retry at regular intervals until it succeeds. Shortly before expiration, the EE SHOULD increase the frequency of retires.

For example, suppose a STAR certificate is issued for 8 days. The EE will first attempt to renew the certificate 4 days before expiration. If that fails it will retry every three hours until only six hours are left before expiration. At that point it will increase the frequency and retry every five minutes. If this is part of the system design, at this point it should also alert the user that something is wrong.

4.2. Availability of the Certificate Authority

While the STAR design does not require 99.999% availability, the CA does need to be available for renewing certificates. Downtimes of more than a quarter of the certificate longevity SHOULD NOT happen. For most modern hardware this is entirely possible even without exotic clustering solutions, but when configuring the system administrators should consider that the longevity of the certificates constrains the required availability of the CA.

When setting the longevity for certificates administrators SHOULD consider how long it takes to recover from a failure of the CA. That length of time can be seconds with a good clustering solution, but can span hours or days without one, especially if the fault happens at a bad time. A failure of a CA should be considered a conceivable occurrence, and longevity should be set so that such a failure does not lead to expiration and outage.

Conversely, if short longevity is required by security targets, the CA should be made more reliable with clustering solutions.

4.3. Clock Skew and the notBefore Field

Despite NTP ([RFC5905]) being over thirty years old and implemented in every major operating system clock skew is a fact of life and many deployed systems don't have the right time. It is also not possible to just mandate the use of NTP because the systems that use STAR certificates are often installed on hosts and networks where NTP is either not configured or blocked. We cannot assume that these systems can enable NTP at will.

Skewed clocks have always been a problem for certificates. Because STAR certificates are always just a few days or hours from expiration they are more sensitive to clock skew. A sufficiently skewed clock can cause three different disfunctions and for STAR certificate such disfunction happens with considerably less skew than with long term certificates:

- A valid certificate may be rejected as not yet valid if the current system time is earlier than its notBefore time. Fortunately this issue can be safely mitigated by setting the notBefore field to a time earlier than the time of issuance.

- A valid certificate may be rejected as expired if the current system time is later than its notAfter time. As long as the clock skew is not too great this is solved by a sensible renewal policy. If as in the example in Section 4.1 the certificate is renewed 4 days before expiration or within a few hours after that, a clock skew of up to 3 days will not be a problem.

- An expired certificate may be accepted if the current system time is earlier than its notAfter time. This is a security issue that is discussed in Section 5.3.

There are several common modes of clock skew:

- The system that doesn't have its clock set at all. These systems might be set to January 1st, 1970 or to some date that was interesting for the hardware vendor. Such systems are incompatible with certificates and MUST NOT be used for STAR certificates.

- The system has its timezone set wrong, and the system time was set so that local time looks good. This limits the clock skew to 24 hours and is generally workable.

- A system that has the time set right but the date set wrong. These are also not usable with certificates.

- o A system that was set to the correct time once but has since drifted away. Computer hardware varies wildly between systems with quartz clocks that drift only a few seconds a month and systems that can lose or gain minutes a day. The former are quite usable, the latter are not.

Because of the prevalence of systems with a relatively small skew it is RECOMMENDED to set the notBefore field to a time 72 hours before the actual issuance date.

End Entities MUST NOT use expired certificates and Relying Parties SHOULD alert whenever an expired certificate is presented. This will help the users keep their host clocks set or encourage them to enable NTP.

4.4. Automatic Renewal

Automatic enrollment and renewal is recommended for any system using certificates. While it is possible to renew certificates manually on time, even organizations with the best of IT departments occasionally miss this: [cert-expires]

With short term certificates, this becomes even more important. Renewing a certificate manually every few days or hours is extremely labor intensive, especially when the system contains hundreds, thousands or more end entities, and the risk of outages becomes a certainty.

This document does not mandate any particular enrollment or renewal mechanism. Any of a myriad of standard and proprietary methods can be used and systems with proprietary methods have been shipping for years. The IETF is in the process of standardizing the ACME protocol for enrollment and renewal ([I-D.ietf-acme-acme]) and an extension is proposed to make it more suitable for STAR certificates ([I-D.ietf-acme-star]). The ANI as described in Section 3.4 is a complete zero touch system design providing and relying on automatic certificate renewal.

4.5. Secure (Re-)Enrollments

When short lived certificates expire, automatic re-enrollment can further help to provide survivable, resilient PKI security. Traditionally, initial enrollments, even with otherwise automated solutions such as EST ([RFC7030]) required a manual interaction, or else the device had to perform TOFU (Trust On First Use) to be automatically enrolled. TOFU is even more problematic for re-enrollments and becomes more problematic, the shorter lived certificates and/or trust anchors are. Consider the risk where

during re-enrollment, the device may already be fully configured and could be taken over by an attacker just having to wait for a short lived certificate device certificate or trust anchor to expire. Or consider devices auto-resetting themselves to factory conditions to avoid this problem and then not having to be re-enrolled, but also be re-configured - in the absence of fully zero-touch provisioning solutions.

ANIs BRSKI protocol ([I-D.ietf-anima-bootstrapping-keyinfra], which introduces extensions to EST), and NetConf Zero Touch ([I-D.ietf-netconf-zerotouch] allow fully automated enrollment and re-enrollment of device certificate and trust anchors through the use of "vouchers" ([I-D.ietf-anima-voucher]). These are new digital artifacts that allow enrolling devices to securely trust domains to (re-)enroll them. They work by providing a signed statement by a representative of the manufacturer of the device, that the device with a specified identity (e.g: IDevID) should trust a particular domain - identified by an initial trust anchor. This allows to overcome the biggest challenge of expired short lived certificates/ trust-anchors.

Furthermore, if the certificate and/or trust anchors are required for security of network connectivity - such as routing protocol security or network layer encryption - to even reach a re-enrollment server, then there is yet another challenge with short lived certificates/ trust-anchors and their higher likelyhood of expiring.

In the case of ANI, network layer security (e.g.: IPsec) is used for protecting network connectivity including to reach the EST renewal server. When certificate/TA are expired, renewal can not be used. Instead though, automatic re-enrollment can be used, which does not rely on generic network layer security, but instead relies on its own proxy service to provide connectivity for such devices that need to re-enroll. Nevertheless, re-enrollment may be a complex operation due to the potential need to involve the above mentioned representative entity of the manufacturer to generate vouchers.

4.6. Future enhancements for renewal/re-enrollment

One easy improvement that is specifically of interest with the use of short-lived device certificates/trust-anchors is a new interpretation of the lifetime of certificates. Today, there is no clear distinction when or how to apply the lifetime, and in result, it is usually assumed to be applicable to all operations relying on those certificates.

In the case of short-lived certificates, the elements performing certificate renewal/re-enrollment can easily have a different

interpretation of the lifetime and may not rely on what the certificate itself says. This allows to turn re-enrollments into renewals and avoid possible complexities or manual steps potentially required for re-enrollment (depending on the system used).

In the case of BRSKI/EST, there is only one TLS connection used for renewal and/or re-enrollment and expiry affects the certificates used on this TLS connection. The server uses EST for renewal or the extended signaling of BRSKI for re-enrollment. When a device with expired, short-lived certificate connects to the BRSKI/EST server, this server could allow to perform only simple EST renewal instead of re-enrollment with a voucher by simply considering the lifetime of the presented (and expired) device certificate to be extended.

This type of re-interpretation requires primarily some generic work to allow this type of interpretation - and then per-solution work to leverage this interpretation. In the case of BRSKI/EST for example, devices would simply use their expired domain certificate to authenticate themselves and perform certificate renewal - instead of using their IDevID and trying to re-enroll (which is a more complex operation with potentially external dependenices against the manufacturer component).

4.7. Certificate Transparency

Certificate Transparency (CT), [RFC6962] is about keeping a log of all issued certificates.

A system that issues a certificate every few days to thousands or end entities will create more records for a CT log than a web host that gets one certificate every year.

TBA: Discussion about this.

5. Security Considerations

STAR certificates eliminate an important security feature of PKI which is the ability to revoke certificates. Revocation allows the administrator to limit the damage done by a rogue node or an adversary who has control of the private key. With STAR certificates expiration replaces revocation so there is a timeliness issue.

It should be noted that revocation also has timeliness issues, because both CRLs and OCSP responses have nextUpdate fields that tell RPs how long they should trust this revocation data. These fields are typically set to hours, days, or even weeks in the future. Any revocation that happens before the time in nextUpdate goes unnoticed by the RP.

Section 5.1 discusses the reasons why a certificate would be revoked if revocation was available and how STAR certificates do the same. Section 5.2 discusses considerations for setting the longevity of a certificate, and Section 5.3 discusses how longevity should be adjusted to deal with clock skew.

More discussion of the security of STAR certificates is available in [Topalovic].

5.1. Reasons for Revocation

There are two types of compromise that require administrators to revoke a certificate:

- A host has lost control of the private key. There are many ways that this can happen: a host can be hacked and a file containing the private key may or may not have been copied; a disk may be replaced and the old one has not been securely disposed of; a fault causes the private key to be erased. In all these cases we would like to revoke the certificate to make sure an adversary cannot use the private key for nefarious purposes. For STAR certificates the only solution is to wait for the certificate to expire and the system is vulnerable until that happens. Longevity should be set so that this risk is acceptable.

- A host may begin doing unintended things, either due to a software fault or due to a malicious takeover. Again without revocation RPs will continue to trust this node until its certificate expires.

When a node "goes rogue" or an adversary gets control of the private key it is important to block renewal or these certificates or else the attack can persist forever. No matter how short-term these short term certificates are, there is a certain window of time when the attacker can use the certificate. This can often be mitigated with application-level measures.

With most systems relying parties are configured with the names of nodes with which they are allowed to communicate. When revocation is not available changing the configuration so that the rogue node cannot connect is RECOMMENDED. This is useful even when revocation is available because timeliness issues are common to both revocation and expiration.

5.2. Longevity and Revocation

There is always a period of time between when a compromise is discovered and when RPs stop trusting the certificate. With revocation this has to do with the time it takes to process the revocation and the span of time between the thisUpdate and nextUpdate fields. With STAR certificates this is controlled by the time it takes to inhibit renewals and the longevity of the certificates.

For this reason it makes sense to set the longevity to a period of time similar to the span of time that we would set for the CRL or OCSP updates. Typically a few days is an appropriate time. For some cases this can be as low as a few hours. Setting the renewal time too short may cause operational problems as discussed in Section 4.3 and Section 4.2. In general longevity should not be set shorter than the availability of the CA allows.

Fortunately modern hardware is powerful enough and reliable enough that even a system with tens of thousands of end entities with longevity of 1-2 days should not suffer an outage because of expired certificates.

5.3. Clock Skew and Security

As discussed in Section 4.3 clock skew can lead to expired certificates being treated as valid. While even the use of NTP may leave clocks with a few seconds of inaccuracy, all installations MUST take steps to limit the clock skew on their hosts.

An upper bound for the amount of skew allowed for hosts in a particular system is one of the parameters for such a system. For systems using NTP this can be 2 seconds. For systems where the clocks are set manually, this tends to be far greater, but without an upper bound no guarantees can be made about the security of certificate use.

This upper bound is also a limit on the target certificate longevity. For example, if hosts and CAs can each have a clock skew of 24 hours then it is impossible to achieve a longevity of under 48 hours. With a reasonable skew and a reasonable target longevity we can achieve our security targets by reducing the certificate longevity by twice the upper bound for skew. So if skew is bounded by 24 hours (the bad timezone case) and target longevity is 7 days, it makes sense to set the longevity on the CA to 5 days.

5.4. CA availability

A successful Denial of Service (DoS) attack against a CA prevents it from issuing certificates. With short-term certificates this could quickly lead to outages as certificates expire.

The important period of time here is the time between when the EE first attempts to renew the certificate and the time that the certificate expires. For example, if the EE attempts to renew the certificates a mere five minutes before expiration, then a five-minute CA outage can lead to an invalid certificate and failed connections.

This issue is no different from DoS attacks against the DP for certificates with revocation. The methods of protection are also similar:

- Certificate renewal should first be attempted plenty of time in advance as recommended in Section 4.1. This will leave enough time for administrators to deal with the attack.

- As for all important infrastructure, network defenses SHOULD be deployed to mitigate DoS attacks.

6. IANA Considerations

There are no requests to IANA in this document.

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>.

[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>.

7.2. Informative References

[cert-expires]
Lennon, M., "Google Lets SMTP Certificate Expire", April 2015, <http://www.securityweek.com/google-lets-smtp-certificate-expire>.

[I-D.ietf-acme-acme]
Barnes, R., Hoffman-Andrews, J., and J. Kasten, "Automatic Certificate Management Environment (ACME)", draft-ietf-acme-acme-07 (work in progress), June 2017.

[I-D.ietf-acme-star]
Sheffer, Y., Lopez, D., Gonzalez de Dios, O., Pastor Perales, A., and T. Fossati, "Use of Short-Term, Automatically-Renewed (STAR) Certificates to Delegate Authority over Web Sites", draft-ietf-acme-acme-07 (work in progress), June 2017.

[I-D.ietf-anima-autonomic-control-plane]
Eckert, T., Behringer, M., and S. Bjarnason, "An Autonomic Control Plane (ACP)", draft-ietf-anima-autonomic-control-plane-13 (work in progress), December 2017.

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-11 (work in progress), February 2018.

[I-D.ietf-anima-reference-model]
Behringer, M., Carpenter, B., Eckert, T., Ciavaglia, L., Pierre, P., Liu, B., Nobre, J., and J. Strassner, "A Reference Model for Autonomic Networking", draft-ietf-anima-reference-model-05 (work in progress), October 2017.

[I-D.ietf-anima-stable-connectivity]
Eckert, T. and M. Behringer, "Using Autonomic Control Plane for Stable Connectivity of Network OAM", draft-ietf-anima-stable-connectivity-10 (work in progress), February 2018.

[I-D.ietf-anima-voucher]
Watsen, K., Richardson, M., Pritikin, M., and T. Eckert, "Voucher Profile for Bootstrapping Protocols", draft-ietf-anima-voucher-07 (work in progress), January 2018.

[I-D.ietf-netconf-zerotouch]
Watsen, K., Abrahamsson, M., and I. Farrer, "Zero Touch Provisioning for Networking Devices", draft-ietf-netconf-zerotouch-20 (work in progress), February 2018.

[RFC4806] Myers, M. and H. Tschofenig, "Online Certificate Status Protocol (OCSP) Extensions to IKEv2", RFC 4806, DOI 10.17487/RFC4806, February 2007, <https://www.rfc-editor.org/info/rfc4806>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <https://www.rfc-editor.org/info/rfc6066>.

[RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, DOI 10.17487/RFC6960, June 2013, <https://www.rfc-editor.org/info/rfc6960>.

[RFC6961] Pettersen, Y., "The Transport Layer Security (TLS) Multiple Certificate Status Request Extension", RFC 6961, DOI 10.17487/RFC6961, June 2013, <https://www.rfc-editor.org/info/rfc6961>.

[RFC6962] Laurie, B., Langley, A., and E. Kasper, "Certificate Transparency", RFC 6962, DOI 10.17487/RFC6962, June 2013, <https://www.rfc-editor.org/info/rfc6962>.

[RFC7030] Pritikin, M., Ed., Yee, P., Ed., and D. Harkins, Ed., "Enrollment over Secure Transport", RFC 7030, DOI 10.17487/RFC7030, October 2013, <https://www.rfc-editor.org/info/rfc7030>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <https://www.rfc-editor.org/info/rfc7296>.

[RFC8192] Hares, S., Lopez, D., Zarny, M., Jacquenet, C., Kumar, R., and J. Jeong, "Interface to Network Security Functions (I2NSF): Problem Statement and Use Cases", RFC 8192, DOI 10.17487/RFC8192, July 2017, <https://www.rfc-editor.org/info/rfc8192>.

[Topalovic] Topalovic, E., Saeta, B., Huang, L., Jackson, C., and D. Boneh, "Towards Short-Lived Certificates", 2012, <http://www.w2spconf.com/2012/papers/w2sp12-final9.pdf>.

Authors' Addresses

Yoav Nir
Dell EMC
9 Andrei Sakharov St
Haifa 3190500
Israel

EMail: ynir.ietf@gmail.com

Thomas Fossati
Nokia

EMail: thomas.fossati@nokia.com

Yaron Sheffer
Intuit

EMail: yaronf.ietf@gmail.com

Toerless Eckert
Huawei USA - Futurewei Technologies Inc.
2330 Central Expy
Santa Clara 95050
USA

EMail: tte+ietf@cs.fau.de