Network Working Group
Internet-Draft
Updates: 4880 (if approved)
Intended status: Informational
Expires: February 29, 2016

W. Koch
g10 Code

August 28, 2015

EdDSA for OpenPGP
draft-koch-eddsa-for-openpgp-03

Abstract

This specification extends OpenPGP with the EdDSA public key algorithm and describes the use of curve Ed25519.

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 February 29, 2016.

Table of Contents

1. Introduction

The OpenPGP specification in [RFC4880] defines the RSA, Elgamal, and DSA public key algorithms. [RFC6637] adds support for Elliptic Curve Cryptography and specifies the ECDSA and ECDH algorithms. Due to patent reasons no point compression was defined.

This document specifies how to use the EdDSA public key signature algorithm [I-D.josefsson-eddsa-ed25519] with the OpenPGP standard. It defines a new signature algorithm named EdDSA and specifies how to use the Ed25519 curve with EdDSA. This algorithm uses a custom point compression method. There are three main advantages of the EdDSA algorithm: It does not require the use of a unique random number for each signature, there are no padding or truncation issues as with ECDSA, and it is more resilient to side-channel attacks.

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].

2. Supported Curves

This document references the Curve "Ed25519" which is the Edwards form of "Curve25519" and specified in the same paper as the "EdDSA" algorithm ([ED25519]). For the full specification see [I-D.josefsson-eddsa-ed25519].

Other curves may be used by using a specific OID for the curve and its EdDSA parameters.

The following public key algorithm IDs are added to expand section 9.1 of [RFC4880], "Public-Key Algorithms":

ID	Description of Algorithm
TBD1	EdDSA public key algorithm

Compliant applications MUST support EdDSA with the curve Ed25519. Applications MAY support other curves as long as a dedicated OID for using that curve with EdDSA is used.

3. Point Format

The EdDSA algorithm defines a specific point compression format. To indicate the use of this compression format and to make sure that the key can be represented in the Multiprecision Integer (MPI) format of [RFC4880] the octet string specifying the point is prefixed with the octet 0x40. This encoding is an extension of the encoding given in [RFC6637] which uses 0x04 to indicate an uncompressed point.

For example, the length of a public key for the curve Ed25519 is 263 bit: 7 bit to represent the 0x40 prefix octet and 32 octets for the native value of the public key.

4. Encoding of Public and Private Keys

The following algorithm specific packets are added to Section 5.5.2 of [RFC4880], "Public-Key Packet Formats", to support EdDSA.

Algorithm-Specific Fields for EdDSA keys:

- a variable length field containing a curve OID, formatted as follows:
  - a one-octet size of the following field; values 0 and 0xFF are reserved for future extensions,
  - octets representing a curve OID, defined in Section 6.
- MPI of an EC point representing a public key Q as described under Point Format above.

The following algorithm specific packets are added to Section 5.5.3 of [RFC4880], "Secret-Key Packet Formats", to support EdDSA.

Algorithm-Specific Fields for EdDSA keys:

o an MPI of an integer representing the secret key, which is a scalar of the public EC point.

The version 4 packet format MUST be used.

5. Message Encoding

Section 5.2.3 of [RFC4880], "Version 4 Signature Packet Format" specifies formats. To support EdDSA no change is required, the MPIs representing the R and S value are encoded as MPIs in the same way as done for the DSA and ECDSA algorithms; in particular the Algorithm-Specific Fields for an EdDSA signature are:

- MPI of EdDSA value r.

- MPI of EdDSA value s.

Note that the compressed version of R and S as specified for EdDSA ([I-D.josefsson-eddsa-ed25519]) is used.

The version 3 signature format MUST NOT be used with EdDSA.

Although that algorithm allows arbitrary data as input, its use with OpenPGP requires that a digest of the message is used as input. See section 5.2.4 of [RFC4880], "Computing Signatures" for details. Truncation of the resulting digest is never applied; the resulting digest value is used verbatim as input to the EdDSA algorithm.

6. Curve OID

The EdDSA key parameter curve OID is an array of octets that defines a named curve. The table below specifies the exact sequence of bytes for each named curve referenced in this document:

OID	Len	Encoding in hex format	Name
1.3.6.1.4.1.11591.15.1	9	2B 06 01 04 01 DA 47 0F 01	Ed25519

See [RFC6637] for a description of the OID encoding given in the second and third columns.

7. Security Considerations

The security considerations of [RFC4880] apply accordingly.

Although technically possible the use of EdDSA with digest algorithms weaker than SHA-256 (e.g. SHA-1) is not suggested.

8. IANA Considerations

IANA is requested to assign an algorithm number from the OpenPGP Public-Key Algorithms range, or the "namespace" in the terminology of [RFC5226], that was created by [RFC4880]. See section 2.

ID	Algorithm	Reference
TBD1	EdDSA public key algorithm	This doc

[Notes to RFC-Editor: Please remove the table above on publication. It is desirable not to reuse old or reserved algorithms because some existing tools might print a wrong description. A higher number is also an indication for a newer algorithm. As of now 22 is the next free number.]

9. Acknowledgments

The author would like to acknowledge the help of the individuals who kindly voiced their opinions on the IETF OpenPGP and GnuPG mailing lists, in particular, the help of Andrey Jivsov, Jon Callas, and NIIBE Yutaka.

10. References

10.1. Normative References

[I-D.josefsson-eddsa-ed25519] Josefsson, S. and N. Moller, "EdDSA and Ed25519", draft-josefsson-eddsa-ed25519-03 (work in progress), May 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4880] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", RFC 4880, November 2007.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC6637] Jivsov, A., "Elliptic Curve Cryptography (ECC) in OpenPGP", RFC 6637, June 2012.

10.2. Informative References

[ED25519] Bernstein, D., Duif, N., Lange, T., Schwabe, P., and B. Yang, "High-speed high-security signatures", Journal of Cryptographic Engineering Volume 2, Issue 2, pp. 77-89, September 2011, <http://dx.doi.org/10.1007/s13389-012-0027-1>.

Appendix A. Test vectors

For help implementing this specification a non-normative example is given. This example assumes that the algorithm id for EdDSA (TBD1) will be 22.

A.1. Sample key

The secret key used for this example is:

D: 1a8b1ff05ded48e18bf50166c664ab023ea70003d78d9e41f5758a91d850f8d2

Note that this is the raw secret key used as input to the EdDSA signing operation. The key was created on 2014-08-19 14:28:27 and thus the fingerprint of the OpenPGP key is:

```
C959 BDBA FA32 A2F8 9A15  3B67 8CFD E121 9796 5A9A
```

The algorithm specific input parameters without the MPI length headers are:

oid: 2b06010401da470f01

q: 403f098994bdd916ed4053197934e4a87c80733a1280d62f8010992e43ee3b2406

The entire public key packet is thus:

```
98 33 04 53 f3 5f 0b 16  09 2b 06 01 04 01 da 47
0f 01 01 07 40 3f 09 89  94 bd d9 16 ed 40 53 19
79 34 e4 a8 7c 80 73 3a  12 80 d6 2f 80 10 99 2e
43 ee 3b 24 06
```

A.2. Sample signature

The signature is created using the sample key over the input data "OpenPGP" on 2015-09-16 12:24:53 and thus the input to the hash function is:

m: 4f70656e504750040016080006050255f95f9504ff0000000c

Using the SHA-256 hash algorithm yields the digest:

d: f6220a3f757814f4c2176ffbb68b00249cd4ccdc059c4b34ad871f30b1740280

Which is fed into the EdDSA signature function and yields this signature:

r: 56f90cca98e2102637bd983fdb16c131dfd27ed82bf4dde5606e0d756aed3366

s: d09c4fa11527f038e0f57f2201d82f2ea2c9033265fa6ceb489e854bae61b404

The entire signature packet is thus:

```
88 5e 04 00 16 08 00 06  05 02 55 f9 5f 95 00 0a
09 10 8c fd e1 21 97 96  5a 9a f6 22 01 00 56 f9
0c ca 98 e2 10 26 37 bd  98 3f db 16 c1 31 df d2
7e d8 2b f4 dd e5 60 6e  0d 75 6a ed 33 66 01 00
d0 9c 4f a1 15 27 f0 38  e0 f5 7f 22 01 d8 2f 2e
a2 c9 03 32 65 fa 6c eb  48 9e 85 4b ae 61 b4 04
```

Appendix B. Point compression flag bytes

This specification introduces the new flag byte 0x40 to indicate the point compression format. The value has been chosen so that the high bit is not cleared and thus to avoid accidental sign extension. Two other values might also be interesting for other ECC specifications:

Flag	Description
0x04	Standard flag for uncompression format
0x40	Native point format of the curve follows
0x41	Only X coordinate follows.
0x42	Only Y coordinate follows.

Appendix C. Changes since -02

- Fixed decription in the test vectors regarding an extra 0x00 byte.
- Small gramar fixes.

- Reference Josefsson's EdDSA I-D instead of the original paper.

Author's Address

Werner Koch
g10 Code

Email: wk@gnupg.org
URI: https://g10code.com