Workgroup: Transport Layer Security
Internet-Draft:
draft-mattsson-tls-compact-ecc-06
Published: 23 February 2024
Intended Status: Standards Track
Expires: 26 August 2024
Authors: J. Preuß Mattsson H. Tschofenig
Ericsson

Compact ECDHE and ECDSA Encodings for TLS 1.3

Abstract

The encodings used in the ECDHE groups secp256r1, secp384r1, and secp521r1 and the ECDSA signature algorithms ecdsa_secp256r1_sha256, ecdsa_secp384r1_sha384, and ecdsa_secp521r1_sha512 have significant overhead and the ECDSA encoding produces variable-length signatures. This document defines new optimal fixed-length encodings and registers new ECDHE groups and ECDSA signature algorithms using these new encodings. The new encodings reduce the size of the ECDHE groups with 33, 49, and 67 bytes and the ECDSA algorithms with an average of 7 bytes. These new encodings also work in DTLS 1.3 and are especially useful in cTLS.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://emanjon.github.io/draft-mattsson-tls-compact-ecc/draft-mattsson-tls-compact-ecc.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-mattsson-tls-compact-ecc/.

Discussion of this document takes place on the Transport Layer Security Working Group mailing list (mailto:tls@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/tls/. Subscribe at https://www.ietf.org/mailman/listinfo/tls/.

Source for this draft and an issue tracker can be found at https://github.com/emanjon/draft-mattsson-tls-compact-ecc.

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 26 August 2024.

Copyright Notice

Table of Contents

1. Introduction

The encodings used in the ECDHE groups secp256r1, secp384r1, and secp521r1 and the ECDSA signature algorithms ecdsa_secp256r1_sha256, ecdsa_secp384r1_sha384, and ecdsa_secp521r1_sha512 have significant overhead and the ECDSA encodings produces variable-length signatures. This document defines new optimal fixed-length encodings and registers new ECDHE groups and ECDSA signature algorithms using these new encodings. The new encodings reduce the size of the ECDHE groups with 33, 49, and 67 bytes and the ECDSA algorithms with an average of 7 bytes. These new encodings also work in DTLS 1.3

[RFC9147] and are especially useful in cTLS [I-D.ietf-tls-ctls]. When secp256r1 and ecdsa_secp256r1_sha256 are used as a replacement for the old encodings they reduce the size of a mutually authenticated TLS handshake with on average 80 bytes. The new encodings have the same security properties and requirements as the old encodings.

2. Conventions and Definitions

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.

3. Compact ECDHE Encoding

The encoding specified in [RFC8446] of the ECDHE groups secp256r1, secp384r1, and secp521r1 [RFC8422] have significant overhead. This document specifies a new optimal fixed-length encoding for the groups. The new encoding is defined as a compression of the UncompressedPointRepresentation structure. Given a UncompressedPointRepresentation structure [RFC8446]

```
struct {
    uint8 legacy_form = 4;
    opaque X[coordinate_length];
    opaque Y[coordinate_length];
} UncompressedPointRepresentation;
```

the legacy_form and Y field are omitted to create a CompactRepresentation structure.

```
struct {
    opaque X[coordinate_length];
} CompactRepresentation;
```

The resulting groups are called secp256r1_compact, secp384r1_compact, and secp521r1_compact. The new encodings have CompactRepresentation structures of length 32, 48, and 66 bytes, and reduce the size with 33, 49, and 67 bytes respectively. For secp256r1_compact, secp384r1_compact, and secp521r1_compact the opaque key_exchange field contains the serialized value of the CompactRepresentation struct.

Value	Description	Recommended	Reference
TBD1	secp256r1_compact	Y	[This-Document]
TBD2	secp384r1_compact	Y	[This-Document]

Value	Description	Recommended	Reference
TBD3	secp521r1_compact	Y	[This-Document]

Table 1: Compact ECDHE Groups

The difference between compact representation [RFC6090] and point compression [SECG]) is that point compression also communicates the sign bit of the y-coordinate along with the x-coordinate while compact representation only transmits the x-coordinate.

3.1. Example Compact ECDHE Encoding

The following shows an example compact ECDHE encoding. Figure 1 shows a 65 bytes secp256r1 UncompressedPointRepresentation structure.

```
04 A6 DA 73 92 EC 59 1E 17 AB FD 53 59 64 B9 98
94 D1 3B EF B2 21 B3 DE F2 EB E3 83 0E AC 8F 01
51 81 26 77 C4 D6 D2 23 7E 85 CF 01 D6 91 0C FB
83 95 4E 76 BA 73 52 83 05 34 15 98 97 E8 06 57
80
```

Figure 1: secp256r1

Figure 2 shows the 32 bytes secp256r1_compact CompactRepresentation structure encoding of the same key share.

```
A6 DA 73 92 EC 59 1E 17 AB FD 53 59 64 B9 98 94
D1 3B EF B2 21 B3 DE F2 EB E3 83 0E AC 8F 01 51
```

Figure 2: secp256r1_compact

3.2. Implementation Considerations for Compact Representation

For compatibility with APIs a compressed y-coordinate might be required. For compatibility with APIs that do not support the full [SECG] format an uncompressed y-coordinate might be required. For point validation an uncompressed y-coordinate is required. Using the notation in [SECG]:

*If a compressed y-coordinate is required, then the value ~yp set to zero can be used. The compact representation described above can in such a case be transformed into the SECG point compressed format by prepending X with the single byte 0x02 (i.e., M = 0x02 || X).

*If an uncompressed y-coordinate is required, then a y-coordinate has to be calculated following Section 2.3.4 of [SECG] or Appendix C of [RFC6090]. Any of the square roots (see [SECG] or

[RFC6090]) can be used. The uncompressed SECG format is M = 0x04 || X || Y.

For example: The curve P-256 has the parameters (using the notation in [RFC6090])

*p = $2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$

*a = -3

*b = 41058363725152142129326129780047268409114441015993725554835256314039467401291

Given an example x:

*x = 11579208918339630209554680715474055844340679510865333639897069777272788799766525

we can calculate y as the square root w = $(x^3 + a \cdot x + b)^{((p + 1)/4)} \pmod p$

*y = 83438718007019280682007586491862600528145125996401575416632522940595860276856

Note that this does not guarantee that (x, y) is on the correct elliptic curve. A full validation according to Section 5.6.2.3.3 of [SP-800-56A] is done by also checking that $0 \le x < p$ and that $y^2 \equiv x^3 + a \cdot x + b \pmod p$. The implementation **MUST** perform public-key validation.

4. Compact ECDSA Encoding

The variable-length encoding of the ECDSA signature algorithms ecdsa_secp256r1_sha256, ecdsa_secp384r1_sha384, and ecdsa_secp521r1_sha512 specified in [RFC8446] have significant overhead.

This document specifies a new optimal fixed-length encoding for the algorithms. The new encoding is defined as a compression of the DER-encoded ECDSA-Sig-Value structure. Given a DER-encoded ECDSA-Sig-Value structure [RFC8422]

```
Ecdsa-Sig-Value ::= SEQUENCE {
    r       INTEGER,
    s       INTEGER
}
```

the SEQUENCE type, INTEGER type, and length fields are omitted and if necessary the two INTEGER value fields are truncated (at most a single zero byte) or left padded with zeroes to the fixed length L.

For secp256r1, secp384r1, and secp521r1, L is 32, 48, and 66 bytes respectively. The resulting signatures are called ecdsa_secp256r1_sha256_compact, ecdsa_secp384r1_sha384_compact, and ecdsa_secp521r1_sha512_compact and has length 64, 96, and 132 bytes respectively. The new encodings reduce the size of the signatures with an average of 7 bytes. For secp256r1_compact, secp384r1_compact, and secp521r1_compact the opaque signature field contains the compressed Ecdsa-Sig-Value.

Value	Description	Recommended	Reference
TBD4	ecdsa_secp256r1_sha256_compact	Y	[This-Document]
TBD5	ecdsa_secp384r1_sha384_compact	Y	[This-Document]
TBD6	ecdsa_secp521r1_sha512_compact	Y	[This-Document]

Table 2: Compact ECDSA Signature Algorithms

4.1. Example Compact ECDSA Encoding

The following shows an example compact ECDSA encoding. Figure 3 shows a 71 bytes DER encoded ecdsa_secp256r1_sha256 ECDSA-Sig-Value structure. The values on the left are the ASN.1 tag (in hexadecimal) and the length (in decimal).

```
30  69: SEQUENCE {
02  33:  INTEGER
          00 D7 A4 D3 4B D5 4F 55 FE E1 A8 96 25 67 8C 3D
          D5 E5 F6 0D AC 73 EC 94 0C 5C 7B 93 04 A0 20 84
          A9
02  32:  INTEGER
          28 9F 59 5E D4 88 B9 AC 68 9A 3D 19 2B 1A 8B B3
          8F 34 AF 78 74 C0 59 C9 80 6A 1F 38 26 93 53 E8
          }
```

Figure 3: ecdsa_secp256r1_sha256

Figure 4 shows the 64 bytes ecdsa_secp256r1_sha256_compact encoding of the same signature.

```
D7 A4 D3 4B D5 4F 55 FE E1 A8 96 25 67 8C 3D D5
E5 F6 0D AC 73 EC 94 0C 5C 7B 93 04 A0 20 84 A9
28 9F 59 5E D4 88 B9 AC 68 9A 3D 19 2B 1A 8B B3
8F 34 AF 78 74 C0 59 C9 80 6A 1F 38 26 93 53 E8
```

Figure 4: ecdsa_secp256r1_sha256_compact

5. Security Considerations

The new encodings are just encodings and have the same security properties and security requirements as the old encodings. Compact representation of a ECDHE key share produces the same shared secret as the uncompressed encoding and does not change any requirements on point validation, the peers **MUST** validate each other's public key shares.

6. IANA Considerations

IANA is requested to update the TLS Supported Groups registry [RFC8447] under the Transport Layer Security (TLS) Parameters heading with the contents of Table 1.

IANA is requested to update the TLS SignatureScheme registry [RFC8447] under the Transport Layer Security (TLS) Parameters heading with the contents of Table 2.

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/rfc/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/rfc/rfc8174>.

[RFC8422] Nir, Y., Josefsson, S., and M. Pegourie-Gonnard, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and Earlier", RFC 8422, DOI 10.17487/RFC8422, August 2018, <https://www.rfc-editor.org/rfc/rfc8422>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/rfc/rfc8446>.

[RFC8447] Salowey, J. and S. Turner, "IANA Registry Updates for TLS and DTLS", RFC 8447, DOI 10.17487/RFC8447, August 2018, <https://www.rfc-editor.org/rfc/rfc8447>.

7.2. Informative References

[I-D.ietf-tls-ctls] Rescorla, E., Barnes, R., Tschofenig, H., and B. M. Schwartz, "Compact TLS 1.3", Work in Progress,

Internet-Draft, draft-ietf-tls-ctls-09, 23 October 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-ctls-09>.

[RFC6090] McGrew, D., Igoe, K., and M. Salter, "Fundamental Elliptic Curve Cryptography Algorithms", RFC 6090, DOI 10.17487/RFC6090, February 2011, <https://www.rfc-editor.org/rfc/rfc6090>.

[RFC9147] Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", RFC 9147, DOI 10.17487/RFC9147, April 2022, <https://www.rfc-editor.org/rfc/rfc9147>.

[SECG] "Standards for Efficient Cryptography 1 (SEC 1)", May 2009, <https://www.secg.org/sec1-v2.pdf>.

[SP-800-56A] Barker, E., Chen, L., Roginsky, A., Vassilev, A., and R. Davis, "Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography", NIST Special Publication 800-56A Revision 3, April 2018, <https://doi.org/10.6028/NIST.SP.800-56Ar3>.

Acknowledgments

The authors want to thank Dan Brown, Scott Fluhrer, and Erik Thormarker for their valuable comments and feedback.

Authors' Addresses

John Preuß Mattsson
Ericsson AB
SE-164 80 Stockholm
Sweden

Email: john.mattsson@ericsson.com

Hannes Tschofenig

Email: Hannes.Tschofenig@gmx.net