Abstract

This document specifies the use of several ECC Brainpool curves for authentication and key exchange in the Transport Layer Security (TLS) protocol version 1.3.

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 October 25, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.
1. Introduction

In [RFC5639], a new set of elliptic curve groups over finite prime fields for use in cryptographic applications was specified. These groups, denoted as ECC Brainpool curves, were generated in a verifiably pseudo-random way and comply with the security requirements of relevant standards from ISO [ISO1] [ISO2], ANSI [ANSI1], NIST [FIPS], and SecG [SEC2].

[RFC8422] defines the usage of elliptic curves for authentication and key agreement in TLS 1.2 and earlier versions, and [RFC7027] defines the usage of the ECC Brainpool curves for authentication and key exchange in TLS. The latter is applicable to TLS 1.2 and earlier versions, but not to TLS 1.3 that deprecates the ECC Brainpool Curve IDs registered for the use of ECC Brainpool Curves in earlier TLS versions.

The negotiation of ECC Brainpool Curves for key exchange in TLS 1.3 according to [RFC8446] requires the definition and assignment of additional NamedGroup IDs. Analogously, the negotiation of ECC Brainpool Curves for authentication requires the definition and assignment of additional SignatureScheme IDs. This document specifies such values for three curves from [RFC5639].

2. Brainpool NamedGroup Types

According to [RFC8446], the name space NamedGroup is used for the negotiation of elliptic curve groups for key exchange during a handshake starting a new TLS session. This document adds new NamedGroup types to three elliptic curves defined in [RFC5639] as follows.
The encoding of ECDHE parameters for sec256r1, secp384r1, and secp521r1 as defined in section 4.2.8.2 of [RFC8446] also applies to this document.

Test vectors for a Diffie-Hellman key exchange using these elliptic curves are provided in Appendix A.

3. Brainpool SignatureScheme Types

According to [RFC8446], the name space SignatureScheme is used for the negotiation of elliptic curve groups for authentication via the "signature_algorithms" extension. This document adds new SignatureScheme types to three elliptic curves defined in [RFC5639] as follows.

    enum {
        ecdsa_brainpoolP256r1tls13_sha256(0x081A),
        ecdsa_brainpoolP384r1tls13_sha384(0x081B),
        ecdsa_brainpoolP512r1tls13_sha512(0x081C)
    } SignatureScheme;

This notation is used to clarify that an ECDSA signature is calculated over the hashed message.

4. IANA Considerations

IANA is requested to update the references for the ECC Brainpool curves listed in the Transport Layer Security (TLS) Parameters registry "TLS Supported Groups" [IANA-TLS] to this document.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>DTLS-OK</th>
<th>Recommended</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>brainpoolP256r1tls13</td>
<td>Y</td>
<td>N</td>
<td>This doc</td>
</tr>
<tr>
<td>32</td>
<td>brainpoolP384r1tls13</td>
<td>Y</td>
<td>N</td>
<td>This doc</td>
</tr>
<tr>
<td>33</td>
<td>brainpoolP512r1tls13</td>
<td>Y</td>
<td>N</td>
<td>This doc</td>
</tr>
</tbody>
</table>

Table 1
IANA is requested to update the references for the ECC Brainpool curves in the Transport Layer Security (TLS) Parameters registry "TLS SignatureScheme" [IANA-TLS] to this document.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>DTLS-OK</th>
<th>Recommended</th>
<th>Reference</th>
</tr>
</thead>
</table>
| 0x081A       | ecdsa_brainpoolP256r
    1tls13_sha256 | Y       | N           | This doc    |
| 0x081B       | ecdsa_brainpoolP384r
    1tls13_sha384 | Y       | N           | This doc    |
| 0x081C       | ecdsa_brainpoolP512r
    1tls13_sha512 | Y       | N           | This doc    |

Table 2

5. Security Considerations

The security considerations of [RFC8446] apply accordingly.

The confidentiality, authenticity and integrity of the TLS communication is limited by the weakest cryptographic primitive applied. In order to achieve a maximum security level when using one of the elliptic curves from Table 1 for key exchange and / or one of the signature algorithms from Table 2 for authentication in TLS, the key derivation function, the algorithms and key lengths of symmetric encryption and message authentication as well as the algorithm, bit length and hash function used for signature generation should be chosen according to the recommendations of [NIST800-57] and [RFC5639]. Furthermore, the private Diffie-Hellman keys should be selected with the same bit length as the order of the group generated by the base point G and with approximately maximum entropy.

Implementations of elliptic curve cryptography for TLS may be susceptible to side-channel attacks. Particular care should be taken for implementations that internally transform curve points to points on the corresponding "twisted curve", using the map \((x',y') = (xZ^2, yZ^3)\) with the coefficient \(Z\) specified for that curve in [RFC5639], in order to take advantage of an an efficient arithmetic based on the twisted curve’s special parameters \((A = -3)\): although the twisted curve itself offers the same level of security as the corresponding random curve (through mathematical equivalence), arithmetic based on small curve parameters may be harder to protect against side-channel attacks. General guidance on resistance of elliptic curve
cryptography implementations against side-channel-attacks is given in [BSI1] and [HMV].

6. References

6.1. Normative References


6.2. Informative References


Appendix A. Test Vectors

This section provides some test vectors for example Diffie-Hellman key exchanges using each of the curves defined in Table 1. In all of the following sections the following notation is used:

\[\begin{align*}
  d_A &: \text{the secret key of party A} \\
  x_{qA} &: \text{the x-coordinate of the public key of party A} \\
  y_{qA} &: \text{the y-coordinate of the public key of party A} \\
  d_B &: \text{the secret key of party B} \\
  x_{qB} &: \text{the x-coordinate of the public key of party B} \\
  y_{qB} &: \text{the y-coordinate of the public key of party B} \\
  x_Z &: \text{the x-coordinate of the shared secret that results from completion of the Diffie-Hellman computation, i.e. the hex representation of the pre-master secret} \\
  y_Z &: \text{the y-coordinate of the shared secret that results from completion of the Diffie-Hellman computation}
\end{align*}\]

The field elements \(x_{qA}, y_{qA}, x_{qB}, y_{qB}, x_Z, y_Z\) are represented as hexadecimal values using the FieldElement-to-OctetString conversion method specified in [SEC1].

A.1. 256 Bit Curve

Curve brainpoolP256r1

\[\begin{align*}
  d_A &= \text{81DB1EE100150FF2EA338D708271BE38300CB54241D79950F77B063039804F1D} \\
  x_{qA} &= \text{44106E913F92BC02A1705D9953A8414DB95E1AAA49E81D9E85F929A8E3100BE5} \\
  y_{qA} &= \text{8AB4846F11CACCBB73CE49CBDD120F5A900A69FD32C272223F789EF10EB089BDC} \\
  d_B &= \text{55E40BC41E37E32AD25C3C6654511FFA8474A91A0032087593852D3E7D76BD3} \\
  x_{qB} &= \text{8D2D688C6CF93E1160AD04CC4429117DC2C41825E1E9FCA0ADDD34E6F1B39F7B}
\end{align*}\]
\[ y_{qB} = 990C57520812BE512641E47034832106BC7D3E8DD0E4C7F1136D7006547C6A \]
\[ x_Z = 89AFC39D41D3B327814B80940B042590F96556EC916AE7939BCE31F3A18BF2B \]
\[ y_Z = 49C27868F4ECA2179BFD7D59B1E3BF34C1DBDE61AE12931648F43E59632504DE \]

A.2. 384 Bit Curve

Curve brainpoolP384r1

\[ dA = 1E20F5E048A5886F1F157C74E91BDE2B98C8B52D58E5003D57053FC4B0BD6 \]
\[ 5D6F15EB5D1EE1610DF870795143627D042 \]
\[ x_qA = 68B665DD91C195800650CDD363C625F4E742E8134667B767B1B47679358 \]
\[ 8F885AB698C852D4A6E77A252D6380FCAF068 \]
\[ y_qA = 55BC91A399EC01DEE36017B7673A931236D2F1F5C83942D049E3FA206 \]
\[ 07493E0D038FF2FD302CA6767D1585F7FAA59 \]
\[ dB = 032640BC6003C59260F7250C3DB58CE647F98E1260ACCE4ACDA3D869F74E \]
\[ 01F8BA5E03239538D6A9831497ABAC96670 \]
\[ x_qB = 4D44326F269A597A5B58BBA565DA5556ED7FD9A8A9EB76C25F46DB69D19 \]
\[ DC8CE6AD18E404B15738B2086DF37E71D1EB4 \]
\[ y_qB = 62D692136DE56CE93BF5FA3188EF58BC8A3A0EC6C1E151A21038A42E91 \]
\[ 85329B5B275903D192F8D4E1F32FE9CC78C48 \]
\[ x_Z = 0BD93A7EA0B3D519D09D8E48D0785FB744A6B355E6304BC51C229FBCE2 \]
\[ 39BBADF6403715C35D4FB2A5444F575D4F42 \]
\[ y_Z = 0DF213417EBE4D84EA5F76F66C56470C489A3478D146DECF6DF0D94BAE9 \]
\[ E598157290F8756066975F1DB34B2324B7BD \]

A.3. 512 Bit Curve

Curve brainpoolP512r1

\[ dA = 16302FF0D4BBB5A8D733DAB7141C1B45ACBC8715939677F6A56850A38BD87B \]
\[ D59B09E80279609FF33EB9D4C061231FB26F92EEB04982A5F1D1764CAD5766542 \]
\[ 2 \]
\[ x_qA = 0A420517E406A0C0ACDCE90FCD71487718D3B953EF7FBEC5F7F27E28C6 \]
\[ 149999397E91E029E6457DB2D3E640668B392C2A7E737A7F0BF04436D11640F0D \]
\[ 9FD \]
y_qA = 72E6882E8DB28AAD36237CD25D580DB23783961C8DC52DFA2EC138AD472A0FCEF3887CF62B623B2A87DE5C588301EA3E5FC269B373B60724F5E82A6AD147FDE7

dB = 230E18E1BCC88A362FA54E4EA3902009292F7F8033624FD471B5D8ACE49D12CFABBC19963DAB8E2F1EBA00BFFB29E4D72D13F2224562F405CB80503666B25429

x_qB = 9D45F66DE5D67E2E6DB6E93A59CE0BB48106097FF78A081DE781CDB31FCE8CCBAAEA8DD4320C4119F1E9CD437A2EAB3731FA9668AB268D871DEDA55A5473199F

y_qB = 2FDC313095BCDD5FB3A91636F07A959C8E86B5636A1E930E8396049CB481961D365CC11453A06C719835475B12CB52FC3C383BCE35E27EF194512B71876285FA

x_Z = A7927098655F1F9976FA50A9D566865DC530331846381C87256BAF3226244B76D36403C024D7BBF0AA0803EAFF405D3D24F11A9B5C0BEF679FE1454B21C4CD1F

y_Z = 7DB71C3DEF63212841C463E881BDCF055523BD368240E6C3143BD8DEF8B3B3223B95E0F53082FF5E412F4222537A43DF1C6D25729DB51620A832BE6A26680A2

Authors’ Addresses

Leonie Bruckert
secunet Security Networks
Ammonstr. 74
01067 Dresden
Germany

Phone: +49 201 5454 3819
EMail: leonie.bruckert@secunet.com

Johannes Merkle
secunet Security Networks
Mergenthaler Allee 77
65760 Eschborn
Germany

Phone: +49 201 5454 3091
EMail: johannes.merkle@secunet.com