Elliptic Curve Cryptography (ECC) Brainpool Curves for Transport Layer Security (TLS)

Abstract

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

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7027.

Copyright Notice

Copyright (c) 2013 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 (http://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

[ RFC5639 ] specifies a new set of elliptic curve groups over finite prime fields for use in cryptographic applications. 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].

[ RFC4492 ] defines the usage of elliptic curves for authentication and key agreement in TLS 1.0 and TLS 1.1; these mechanisms may also be used with TLS 1.2 [RFC5246]. While the ASN.1 object identifiers defined in [ RFC5639 ] already allow usage of the ECC Brainpool curves for TLS (client or server) authentication through reference in X.509 certificates according to [ RFC3279 ] and [ RFC5480 ], their negotiation for key exchange according to [ RFC4492 ] requires the definition and assignment of additional NamedCurve IDs. This document specifies such values for three curves from [ RFC5639 ].

2. Brainpool NamedCurve Types

According to [ RFC4492 ], the name space NamedCurve is used for the negotiation of elliptic curve groups for key exchange during a handshake starting a new TLS session. This document adds new NamedCurve types to three elliptic curves defined in [ RFC5639 ] as follows:

```c
enum {
    brainpoolP256r1(26),
    brainpoolP384r1(27),
    brainpoolP512r1(28)
} NamedCurve;
```

These curves are suitable for use with Datagram TLS [ RFC6347 ].
Test vectors for a Diffie-Hellman key exchange using these elliptic curves are provided in Appendix A.

3. IANA Considerations

IANA has assigned numbers for the ECC Brainpool curves listed in Section 2 in the "EC Named Curve" [IANA-TLS] registry of the "Transport Layer Security (TLS) Parameters" registry as follows:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>DTLS-OK</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>brainpoolP256r1</td>
<td>Y</td>
<td>RFC 7027</td>
</tr>
<tr>
<td>27</td>
<td>brainpoolP384r1</td>
<td>Y</td>
<td>RFC 7027</td>
</tr>
<tr>
<td>28</td>
<td>brainpoolP512r1</td>
<td>Y</td>
<td>RFC 7027</td>
</tr>
</tbody>
</table>

Table 1

4. Security Considerations

The security considerations of [RFC5246] apply to the ECC Brainpool curves described in this document.

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 authentication and/or key exchange 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 \((x',y') = (x*Z^2, y*Z^3)\) with the coefficient \(Z\) specified for that curve in [RFC5639], in order to take advantage of 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), an 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].

5. References

5.1. Normative References


5.2. 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. The following notation is used in the subsequent sections:

- \( d_A \): the secret key of party A
- \( x_{qA} \): the x-coordinate of the public key of party A
- \( y_{qA} \): the y-coordinate of the public key of party A
- \( d_B \): the secret key of party B
- \( x_{qB} \): the x-coordinate of the public key of party B
- \( y_{qB} \): the y-coordinate of the public key of party B
- \( x_Z \): 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
- \( y_Z \): the y-coordinate of the shared secret that results from completion of the Diffie-Hellman computation

The field elements \( x_{qA}, y_{qA}, x_{qB}, y_{qB}, x_Z, \) and \( y_Z \) are represented as hexadecimal values using the FieldElement-to-OctetString conversion method specified in [SEC1].
A.1. 256-Bit Curve

Curve brainpoolP256r1

\[ d_A = 81DB1EE100150FF2EA338D708271BE38300CB54241D79950F77B063039804F1D \]

\[ x_{qA} = 44106E913F92BC02A1705D9953A8414DB95E1AAA49E81D9E85F929A8E3100BE5 \]

\[ y_{qA} = 8AB4846F11CACCB73CE49CBDD120F5A900A69FD32C272223F789EF10EB089BDC \]

\[ d_B = 55E40BC41E37E3E2AD25C3C6654511FFA8474A91A0032087593852D3E7D76BD3 \]

\[ x_{dB} = 8D2D688C6CF93E1160AD04CC4429117DC2C41825E1E9FCA0ADDD34E6F1B39F7B \]

\[ y_{dB} = 990C57520812B512641E47034832106BC7D3E8DD0E4C7F1136D7006547CEC6A \]

\[ x_z = 89AFC39D41D3B327814B80940B042590F96556EC91E6AE7939BCE31F3A18BF2B \]

\[ y_z = 49C27868F4ECA2179BFD7D59B1E3BF34C1DBDE61AE12931648F43E59632504DE \]
A.2. 384-Bit Curve

Curve brainpoolP384r1

dA = 1E20F5E048A586F1F157C74E91BDE2B98C8B52D58E5003D57053FC4B0BD6 5D6F15EB5D1EE1610DF870795143627D042

x_qA = 68B665DD91C195800650CDD363C625F4E742E8134667B767B1B47679358 8F885AB698C852D4A6E77A252D6380FCAF068

y_qA = 55BC91A39C9EC01DEE36017B7D673A931236D2F1F5C83942D049E3FA206 07493E0D038FF2FD30C2AB67D15C85F7FAA59

dB = 032640BC6003C59260F7250C3DB58CE647F98E1260ACCE4ACDA3DD869F74E 01F8BA5E0324309DB6A9831497ABAC96670

x_qB = 4D44326F269A597A5B58BBA565DA5556ED7FD9A8A9EB76C25F46DB69D19 DC8CE6AD18E404B15738B2086DF37E71D1EB4

y_qB = 62D692136DE56CBE93BF5FA3188EF58BC8A3A0EC6C1E151A21038A42E91 85329B5B275903D192F8D4E1F32FE9CC78C48

x_Z = 0BD9D3A7E0B3D519D09D8E48D0785FB744A6B355E6304BC51C229FBBCE2 39BBAD6403715C35D4FB2A5444F575D4F42

y_Z = 0DF213417EB4E40A5F76F66C56470C489A3478D146DECF6DF0D94BAE9 E598157290F8756066975F1DB34B2324B7BD
A.3. 512-Bit Curve

Curve brainpoolP512r1

dA = 16302FF0DBBB5A8D733DAB7141C1B45ACBC87159396776A56850A38DB87BD59B09E80279609FF33EB94C061231F26F92E8B04982A5F1D1764CAD5766542D

x_qA = 0A420517E406AAC0ACDCE90FCD71487718D3B953EF7FBEC57F7F27E28C6149999397E91E029E06457DB2D3E640668B392CA7E737A7F0BF04436D11640FDD2

y_qA = 72E68728EDB8A2D36237CD25D580DB23783961C8DC52DFA2EC138AD472A0FCEF3887CF62B623B2A87DE5C588301EA3E5FC269B373B60724F5E82A6AD147FD9FD

dB = 230E18E1BCC8A3627AF5A4EB43A902009292F7F8033624FD471DB5D8ACE49D12CFABBAC9963DAB8E2F1EBA00BFFB29E4D72D13F2224562F405CB80503666B25428

x_qB = 72E68728EDB8A2D36237CD25D580DB23783961C8DC52DFA2EC138AD472A0FCEF3887CF62B623B2A87DE5C588301EA3E5FC269B373B60724F5E82A6AD147FD9FD

y_qB = 72E68728EDB8A2D36237CD25D580DB23783961C8DC52DFA2EC138AD472A0FCEF3887CF62B623B2A87DE5C588301EA3E5FC269B373B60724F5E82A6AD147FD9FD

x_Z = A7927098655F1F9976FA5A9D566865DC530331846381C87256BAF3226244B76D36403C024D7BBF0AA0803EAF405D3D24F11A95C0BF679FE1454B21C4CD1F

y_Z = 7DB71C3DEF63212841C463E881BDCF055523BD368240E6C3143BD8DEFFB3B3223B95E0F53082FF5E412F4222537A43DF1C6D25729DDB51620A832BE6A26680A2
Authors’ Addresses

Johannes Merkle  
secunet Security Networks  
Mergenthaler Allee 77  
65760 Eschborn  
Germany  
Phone: +49 201 5454 3091  
EMail: johannes.merkle@secunet.com

Manfred Lochter  
Bundesamt fuer Sicherheit in der Informationstechnik (BSI)  
Postfach 200363  
53133 Bonn  
Germany  
Phone: +49 228 9582 5643  
EMail: manfred.lochter@bsi.bund.de