

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: August 19, 2016

O. Sury
CZ.NIC
R. Edmonds
Farsight Security, Inc.
February 16, 2016

Ed25519 for DNSSEC
draft-ietf-curdle-dnskey-ed25519-01

Abstract

This document describes how to specify Ed25519 keys and signatures in DNS Security (DNSSEC). It uses the Edwards-curve Digital Security Algorithm (EdDSA) with the Ed25519 parameter choice.

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 19, 2016.

Copyright Notice

Copyright (c) 2016 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.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. DNSKEY and RRSIG Resource Records for Ed25519	3
4. Examples	3
5. Acknowledgements	4
6. IANA Considerations	5
7. Implementation Status	5
8. Security Considerations	6
9. References	6
9.1. Normative References	6
9.2. Informative References	6
Authors' Addresses	7

1. Introduction

DNSSEC, which is broadly defined in [RFC4033], [RFC4034], and [RFC4035], uses cryptographic keys and digital signatures to provide authentication of DNS data. Currently, the most popular signature algorithm in use is RSA. [RFC5933] and [RFC6605] later defined the use of GOST and NIST specified elliptic curve cryptography in DNSSEC.

This document defines the use of DNSSEC's DS, DNSKEY, and RRSIG resource records (RRs) with a new signing algorithm, Edwards-curve Digital Signature Algorithm (EdDSA) with the Ed25519 parameter choice. A more thorough description of EdDSA and Ed25519 can be found in [I-D.irtf-cfrg-eddsa].

Concerns about the real-world security of elliptic curve cryptography have emerged since ECDSA was standardized for DNSSEC. The only two curves standardized for use with ECDSA in DNSSEC, NIST P-256 and NIST P-384, fail several of the [SafeCurves] security criteria and are considered "unsafe". This document adds an additional elliptic curve algorithm and parameter choice to DNSSEC, allowing additional flexibility.

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.

Ed25519 has a 128-bit security target, which is considered to be equivalent in strength to RSA with ~3000-bit keys. Ed25519 public keys are 256 bits (32 bytes) long while signatures are 512 bits (64 bytes) long.

The usage of elliptic curve cryptography in DNSSEC has advantages and disadvantages relative to RSA as already described in [RFC6605]. Even when compared to the use of RSA at reduced relative strengths (for instance, 1024- or 2048-bit RSA), Ed25519 still requires substantially smaller keys and signatures. The authors of the study Making the Case for Elliptic Curves in DNSSEC [ECCSIZE] came to the conclusion that using elliptic curve cryptography rather than RSA in DNSSEC can effectively prevent fragmentation of DNSSEC responses as well as significantly reduce the amplification attack potential in DNSSEC.

Signing with Ed25519 is significantly faster than signing with either equivalently strong RSA or the two existing curves standardized for use with the ECDSA algorithm in DNSSEC, while the validation of RSA signatures is still significantly faster than the validation of Ed25519 signatures. However, the authors of the TBD [ECCSPEED] study came to the conclusion that even if the deployment of elliptic curve cryptography in DNSSEC grows to cover 100% of the name space, a resolver will still be able to perform validation using a single CPU core.

2. 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].

3. DNSKEY and RRSIG Resource Records for Ed25519

An Ed25519 public key consists of a 32-byte value, which is encoded into the Public Key field of a DNSKEY resource record as a simple bit string. The generation of a public key is defined in Chapter 5.1.5 in [I-D.irtf-cfrg-eddsa].

An Ed25519 signature consists of a 64-byte value, which is encoded into the Signature field of an RRSIG resource record as a simple bit string. The Ed25519 signature algorithm is described in Chapter 5.1.6 in [I-D.irtf-cfrg-eddsa].

The algorithm number associated with the use of Ed25519 in DS, DNSKEY and RRSIG resource records is TBD. This registration is fully defined in the IANA Considerations section.

4. Examples

This section needs an update after the algorithm for Ed25519 is assigned.

```
Private-key-format: v1.2
Algorithm: TBD (ED25519)
PrivateKey: ODIyNjAzODQ2MjgwODAxMjI2NDUxOTAyMDQxNDIyNjI=
# corresponding to 82260384628080122645190204142262 INT
```

```
example.com. 3600 IN DNSKEY 257 3 TBD (
    102Woi0iS8Aa25FQkUd9RMzZHJpBoRQwAQEX1SxZJA4= )
```

```
example.com. 3600 IN DS 3613 TBD 2 (
    3aa5ab37efce57f737fc1627013fee07bdf241bd10f3
    b1964ab55c78e79a304b )
```

```
www.example.com. 3600 IN A 192.0.2.1
www.example.com. 3600 IN RRSIG A TBD 3 3600 (
    20150820000000 20150730000000 3613 example.com.
    cvTRVrU7dwnemQuBq9/E4t1IiRpvWcEmYdzqs6SCQxw6
    qmczBBQGldssMx1TCJnwsEs9ZuA2phPzuJNoon9BCA== )
```

```
Private-key-format: v1.2
Algorithm: TBD (ED25519)
PrivateKey: DSSF3o0s0f+ElWzj9E/Osxx8hLpk55chkmx0LYN5WiY=
```

```
example.com. 3600 IN DNSKEY 257 3 TBD (
    zPnZ/QwEe7S8C5SPz2OfS5RR40Atk2/rYnE9xHIEijs= )
```

```
example.com. 3600 IN DS 55648 TBD 2 (
    96401675bc7ecdd541ec0f70d69238c7b95d3bd4de1e
    231a068ceb214d02a4ed )
```

```
www.example.com. 3600 IN A 192.0.2.1
www.example.com. 3600 IN RRSIG A TBD 3 3600 (
    20150820000000 20150730000000 35452 example.com.
    yUGb9rCNiuhDaRJbuhYHj89Y/3Pi8KWUm7lOt00ivVRGvgulmVX8Dgpe
    AFyMP2MKXJrqYJr+ViiCIDwcOIbPAQ==)
```

5. Acknowledgements

Some of the material in this document is copied liberally from [RFC6605].

The authors of this document wish to thank Jan Vcelak, Pieter Lexis and Kees Monshouwer for a review of this document.

6. IANA Considerations

This document updates the IANA registry "Domain Name System Security (DNSSEC) Algorithm Numbers". The following entry has been added to the registry:

Number	TBD
Description	Ed25519
Mnemonic	ED25519
Zone Signing	Y
Trans. Sec.	*
Reference	This document

* There has been no determination of standardization of the use of this algorithm with Transaction Security.

7. Implementation Status

(Note to the RFC Editor: please remove this entire section as well as the reference to RFC 6982 before publication.)

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC6982]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC6982], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

TODO: Fill out this section.

8. Security Considerations

The security level of Ed25519 is slightly under the standard 128-bit level ([RFC7748]). Security considerations listed in [RFC7748] also apply to the usage of Ed25519 in DNSSEC. Such an assessment could, of course, change in the future if new attacks that work better than the ones known today are found.

9. References

9.1. Normative References

- [I-D.irtf-cfrg-eddsa] Josefsson, S. and I. Liusvaara, "Edwards-curve Digital Signature Algorithm (EdDSA)", draft-irtf-cfrg-eddsa-02 (work in progress), January 2016.
- [RFC2119] 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>>.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <<http://www.rfc-editor.org/info/rfc4033>>.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <<http://www.rfc-editor.org/info/rfc4034>>.
- [RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <<http://www.rfc-editor.org/info/rfc4035>>.
- [RFC7748] Langley, A., Hamburg, M., and S. Turner, "Elliptic Curves for Security", RFC 7748, DOI 10.17487/RFC7748, January 2016, <<http://www.rfc-editor.org/info/rfc7748>>.

9.2. Informative References

- [ECCSIZE] van Rijswijk-Deij, R., Speroto, A., and A. Pras, "Making the Case for Elliptic Curves in DNSSEC", 2015, <<http://www.sigcomm.org/ccr/papers/2015/October/0000000.0000002>>.

- [ECCSPEED] van Rijswijk-Deij, R. and K. Hageman, "TBD", 2016, <TBD>.
- [RFC5933] Dolmatov, V., Ed., Chuprina, A., and I. Ustinov, "Use of GOST Signature Algorithms in DNSKEY and RRSIG Resource Records for DNSSEC", RFC 5933, DOI 10.17487/RFC5933, July 2010, <<http://www.rfc-editor.org/info/rfc5933>>.
- [RFC6605] Hoffman, P. and W. Wijngaards, "Elliptic Curve Digital Signature Algorithm (DSA) for DNSSEC", RFC 6605, DOI 10.17487/RFC6605, April 2012, <<http://www.rfc-editor.org/info/rfc6605>>.
- [RFC6982] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", RFC 6982, DOI 10.17487/RFC6982, July 2013, <<http://www.rfc-editor.org/info/rfc6982>>.
- [SafeCurves] Bernstein, D. and T. Lange, "SafeCurves: choosing safe curves for elliptic-curve cryptography", 2016, <<http://safecurves.cr.yp.to/>>.

Authors' Addresses

Ondrej Sury
CZ.NIC
Milesovska 1136/5
Praha 130 00
CZ

Phone: +420 222 745 111
Email: ondrej.sury@nic.cz

Robert Edmonds
Farsight Security, Inc.
177 Bovet Rd #180
San Mateo, California 94402
US

Phone: +1 650 489 7919
Email: edmonds@fsi.io

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: September 9, 2016

O. Sury
CZ.NIC
R. Edmonds
Farsight Security, Inc.
March 8, 2016

Ed448 for DNSSEC
draft-ietf-curdle-dnskey-ed448-00

Abstract

This document describes how to specify Ed448 keys and signatures in DNS Security (DNSSEC). It uses the Edwards-curve Digital Security Algorithm (EdDSA) with the Ed448 parameter choice.

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 September 9, 2016.

Copyright Notice

Copyright (c) 2016 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.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. DNSKEY and RRSIG Resource Records for Ed448	3
4. Examples	4
5. Acknowledgements	4
6. IANA Considerations	4
7. Implementation Status	5
8. Security Considerations	5
9. References	6
9.1. Normative References	6
9.2. Informative References	6
Authors' Addresses	7

1. Introduction

DNSSEC, which is broadly defined in [RFC4033], [RFC4034], and [RFC4035], uses cryptographic keys and digital signatures to provide authentication of DNS data. Currently, the most popular signature algorithm in use is RSA. [RFC5933] and [RFC6605] later defined the use of GOST and NIST specified elliptic curve cryptography in DNSSEC.

This document defines the use of DNSSEC's DS, DNSKEY, and RRSIG resource records (RRs) with a new signing algorithm, Edwards-curve Digital Signature Algorithm (EdDSA) with the Ed448 parameter choice. A more thorough description of EdDSA and Ed448 can be found in [I-D.irtf-cfrg-eddsa].

Concerns about the real-world security of elliptic curve cryptography have emerged since ECDSA was standardized for DNSSEC. The only two curves standardized for use with ECDSA in DNSSEC, NIST P-256 and NIST P-384, fail several of the [SafeCurves] security criteria and are considered "unsafe". This document adds an additional elliptic curve algorithm and parameter choice to DNSSEC, allowing additional flexibility.

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.

Ed448 has a 224-bit security target, which is considered to be equivalent in strength to RSA with ~15000-bit keys. Ed448 public keys are 456 bits (57 bytes) long while signatures are 912 bits (114 bytes) long.

The usage of elliptic curve cryptography in DNSSEC has advantages and disadvantages relative to RSA as already described in [RFC6605]. Even when compared to the use of RSA at reduced relative strengths (for instance, 1024- or 2048-bit RSA), Ed448 still requires substantially smaller keys and signatures. The authors of the study Making the Case for Elliptic Curves in DNSSEC [ECCSIZE] came to the conclusion that using elliptic curve cryptography rather than RSA in DNSSEC can effectively prevent fragmentation of DNSSEC responses as well as significantly reduce the amplification attack potential in DNSSEC.

Ed448 is provided for those applications with relaxed performance requirements and where there is a desire to hedge against analytical attacks on elliptic curves. Still signing with Ed448 is significantly faster than signing with either equivalently strong RSA or the two existing curves standardized for use with the ECDSA algorithm in DNSSEC, while the validation of RSA signatures is still significantly faster than the validation of Ed448 signatures. However, the authors of the TBD [ECCSPEED] study came to the conclusion that even if the deployment of elliptic curve cryptography in DNSSEC grows to cover 100% of the name space, a resolver will still be able to perform validation using a single CPU core.

2. 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].

3. DNSKEY and RRSIG Resource Records for Ed448

An Ed448 public key consists of a 57-byte value, which is encoded into the Public Key field of a DNSKEY resource record as a simple bit string. The generation of a public key is defined in Chapter 5.2.5 in [I-D.irtf-cfrg-eddsa].

An Ed448 signature consists of a 114-byte value, which is encoded into the Signature field of an RRSIG resource record as a simple bit string. The Ed448 signature algorithm is described in Chapter 5.2.6 and verification of the Ed448 signature is described in Chapter 5.2.7 in [I-D.irtf-cfrg-eddsa].

The algorithm number associated with the use of Ed448 in DS, DNSKEY and RRSIG resource records is TBD. This registration is fully defined in the IANA Considerations section.

4. Examples

This section needs an update after the algorithm for Ed448 is assigned.

```
Private-key-format: v1.2
Algorithm: TBD (ED448)
PrivateKey: TBD
```

```
example.com. 3600 IN DNSKEY 257 3 TBD (
    TBD )
```

```
example.com. 3600 IN DS 3613 TBD 2 (
    TBD )
```

```
www.example.com. 3600 IN A 192.0.2.1
www.example.com. 3600 IN RRSIG A TBD 3 3600 (
    20150820000000 20150730000000 3613 example.com.
    TBD )
```

```
Private-key-format: v1.2
Algorithm: TBD (ED448)
PrivateKey: TBD
```

```
example.com. 3600 IN DNSKEY 257 3 TBD (
    TBD )
```

```
example.com. 3600 IN DS 55648 TBD 2 (
    TBD )
```

```
www.example.com. 3600 IN A 192.0.2.1
www.example.com. 3600 IN RRSIG A TBD 3 3600 (
    20150820000000 20150730000000 35452 example.com.
    TBD )
```

5. Acknowledgements

Some of the material in this document is copied liberally from [RFC6605].

The authors of this document wish to thank Jan Vcelak, Pieter Lexis and Kees Monshouwer for a review of this document.

6. IANA Considerations

This document updates the IANA registry "Domain Name System Security (DNSSEC) Algorithm Numbers". The following entry has been added to the registry:

Number	TBD
Description	Ed448
Mnemonic	ED448
Zone Signing	Y
Trans. Sec.	*
Reference	This document

* There has been no determination of standardization of the use of this algorithm with Transaction Security.

7. Implementation Status

(Note to the RFC Editor: please remove this entire section as well as the reference to RFC 6982 before publication.)

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC6982]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC6982], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

TODO: Fill out this section.

8. Security Considerations

The security level of Ed448 is slightly under the standard 128-bit level ([RFC7748]). Security considerations listed in [RFC7748] also apply to the usage of Ed448 in DNSSEC. Such an assessment could, of course, change in the future if new attacks that work better than the ones known today are found.

9. References

9.1. Normative References

- [I-D.irtf-cfrg-eddsa]
Josefsson, S. and I. Liusvaara, "Edwards-curve Digital Signature Algorithm (EdDSA)", draft-irtf-cfrg-eddsa-03 (work in progress), March 2016.
- [RFC2119] 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>>.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <<http://www.rfc-editor.org/info/rfc4033>>.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <<http://www.rfc-editor.org/info/rfc4034>>.
- [RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <<http://www.rfc-editor.org/info/rfc4035>>.
- [RFC7748] Langley, A., Hamburg, M., and S. Turner, "Elliptic Curves for Security", RFC 7748, DOI 10.17487/RFC7748, January 2016, <<http://www.rfc-editor.org/info/rfc7748>>.

9.2. Informative References

- [ECCSIZE] van Rijswijk-Deij, R., Speroto, A., and A. Pras, "Making the Case for Elliptic Curves in DNSSEC", 2015, <<http://www.sigcomm.org/ccr/papers/2015/October/0000000.0000002>>.
- [ECCSPEED]
van Rijswijk-Deij, R. and K. Hageman, "TBD", 2016, <TBD>.
- [RFC5933] Dolmatov, V., Ed., Chuprina, A., and I. Ustinov, "Use of GOST Signature Algorithms in DNSKEY and RRSIG Resource Records for DNSSEC", RFC 5933, DOI 10.17487/RFC5933, July 2010, <<http://www.rfc-editor.org/info/rfc5933>>.

- [RFC6605] Hoffman, P. and W. Wijngaards, "Elliptic Curve Digital Signature Algorithm (DSA) for DNSSEC", RFC 6605, DOI 10.17487/RFC6605, April 2012, <<http://www.rfc-editor.org/info/rfc6605>>.
- [RFC6982] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", RFC 6982, DOI 10.17487/RFC6982, July 2013, <<http://www.rfc-editor.org/info/rfc6982>>.
- [SafeCurves] Bernstein, D. and T. Lange, "SafeCurves: choosing safe curves for elliptic-curve cryptography", 2016, <<http://safecurves.cr.yp.to/>>.

Authors' Addresses

Ondrej Sury
CZ.NIC
Milesovska 1136/5
Praha 130 00
CZ

Phone: +420 222 745 111
Email: ondrej.sury@nic.cz

Robert Edmonds
Farsight Security, Inc.
155 Bovet Rd #476
San Mateo, California 94402
US

Phone: +1 650 489 7919
Email: edmonds@fsi.io

Internet Engineering Task Force
Internet-Draft
Updates: 4253, 4419, 4432, 4462, 5656
(if approved)
Intended status: Standards Track
Expires: September 15, 2016

M. Baushke
Juniper Networks, Inc.
March 14, 2016

Key Exchange (KEX) Method Updates and Recommendations for Secure Shell
(SSH)
draft-ietf-curdle-ssh-kex-sha2-03

Abstract

This document adds recommendations for adoption of ssh-curves from the [I-D.ietf-curdle-ssh-curves], adds some new Modular Exponential (MODP) Groups, and deprecates some previously specified Key Exchange Method algorithm names for the Secure Shell (SSH) protocol. It also updates [RFC4253], [RFC4419], [RFC4462], and [RFC5656] by specifying the set key exchange algorithms that currently exist and which ones MUST, SHOULD, MAY, and SHOULD NOT be implemented. New key exchange methods use the SHA-2 family of hashes.

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 September 15, 2016.

Copyright Notice

Copyright (c) 2016 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. Overview and Rationale

Secure Shell (SSH) is a common protocol for secure communication on the Internet. In [RFC4253], SSH originally defined the Key Exchange Method Name `diffie-hellman-group1-sha1` which used [RFC2409] Oakley Group 1 (a MODP group with 768 bits) and SHA-1 [RFC3174]. Due to recent security concerns with SHA-1 [RFC6194] and with MODP groups with less than 2048 bits [NIST-SP-800-131Ar1] implementer and users request support for larger MODP group sizes with data integrity verification using the SHA-2 family of secure hash algorithms as well as MODP groups providing more security.

The United States Information Assurance Directorate (IAD) at the National Security Agency (NSA) has published a FAQ [MFQ-U-OO-815099-15] suggesting that the use of Elliptic Curve Diffie-Hellman (ECDH) using the `nistp256` curve and SHA-2 based hashes less than SHA2-384 are no longer sufficient for transport of Top Secret information. It is for this reason that this draft moves `ecdh-sha2-nistp256` from a REQUIRED to OPTIONAL as a key exchange method. This is the same reason that the stronger MODP groups being introduced are using SHA2-512 as the hash algorithm. Group14 is already present in most SSH implementations and most implementations already have a SHA2-256 implementation, so `diffie-hellman-group14-sha256` is provided as an easy to implement and faster to use key exchange for small embedded applications.

It has been observed in [safe-curves] that the NIST recommended Elliptic Curve Prime Curves (P-256, P-384, and P-521) are perhaps not the best available for Elliptic Curve Cryptography (ECC) Security. For this reason, none of the [RFC5656] curves are marked as a MUST implement. However, the requirement that "every compliant SSH ECC implementation MUST implement ECDH key exchange" is now taken to mean that if `ecdsa-sha2-[identifier]` is implemented, then `ecdh-sha2-[identifier]` MUST be implemented.

Please send comments on this draft to curdle@ietf.org.

2. 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].

3. Key Exchange Algorithms

This memo adopts the style and conventions of [RFC4253] in specifying how the use of new data key exchange is indicated in SSH.

A new set of Elliptic Curve Diffie-Hellman ssh-curves exist. The curve25519-sha256 MUST be adopted where possible.

As a hedge against uncertainty raised by the NSA IAD FAQ publication, three new MODP Diffie-Hellman based key exchanges are proposed for inclusion in the set of key exchange method names as well as the curve448-sha512 curve.

The following new key exchange algorithms are defined:

Key Exchange Method Name	Note
diffie-hellman-group14-sha256	MAY/OPTIONAL
diffie-hellman-group16-sha512	SHOULD/RECOMMENDED
diffie-hellman-group18-sha512	MAY/OPTIONAL

Figure 1

The SHA-2 family of secure hash algorithms are defined in [FIPS-180-4].

The method of key exchange used for the name "diffie-hellman-group14-sha256" is the same as that for "diffie-hellman-group14-sha1" except that the SHA2-256 hash algorithm is used.

The group16 and group18 names are the same as those specified in [RFC3526] 4096-bit MODP Group 16 and 8192-bit MODP Group 18.

The SHA2-512 algorithm is to be used when "sha512" is specified as a part of the key exchange method name.

4. IANA Considerations

This document augments the Key Exchange Method Names in [RFC4253]. It downgrades the use of SHA-1 hashing for key exchange methods in [RFC4419], [RFC4432], and [RFC4462]. It also moves from MUST to MAY the ecdh-sha2-nistp256 given in [RFC5656].

It is desirable to also include the ssh-curves from the [I-D.ietf-curdle-ssh-curves] in this list. The "curve25519-sha256" is currently available in some Secure Shell implementations under the name "curve25519-sha256@libssh.org" and is the best candidate for a fast, safe, and secure key exchange method.

IANA is requested to update the SSH algorithm registry with the following entries:

Key Exchange Method Name	Reference	Note
diffie-hellman-group-exchange-sha1	RFC4419	SHOULD NOT
diffie-hellman-group-exchange-sha256	RFC4419	MAY
diffie-hellman-group1-sha1	RFC4253	SHOULD NOT
diffie-hellman-group14-sha1	RFC4253	SHOULD
ecdh-sha2-nistp256	RFC5656	MAY
ecdh-sha2-nistp384	RFC5656	SHOULD
ecdh-sha2-nistp521	RFC5656	SHOULD
ecdh-sha2-*	RFC5656	MAY
ecmqv-sha2	RFC5656	MAY
gss-gex-sha1-*	RFC4462	SHOULD NOT
gss-group1-sha1-*	RFC4462	SHOULD NOT
gss-group14-sha1-*	RFC4462	MAY
gss-*	RFC4462	MAY
rsa1024-sha1	RFC4432	SHOULD NOT
rsa2048-sha256	RFC4432	MAY
diffie-hellman-group14-sha256	This Draft	MAY
diffie-hellman-group16-sha512	This Draft	SHOULD
diffie-hellman-group18-sha512	This Draft	MAY
curve25519-sha256	ssh-curves	MUST
curve448-sha512	ssh-curves	MAY

Figure 2

The Note in the above table is an implementation suggestion/recommendation for the listed key exchange method. It is up to the end-user as to what algorithms they choose to be able to negotiate.

The guidance of this document is that the SHA-1 algorithm hashing SHOULD NOT be used. If it is used, it should only be provided for backwards compatibility, should not be used in new designs, and should be phased out of existing key exchanges as quickly as possible because of its known weaknesses. Any key exchange using SHA-1 SHOULD NOT be in a default key exchange list if at all possible. If they are needed for backward compatibility, they SHOULD be listed after all of the SHA-2 based key exchanges.

The RFC4253 REQUIRED diffie-hellman-group14-sha1 method SHOULD be retained for compatibility with older Secure Shell implementations.

It is intended that this key exchange be phased out as soon as possible.

5. Acknowledgements

Thanks to the following people for review and comments: Denis Bider, Peter Gutmann, Damien Miller, Niels Moeller, Matt Johnston, Iwamoto Kouichi, Simon Josefsson, Dave Dugal, Daniel Migault.

Thanks to the following people for code to implement interoperable exchanges using some of these groups as found in an this draft: Darren Tucker for OpenSSH and Matt Johnston for Dropbear. And thanks to Iwamoto Kouichi for information about RLogin, Tera Term (ttssh) and Poderosa implementations also adopting new Diffie-Hellman groups based on this draft.

6. Security Considerations

The security considerations of [RFC4253] apply to this document.

The security considerations of [RFC3526] suggest that these MODP groups have security strengths given in this table. They are based on [RFC3766] Determining Strengths For Public Keys Used For Exchanging Symmetric Keys.

Group modulus security strength estimates (RFC3526)

Group	Modulus	Strength Estimate 1		Strength Estimate 2	
		in bits	exponent size	in bits	exponent size
14	2048-bit	110	220-	160	320-
15	3072-bit	130	260-	210	420-
16	4096-bit	150	300-	240	480-
17	6144-bit	170	340-	270	540-
18	8192-bit	190	380-	310	620-

Figure 3

Many users seem to be interested in the perceived safety of using larger MODP groups and hashing with SHA2-based algorithms.

7. References

7.1. Normative References

- [FIPS-180-4]
National Institute of Standards and Technology, "Secure Hash Standard (SHS)", FIPS PUB 180-4, August 2015, <<http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf>>.
- [RFC2119] 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>>.
- [RFC3526] Kivinen, T. and M. Kojo, "More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)", RFC 3526, DOI 10.17487/RFC3526, May 2003, <<http://www.rfc-editor.org/info/rfc3526>>.
- [RFC4253] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Transport Layer Protocol", RFC 4253, DOI 10.17487/RFC4253, January 2006, <<http://www.rfc-editor.org/info/rfc4253>>.

7.2. Informative References

- [I-D.ietf-curdle-ssh-curves]
Adamantiadis, A. and S. Josefsson, "Secure Shell (SSH) Key Exchange Method using Curve25519 and Curve448", draft-ietf-curdle-ssh-curves-00 (work in progress), March 2016.
- [MFQ-U-OO-815099-15]
"National Security Agency/Central Security Service", "CNSA Suite and Quantum Computing FAQ", January 2016, <<https://www.iad.gov/iad/library/ia-guidance/ia-solutions-for-classified/algorithm-guidance/cnsa-suite-and-quantum-computing-faq.cfm>>.
- [NIST-SP-800-131Ar1]
Barker, and Roginsky, "Transitions: Recommendation for the Transitioning of the Use of Cryptographic Algorithms and Key Lengths", NIST Special Publication 800-131A Revision 1, November 2015, <<http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-131Ar1.pdf>>.

- [RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, DOI 10.17487/RFC2409, November 1998, <<http://www.rfc-editor.org/info/rfc2409>>.
- [RFC3174] Eastlake 3rd, D. and P. Jones, "US Secure Hash Algorithm 1 (SHA1)", RFC 3174, DOI 10.17487/RFC3174, September 2001, <<http://www.rfc-editor.org/info/rfc3174>>.
- [RFC3766] Orman, H. and P. Hoffman, "Determining Strengths For Public Keys Used For Exchanging Symmetric Keys", BCP 86, RFC 3766, DOI 10.17487/RFC3766, April 2004, <<http://www.rfc-editor.org/info/rfc3766>>.
- [RFC4419] Friedl, M., Provos, N., and W. Simpson, "Diffie-Hellman Group Exchange for the Secure Shell (SSH) Transport Layer Protocol", RFC 4419, DOI 10.17487/RFC4419, March 2006, <<http://www.rfc-editor.org/info/rfc4419>>.
- [RFC4432] Harris, B., "RSA Key Exchange for the Secure Shell (SSH) Transport Layer Protocol", RFC 4432, DOI 10.17487/RFC4432, March 2006, <<http://www.rfc-editor.org/info/rfc4432>>.
- [RFC4462] Hutzelman, J., Salowey, J., Galbraith, J., and V. Welch, "Generic Security Service Application Program Interface (GSS-API) Authentication and Key Exchange for the Secure Shell (SSH) Protocol", RFC 4462, DOI 10.17487/RFC4462, May 2006, <<http://www.rfc-editor.org/info/rfc4462>>.
- [RFC5656] Stebila, D. and J. Green, "Elliptic Curve Algorithm Integration in the Secure Shell Transport Layer", RFC 5656, DOI 10.17487/RFC5656, December 2009, <<http://www.rfc-editor.org/info/rfc5656>>.
- [RFC6194] Polk, T., Chen, L., Turner, S., and P. Hoffman, "Security Considerations for the SHA-0 and SHA-1 Message-Digest Algorithms", RFC 6194, DOI 10.17487/RFC6194, March 2011, <<http://www.rfc-editor.org/info/rfc6194>>.
- [safe-curves] Bernstein, and Lange, "SafeCurves: choosing safe curves for elliptic-curve cryptography.", February 2016, <<https://safecurves.cr.yp.to/>>.

Author's Address

Mark D. Baushke
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089-1228
US

Phone: +1 408 745 2952
Email: mdb@juniper.net
URI: <http://www.juniper.net/>

SIPCORE Working Group
Internet-Draft
Intended Status: Standards Track
Expires: January 7, 2017

Fuwen Liu
Minpeng Qi
Zuo Min
Chinamobile
July 7, 2016

SIP Authentication using the EC-SRP5 Protocol
draft-liu-sipcore-ec-srp5-03

Abstract

This document specifies how the elliptic curve secure remote protocol (EC-SRP) is applied to SIP authentication. SIP Client and server perform mutual authenticate by using the modern 'zero knowledge' method without disclosing the password in the process. It has low computation complexity and low bandwidth consumption due to the use of elliptical curve cryptography. This makes it more suitable for resource-constrained environments, e.g. wireless network. The security of the scheme is based on the computational intractability of the elliptic curve discrete logarithm problem. It is resilient to various kinds of attacks, including off-line dictionary attacks.

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/lid-abstracts.html>

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

Copyright and License Notice

Copyright (c) 2014 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.

Table of Contents

1	Introduction	3
1.1	Terminology	4
2	EC-SRP5 Protocol in SIP Authentication	4
2.1	Notation	5
2.2	Password Verifier	5
2.3	Protocol Overview	5
2.3.1	Initial Request	7
2.3.2	Response	7
2.3.3	Request	7
2.3.4	Confirmation	7
3	Security Considerations	8
3.1	Off-line dictionary attack resistance	8
3.2	On-line dictionary attack resistance	8
3.3	Man-in-the middle attack resistance	8
3.4	Replay attack resistance	9
4	Elliptic Curve Index	9
5	Acknowledgments	10
6	References	11
	Authors' Addresses	12
	Appendix A: Algorithm ECPEPKGP-SRP5-SERVER	13
	Appendix B: Algorithm ECSVDP-SRP5-CLIENT	13
	Appendix C: Algorithm ECSDVP-SRP5-SERVER	14

1 Introduction

SIP [1] applies HTTP digest authentication [2] [19] by default to performing user authentication. It is designed on the basis of the challenge-response mechanism, where the server presents the client a challenge (randomly-generated number), and the client responds with a valid answer which is generated by hashing the challenge in conjunction with the password. This is a weak authentication because it is possible for an attacker to recover the used password. Although passwords are not transmitted in a clear form over the insecure network, an adversary is still able to acquire the correct password by using a special variant of the brute-force attack: the off-line dictionary attack [3]. This results from the low entropy of a human-chosen password. The length of passwords mostly used in practice is rarely longer than 8 characters. It has merely about 30 bits of entropy (2^{30}) if the password is chosen by a human [4].

A Key-Derivation Authentication Scheme [5] has been proposed for SIP authentication. It creates a master-key by using a key-derivation function (KDF), whose inputs include a password, a salt, a key length, and an iteration count. A good example of KDF is HMAC [6]. The major difference between the HTTP digest authentication and the Key-Derivation Authentication is that the former performs the HMAC computation only once, while the latter computes HMAC n times, where n is the iteration count whose default value is 1000. This method could slow down the speed of off-line dictionary attacks. But it is not cryptographically secure as it needs just more 999 HMAC computations compared to the HTTP digest authentication when checking the correctness of a guessed password. Accordingly, the attacker can recover the password in a reasonable time using the off-line dictionary attacks.

Bellovin and Merritt first introduced an innovative password-based protocol, called DH-EKE (Diffie-Hellman Encrypted Key Exchange) protocol [7], to foil off-line dictionary attacks. Its basic idea is that two parties exchange ephemeral DH public keys encrypted with a shared password. Only the parties who know the password are able to authenticate each other and agree upon a session key for securing the communication. The computation complexity of off-line dictionary attacks on the DH-EKE protocol is equal to that of solving the discrete logarithm problem because the password is entangled with the ephemeral DH public key.

Inspired by the DH-EKE protocol, numerous password-based authentication key agreement protocols have been developed. Typical examples are the PAK (Password Authenticated Key exchange) protocol [8], SPEKE (Secure Password Exponential Key Exchange) protocol [9], AMP (Authentication via Memorable Password) protocol [10], and

SRP (Secure Remote Password) protocol [11]. They have been adopted as the standards by the IEEE computer society, including the elliptical curve (EC) variants of these protocols[12]. The EC-SRP5 protocol [13], a EC variant of the SRP protocol is chosen for SIP authentication in the demo, since it is much more efficient than the original SRP protocol regarding the computation and bandwidth consumption due to the use of elliptic curve cryptography.

Elliptic curve cryptography systems [14] are constructed by using elliptic curve over finite fields, which supersedes the conventional asymmetrical cryptographic algorithms, e.g., RSA and DH, in terms of computational and communicational burdens. ECC-based cryptographic systems require shorter key length and less computing power than conventional systems based on discrete logarithm problem (DLP). This is because the algorithms to solve the ECDLP run in a fully exponential time, while the sub-exponential time algorithms are available to address the discrete logarithm problem. The following table gives approximate comparable key sizes for symmetric- and asymmetric-key crypto systems based on the best-known algorithms for attacking them [15].

Symmetric	ECC	DH/DSA/RSA
112	224- 255	2048
128	256-383	3072
192	384-511	7680
256	511+	15360

Table 1: Comparable Key Sizes (in bits)

As shown in Table 1, compared to currently prevalent crypto systems such as RSA, ECC offers equivalent security with smaller key sizes. Smaller key sizes result in savings for power, memory, bandwidth, and computational cost that make ECC especially attractive for constrained environments, such as wireless environments. Another advantage of ECC is that some elliptic curves such as Curve25519 and Curve448[22] are resistant to a wide range of side-channel attacks, since they use constant-time implementation and an exception-free scalar multiplication.

1.1 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 [RFC2119].

2 EC-SRP5 Protocol in SIP Authentication

2.1 Notation

The terms used in the document are listed as follows:

ECI: elliptic curve index
G: a base point (xG, yG) on an elliptic curve
s: salt
Tc: client's temporary private key
Ts: server's temporary private key
Wc: client's public key
Ws: server's public key
Cc: client's confirmation value
Cs: server's confirmation value
Pw: password
v: password verifier
Z: shared secret between client and server
SIP-URI: Uniform Resource Identifier for SIP
containing user name and domain name

The | symbol denotes string concatenation, the * operator is the scalar point multiplication operation in an EC group, and the . operator is the integer multiplication.

2.2 Password Verifier

The password verifier is computed based on the salt s, uniform resource identifier SIP-URI, password Pw, and elliptic curve index ECI. The SHA-256 hash algorithm[18] is used as the hash function.

$$i = \text{OS2IP}(\text{SHA-256}(s \parallel \text{SHA-256}(\text{SIP-URI} \parallel ":" \parallel \text{Pw} \parallel \text{ECI})))$$
$$v = i * G$$

where OS2IP means octet string to integer conversion primitive, the derived password verifier v is actually a point on the elliptic curve indicated by the ECI.

The server then stores the following information in the database for each user:

- o SIP-URI
- o salt s
- o elliptic curve index ECI
- o password verifier v

2.3 Protocol Overview

The following flows describe the EC-SRP5 based SIP authentication mechanism at a high-level. The four messages are exchanged between the client and the server during the authentication procedure, which

are Initial Request, Response, Request, and Confirmation, respectively.

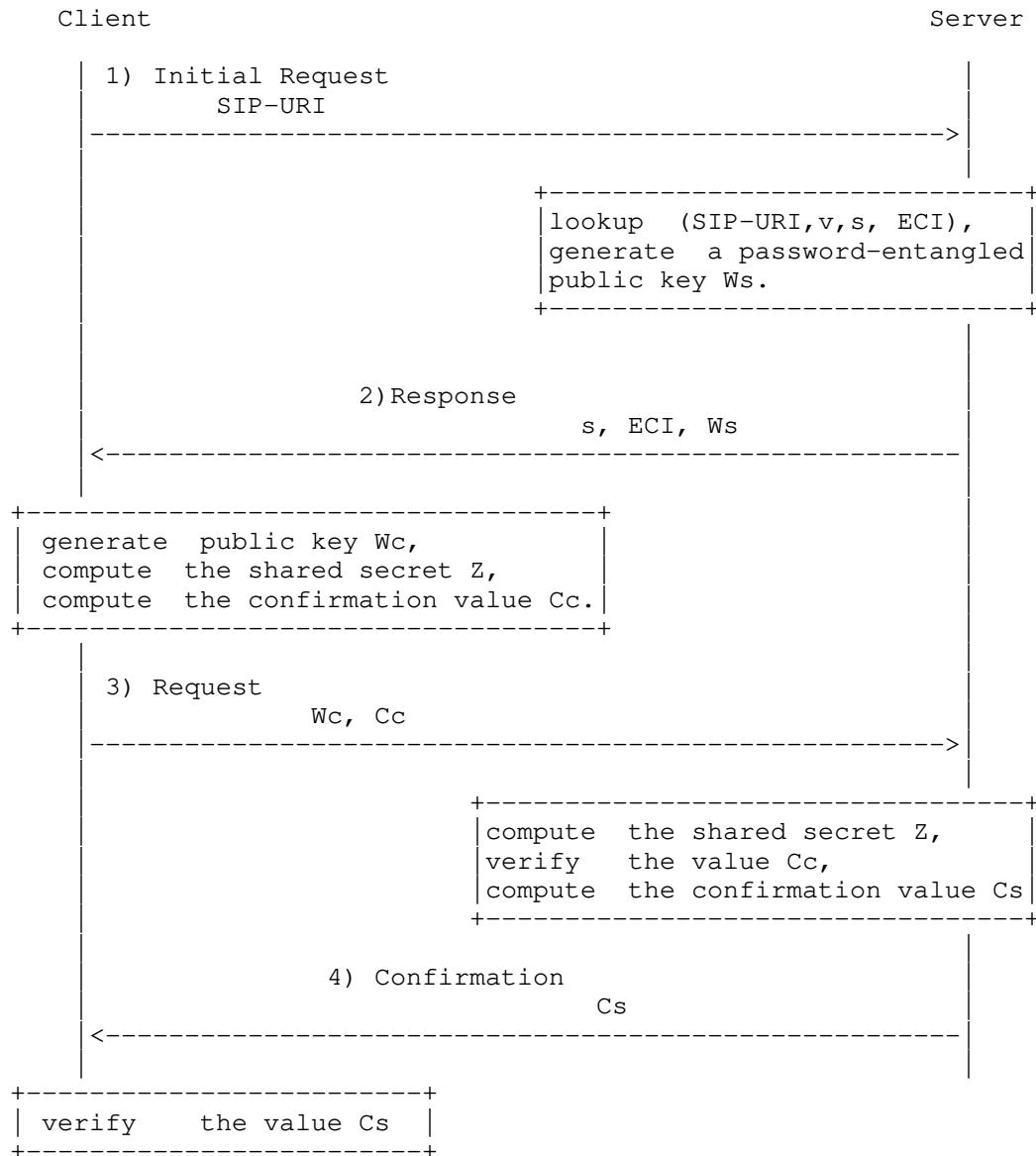


Fig.1 SIP Authentication procedure based on EC-SRP5 protocol

2.3.1 Initial Request

Authentication is initiated by the client to send the Initial Request message to the server, which contains the SIP-URI.

2.3.2 Response

When receiving the Initial Request from the client, the server lookups the database using the SIP-URI as the key for search, and fetches the password verifier v , salt s , and elliptic curve index ECI associated with the SIP-URI. The server generates a temporary private key T_s by randomly selecting an integer in the range $[1, r-1]$, where r is the order of the base point G . After that, a password-entangled public key W_s is computed by using the algorithm ECPEKGP-SRP5-SERVER (T_s, v), which is specified in Appendix A. The server then responds the Initial Request with the Response message containing the salt s , elliptic curve index ECI, and the public key W_s .

2.3.3 Request

When obtaining the Response message from the server, the client generates a temporary private key T_c by randomly choosing an integer from the interval $[1, r-1]$. The client's public key is created by calculating $W_c = T_c * G$. Then client checks the password-entangled public key W_s to validate whether it is a non-identity element of the parent group. This serves to prevent the simple substitution attacks[20]. If it is not true, the client MUST stop the authentication process. Otherwise the client derives the shared secret Z using the algorithm ECSVDP-SRP5-CLIENT (T_c, P_w, W_c, W_s, s) specified in Appendix B. The password verifier v is created as specified in Section 2.2 by using the password P_w , salt s , and elliptic curve index ECI. Then the client creates the confirmation value C_c by computing $C_c = \text{SHA-256}(\text{hex}(04), W_c, W_s, Z, v)$, in order to confirm the possession of the shared secret Z to the server. Then the client acknowledges the server with the Request message containing the public key W_c and confirmation value C_c .

2.3.4 Confirmation

When receiving the Request message from the client, the server verifies whether the public key W_c is a non-identity element of the parent group. If it is false, the server MUST abort the authentication. Otherwise the server computes the shared secret Z by applying the algorithm ECSDVP-SRP5-SERVER(T_s, v, W_c, W_s), which is detailed in Appendix C. Then the server calculates the expected confirmation value C_c' with respect to the client using $C_c' = \text{SHA-256}(\text{hex}(04), W_c, W_s, Z, v)$. If the expected confirmation value C_c' is not identical to the received confirmation value C_c , the server MUST terminate the authentication process. Otherwise the client is successfully authenticated by the server. Then server generated the

confirmation value C_s by computing $C_s = \text{SHA-256}(\text{hex}(03), W_c, W_s, Z, v)$, and sends it to the client. This serves to confirm the possession of the shared secret Z to the client.

Once obtaining the Confirmation message from the server, the client computes the expected confirmation value C_s' using $C_s' = \text{SHA-256}(\text{hex}(03), W_c, W_s, Z, v)$ to verify the received confirmation value C_s . If the expected confirmation value C_s' is not identical to the received confirmation value C_s , the client MUST abort the authentication. Otherwise the client authenticates the server successfully. At this point, the client and the server have completed the mutual authentication.

3 Security Considerations

3.1 Off-line dictionary attack resistance

The messages exchanged in the protocol are usually available to an eavesdropper. The message related to the password information is just the server's password-entangled public key W_s . An attacker, however, is not able to derive the password from the message W_s . This is because W_s is the addition of the two points in the group, i.e. $W_s = T_s * G + e_1$, see Appendix A. Password verifier is used as input selector value to choose a pseudo-random element e_1 of a group. The element e_1 is shadowed by adding the point $T_s * G$, which has high-grade entropy as T_s is the order of base point G which usually exceeds 192bits long. In this way, an attacker can not access the sensitive password information from the eavesdropped message W_s .

3.2 On-line dictionary attack resistance

An adversary launches on-line dictionary attacks by running the protocol with an honest party using a guessed password. Each time the protocol abandons the active adversary can eliminate one password. The attack itself is not a great threat to the use of the protocol, since this attack is trivial to detect in the sever by checking the confirmation value C_c . To prevent an attacker from guessing more passwords, the server usually blocks the user authentication when the times of authentication failure reach the default value set in advance.

3.3 Man-in-the middle attack resistance

The man-in-the middle (MITM) attack is that an adversary replaces the exchanged public keys W_s and W_c with its own public keys W_s' and W_c' in the middle, respectively. The object of the MITM attack is to fool the client and the server to believe that they communicate with each other using the shared secret Z . Actually the client talks to the

adversary with the shared secret Z' , and the server talks to the adversary with the shared secret Z'' . The EC-SRP5 protocol thwarts the MITM attacks by generating and verifying the confirmation value C_c and C_s in the client's side and server's side, respectively. In both client and server, the received confirmation value will not equal to the computed confirmation value when the MITM attack is launched, because the client and server do not have the shared secret Z .

- 3.4 Replay attack resistance The replay attack is that an adversary simply takes a previously sent message, and resends it later in an attempt to gain access to a network or resource. Provided that an adversary replays the messages sent by the server before, which are the Response message (s , ECI , W_s) and the Confirmation message (C_s).

The shared secret Z is computed based on the client's temporary private key T_c . In each authentication process the client will randomly generate a private key, which is completely different from the private keys used in past authentication processes. This implies that each authentication session has its unique shared secret Z . The confirmation value C_s thus will vary with the authentication session. As a result, the client can detect the replay attack by comparing C_s with the expected confirmation value C_s' .

4 Elliptic Curve Index

It is RECOMMENDED that the following elliptic curves are used, which are specified in [16] as well as in [21].

Description	ECI
secp224k1	1.3.132.0.32
secp224r1	1.3.132.0.33
secp256k1	1.3.132.0.10
secp256r1	1.2.840.10045.3.1.7
secp384r1	1.3.132.0.34
secp521k1	1.3.132.0.35
brainpoolP256r1	1.3.132.0.26
brainpoolP384r1	1.3.132.0.27
brainpoolP512r1	1.3.132.0.28

The elliptic curve identifier (ECI) is the string in the second column of the table, the ASCII representation of the object identifier (OID) of the curve.

5. Acknowledgments

The authors would like to thank Paul Kyzivat for his insight reviews and constructive suggestions, thanks also go to Xiaojun Zhuang, Ivy Guo, and Cathy Wang for their useful comments and suggestions.

6 References

- [1] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [2] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart: HTTP Authentication: Basic and Digest Access Authentication. IETF RFC 2617, June 1999.
- [3] T. Wu: A Real-World Analysis of Kerberos Password Security. Proceedings of the ISOC Symposium on Network and Distributed System Security, 1999.
- [4] W. E. Burr, D. F. Dodson, E. M. Newton, R. A. Perlner, W. T. Polk, S. Gupta, and E. A. Nabb : Electronic Authentication Guideline. NIST Special publication 800-63-2, August 2013.
- [5] R. Shekh-Yusef: Key-Derivation Authentication Scheme. IETF draft draft-yusef-sipcore-key-derivation-00, October 10, 2014.
- [6] H. Krawczyk, M. Bellare, and R. Canetti: HMAC: Keyed-Hashing for Message Authentication, RFC 2104, Febr. 1997.
- [7] S. Bellovin and M. Merritt: Encrypted Key Exchange: Password-based Protocols Secure against Dictionary Attacks. Proceedings of the IEEE Symposium on Research in Security and Privacy, May 1992.
- [8] P. MacKenzie: The PAK Suite: Protocols for Password-Authenticated Key Exchange. DIMACS Technical Report 002-46, October 2002.
- [9] D. Jablon: Strong Password-Only Authenticated Key Exchange. Computer Communication Review, ACM SIGCOMM 26 (1996) 5: 5-26.
- [10] T. Kwon: Authentication and Key Agreement via Memorable Password. NDSS 2001 Symposium Conference Proceedings, February 2001.
- [11] T. Wu: The Secure Remote Password Protocol. Proceedings of the 1998 Internet Society Network and Distributed System Security Symposium, San Diego, March 1998, pp. 97-111.
- [12] Wang, Y., "IEEE P1363.2 Submission / D2001-06-29," A contribution by Yongge Wang for P1363.2 giving an elliptic curve version of the SRP protocol, June 29, 2001.

- [13] IEEE P1363.2: Password-Based Public-Key Cryptography. September 2008.
- [14] D. Hankerson, A. Menezes, S. Vanstone: Guide to Elliptic Curve Cryptography. Springer, 2003.
- [15] NIST: Recommendation on Key Management, SP 800-57, August 2005.
- [16] SEC 2: Recommended Elliptic Curve Domain Parameters. Version 2.0, Jan. 2010.
- [17] IEEE 1363-2000: IEEE Standard Specifications for Public-Key Cryptography.
- [18] FIPS PUB 180-4: Secure Hash Standard. March 2012.
- [19] J. Franks, P. Hallam-Baker, J. Hostetler, P. Leach, A. Luotonen, E. Sink, L. Stewart: An Extension to HTTP : Digest Access Authentication. IETF RFC 2069, January 1997.
- [20] J. F. Raymond¹ and A. Stiglic: Security Issues in the Diffie-Hellman Key Agreement Protocol.
<http://crypto.cs.mcgill.ca/~stiglic/>
- [21] J. Merkle and M. Lochter: Elliptic Curve Cryptography (ECC) Brainpool Curves for Transport Layer Security (TLS). RFC 7027, October 2013.
- [22] A. Langley, M. Hamburg and S. Turner: Elliptic Curves for Security. RFC 7748, January 2016.

Authors' Addresses

Fuwen Liu
China Mobile
32 Xuanwumenxi Ave, Xicheng District
Beijing 100032
China

Email: liufuwen@chinamobile.com

Minpeng Qi
China Mobile
32 Xuanwumenxi Ave, Xicheng District
Beijing 100032
China

Email: qiminpeng@chinamobile.com

Min Zuo
China Mobile
32 Xuanwumenxi Ave, Xicheng District
Beijing 100032
China

Email: zuomin@chinamobile.com

Appendix A: Algorithm ECPEPKGP-SRP5-SERVER

ECPEPKGP-SRP5-SERVER is Elliptic Curve Password-Entangled Public Key Generation Primitive for server. The algorithm ECPEPKGP-SRP5-SERVER (T_s , v) is used to generate a elliptic curve password-entangled public key W_s , where the inputs are server's temporary private key T_s and password verifier v . The following steps are needed to compute the public key W_s :

- (1) Compute octet string $o1 = \text{GE2OSP-X}(v)$
- (2) Compute group element $e1 = \text{ECREDP}(o1)$
- (3) Compute group element $W_s = T_s * G + e1$
- (4) Output W_s as the password-entangled public key

Where GE2OSP-X is used to convert group elements into octet strings. ECREDP is Elliptic Curve Random Element Derivation Primitive [17]. The primitive uses a hash function of a password-based input selector value to select a pseudo-random element of a group to be used in the password-based authenticated key agreement scheme, in order to prevent collisions and obscure exponential relationships of output values.

Appendix B: Algorithm ECSVDP-SRP5-CLIENT

ECSVDP-SRP5-CLIENT is Elliptic Curve Password-Entangled Secret Value Derivation Primitive for client. The algorithm ECSVDP-SRP5-CLIENT (T_c , P_w , W_c , W_s , s) derives a shared secret value Z from the temporary private key T_c , password P_w , the client's public key W_c , server's password entangled public key W_s , and salt s . It has the following sequence of steps:

- (1) Compute octet string $o1 = \text{GE2OSP-X}(Wc)$
- (2) Compute octet string $o2 = \text{GE2OSP-X}(Ws)$
- (3) Compute octet string $o3 = \text{SHA-256}(o1 || o2)$
- (4) compute an integer $i2 = \text{OS2IP}(o3)$
- (5) Compute octet string $o4 = \text{GE2OSP-X}(v)$
- (6) Compute group element $e1 = \text{ECREDP}(o4)$
- (7) Compute group element $e2 = Ws - e1$
- (8) Compute $i3 = \text{OS2IP}(\text{SHA-256}(s || \text{SHA-256}(\text{SIP-URI} || ":" || Pw || ECI)))$
- (9) Compute group element $zg = (Tc + (i2.i3)) * e2$
- (10) Compute field element $z = \text{GE2SVFEP}(zg)$
- (11) Compute shared secret value $Z = \text{FE2OSP}(z)$
- (12) Output Z

Where GE2SVFEP is the primitive for group element to secret value field element conversion, FE2OSP is field element to octet string conversion primitive.

Appendix C: Algorithm ECSDVP-SRP5-SERVER

ECSDVP-SRP5-SERVER is Elliptic Curve Password-Entangled Secret Value Derivation Primitive for server. The algorithm ECSDVP-SRP5-SERVER (Ts, v, Wc, Ws) derives a shared secret value Z from the temporary private key Ts , password verifier v , the client's public key Wc , and server's password entangled public key Ws . It has the following sequence of steps:

- (1) Compute octet string $o1 = \text{GE2OSP-X}(Wc)$
- (2) Compute octet string $o2 = \text{GE2OSP-X}(Ws)$
- (3) Compute octet string $o3 = \text{SHA-256}(o1 || o2)$
- (4) compute an integer $i2 = \text{OS2IP}(o3)$
- (5) Compute group element $zg = Ts * (Wc + i2 * v)$
- (6) Compute field element $z = \text{GE2SVFEP}(zg)$
- (7) Compute shared secret value $Z = \text{FE2OSP}(z)$
- (8) Output Z