Network Working Group
Internet-Draft
Updates: 7296 (if approved)
Intended status: Standards Track
Expires: April 18, 2016

T. Kivinen
INSIDE Secure
P. Wouters
Red Hat
H. Tschofenig

October 16, 2015

Generic Raw Public Key Support for IKEv2
draft-kivinen-ipsecme-oob-pubkey-14.txt

Abstract

The Internet Key Exchange Version 2 (IKEv2) protocol did have support for raw public keys, but it only supported RSA Raw public keys. In constrained environments it is useful to make use of other types of public keys, such as those based on Elliptic Curve Cryptography. This document updates RFC 7296, adding support for other types of raw public keys to IKEv2.

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 April 18, 2016.

Table of Contents

1. Introduction

This document replaces an algorithm-specific version of raw public keys of Internet Key Exchange Version 2 (IKEv2) [RFC7296] with a generic version of raw public keys that is algorithm agnostic.

In [RFC5996] IKEv2 had support for PKCS #1 encoded RSA keys, i.e., a DER-encoded RSAPublicKey structure (see [RSA] and [RFC3447]). Other raw public key types are, however, not supported. In [RFC7296] this feature was removed, and this document adds support for raw public keys back to IKEv2 in a more generic way.

DNSSEC allows public keys to be associated with domain names for usage with security protocols like IKEv2 and Transport Layer Security (TLS) [RFC5246] but it relies on extensions in those protocols to be specified.

The Raw Public Keys in Transport Layer Security specification ([RFC7250]) adds generic support for raw public keys to TLS by re-using the SubjectPublicKeyInfo format from the X.509 Public Key Infrastructure Certificate profile [RFC5280].

This document is similar to the Raw Public Keys in Transport Layer Security specification and applies the concept to IKEv2 to support all public key formats defined by PKIX. This approach also allows future public key extensions to be supported without the need to introduce further enhancements to IKEv2.

To support new types of public keys in IKEv2 the following changes are needed:

- o A new Certificate Encoding format needs to be defined for carrying the SubjectPublicKeyInfo structure. Section 3 specifies this new encoding format.

- o A new Certificate Encoding type needs to be allocated from the IANA registry. Section 5 contains this request to IANA.

The base IKEv2 specification includes support for RSA and DSA signatures, but the Signature Authentication in IKEv2 [RFC7427] extended IKEv2 so that signature methods over any key type can be used. Implementations using raw public keys SHOULD use the Digital Signature method described in the RFC7427.

When using raw public keys, the authenticated identity is not usually the identity from the ID payload, but instead the public key itself is used as identity for the other end. This means that ID payload contents might not be useful for authentication purposes. It might still be used for policy decisions, for example to simplify the policy lookup etc. Alternatively, the ID_NULL type [RFC7619] can be used to indicate that the ID payload is not relevant to this authentication.

2. 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 [RFC2119].

3. Certificate Encoding Payload

Section 3.6 of RFC 7296 defines the Certificate payload format as shown in Figure 1.

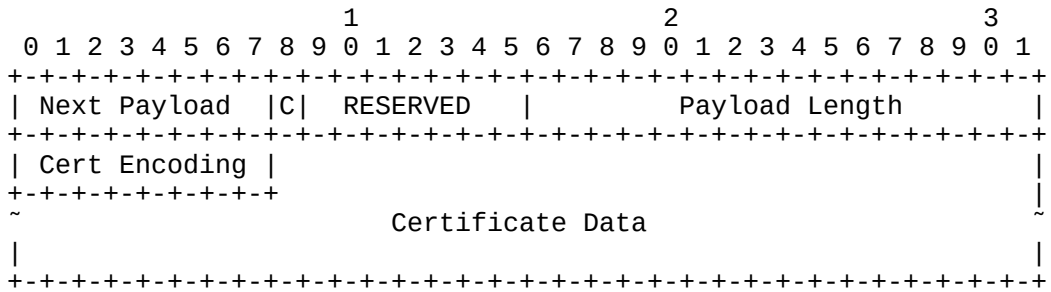

Figure 1: Certificate Payload Format.

To support raw public keys, the field values are as follows:

- Certificate Encoding (1 octet) - This field indicates the type of certificate or certificate-related information contained in the Certificate Data field.

Certificate Encoding	Value
Raw Public Key	TBD

- Certificate Data (variable length) - Actual encoding of the certificate data.

In order to provide a simple and standard way to indicate the key type when the encoding type is 'Raw Public Key', the SubjectPublicKeyInfo structure of the PKIX certificate is used. This is a very simple encoding, as most of the ASN.1 part can be included literally, and recognized by block comparison. See [RFC7250] Appendix A for a detailed breakdown. In addition, Appendix A has several examples.

In addition to the Certificate payload, the Cert Encoding for Raw Public Key can be used in the Certificate Request payload. In that case the Certification Authority field MUST be empty if the "Raw Public Key" certificate encoding is used.

For RSA keys, the implementations MUST follow the public key processing rules of section 1.2 of the Additional Algorithms and Identifiers for RSA Cryptography for PKIX ([RFC4055]) even when the SubjectPublicKeyInfo is not part of a certificate, but rather sent as a Certificate Data field. This means that RSASSA-PSS and RSASSA-PSS-params inside the SubjectPublicKeyInfo structure MUST be sent when applicable.

4. Security Considerations

An IKEv2 deployment using raw public keys needs to utilize an out-of-band public key validation procedure to be confident in the authenticity of the keys being used. One way to achieve this goal is to use a configuration mechanism for provisioning raw public keys into the IKEv2 software. "Smart object" deployments are likely to use such preconfigured public keys.

Another approach is to rely on secure DNS to associate public keys with domain names using the IPSECKEY DNS RRtype [RFC4025]. More information can be found in DNS-Based Authentication of Named Entities (DANE) [RFC6394].

This document does not change the security assumptions made by the IKEv2 specification since "Raw RSA Key" support was already available in IKEv2 in [RFC5996]. This document only generalizes raw public key support.

5. IANA Considerations

This document allocates a new value from the IKEv2 Certificate Encodings registry:

TBD Raw Public Key

6. Acknowledgements

This document reproduces some parts of the similar TLS document ([RFC7250]).

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, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <http://www.rfc-editor.org/info/rfc5280>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <http://www.rfc-editor.org/info/rfc7296>.

[RFC7427] Kivinen, T. and J. Snyder, "Signature Authentication in the Internet Key Exchange Version 2 (IKEv2)", RFC 7427, DOI 10.17487/RFC7427, January 2015, <http://www.rfc-editor.org/info/rfc7427>.

[RFC7619] Smyslov, V. and P. Wouters, "The NULL Authentication Method in the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 7619, DOI 10.17487/RFC7619, August 2015, <http://www.rfc-editor.org/info/rfc7619>.

7.2. Informative References

[RFC3447] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, DOI 10.17487/RFC3447, February 2003, <http://www.rfc-editor.org/info/rfc3447>.

[RFC4025] Richardson, M., "A Method for Storing IPsec Keying Material in DNS", RFC 4025, DOI 10.17487/RFC4025, March 2005, <http://www.rfc-editor.org/info/rfc4025>.

[RFC4055] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4055, DOI 10.17487/RFC4055, June 2005, <http://www.rfc-editor.org/info/rfc4055>.

[RFC4754] Fu, D. and J. Solinas, "IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA)", RFC 4754, DOI 10.17487/RFC4754, January 2007, <http://www.rfc-editor.org/info/rfc4754>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", RFC 5480, DOI 10.17487/RFC5480, March 2009, <http://www.rfc-editor.org/info/rfc5480>.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, DOI 10.17487/RFC5996, September 2010, <http://www.rfc-editor.org/info/rfc5996>.

[RFC6394] Barnes, R., "Use Cases and Requirements for DNS-Based Authentication of Named Entities (DANE)", RFC 6394, DOI 10.17487/RFC6394, October 2011, <http://www.rfc-editor.org/info/rfc6394>.

[RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <http://www.rfc-editor.org/info/rfc7250>.

[RSA] R. Rivest, , A. Shamir, , and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", February 1978.

Appendix A. Examples

This appendix provides examples of the actual payloads sent on the wire.

A.1. ECDSA Example

This first example uses the 256-bit ECDSA private/public key pair defined in section 8.1 of the IKEv2 ECDSA document [RFC4754].

The public key is as follows:

- Algorithm : id-ecPublicKey (1.2.840.10045.2.1)
- Fixed curve: secp256r1 (1.2.840.10045.3.1.7)
- Public key x coordinate : cb28e099 9b9c7715 fd0a80d8 e47a7707 9716cbbf 917dd72e 97566ea1 c066957c
- Public key y coordinate : 2b57c023 5fb74897 68d058ff 4911c20f dbe71e36 99d91339 afbb903e e17255dc

The SubjectPublicKeyInfo ASN.1 object is as follows:

```
0000 :     SEQUENCE
0002 :       SEQUENCE
0004 :         OBJECT IDENTIFIER  id-ecPublicKey (1.2.840.10045.2.1)
000d :         OBJECT IDENTIFIER  secp256r1 (1.2.840.10045.3.1.7)
0017 :       BIT STRING  (66 bytes)
00000000: 0004 cb28 e099 9b9c 7715 fd0a 80d8 e47a
00000010: 7707 9716 cbbf 917d d72e 9756 6ea1 c066
00000020: 957c 2b57 c023 5fb7 4897 68d0 58ff 4911
00000030: c20f dbe7 1e36 99d9 1339 afbb 903e e172
00000040: 55dc
```

The first byte (00) of the bit string indicates that there is no "number of unused bits", and the second byte (04) indicates uncompressed form ([RFC5480]). Those two octets are followed by the values of X and Y.

The final encoded SubjectPublicKeyInfo object is as follows:

```
00000000: 3059 3013 0607 2a86 48ce 3d02 0106 082a
00000010: 8648 ce3d 0301 0703 4200 04cb 28e0 999b
00000020: 9c77 15fd 0a80 d8e4 7a77 0797 16cb bf91
00000030: 7dd7 2e97 566e a1c0 6695 7c2b 57c0 235f
00000040: b748 9768 d058 ff49 11c2 0fdb e71e 3699
00000050: d913 39af bb90 3ee1 7255 dc
```

This will result in the final IKEv2 Certificate Payload:

```
00000000: NN00 0060 XX30 5930 1306 072a 8648 ce3d
00000010: 0201 0608 2a86 48ce 3d03 0107 0342 0004
00000020: cb28 e099 9b9c 7715 fd0a 80d8 e47a 7707
00000030: 9716 cbbf 917d d72e 9756 6ea1 c066 957c
00000040: 2b57 c023 5fb7 4897 68d0 58ff 4911 c20f
00000050: dbe7 1e36 99d9 1339 afbb 903e e172 55dc
```

Where NN is the next payload type (i.e., the type of the payload that immediately follows this Certificate payload).

Note to the RFC editor / IANA, replace the XX above with the newly allocated Raw Public Key number (in hex notation), and remove this note.

A.2. RSA Example

This second example uses a random 1024-bit RSA key.

The public key is as follows:

- Algorithm : rsaEncryption (1.2.840.113549.1.1.1)

- Modulus n (1024 bits, decimal):
  13235620711627409124170755510255990457003972512968992059076067098474693867078469765406633930292745175632738983925375171294852777599627772780735262903298211841009721044682579432931952695408402169276996518188784375861544353691437281683053790189766153444138644776266465646382496723290499691435609390077675483541
- Modulus n (1024 bits, hexadecimal): bc7b4347 49c7b386 00bfa84b 44f88187 9a2dda08 d1f0145a f5806c2a ed6a6172 ff0dc3d4 cd601638 e8ca348e bdca5742 31cadc97 12e209b1 fddba58a 8c62b369 038a3d1e aa727c1f 39ae49ed 6ebc30f8 d9b52e23 385a4019 15858c59 be72f343 fb1eb87b 16ffc5ab 0f8f8fe9 f7cb3e66 3d8fe9f9 ecfa1230 66f36835 8ceaefd3

   o Exponent e (17 bits, decimal): 65537

   o Exponent e (17 bits, hexadecimal): 10001

   The SubjectPublicKeyInfo ASN.1 object is as follows:

```
0000 :     SEQUENCE
0003 :       SEQUENCE
0005 :         OBJECT IDENTIFIER rsaEncryption (1.2.840.113549.1.1.1)
0016 :         NULL
0018 :       BIT STRING  (141 bytes)
00000000: 0030 8189 0281 8100 bc7b 4347 49c7 b386
00000010: 00bf a84b 44f8 8187 9a2d da08 d1f0 145a
00000020: f580 6c2a ed6a 6172 ff0d c3d4 cd60 1638
00000030: e8ca 348e bdca 5742 31ca dc97 12e2 09b1
00000040: fddb a58a 8c62 b369 038a 3d1e aa72 7c1f
00000050: 39ae 49ed 6ebc 30f8 d9b5 2e23 385a 4019
00000060: 1585 8c59 be72 f343 fb1e b87b 16ff c5ab
00000070: 0f8f 8fe9 f7cb 3e66 3d8f e9f9 ecfa 1230
00000080: 66f3 6835 8cea efd3 0203 0100 01
```

   The first byte (00) of the bit string indicates that there is no "number of unused bits". Inside that bit string there is an ASN.1 sequence having 2 integers. The second byte (30) indicates that this is beginning of the sequence, and the next byte (81) indicates the length does not fit in 7 bits, but requires one byte, so the length is in the next byte (89). Then starts the first integer with tag (02) and length (81 81). After that we have the modulus (prefixed with 0 so it will not be a negative number). After the modulus there follows the tag (02) and length (03) of the exponent, and the last 3 bytes are the exponent.

   The final encoded SubjectPublicKeyInfo object is as follows:

```
00000000: 3081 9f30 0d06 092a 8648 86f7 0d01 0101
00000010: 0500 0381 8d00 3081 8902 8181 00bc 7b43
00000020: 4749 c7b3 8600 bfa8 4b44 f881 879a 2dda
00000030: 08d1 f014 5af5 806c 2aed 6a61 72ff 0dc3
00000040: d4cd 6016 38e8 ca34 8ebd ca57 4231 cadc
00000050: 9712 e209 b1fd dba5 8a8c 62b3 6903 8a3d
00000060: 1eaa 727c 1f39 ae49 ed6e bc30 f8d9 b52e
00000070: 2338 5a40 1915 858c 59be 72f3 43fb 1eb8
00000080: 7b16 ffc5 ab0f 8f8f e9f7 cb3e 663d 8fe9
00000090: f9ec fa12 3066 f368 358c eaef d302 0301
000000a0: 0001
```

   This will result in the final IKEv2 Certificate Payload:

```
00000000: NN00 00a7 XX30 819f 300d 0609 2a86 4886
00000010: f70d 0101 0105 0003 818d 0030 8189 0281
00000020: 8100 bc7b 4347 49c7 b386 00bf a84b 44f8
00000030: 8187 9a2d da08 d1f0 145a f580 6c2a ed6a
00000040: 6172 ff0d c3d4 cd60 1638 e8ca 348e bdca
00000050: 5742 31ca dc97 12e2 09b1 fddb a58a 8c62
00000060: b369 038a 3d1e aa72 7c1f 39ae 49ed 6ebc
00000070: 30f8 d9b5 2e23 385a 4019 1585 8c59 be72
00000080: f343 fb1e b87b 16ff c5ab 0f8f 8fe9 f7cb
00000090: 3e66 3d8f e9f9 ecfa 1230 66f3 6835 8cea
000000a0: efd3 0203 0100 01
```

Where NN is the next payload type (i.e., the type of the payload that immediately follows this Certificate payload).

Note to the RFC editor / IANA, replace the XX above with the newly allocated Raw Public Key number, and remove this note.

Authors' Addresses

Tero Kivinen
INSIDE Secure
Eerikinkatu 28
HELSINKI FI-00180
FI

Email: kivinen@iki.fi

Paul Wouters
Red Hat

Email: pwouters@redhat.com

Hannes Tschofenig

Email: Hannes.Tschofenig@gmx.net
URI: http://www.tschofenig.priv.at