DIME
Internet-Draft
Intended status: Standards Track
Expires: September 6, 2012

J. Korhonen
H. Tschofenig
Nokia Siemens Networks

March 5, 2012

Diameter End-to-End Security: Keyed Message Digests, Digital Signatures, and Encryption

draft-korhonen-dime-e2e-security-00.txt

Abstract

This document defines an extension for end to end authentication, integrity and confidentiality protection of Diameter Attribute Value Pairs. The solutions focuses on protecting Diameter Attribute Value Pairs and leaves the key distribution solution to a separate specification. The integrity protection can be introduced in a backward compatible manner to existing application. The confidentiality protection requires an explicit support from an application, thus is applicable only for newly defined applications.

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].

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 6, 2012.

Copyright Notice

Copyright (c) 2012 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

The Diameter base protocol [I-D.ietf-dime-rfc3588bis] leverages IPsec and TLS for mutual authentication between neighboring Diameter nodes and for channel security offering data origin authentication, integrity and confidentiality protection. The Diameter base protocol, however, also defines Diameter agents, namely Relay Agents, Proxy Agents, Redirect Agents, and Translation Agents.

Relay Agents are Diameter agents that accept requests and route messages to other Diameter nodes based on information found in the messages. Since Relays do not perform any application level processing, they provide relaying services for all Diameter applications.

Similarly to Relays, Proxy Agents route Diameter messages using the Diameter routing table. However, they differ since they modify messages to implement policy enforcement.

Redirect Agents do not relay messages, and only return an answer with the information necessary for Diameter agents to communicate directly, they do not modify messages. Redirect Agents do not have negative impacts on end-to-end security and are therefore not considered in this document.

A Translation Agent is a device that provides translation between two protocols. To offer end-to-end security across different protocol requires the ability to convey and process the AVPs defined in this document by both end points. Since such support is very likely not available this document does not cover this functionality.

This Diameter extension specifies how AVP authentication, integrity and confidentiality protection can be offered using either symmetric or asymmetric cryptography. As a solution mechanism Javascript Object Signing and Encryption (JOSE) is utilized. JOSE offers a simple encoding with small set of features ideal for the purpose of Diameter.

This document focuses on protecting Diameter AVP and leaves the key distribution solution to a separate specification. To offer the functionality two AVPs are defined: Signed-Data and Encrypted-Data.

2. AVP Encoding

2.1. Signed-Data AVP

The Signed-Data AVP (AVP Code TBD) is of type OctetString and utilizes the JSON Web signature (JWS) mechanism defined in [I-D.ietf-jose-json-web-signature].

JWS represents digitally signed or HMACed content using JSON data structures. The representation in [I-D.ietf-jose-json-web-signature] consists of three parts: the JWS Header, the JWS Payload, and the JWS Signature. The three parts are represented as the concatenation of the encoded strings in that order, with the three strings being separated by period ('.') characters. For the JWS Payload we define a new JSON object that contains an array of AVP code number and a hash of AVP pairs. The JWS Signature then covers the all APVs to be signed or HMACed. Both JWS Payload and signature MUST use the same hash algorithm of the cryptographic algorithm indicated in the JWS Header.

To package a set of AVPs for signing, each AVP octet representation to be protected are first individually hashed and encoded into the JSON object with its AVP code number, as shown in Figure 1:

```
{ "avp":
  [
    {"code" : 123,
     "hash": "NzY0YjIwYTgyNjE1NjYzNzBkMjExZTUyZmQwNTA5Njc="
    },
    {"code" : 321,
     "hash": "OWQ3NjMyNzViNGVmNjQzMGVmNTg4Y2JjMDRiNzU5OGY="
    }
  ]
}
```

Figure 1: Example JWS Payload

The entire AVP MUST be input to the hash calculation, from the first byte of the AVP code to the last byte of the AVP data, including all other fields, length, reserved/flags, and optional vendor IDs, and padding. The AVP MUST be input to the hash calculation in network byte order. The "code" in the Figure 1 is an integer value containing the AVP code number, and the "hash" is the Base64 encoded hash of the AVP.

The JWS Signature is calculated over the entire JWS Payload and then the all three JWS parts are placed in the Signed-Data AVP. There can be multiple Signed-Data AVPs in a Diameter message. The AVP code in

the JWS Payload is to indicate which AVP this hash possibly refers to. If there are multiple instances of the same AVP in the Diameter message, there is no other way than make the verification against all of those. It is possible that the message sender only hashed one AVP of the same type and, therefore, the receiver MUST verify the hash against all occurrences of the AVP of the same code number. Such flexibility is added there to allow reordering of the AVPs and addition or deletion of new AVPs by intermediating agents.

If a receiver detects errors with the processing of the Signed-Data AVP it MAY return one of the errors defined in Section 3. If a receiver does not find any AVP the Signed-Data AVP has a signature for, it MAY also return one of the errors defined in Section 3.

When AVPs are to be both encrypted and signed, the Encrypted-Data AVP MUST be created first. This means that signing is "outside" encryption.

Here is an example: Imagine the following AVPs from the QoS-Resources AVP in the QoS-Install Request (defined in RFC 5866 [RFC5866] message shall be signed. The resulting example message has the following structure:

```
<QoS-Install-Request> ::= < Diameter Header: 327, REQ, PXY >
                          < Session-Id >
                          { Auth-Application-Id }
                          { Origin-Host }
                          { Origin-Realm }
                          { Destination-Realm }
                          { Auth-Request-Type }
                          [ Signed-Data ]
                        * [ QoS-Resources ]
                          ...
```

Example Diameter Message with Signed-Data AVP

The Signed-Data AVP in this example may contain a JWS Header that indicates the use of the HMAC SHA-256 algorithm with the key id 'abc123'. The protected AVPs are Session-Id, Origin-Host and Origin-Realm. The calculated HMAC SHA-256 values are for example purposes only (i.e., are not real):

JWS Header:

```
{  "typ":"JWT",
   "alg":"HS256",
   "kid":"abc123"
}
```

JWS Payload:

```
{ "avp":
  [
    {"code" : 263,      // Session-Id
     "hash": "OWQwZTA0OTViYThjMDMxMmI2Mjc0YzUyN2Q1MWEwNDg="
    },
    {"code" : 264,      // Origin-Host
     "hash": "MzljYTg4ZmZhYTVhNmZmOTAyOWVkOTViYTUzNGUwMjg="
    },
    {"code" : 296,      // Origin-Realm
     "hash": "MjAyNzMwYWNhNmUzYTE4MDJmNDRhNjMzZjI1MGY2ZmU="
    }
  ]
}
```

JWS Signature:

```
"wv3yJxyrhYJkCcDjK63elFkEvAV9dsSUNBf5Cu1ref8="
```

Combined example JWS (with line breaks for display purposes only):

```
eyAgInR5cCI6IkpXVCIsDQogICAgImFsZyI6IkhTMjU2IiwNCiAgICAia2l
kIjoiYWJjMTIzIg0KfQ==
.
eyAiYXZwIjoNCiAgIFsNCiAgICAgeyJjb2RlIiA6IDI2MywgICAgICAvLyB
TZXNzaW9uLUlkDQogICAgICAiaGFzaCI6ICJPV1F3WlRBME9UVmlZVGhqTU
RNeE1tSTJNamMwWXpVeU4yUTFNV0V3TkRnPSINCiAgICAgfSwNCiAgICAge
yJjb2RlIiA6IDI2NCwgICAgICAvLyBPcmlnaW4tSG9zdA0KICAgICAgImhh
c2giOiAiTXpsallUZzRabVpoWVRWaE5tWm1PVEF5T1dWa09UVmlZVFV6Tkd
Vd01qZz0iDQogICAgIH0sDQogICAgIHsiY29kZSIgOiAyOTYsICAgICAgLy
8gT3JpZ2luLVJlYWxtDQogICAgICAiaGFzaCI6ICJNakF5TnpNd1lXTmhOb
VV6WVRFNE1ESm1ORFJoTmpNelpqSTFNR1kyWm1VPSINCiAgICAgfQ0KICAg
XQ0KIH0=
.
wv3yJxyrhYJkCcDjK63elFkEvAV9dsSUNBf5Cu1ref8=
```

Example JWS Header, Payload and Signature

2.2. Encrypted-Data AVP

The Encrypted-Data AVP (AVP Code TBD) is of type OctetString and contains the JSON Web Encryption (JWE) [I-D.ietf-jose-json-web-encryption] data structure and consists of three parts: the JWE Header, the JWE Encrypted Key, and the JWE Ciphertext. The three parts are represented as the concatenation of the encoded strings in that order, with the three strings being separated by period ('.') characters. JWE does not add a content integrity check if not provided by the underlying encryption algorithm.

A single AVP or an entire list of AVPs MUST be input to the encryption process, from the first byte of the AVP code to the last byte of the AVP data, including all other fields, length, reserved/ flags, and optional vendor IDs, and padding. The AVP MUST be input to the encryption process in network byte order, and the encryptor is free to order AVPs whatever way it chooses. When AVPs are to be both encrypted and authenticated, the Encrypted-Data AVP MUST be created first.

Note that the usage of the Encrypted-Data AVP requires explicit support by the Diameter application since a receiving Diameter node must first decrypt the content of the Encrypted-Data AVP in order to evaluate the AVPs carried in the message. In case that a Diameter node is unable to understand the Encrypted-Data AVP and ignores the AVP then two possible outcomes are possible: First, if the encrypted AVPs are optional then their content is not considered by the receiving Diameter server without any indication to the sender that they have not been processes. Worse, in the second case when the encrypted AVPs are mandatory to be processed then the receiving Diameter node will return an error that may not inform the sender about the failure to decrypt the Encrypted-Data AVP. Consequently, the usage of the Encrypted-Data AVP may require changes to the ABNF definition of a Diameter application.

If a receiver detects that the contents of the Encrypted-Data AVP is invalid, it SHOULD return the new Result-Code AVP value defined in Section 3.

3. Result-Code AVP Values

This section defines new Diameter result code values for usage with Diameter applications.

3.1. Transient Failures

Errors that fall within the transient failures category are used to inform a peer that the request could not be satisfied at the time it was received, but MAY be able to satisfy the request in the future.

DIAMETER_KEY_UNKNOWN (TBD)

This error code is returned when a Signed-Data or an Encrypted-Data AVP is received that was generated using a key that cannot be found in the key store. This error may, for example, be caused if one of the endpoints of an end-to-end security association lost a previously agreed upon key, perhaps as a result of a reboot. To recover a new end-to-end key establishment procedure may need to be invoked.

3.2. Permanent Failures

Errors that fall within the permanent failures category are used to inform the peer that the request failed, and should not be attempted again.

DIAMETER_DECRYPTION_ERROR (TBD)

This error code is returned when an Encrypted-Data AVP is received and the decryption fails for an unknown reason.

DIAMETER_SIGNATURE_ERROR (TBD)

This error code is returned when a Signed-Data AVP is received and the verification fails for an unknown reason.

4. IANA Considerations

IANA is requested to allocate AVP codes for the following AVPs:

AVP Name	AVP Code	Section Defined	Data Type
Signed-Data	TBD	3.1	OctetString
Encrypted-Data	TBD	3.2	OctetString

This specification additionally defines a few Result-Code AVP values, see Section 3.

5. Security Considerations

The purpose of this document is to offer end-to-end security mechanisms for calculating keyed message digest, for signing, and for encryption of Diameter AVPs.

6. Acknowledgements

We would like to thank the authors of [I-D.ietf-aaa-diameter-e2e-sec] for their work on CMS end-to-end security for Diameter. Their document inspired us.

7. References

7.1. Normative References

[I-D.ietf-dime-rfc3588bis]
Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", draft-ietf-dime-rfc3588bis-29 (work in progress), August 2011.

[I-D.ietf-jose-json-web-encryption]
Hildebrand, J., Rescorla, E., and M. Jones, "JSON Web Encryption (JWE)", draft-ietf-jose-json-web-encryption-00 (work in progress), January 2012.

[I-D.ietf-jose-json-web-signature]
Sakimura, N., Jones, M., and J. Bradley, "JSON Web Signature (JWS)", draft-ietf-jose-json-web-signature-00 (work in progress), January 2012.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

7.2. Informational References

[I-D.ietf-aaa-diameter-e2e-sec]
Calhoun, P., "Diameter End-2-End Security Extension", 2001.

[RFC5866] Sun, D., McCann, P., Tschofenig, H., Tsou, T., Doria, A., and G. Zorn, "Diameter Quality-of-Service Application", RFC 5866, May 2010.

Authors' Addresses

Jouni Korhonen
Nokia Siemens Networks
Linnoitustie 6
FI-02600 Espoo
FINLAND

Email: jouni.nospam@gmail.com

Hannes Tschofenig
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600
Finland

Phone: +358 (50) 4871445
Email: Hannes.Tschofenig@gmx.net
URI: http://www.tschofenig.priv.at