KARP Working Group
Internet Draft
Intended status: Standards Track
Expires: March, 2011
September 2010

Manav Bhatia
Alcatel-Lucent

Non IPSec Authentication mechanism for OSPFv3

draft-bhatia-karp-non-ipsec-ospfv3-auth-01.txt

Abstract

Currently a few routing protocols use IPSec for authenticating their protocol packets. There are known issues with using IPSec with the routing protocols and this draft proposes an alternative Generic Authentication mechanism that can be used so that these protocols do not depend upon IPSec for security. The mechanism introduced in this draft is generic and can be used by any protocol that currently uses IPSec for authentication.

While this mechanism is generic, this draft specifically looks at OSPFv3 and how it can use the mechanism described herein.

Conventions used in this document

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]

Table of Contents

1. Introduction

Routing protocols like Open Shortest Path First Version 3 (OSPFv3) and Protocol Independent Multicast Sparse Mode (PIM-SIM) (for the link-local messages) use IPSec [RFC4301] to ensure authentication of their control packets.

There however are some environments (mobile ad-hoc), where IPSec is difficult to configure and maintain, and this mechanism cannot be used.

The anti-replay feature available in both Authentication Header (AH) [RFC4302] and Encapsulating Security Payload (ESP) [RFC4303] is disabled by the receiver for a manually keyed Security Association (SA). This means that the Anti-Replay Window as described in [RFC4301] is completely ignored when receiving IPSec packets.

Since routing protocols mostly use manually keyed SAs, they cannot make use of the Anti-Reply feature provided by IPSec. This makes the protocols vulnerable to replay attacks.

Routing protocols need IPSec for data integrity and rarely employ it for confidentiality, therefore most specifications [RFC4552] [RFC5796] that describe how routing protocols need to use IPSec for security mandate using ESP-NULL.

So while the routing protocols could be using ESP-NULL, which means that routing packets are being sent in clear, there is no deterministic way to differentiate between encrypted and unencrypted ESP packets by simply examining the packet. This can pose some challenge to a device that wants to prioritize certain control traffic over the other. IP Security Maintenance and Extensions (IPSecME) Working group has documented two approaches to enable intermediate security devices to distinguish between encrypted and unencrypted ESP traffic - Wrapped Encapsulating Security Payload (WESP) [RFC5840] and the heuristics approach [RFC5879].

While this issue of traffic prioritization on the receiving nodes can be solved by employing certain implementation tricks, it is an issue that should get addressed.

This draft provides a generic mechanism that routing protocols can use for data integrity verification while fixing the above stated issues.

1.1. OSPFv3

Unlike OSPF (Open Shortest Path First) Version 2 [RFC2328] OSPF for IPv6 (OSPFv3) [RFC5340], does not have Auth Type and Authentication fields in its headers for authenticating the protocol packets. It instead relies on the IPv6 Authentication Header (AH) [RFC4302] and IPv6 Encapsulating Security Payload (ESP) [RFC4303] to provide integrity, authentication, and/or confidentiality.

[RFC4552] describes how IPv6 AH/ESP extension headers can be used to provide authentication/confidentiality to OSPFv3.

[RFC4552] discusses, at length, the reasoning behind using manually configured keys, rather than some automated key management protocol such as IKEv2 [RFC5996]. The primary problem is the lack of suitable key management mechanism, as OSPF adjacencies are formed on a one-to-many basis and most key management mechanisms are designed for a one-to-one communication model. This forces the system administrator to use manually configured security associations (SAs) and cryptographic keys to provide the authentication and, if desired, confidentiality services.

Regarding replay protection [RFC4552] states that:

> As it is not possible as per the current standards to provide complete replay protection while using manual keying, the proposed solution will not provide protection against replay attacks.

Since there is no replay protection provided there are a number of vulnerabilities in OSPFv3 which have been discussed in [crypto-issues].

These can be fixed if we move to a non IPSec method for authenticating the OSPFv3 protocol packets.

Lastly, there is also an issue with using IPSec for authenticating OSPFv3 packets where prioritizing certain protocol packets over the others becomes difficult.

This draft proposes a new mechanism that works similar to OSPFv2 for providing authentication to the OSPFv3 packets and attempts to solve the problems described above for OSPFv3.

Additionally this document describes how HMAC-SHA authentication can be used for OSPFv3.

By definition, HMAC ([RFC2104], [FIPS-198]) requires a cryptographic hash function. This document proposes to use any one of SHA-1, SHA-256, SHA-384, or SHA-512 [FIPS-180-3] to authenticate the OSPFv3 packets.

It is believed that [RFC2104] is mathematically identical to [FIPS-198] and it is also believed that algorithms in [RFC4634] are mathematically identical to [FIPS-180-3].

2. Basic Operation

In order to provide authentication for OSPFv3 packets, implementations MUST support the Generic Authentication extension header described in the subsequent sections.

This will only be used to provide data integrity and cannot be used for confidentiality. If the latter is required then implementations MUST use ESP as described in [RFC4552].

Using this authentication scheme, a session key (either statically configured or derived from some master key) is used on all the routers attached to a common network. For each OSPFv3 protocol packet, this key is used to generate and verify a "message digest". This digest is carried inside the Generic Authentication extension header. The message digest is a one-way function of the OSPFv3 protocol packet and the secret key. Since the secret key is never sent over the network in the clear, protection is provided against passive attacks [RFC1704].

The algorithms used to generate and verify the digest are specified implicitly by the key. In addition, a non decreasing sequence number is included in the Generic Authentication Header carried along with each OSPFv3 protocol packet to protect against replay attacks.

3. OSPFv3 Security Association

An OSPFv3 Security Association contains a set of parameters shared between any two legitimate OSPFv3 speakers.

Parameters associated with an OSPFv3 SA:

Key Identifier (Key ID)

This is a 32-bit unsigned integer used to uniquely identify an OSPFV3 SA, as manually configured by the network operator.

The receiver determines the active SA by looking at the Key ID field in the incoming protocol packet.

The sender based on the active configuration, selects the Security Association to use and puts the correct Key ID value associated with the Security Association in the OSPFV3 protocol packet. If multiple valid and active OSPFV3 Security Associations exist for a given outbound interface at the time an OSPFV3 packet is sent, the sender may use any of those security associations to protect the packet.

Using Key IDs makes changing keys while maintaining protocol operation convenient. Each key ID specifies two independent parts, the authentication protocol and the authentication key, as explained below.

Normally, an implementation would allow the network operator to configure a set of keys in a key chain, with each key in the chain having fixed lifetime. The actual operation of these mechanisms is outside the scope of this document.

Note that each key ID can indicate a key with a different authentication protocol. This allows multiple authentication mechanisms to be used at various times without disrupting an OSPFv3 peering, including the introduction of new authentication mechanisms.

Authentication Algorithm

This signifies the authentication algorithm to be used with the OSPFv3 SA. This information is never sent in cleartext over the wire. Because this information is not sent on the wire, the implementer chooses an implementation specific representation for this information.

At present, the following values are possible:

HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512.

Authentication Key

This value denotes the cryptographic authentication key associated with the OSPFv3 SA. The length of this key is variable and depends upon the authentication algorithm specified by the OSPFv3 SA.

4. Authentication Procedure

4.1. Generic Authentication Header

The Generic Authentication Header uses the Next_Header (TBD via IANA) in the immediately preceding header, and has the following format:

```
0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |   Header Len  |             0                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Key ID                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Cryptographic Sequence Number                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|            Authentication Data (Variable)                     |
~                                                               ~
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        Padding (Optional)                     |
~                                                               ~
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1

Next Header

8 bit selector that identifies the type of header immediately following the Generic Authentication Extension Header. Uses the same values as the IPv6 Next Header field [RFC2460].

Header Len

8-bit selector that indicates the length in 8 octet units of the extension header, not including the first 8 octets.

Reserved

16-bit reserved field. The value MUST be initialized to zero by the sender, and MUST be ignored by the receiver.

Key ID

32-bit field that identifies the algorithm and the secret key used to create the message digest carried inside the extension header.

Cryptographic Sequence Number

32-bit non-decreasing sequence number that is used to guard against replay attacks. This provides a long term protection however it is still possible to replay packets using this mechanism until the sequence number changes. This field is initialized to zero and is also set to zero whenever the originating router state is transitioned to "Down" state. Whenever a protocol packet carrying this extension header is accepted as authentic, the receiving router, must update the cryptographic sequence number that it maintains for the originating router to the received packet's sequence number.

This specification does not provide a rollover procedure for the cryptographic sequence number. When the cryptographic sequence number that the router is sending hits the maximum value, the router should reset the cryptographic sequence number that it is sending back to 0. After this is done, the router's neighbors will reject the router's protocol packets for some period till the protocol employing the use of generic authentication times out and tries reestablishing the adjacencies. In case of OSPFv3 this period would be the RouterDeadInterval. However, it is expected that most implementations will use "seconds since reboot" (or "seconds since 1960", etc.) as the cryptographic sequence number. Such a choice will essentially prevent rollover, since the cryptographic sequence number field is 32 bits in length.

A protocol that wants to use the Generic Extension header MAY not use the Anti-Replay feature if it does not require it. In such cases this field MUST be set to zero and MUST be ignored by the receiving router.

Authentication Data

Variable data that is carrying the digest of the protocol packet.

Padding

This MUST be used with IPv6 in order to preserve IPv6 8-octet alignment. If HMAC-SHA-1 is being used as the authentication algorithm then the authentication data is of 20 bytes. Add to this 1 byte of the next header, 1 byte of the header length, the 2 reserved bytes, 4 bytes for the cryptographic sequence number and 4 bytes of the Key ID and we get 32 bytes. Since this is already aligned to an 8 octet boundary no padding is required. However, if the authentication algorithm is HMAC-SHA-256 then the total size comes to 44 bytes, which is not aligned. In this case 4 bytes of padding is used.

The following diagram illustrates the OSPFv3 packet before and after applying this extension header.

Packet format before applying Generic Authentication Header:

```
      +----------------+-----------------------------+
      |orig IP header  |      OSPFv3  Payload        |
      |(any options)   |                             |
      +----------------+-----------------------------+
```

Packet format after applying Generic Authentication Header:

```
      +----------------+--------------+-----------------+
      |orig IP header  | Generic Auth | OSPFv3  Payload  |
      |(any options)   |    header    |                  |
      +----------------+--------------+-----------------+
                       |<----------- integrity --------->|
```

4.2. Cryptographic Authentication Procedure

As noted earlier the algorithms used to generate and verify the message digest are specified implicitly by the secret key. This specification discusses the computation of Cryptographic Authentication data when any of the NIST SHS family of algorithms is used in the Hashed Message Authentication Code (HMAC) mode.

The currently valid algorithms (including mode) for Cryptographic Authentication include:

```
           HMAC-SHA-1
           HMAC-SHA-256
           HMAC-SHA-384
           HMAC-SHA-512
```

Of the above, implementations of this specification MUST include support for at least:

```
       HMAC-SHA-256
```

and SHOULD include support for:

```
       HMAC-SHA-1
```

and MAY also include support for:

```
       HMAC-SHA-384
```

```
          HMAC-SHA-512
```

4.3. Cryptographic Aspects

In the algorithm description below, the following nomenclature, which is consistent with [FIPS-198], is used:

H is the specific hashing algorithm (e.g. SHA-256).

K is the Authentication Key for the routing protocol (OSPFv3 in this case) security association.

Ko is the cryptographic key used with the hash algorithm.

B is the block size of H, measured in octets rather than bits.

Note that B is the internal block size, not the hash size.

```
For SHA-1 and SHA-256:   B == 64
For SHA-384 and SHA-512: B == 128
```

L is the length of the hash, measured in octets rather than bits.

XOR is the exclusive-or operation.
Opad is the hexadecimal value 0x5c repeated B times.
Ipad is the hexadecimal value 0x36 repeated B times.
Apad is the hexadecimal value 0x878FE1F3 repeated (L/4) times.

Implementation Note:

This definition of Apad means that Apad is always the same
length
as the hash output.

(1)Preparation of the Key

In this application, Ko is always L octets long.

If the Authentication Key (K) is L octets long, then Ko is equal to K. If the Authentication Key (K) is more than L octets long, then Ko is set to H(K). If the Authentication Key (K) is less than L octets long, then Ko is set to the Authentication Key (K) with zeros appended to the end of the Authentication Key (K) such that Ko is L octets long.

(2)First Hash

First, the protocol packet's Generic Authentication Extension

Header Data field is filled with the value Apad.

Then, a first hash, also known as the inner hash, is computed as follows:

```
        First-Hash = H(Ko XOR Ipad || (Protocol Packet))
```

(3)Second Hash

Then a second hash, also known as the outer hash, is computed as follows:

```
        Second-Hash = H(Ko XOR Opad || First-Hash)
```

(4)Result

The result Second-Hash becomes the authentication data that is sent in the Authentication Data field of the Generic Authentication extension header. The length of the authentication data is always identical to the message digest size of the specific hash function H that is being used.

This also means that the use of hash functions with larger output sizes will also increase the size of the protocol packet as transmitted on the wire.

4.4. Procedures at the Sending Side for OSPFv3

An appropriate OSPFv3 SA is selected for use with an outgoing OSPFv3 protocol packet. This is done based on the active key at that instant. If OSPFV3 is unable to find an active key, then the packet MUST be discarded.

If OSPFV3 is able to find the active key, then the key gives the authentication algorithm (HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 or HMAC-SHA-512) that needs to be applied on the packet.

An implementation MUST construct a pseudo Generic Authentication extension header before it begins the authentication process. It must set the Next Header to 89, to indicate an OSPF packet (or some other value if the Upper layer protocol is something else). The authentication data is computed as explained in the previous section.

The Header length is set as per the authentication algorithm that is being used. It is, for example, set to 3 for HMAC-SHA-1 and 5 for HMAC-SHA-256.

The key ID and the cryptographic sequence number is filled. Appropriate padding, based on the authentication algorithm being employed, must be used.

The result of the authentication algorithm is placed in the Authentication data.

4.5. Procedures at the Receiving Side for OSPFv3

The appropriate OSPFv3 SA is identified by looking at the Key ID from the Generic Authentication extension header from the incoming OSPFv3 protocol packet.

Using the RouterID carried in the OSPFv3 header, the correct sending neighbor MUST be identified. If a cryptographic sequence number is found in the Generic Extension Header and is less than the cryptographic sequence number recorded in the sending neighbor's data structure, that OSPFv3 packet MUST be discarded. Implementations MAY want to log this event.

Authentication algorithm dependent processing needs to be performed, using the algorithm specified by the appropriate OSPFv3 SA for the received packet.

Before an implementation performs any processing it needs to save the values of the Authentication Value field in the Generic Authentication extension header.

It should then set the Authentication Value field with Apad before the authentication data is computed. The calculated data is compared with the received authentication data in the packet and the packet is discarded if the two do not match. In such a case, an error event SHOULD be logged.

5. Generic Authentication Mechanism

The extension header described in this document can be used by any upper layer protocol that desires integrity protection. All it needs to do is to compute the digest over that protocol packet and carry it inside the Generic Authentication extension header as described in this document.

6. Security Considerations

The document proposes extensions to OSPFv3 which would make it more secure than what it is today. It does not provide confidentiality as a routing protocol contains information

that does not need to be kept secret. It does, however, provide means to authenticate the sender of the packets which is of interest to us.

It should be noted that authentication method described in this document is not being used to authenticate the specific originator of a packet, but is rather being used to confirm that the packet has indeed been issued by a router which had access to the password.

The mechanism described here is not perfect and does not need to be perfect. Instead, this mechanism represents a significant increase in the work function of an adversary attacking the OSPFv3 protocol, while not causing undue implementation, deployment, or operational complexity.

7. IANA Considerations

The Generic Authentication extension header number is assigned by IANA out of the IP Protocol Number space (and as recorded at the IANA web page at http://www.iana.org/assignments/protocol-numbers) is: TBD.

8. References

8.1. Normative References

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

[RFC2460] Deering, S., et. al, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC4301] Kent, S. and Seo, K., "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[FIPS-180-3] US National Institute of Standards & Technology, "Secure Hash Standard (SHS)", FIPS PUB 180-3, October 2008.

[FIPS-198] US National Institute of Standards & Technology, "The Keyed-Hash Message Authentication Code (HMAC)", FIPS PUB 198, March 2002.

8.2. Informative References

[RFC1704] Haller, N. and R. Atkinson, "On Internet Authentication", RFC 1704, October 1994.

[RFC2104] Krawczk, H., "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[RFC2328] Moy, J., "OSPF Version 2", RFC 2328, April 1998.

[RFC5340] Coltun, R., et. al., "OSPF for Ipv6", RFC 5340, July 2008

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, December 2005.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC5996] Kaufman, C., et. al., "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

[RFC4552] Gupta, M. and Melam, N., "Authentication/Confidentiality for OSPFv3", RFC 4552, June 2006

[RFC4634] Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and HMAC-SHA)", RFC 4634, July 2006.

[RFC5796] Atwood, W., Islam, S. and Siami, M, "Authentication and Confidentiality in Protocol Independent Multicast Sparse Mode (PIM-SM) Link-Local Messages", RFC 5796, March 2010

[RFC5840] Grewal, K., Montenegro, G. and Bhatia, M., "Wrapped Encapsulating Security Payload (ESP) for Traffic Visibility", RFC 5840, April 2010

[RFC5879] Kivinen, T. and McDonald, D., "Heuristics for Detecting ESP-NULL Packets", RFC 5879, May 2010

[crypto-issues] Bhatia, M., et. al., "Issues with existing Cryptographic Protection Methods for Routing Protocols", Work in Progress

Author's Addresses

Manav Bhatia
Alcatel-Lucent
Bangalore
India

Phone:
Email: manav.bhatia@alcatel-lucent.com