KARP Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 18, 2011

M. Bhatia
Alcatel-Lucent
S. Hartman
Painless Security
D. Zhang
Huawei Technologies co., LTD.
February 14, 2011

Security Extension for OSPFv2 when using Manual Key Management
draft-bhatia-karp-ospf-ip-layer-protection-03

Abstract

The current OSPFv2 cryptographic authentication mechanism as defined in the OSPF standards is vulnerable to both inter-session and intra-session replay attacks when its uses manual keying. Additionally, the existing cryptographic authentication schemes do not cover the IP header. This omission can be exploited to carry out various types of attacks.

This draft proposes an authentication scheme based on a challenge-response mechanism that will protect OSPFv2 from both inter and intra replay attacks when its using manual keys for securing its protocol packets. For comparison, an approach based on making sequence numbers unique is presented. Later we also describe some changes in the cryptographic hash computation so that we eliminate most attacks that result because of OSPFv2 not protecting the IP header.

Table of Contents

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 [RFC2119].

When used in lower case, these words convey their typical use in common language, and are not to be interpreted as described in RFC2119 [RFC2119].

1. Introduction

The OSPFv2 cryptographic authentication mechanism as described in [[RFC2328]] uses per-packet sequence numbers to provide protection against replay attacks. The sequence numbers increase monotonically so that the attempts to replay the stale packets can be thwarted. The sequence number values are maintained as a part of adjacency states. Therefore, if an adjacency is broken down, the associated sequence numbers get re-initiated and the neighbors start all over again. Additionally, the cryptographic authentication mechanism does not specify how to deal with the rollover of a sequence number when it reaches its maximum limit. These omissions can be taken advantage of by attackers to implement various replay attacks ([RFC6039]). In order to address these issues, we propose a challenge/ response mechanism that introduces two additional random numbers to help routers generate distinguishable new states when the sequence numbers need to be re-initiated. Compared with the cryptographic authentication mechanism proposed in [RFC5709], the solution proposed does not impose any more security presumption.

The cryptographic authentication as described in [RFC2328] and later updated in [RFC5709] does not include the IP header. This also can be exploited to launch several attacks as the source address in the IP header is no longer protected. The OSPF specification, in certain cases, requires the implementations to look at the source address carried in the IP header to determine the neighbor the packet was recieved from. Changing the source address of a packet can thus, confuse the reciever which can be exploited to produce a number of denial of service attacks [RFC6039]. If the packet is interpreted as coming from a different neighbor, the sequence number received from the neighbor may be updated. This may disrupt communication with the legitimate neighbor. Hello packets may be reflected to cause a neighbor to appear to have one-way communication. Old Database descriptions may be reflected in cases where the per-packet sequence numbers are sufficiently divergent in order to disrupt an adjacency [I-D.hartman-ospf-analysis]. This is referred to as the IP layer issue in [I-D.ietf-karp-threats-reqs].

[RFC2328] states that implementations MUST offer keyed MD5 authentication. It is likely that this will be deprecated in favor of the stronger algorithms described in [RFC5709] in future deployments [RFC6094].

This draft proposes a simple change in the cryptographic authentication mechanism, as currently described in [RFC5709], to prevent such IP layer attacks.

2. A Challenge and Response Solution

In OSPFv2, a non-decreasing sequence member is associated with each OSPF packet sent from a router in order to prevent replay attacks. However, as illustrated in [I-D.hartman-ospf-analysis] and [RFC6039], in the circumstances where automatic key management mechanisms are unavailable, any re-initiation of sequence numbers can potentially be taken advantage of to perform replay attacks. In this section, we introduce an extension of the OSPFv2 protocol, which uses challenge/ response to benefit the verification of the freshness of OSPF packets when the sequence numbers of routers are re-initiated. This solution eliminates the reliance on automatic key management mechanisms. However, it is assumed that a traffic key is shared between two communicating routers so that an attacker can play antique packets but lacks the capability to modify packets without being detected.

In this protocol, two random numbers (Session ID and Nonce) are introduced. The session ID is used to identify the session a packet is within and thus makes inter-session replay attacks difficult. The nonce is used to challenge the liveness of communicating routers so that states need not be maintained with routers that are not currently neighbors. In combination with the sequence number, the session ID can effectively resist intra-session replay attacks. When the sequence space is exhausted, a router simply chooses a new session ID.

Figure 1 illustrates how two routers A and B, challenge each other's liveness when they are initially connected to a link. First, A selects a new session ID (X1) and a new nonce (N1), and sends them out within a hello packet (see step 1). Particularly, X1and N1 are encapsulated in the OSPF header of the packet. Note that if A is on a multicast LAN, the packet is sent using multicast. Similarly, B sends a hello packet with its new session ID (X2) and Nonce (N2) (step 2). Upon receiving the hello packet from B, A sends a hello packet with X1 and N1. In the neighbor field of the packet, the router ID of B, X2, and N2 is encapsulated (Step 3). Upon receiving the packet sent in step 3, B can ensure the freshness of the packet if the attached session ID and nonce values of both routers are correct.

In the same way, after receiving the hello packet from A, B sends a hello packet with X2 and N2 in the OSPF header, and in the neighbor field of the packet, the router ID of A, X1, and N1 is listed to identify that A has been discovered. After receiving the packet, A can make sure the packet is fresh if the session IDs and the nonce of both routers contained in the packet are correct. After A and B discover each other, they start exchanging their database information (steps 5 and 6). During the exchange, every packet from Router A is

associated with X1 and N1, while every packet from Router B is associated with X2 and N2. Each of these packets also contains a sequence number as part of the cryptographic authentication option. The sequence number MUST increase for every packet sent.

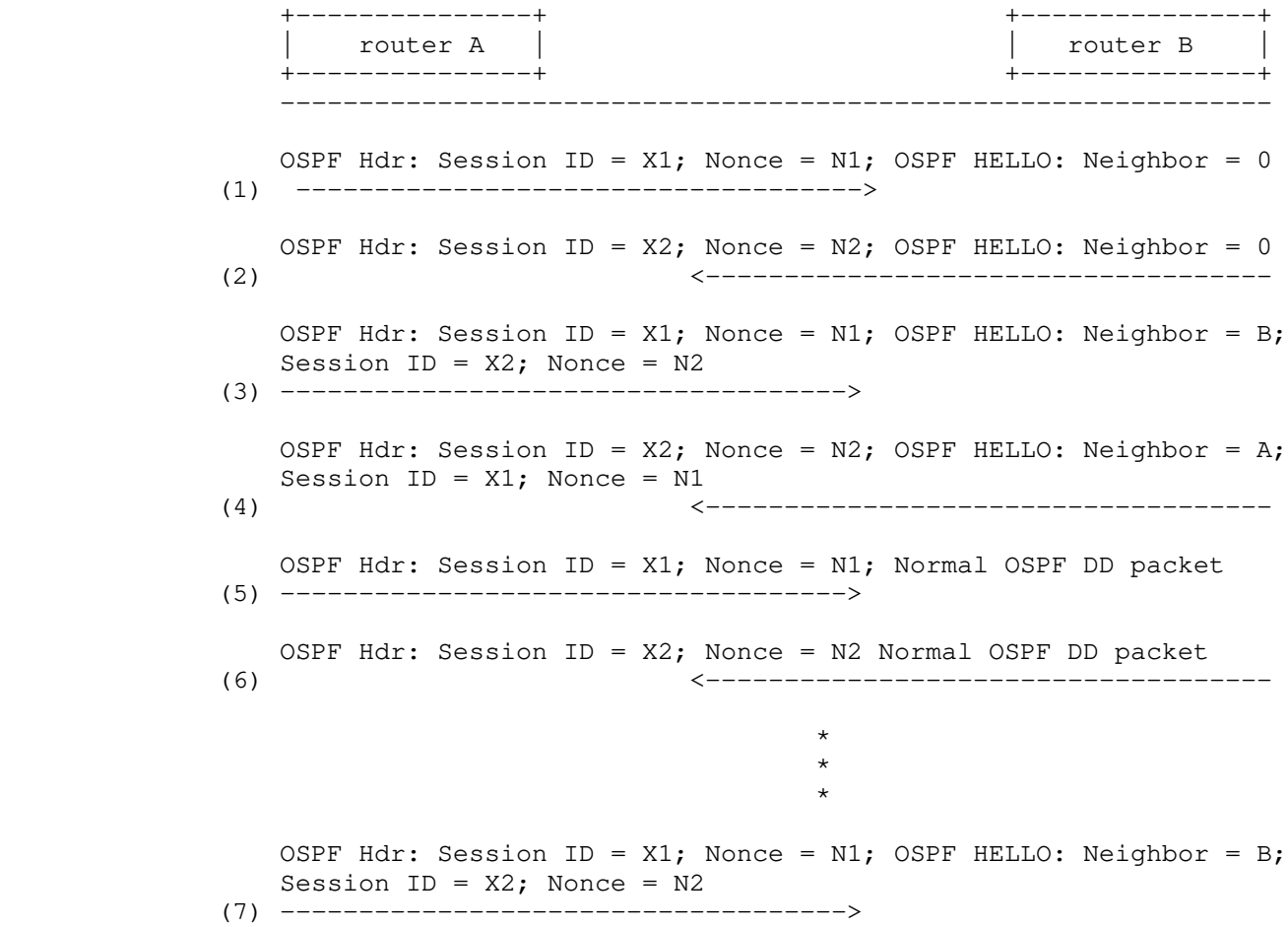


Figure 1 Scenario: two Routers coming up on a LAN

After A and B have generated a neighbor relationship, assume another router, C, is connected to the link. C finds the existence of A and intends to become a neighbor of A. The packets exchanged during this process are illustrated in Figure 2. Firstly, C selects a new session ID (X3) and a new nonce value (N3), and sends them out within a blank hello packet (see the second step of Figure 2). After receiving this packet, A sends out a hello packet with the information of C (router ID, X3, and N3) in the neighbor field.

Because A is challenging the liveness of a new neighbor, A selects a new nonce N1' and encapsulates it in the OSPF header of the hello packet to challenge whether the packet sent in step 2 is really from C. After receiving the packet from A, C can make sure the packet is valid since it consists of its current session ID and nonce (e.g., X3 and N3). Thus, C replies to A with a hello packet including the information of A (e.g., X1 and N1') in the neighbor field. After receiving this packet and checking the correctness of X1 and N1', A can ensure that the packet is fresh and C is currently online.

It worthwhile to note that during the challenge and response the hello packets sent immediately amnogst routers.

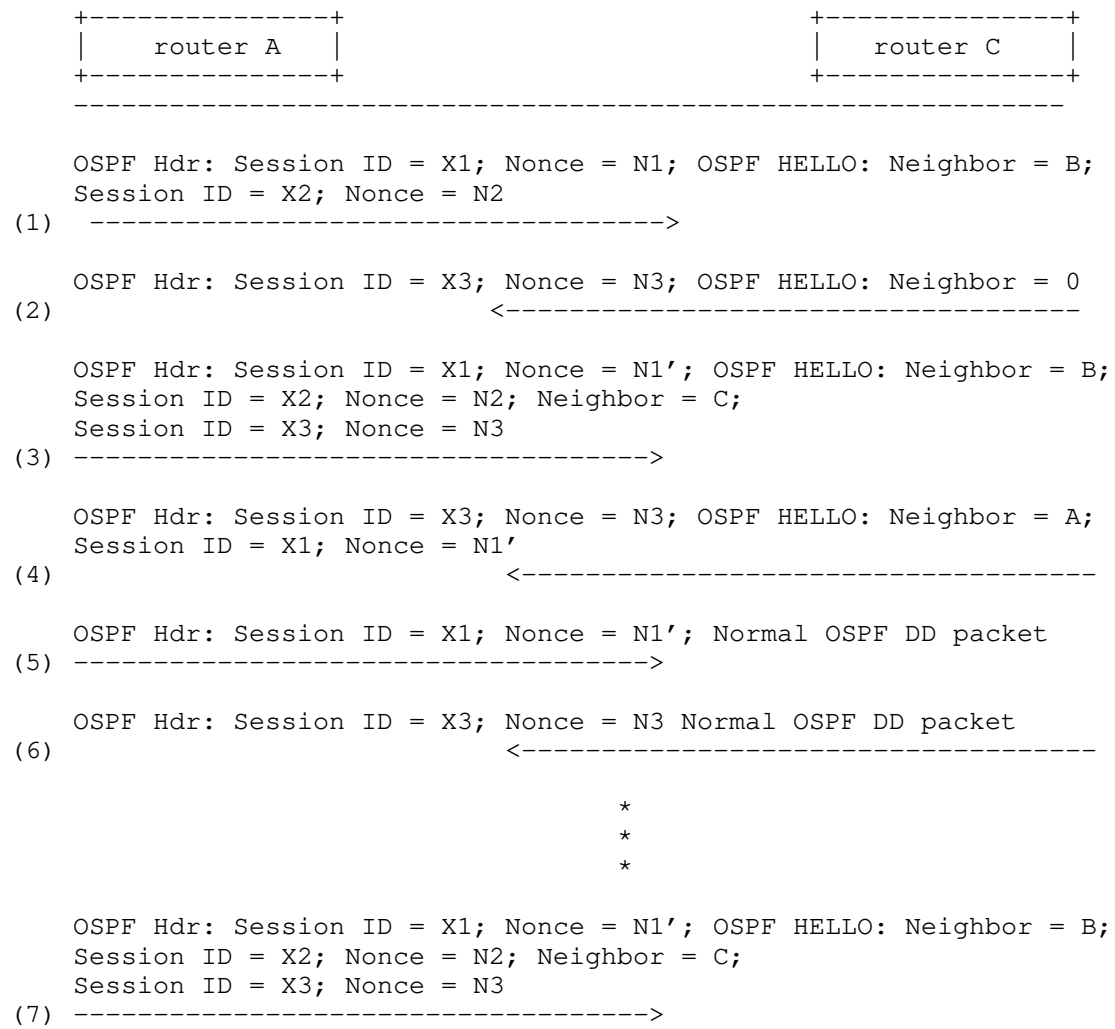


Figure 2. Scenario: another Router C comes up on that LAN

Figure 3 illustrates the scenario in which router A is rebooted. After the reboot, A lost its state and selects a new session ID (X4) and a new nonce value (N4). However, B still maintains the earlier session ID and nonce values of A (X1 and N1). In step 1, A sends a blank hello packet out with its new session ID and nonce value. After receiving the hello packet, B realizes that the session ID and the nonce value of A in the OSPF header are different from the ones maintained in its database. In order to distinguish a reboot from a replay of an old packet, B selects a new nonce value, N2', and

transports it as well as its session ID (X2) in a hello packet to check whether the packet is from A. In the neighbor field of this packet, B continues listing A with the earlier session ID and nonce values (i.e., X1 and N1). Therefore, if an attacker attempts to send an antique packet to masquerade as A, A would update its database with the new nonce of B and send a hello packet with its existing Session ID and nonce values (X1 and N1). In step 3, B receives a new hello packet consisting of B's new nonce value from A. Since this packet lists B with the new nonce value in the neighbors field of the hello and since the nonce is new, this packet cannot be a replay. Now, B can safely assume that A has indeed restarted and can start using the new session ID and the nonce values sent by A in the neighbor field of its hellos.

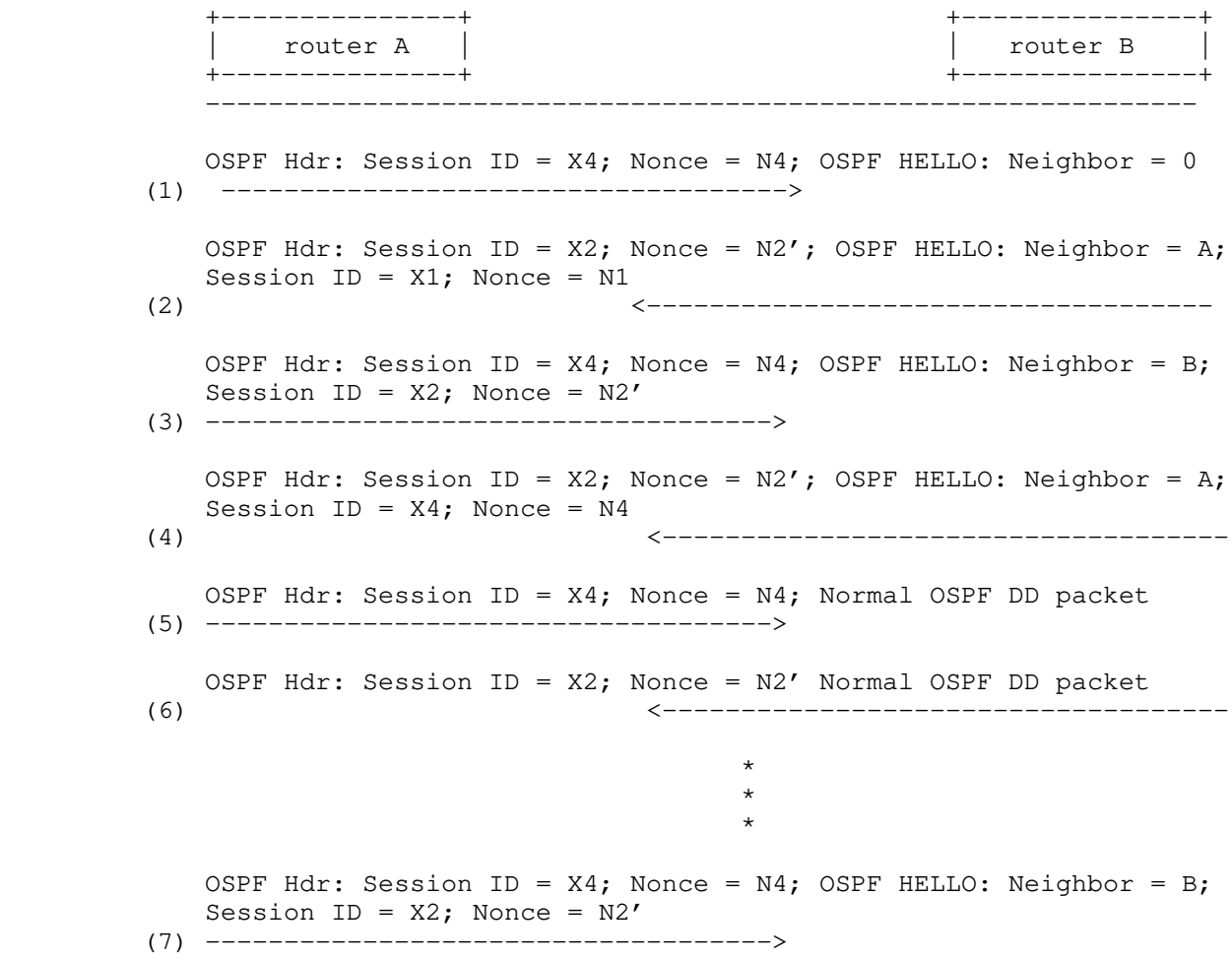


Figure 3. Scenario: Router A Reboot

2.1. Neighbor State Required

This authentication type requires the following additional fields be stored per neighbor:

- o The session ID most recently received from a neighbor
- o The nonce most recently received from a neighbor; this only needs to be kept up-to-date when the session ID changes or when establishing an adjacency

- o A set of sequence numbers for the neighbor; if packets are sometimes processed out of order, then a sequence number MAY be maintained for each type of packet

2.2. Receiver Behavior

This section describes how OSPF receivers will handle the reception of packets with the nonce and session ID.

If a packet is received for a neighbor in at least the 2-way state, then the session ID is compared to the one stored in the neighbor table. If the session ID does not match the session ID recorded with the neighbor, and the packet is not a hello, the packet is discarded. If the packet is a hello, then rules for hellos in following paragraphs apply. Otherwise, if the session ID matches, then if the sequence number in the cryptographic authentication option is not strictly greater than the sequence number associated with the neighbor for this type of packet, then the packet is discarded. If the cryptographic verification of the checksum fails, the packet is discarded. Otherwise, the packet is accepted by the cryptographic authentication and the sequence number associated with the neighbor for this packet type is updated to be the sequence number in the packet. The router MAY update the nonce associated with the neighbor to a nonce in a received hello packet. Updating the nonce is optional because the adjacency is already established. One case where a router implementationd would want to update nonces is where the router has recently changed session IDs without dropping all adjacencies. Such a session ID change is likely to be rare, either the result of a reboot that preserved adjacencies but might not preserve sequence numbers or running out of sequence number space.

If a hello is received for a neighbor that is not found or that has not reached 2-way state the following steps are performed. If a neighbor structure exists for the neighbor and the session ID match that stored in the neighbor structure, then the packet is processed as follows. The sequence number is checked and MUST be strictly greater than the sequence number in the neighbor structure. The cryptographic authentication is verified. If this router is listed in the set of neighbors in the hello packet, the nonce and session ID MUST match this router's current nonce and session ID. If any of these checks fail, the packet is discarded. Otherwise the packet is accepted past cryptographic processing.

By this point, the router has received a hello packet. Either no neighbor structure exists or the session ID has changed. Before permitting communication with this router, its liveness needs to be challenged. If a neighbor has been deleted (because of a timeout) since the last nonce trigger, then a nonce trigger (see Section 2.3is

performed and the packet is discarded. If this router is listed in the list of neighbors, it MUST be listed with its current session ID and nonce otherwise the packet is discarded. If verification of the cryptographic checksum fails, the packet is discarded. If the neighbor is already in 2-way state or greater and this router is not listed in the set of neighbors, the packet is discarded. Otherwise, the session ID, nonce and all sequence numbers associated with the neighbor are updated from the packet and the packet is accepted by cryptographic authentication processing.

2.3. Nonce Triggers

The router keeps track of whether a nonce trigger has happened since the last time a neighbor is deleted.

In order to test liveness, a router updates its current nonce to a new value. As a side effect, all routers on the link that do not already have an adjacency with this router will update the nonce associated with this router. More importantly, though, the router we are testing liveness with will update the nonce in its hello entry for this router. That will allow this router to confirm that the session ID is correct and corresponds to current replay state.

As part of a nonce trigger, the router updates its current nonce. If a hello has not been sent too recently, then a hello is sent with the new nonce. The nonce trigger state is updated to indicate that no new neighbors have been deleted since the last nonce trigger.

3. Packet Format

In the challenge/ response mechanism, every OSPFv2 packet MUST carry the current Session ID and the associated Nonce value. This section describes how this information is carried in the OSPFv2 packets.

The OSPF packet header includes an authentication type field, and 64-bits of data for use by the appropriate authentication scheme (determined by the type field). Authentication types 0, 1 and 2 are defined in [RFC2328]. This document defines Authentication type 3.

When using this authentication scheme the 64 bit Authentication field in the OSPF packet header remains unchanged and is the same as defined in Section D.3 of [RFC2328]. NOTE to the WG: We can also increase the size of the Key ID. Currently it has been kept as, but nothing prevents us from changing this.

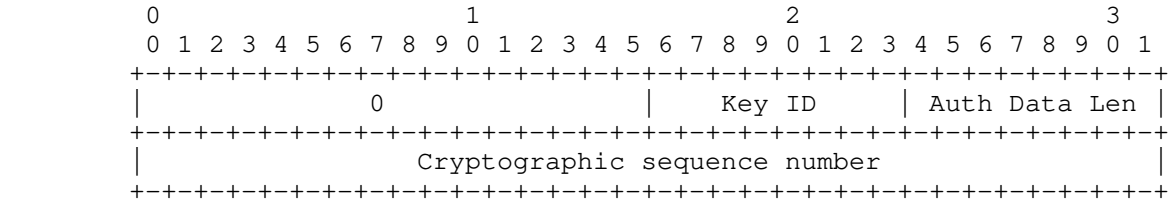


Figure 4.Usage of the Authentication field in the OSPF header when this mechanism is employed

The Session ID and the Nonce information is placed before the message digest that is appended to the OSPF packet. In this case too, the final Authentication data is not actually considered part of the OSPF protocol packet.

3.1. Extensions to OSPF packets

This section describes the new OSPFv2 packet format when this authentication scheme is being used.

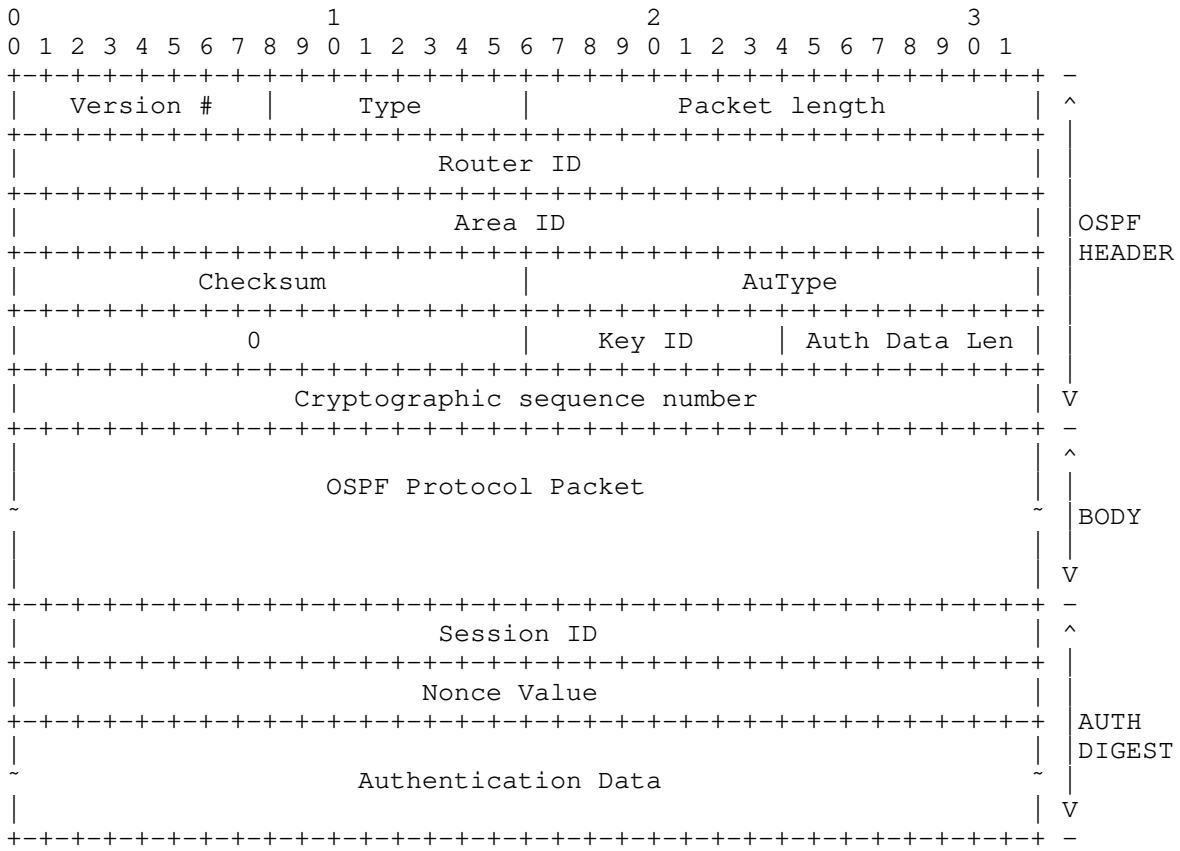
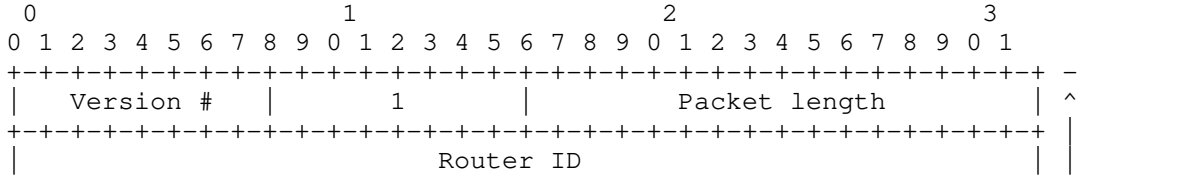


Figure 5.OSPFv2 Packet view

3.2. Extension of Hello Packet

The following figure shows an OSPF HELLO packet when this authentication scheme is being used. The HELLO payload has been modified to include each neighbor's Session ID and the Nonce value. The authentication data, as described above, carries the router's current Session ID and the Nonce value.



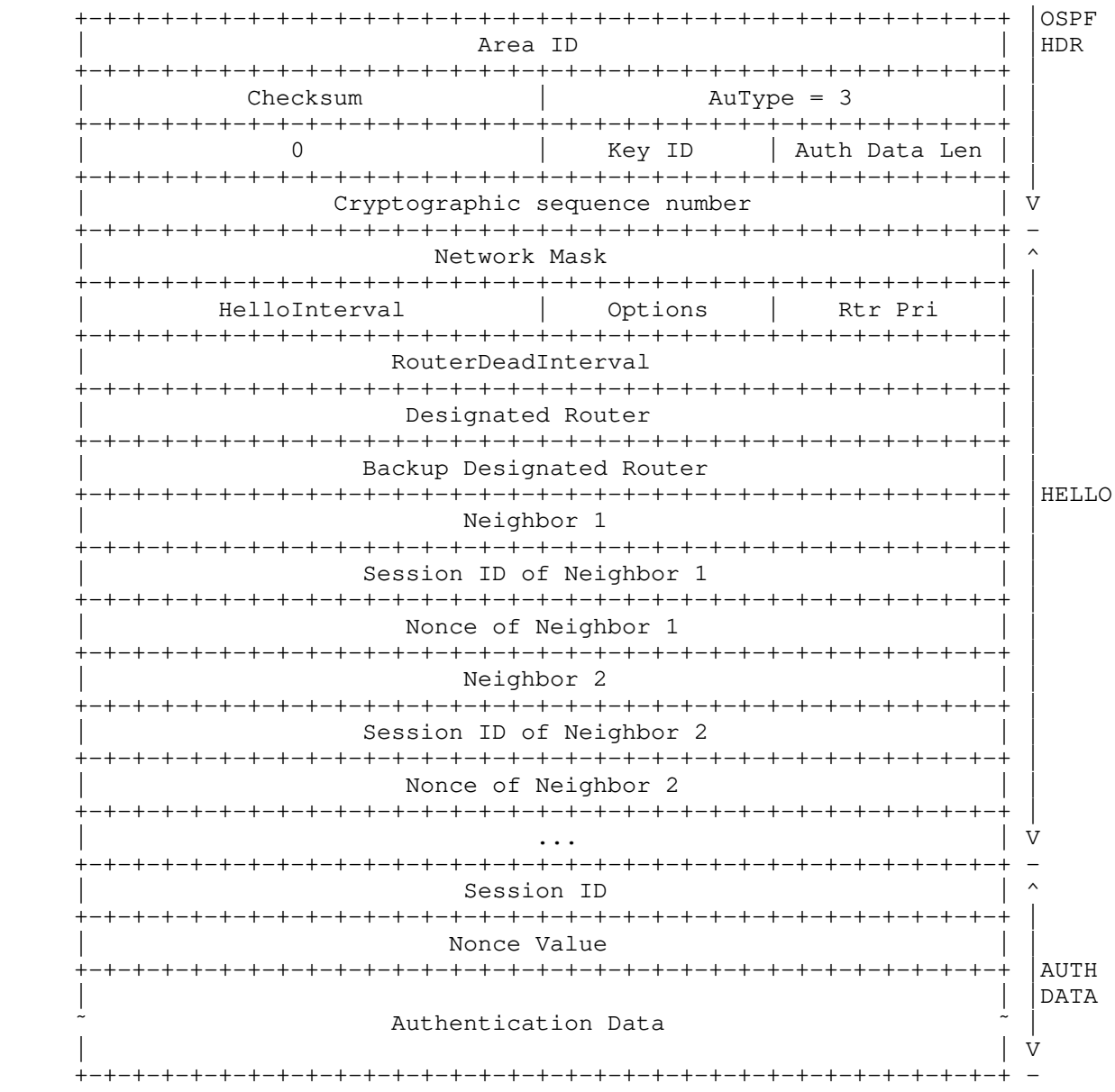


Figure 6.Extension of Protocol Packet

4. Key Selection in Processing OSPF Packets

This section introduces how the proposed security solution looks up long lived keys from key tables [I-D.ietf-karp-crypto-key-table]. Generally, a proper key selected to process an OSPFv2 packet should satisfy the requirements listed as follows:

the key is in its valid period; and

the key can be used for the desired security algorithm.

In the remainder of this section, other requirements that a selected key should particularly satisfy are depicted in different scenarios.

4.1. Key Selection in Sending Unicast OSPF Packets

Assume that a router R1 tries to send a unicast OSPF packet from its interface I1 to the interface R2 of a remote router R2 using security protocol P via interface I at time T. Firstly consider the circumstances where R1 and R2 are not connected with a virtual link. R1 then needs to select a long long-lived symmetric key from its key table. Because the key should be shared by the by both R1 and R2 to protect the communication between I1 and I2, the key should satisfy the following requirements:

the Peer field includes the router ID of R2;

the PeerKeyID field is not "unknown";

the Interfaces field includes I1; and

the Direction field is either "out" or "both".

When R1 and R2 are at the ends of a virtual link, the condition is a little more complex. Because the virtual link can be regarded as an unnumbered point-to-point network, the IP address of the interface actually used to send the packet (i.e., I1) is discovered during the routing table build process. Therefore, when the system operator deploys the keys to protect the virtual link, I1 has not been specified yet. Therefore, the key should be identified by the router IDs rather than by the interface originating the packet, and the third requirement introduced above should be changed to "the Interface field includes the router ID".

4.2. Key Selection in Sending Multicast OSPF Packets

If a router R1 sends an OSPF packet from its interface I1 to a multicast address (e.g., AllSPFRouters, AllDRouters), it needs to

select a key according to the following requirements:

the Peer field includes the multicast address;

the PeerKeyID field is "group";

the Interfaces field includes I1; and

the Direction field is either "out" or "both".

4.3. Key Selection on Receiving OSPF Packets

When Cryptographic Authentication is employed, the ID of the adopted key is encapsulated within the authentication field of an OSPF packet header. Using this ID, it is relatively easy for a receiver to locate the key. The requirement is relatively simple:

the Peer field includes the router ID of the sender; and

the PeerKeyID field includes the key ID obtained from the authentication field

5. Existing Cryptographic Authentication Mechanism

The overall cryptographic authentication process defined in [RFC5709] remains unchanged. To reduce the potential for confusion, this section minimises the repetition of text from RFC 5709 and is incorporated here by reference [RFC5709].

RFC 5709, Section 3.3, describes how the cryptographic authentication must be computed. It requires OSPFv2 packet's Authentication Trailer (which is the appendage described in RFC 2328, Section D.4.3, Page 233, items (6)(a) and (6)(d)) to be filled with the value Apad where Apad is a hexadecimal constant value 0x878FE1F3 repeated (L/4) times, where L is the length of the hash being used and is measured in octets rather than bits.

6. Mechanism to secure the IP header

This document updates the definition of Apad which is currently a constant defined in [RFC5709] to the source address thats carried in the IP header of the OSPFv2 protocol packet. Routers at the sending side must initialize Apad to a value of the source address that would be used when sending out the OSPFv2 packet, repeated L/4 times, where L is the length of the hash, measured in octets. The basic idea is to incorporate the source address from the IP header in the cryptographic authentication computation so that any change there can be detected.

At the recieving end implementations MUST initialize Apad as the source address that exists in the IP Header of the incoming OSPFv2 protocol packet, repeated L/4 times, instead of the constant that's currently defined in [RFC5709]. Besides changing the value of Apad this document does not introduce any other changes to the authentication mechanism described in [RFC5709].

This would prevent all attacks where a rogue OSPF router changes the source address of the protocol packet and reflects it on some other interface as the authentication check would fail and all such packets would get rejected.

7. Alternative Boot Count Approach

During discussion of the challenge/response authentication approach, a desire was expressed to have a simpler alternative to consider. This section presents an alternative that obtains most advantages of the challenge/response mechanism. Instead of adding nonces and session IDs, OSPF implementations are required to keep a count of the number of times they have booted in non-volatile storage. This requirement is also placed on agents by the SNMPv3 security architecture; the same boot count can be used both for SNMP and for this OSPF mechanism.

The OSPF sequence number is extended to be 64-bits rather than 32-bits. The most significant 32-bits are the boot count. The least significant 32-bits is a counter that increases for every packet sent.

A receiver verifies that the sequence number on a received packet is strictly greater than the sequence number of the previous packet received.

Requiring that each packet have a strictly greater sequence number is a change from the current OSPF security model. However this change is required for a number of the security guarantees.

This mechanism requires fewer changes to the OSPF packet than the challenge/response mechanism. Also, the implementation complexity is somewhat less.

However there are disadvantages. First, this mechanism requires that the boot count be maintained succesfully in nonvolatile storage. If the boot count ever goes backwards without changing the encryption key, then all the attacks against the current OSPF protocol become possible against this protocol until the time that the boot count reaches a value greater than the largest value ever used for this client. This can be particularly problematic if equipment is replaced, using a router ID that has been used previously on a link but with a fresh boot count.

Another disadvantage is that the boot count mechanism does not protect against a session replayed while a router is down. If a router crashes or is taken out of service, then an attacker can replay packets as soon as the adjacencies with the router time out. The vulnerabilities of this have not been fully analyzed. Potential vulnerabilities include attacks on the designated router election process and replays of complete sessions. So far it looks like it is not likely that an attacker could bring up a replayed session far enough to inject routes from a down router.

8. Security Considerations

This document attempts to fix the manual key management procedure that currently exists within OSPFv2, as part of the Phase 1 of the KARP Working Group. This therefore, only considers manual key management mechanism to be used for OSPFv2. Any solution that takes advantage of the automatic key management mechanism is beyond the scope of this document.

This document also provides a solution to prevent certain denial of service attacks that can be launched by changing the source address in the IP header of the OSPFv2 protocol packet.

9. IANA Considerations

This document requests a new Auth Type to be defined for OSPFv2. It currently uses 3 to foster pre-standard deployments.

10. Acknowledgements

The authors would like to thank Acee Lindem for valuable contributions and helping to understand the tradeoffs surrounding various solutions.

11. References

11.1. Normative References

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC5709] Bhatia, M., Manral, V., Fanto, M., White, R., Barnes, M., Li, T., and R. Atkinson, "OSPFv2 HMAC-SHA Cryptographic Authentication", RFC 5709, October 2009.

11.2. Informative References

[I-D.hartman-ospf-analysis] Hartman, S. and D. Zhang, "Analysis of OSPF Security According to KARP Design Guide", draft-hartman-ospf-analysis-02 (work in progress), December 2010.

[I-D.ietf-karp-crypto-key-table] Housley, R. and T. Polk, "Database of Long-Lived Symmetric Cryptographic Keys", draft-ietf-karp-crypto-key-table-00 (work in progress), November 2010.

[I-D.ietf-karp-threats-reqs] Lebovitz, G., Bhatia, M., and R. White, "The Threat Analysis and Requirements for Cryptographic Authentication of Routing Protocols' Transports", draft-ietf-karp-threats-reqs-01 (work in progress), October 2010.

[RFC6039] Manral, V., Bhatia, M., Jaeggli, J., and R. White, "Issues with Existing Cryptographic Protection Methods for Routing Protocols", RFC 6039, October 2010.

[RFC6094] Bhatia, M. and V. Manral, "Summary of Cryptographic Authentication Algorithm Implementation Requirements for Routing Protocols", RFC 6094, February 2011.

Authors' Addresses

Manav Bhatia
Alcatel-Lucent
Bangalore,
India

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

Sam Hartman
Painless Security

Email: hartmans@painless-security.com

Dacheng Zhang
Huawei Technologies co., LTD.
Beijing,
China

Phone:
Fax:
Email: zhangdacheng@huawei.com
URI:

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 8, 2011

M. Bhatia
Alcatel-Lucent
S. Hartman
Painless Security
D. Zhang
Huawei

March 7, 2011

A Generic Mechanism to solve Inter-Session Replay Attacks for Routing and Signaling Protocols
draft-bhz-karp-inter-session-replay-00

Abstract

This draft proposes a common solution for routing protocols to enhance their capability in tolerating inter-session replay attacks when using manual keys for securing their protocol packets.

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 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 8, 2011.

Copyright Notice

Table of Contents

1. Introduction

A replay attack is a network attack where an adversary intercepts a valid message transmission and retransmits it sometime later. In certain types of replay attacks, the retransmitted message may also be carefully tampered with. [RFC6039] demonstrates that nearly all the routing protocols and their security mechanisms are vulnerable to replay attacks to some extent. These attacks permit attackers multiple capabilities. Often, by replaying packets, attackers can create a disruption, causing routing information to be removed or signaling to fail because of the attack. Other replays permit an attacker to mask network failures. For example an attacker can maintain an adjacency even when a link or router has failed, allowing the attacker to observe traffic or forcing traffic to be blackholed. Another class of replay attacks permits an attacker to inject old routing information, possibly in place of routing information from a router that is currently down. Successful replay attacks on routing protocols can introduce incorrect routing information into the victims' routing tables, can break their adjacencies, and can eventually disrupt network communication.

Replays may be effective even with very little effort on the part of an attacker. For instance, replaying an OSPF Hello packet with an empty neighbor list can cause all the neighbor adjacencies with the router which originally sent the packet to be reset. All the existing security mechanisms for routing protocols use a non-decreasing cryptographic sequence number to deal with replay attacks. However, this leaves the routers still vulnerable to inter-connection replay attacks where the packets from one session are re-sent and accepted during a later session. None of the existing authentication mechanisms in the routing protocols can prevent this without the assistance of automatic key management mechanisms.

Providing routing protocols with an inter-session replay protection is one of the threats that has been recognized in scope for the work being done in the KARP WG and has been documented in [I-D.ietf-karp-threats-reqs]. This document proposes to provide a generic solution that can be implemented as part of the KARP framework that can be used by all routing and signaling protocols to prevent inter-session replay attacks.

This document proposes introducing a boot count, denoted as the KARP Boot Count (KBC), to enhance the capability of routing protocols in tolerating inter-session replay attacks. KBC is used to record the number of times a router has cold-booted. As a part of the KARP infrastructure, the value of this count must be maintained by all the implementations compliant to this standard in their non-volatile memory.

The following sections explain why the existing security and authentication mechanisms cannot protect the routing and the signaling protocols against inter-session replay attacks. The proposed solution is then introduced and we explain how unlike the existing anti-replay mechanisms, this solution will also work well with automated key management techniques.

2. Existing Mechanisms

Most routing protocols (e.g., OSPF, BFD, and RIP) and signaling protocols (LDP, RSVP, etc) include a non-decreasing cryptographic sequence number within the authentication data of each new packet that a router originates. The receivers keep track of this sequence number and only accept a protocol packet if it carries a cryptographic sequence number that is greater than or equal to the cryptographic sequence number carried in the last valid protocol packet. Using this mechanism, receivers can trivially protect the router against simple replay attacks.

[RFC2328] uses a 32-bit non-decreasing crypto sequence number for every OSPFv2 packet. Once a router has increased its sequence number, an attacker cannot replay an old packet to a neighbor that has an active adjacency without being detected. Note that the sequence numbers are not required to increase for each packet. Additionally, OSPFv2 provides a per-LSA sequence number to prevent an old LSA from being installed.

OSPFv3 [RFC5340] relies on the IP Authentication Header (AH) [RFC4302] and the IP Encapsulating Security Payload (ESP) [RFC4303] to cryptographically sign routing information passed between routers.

[RFC4552] describes the authentication mechanism that OSPFv3 uses. It discusses, at length, the reasoning behind using manually configured keys, rather than some automated key management protocol such as IKEv2 [RFC4306]. The primary problem is the lack of a suitable key management mechanism, as OSPFv3 adjacencies are formed on a one-to-many basis and most key management mechanisms are designed for a one-to-one communication model. Since [RFC4552] uses manual keying it clearly states that it provides no protection against replay attacks. This can be exploited in several ways as described in [RFC6039].

The OSPF WG is currently working on an alternate mechanism [I-D.ietf-ospf-auth-trailer-ospfv3] to protect OSPFv3 protocol packets that does not depend upon IPsec for authentication. This draft proposes a new mechanism that works similar to OSPFv2 [RFC5709] for providing authentication to the OSPFv3 packets and as a side

effect also solves the replay protection problems that exists in OSPFv3.

As part of the solution OSPFv3 routers append a special data block, referred to as, the authentication trailer to the end of the OSPFv3 packets. It contains a 32-bit non decreasing cryptographic sequence number that is used to protect against the replay attacks.

Bidirectional Forwarding Detection (BFD) is specified in [RFC5880]. There is a 32-bit cryptographic sequence number associated with every BFD packet that is used to protect against replay attacks. Note that the sequence number is incremented for each successive packet transmitted within a session for Meticulous Keyed (MD5 or SHA-1) Authentication. When using Keyed (MD5 or SHA-1) Authentication (the non-meticulous variant), the receiver of a packet only requires the sequence number of the packet to be greater than or equal to the last sequence number received.

In order to improve the anti-replay capability of RSVP, a 64-bit monotonically increasing sequence number is associated with every RSVP packet[RFC2747].

3. Inter-Session Replay Attacks

In the security mechanisms where the per-packet sequence numbers only need to be updated occasionally, replay attacks can be quite intuitive. For instance, an attacker can replay the last OSPFv2 packet without being detected since a router executing OSPFv2 accepts packets with sequence number greater than or equal to what they had last received. Of course, this issue can be easily addressed by mandating that protocols must only accept protocol packets if they come with a sequence number that is greater than what they have received till now. However, even if the sequence numbers are monotonically increased, the security mechanisms for routing protocols are still vulnerable to "inter-session" replay attacks if automatic key management mechanisms are unavailable. In normal conditions, it will take a very long period for a sequence number to reach its maximum. However, on many occasions (e.g., reboot), a router may re-initialize its sequence number. In this case, the sequence number of new packets is less than the sequence number of packets previously sent on the link. If an adversary replays the packets intercepted before the re-initialization, it is difficult for the victims to distinguish a replayed packet from the valid ones.

4. Proposal

The basic idea of the proposed solution is to guarantee that the sequence number of a router will always monotonically increase even after a cold reboot. The first part of the solution requires that the sequence numbers increase for every packet, updating the requirement of protocols such as OSPFv2 that only require non-decreasing behavior. This also means that BFD should use the meticulous version of the authentication mechanism as against the regular, since the former requires the cryptographic sequence number to increase for each successive packet that is transmitted for a session. It is insufficient to update the behavior of senders in this regard: receivers MUST check that sequence numbers increase for every packet.

The second part of the solution requires routing protocol implementations to maintain a KARP boot count (KBC) that records the number of times the router has cold booted in a non-volatile storage, similar to how it is done in the SNMPv3 security architecture. In fact, the same boot count MAY also be shared by SNMPv3 and the KARP infrastructure. Before sending out a packet, the routing protocols can request for this count value and can append it before the sequence space that it maintains. How each routing protocol achieve this is an implementation specific issue and beyond the scope of this document.

If the sequence number of a routing protocol (e.g., RSVP) is 64 bits, the sequence space is then broken down to two halves. The most significant 32-bits would indicate the KARP boot count. The least significant 32-bits is a counter that increases for every packet sent.

If the cryptographic sequence number of a routing protocol is 32 bits, it is recommended to extend the sequence number space to 64 bits. The most significant 32-bits would indicate the KARP boot count. The least significant 32-bits would carry the current sequence number that protocols maintain, which increases with each successive packet transmitted within a session. Upon receiving a packet, the receiver MUST verify that the sequence number in the packet is strictly greater than the sequence number of the previous packets received.

In the later case, if an implementation does not intend to expand the length of the sequence number, it could divide this 32-bit cryptographic sequence number space into a 7-bit and a 25-bit field. The most significant 7-bits could then indicate the KARP boot count. The least significant 25-bits is a counter that increases for every packet sent.

This solution assumes that boot counts never wrap within the lifetime of a particular encryption key. Also, the solution assumes that nonvolatile storage is always updated on a boot. Under these assumptions, a sequence number will not be re-used. This is sufficient to guarantee that while two routers are exchanging communications, packets from an old session cannot be replayed. However it does not demonstrate freshness. Many routing protocols discard replay state when an adjacency is dropped or when a router reboots. Once this state is discarded, an attacker can successfully replay packets from an old session. See the discussion in Section 5.

5. Security Considerations

This solution does not try to provide guarantees of freshness: it does not protect against the replay of an antique session while a router is down. For instance, if an OSPF router is taken out of service for some reason, an attacker can replay packets as soon as the adjacencies with the router time out. Actually, this issues is a common problem encountered by all existing anti-replay solutions for routing protocols. To address this issue, the liveliness of routers would need to be checked before the generation of any adjacency. The challenge/response solution is proposed in[I-D.bhatia-karp-ospf-ip-layer-protection] to address this issue.

Updates to routing protocols that use this solution need to discuss residual attacks, particularly those resulting from the lack of freshness guarantees. For example this solution would likely be insufficient for RIPv2 because as soon as a router goes down, old packets from that router could be used to inject routing information. However attacks against a link-state protocol may be quite limited and this solution may be appropriate.

The security of this solution depends on the boot count always increasing for each new boot unless the key changes. This creates significant operational requirements. If equipment is replaced but its router identity (an IP address for several protocols) is re-used, then the key MUST be changed or the boot count preserved from the old equipment. Failure to take one of these steps permits attackers to replay packets from the old equipment until the boot count of the new equipment catches up with that of the old equipment. This will very likely permit an attacker to disrupt adjacencies between the new equipment and other routers. More serious attacks may be possible as well.

6. IANA Considerations

The implementations that decide to extend their sequence space from 32 bits to 64 bits need to require a new Auth Type from IANA as this will be incompatible with the earlier authentication mechanisms.

7. Acknowledgements

The funding for Sam Hartman's work on this draft is provided by Huawei.

8. References

8.1. Normative References

[RFC2082] Baker, F., Atkinson, R., and G. Malkin, "RIP-2 MD5 Authentication", RFC 2082, January 1997.

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC2747] Baker, F., Lindell, B., and M. Talwar, "RSVP Cryptographic Authentication", RFC 2747, January 2000.

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

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, July 2008.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, June 2010.

8.2. Informative References

[I-D.bhatia-karp-ospf-ip-layer-protection]
Bhatia, M., Hartman, S., and D. Zhang, "Security Extension for OSPFv2 when using Manual Key Management", draft-bhatia-karp-ospf-ip-layer-protection-03 (work in progress), February 2011.

[I-D.ietf-karp-threats-reqs]
Lebovitz, G., Bhatia, M., and R. White, "The Threat Analysis and Requirements for Cryptographic Authentication

of Routing Protocols' Transports",
draft-ietf-karp-threats-reqs-01 (work in progress),
October 2010.

[I-D.ietf-ospf-auth-trailer-ospfv3]
Bhatia, M., Manral, V., and A. Lindem, "Supporting
Authentication Trailer for OSPFv3",
draft-ietf-ospf-auth-trailer-ospfv3-03 (work in progress),
February 2011.

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

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

[RFC4306] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol",
RFC 4306, December 2005.

[RFC5709] Bhatia, M., Manral, V., Fanto, M., White, R., Barnes, M.,
Li, T., and R. Atkinson, "OSPFv2 HMAC-SHA Cryptographic
Authentication", RFC 5709, October 2009.

[RFC6039] Manral, V., Bhatia, M., Jaeggli, J., and R. White, "Issues
with Existing Cryptographic Protection Methods for Routing
Protocols", RFC 6039, October 2010.

Authors' Addresses

Manav Bhatia
Alcatel-Lucent
India

Email: manav.bhatia@alcatel-lucent.com

Sam Hartman
Painless Security
USA

Email: hartmans@painless-security.com

Dacheng Zhang
Huawei
China

Email: zhangdacheng@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 15, 2011

S. Hartman
Painless Security
D. Zhang
Huawei

March 14, 2011

Multicast Router Key Management Protocol (MRKMP)
draft-hartman-karp-mrkmp-01.txt

Abstract

Several routing protocols engage in one-to-many communication. In order to authenticate these communications using symmetric cryptography, a group key needs to be established. This specification defines a group protocol for establishing and managing such keys.

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, 2011.

Copyright Notice

Table of Contents

1. Introduction

Many routing protocols such as OSPF and IS-Is use a one-to-many or multicast model of communications. The same message is sent to a number of recipients.

These protocols have cryptographic authentication mechanisms that use a key shared among all members of a communicating group in order to protect messages sent within that group. From a security standpoint, all routers in a group are considered equal. Protecting against a misbehaving router that is part of the group is out of scope for this protocol.

Routers need to be provisioned with some credentials for a one-to-one authentication protocol. Preshared keys or asymmetric keys and an authorization list are expected to be common deployments.

The members of a group elect a Group Controller/Key Server (GCKS). Potentially any member of the group may act as a GCKS. Since protecting against misbehaving routers is out of scope, there is no need to protect against an entity that is not currently the GCKS impersonating the GCKS.

To prove membership in the group, a router authenticates using its provisioned credentials to the current GCKS. If successful, the router is given the current key material for the group. Group size is relatively small and need for forced eviction of members is rare. If a GCKS needs to evict a member, then it can simply re-authenticate with the existing members and provide them new key material.

1.1. Terminology

1.2. Relationship to IKEv2

IKEv2 provides a protocol for authenticating IPsec security associations between two peers. It currently provides no group keying. IKev2 is attractive as a basis for this protocol because while it is much simpler than IKE, it provides all the needed flexibility in one-to-one authentication.

Unlike IKE, IKEv2 is explicitly designed for IPsec. The document does not separate handling of aspects of the protocol that would be needed for IPsec from those that apply to general key management. IPsec specific rules are combined with more general requirements. While concepts and protocol payloads can be used in a different key management protocol, the current structure of IKEv2 does not provide a mechanism for applying IKEv2 to a domain of interpretation other than IPsec. In addition, the complexity required in the IKE

specification when compared to IKEv2 suggests that the generality of IKE may not be worth the complexity cost.

So this protocol borrows concepts and payloads from IKEv2 but does not normatively depend on the IKEv2 specification.

1.3. Relationship to GDOI

The IPsec GDOI provides a protocol that is structurally very similar to this one. As specified, IKE can be used to provide phase 1 authentication to a GCKS. After that, GDOI provides phase 2 messages to establish key-encryption keys and traffic keys. Key management operations can be accomplished via GDOI messages sent to the group after the phase 2 exchange.

GDOI is defined for IKE not for IKEv2. In addition, GDOI's phase 2 uses its own hashing mechanism and nonce mechanism to provide integrity protection and replay protection. Like IKE, GDOI has significant complexity to support phase 2 identities that are different than the phase 1 identity. GDOI requires a GCKS to have a signature key used to sign GDOI messages. Since attacks caused by members of the group masquerading as the GCKS are out of scope, this is significant unnecessary complexity in the protocol.

So, this protocol can be thought of as a simplified GDOI based on IKEv2 rather than IKE. However, integrity and replay mechanisms are taken from IKEv2. Support for phase 2 identities is removed as unneeded complexity. Security for the group key management messages is provided using symmetric primitives rather than asymmetric signatures. Phase 1 authentication will often still involve asymmetric signatures.

2. Overview

2.1. Types of Keys

MRKMP manipulates several different types of symmetric keys:

preshared: Preshared keys are one mechanism for authenticating one router to another during the initial exchange. These keys are configured by some mechanism such as manual configuration or a management application outside of the scope of MRKMP.

peer key management key: Routers share a key with the GCKS that is a result of the mrkmp_init exchange.

KEK: A Key encryption Key (KEK) is a key used to encrypt group key management messages to the current members of a group. A KEK is learned as the product of establishing an MRKMP association or through a group key management message encrypted in a previous KEK. A KEK has an explicit expiration but may also be retired by a message encrypted in the KEK sent by the GCKS.

protocol master key: A protocol master key is the key exported by MRKMP for use by a routing protocol such as OSPF or IS-IS. The Protocol master key is the key that would be manually configured if a routing protocol is used without key management.

transport key: The transport key is the key used to integrity protect routing messages in a protocol such as IS-IS or OSPF. In today's routing protocol cryptographic authentication mechanisms the transport key is the same as the protocol master key. A disadvantage of this approach is that replay prevention is challenging with this architecture. Ideally some key derivation step would be used to establish a fresh transport key among all the participants in the group.

2.1.1. Key Encryption Key

When a router wishes to join a group, the router performs the mrkmp_init and mrkmp_auth exchange with a GCKS. During this process the router can establish an association with a specific group. Part of that association will be delivery of a KEK and associated parameters.

Group key management messages are sent to a group address not unicast to an individual peer. The group key management messages are protected using the KEK. The group key management messages need to provide both integrity and confidentiality protection using the KEK.

As part of establishing the association, the router joining the group is given an expiration time for the KEK. A group key management message may establish a new KEK with new parameters.

From time to time, a GCKS may wish to either force early expiration of a KEK or allow a KEK to expire. Protocol master keys are permitted to be valid for somewhat longer than the KEK that created them so as to avoid disrupting routing when this happens. When a KEK is retired or expires without being replaced by a new KEK announced in the old KEK, group members need to perform a new initial exchange to the GCKS. This is useful for example if a router is no longer authorized to be part of the group.

Other mechanisms such as LKH (section 5.4 [RFC2627]) could be used to permit removal of a group member while avoiding new initial authentications. However these mechanisms come at a complexity cost that is not justified for a small number of routers participating in a single multicast link.

2.1.2. Protocol Keys

Current routing protocols directly use the protocol master key to integrity protect messages. One advantage for this approach is that the initial hello messages used for discovery and capability exchange can be protected using the same mechanism as other messages. Typically a sequence number is used for replay detection. Without changing the key, the existing protocols are vulnerable to a number of serious denial of service attacks from replays.

The MRKMP can solve this replay problem by changing the protocol master key whenever a peer is about to exhaust its sequence number space or whenever a peer loses information about what sequence numbers it used. This could potentially involve changing the protocol master key whenever a router reboots that was part of the group using the current protocol master key. Since key changes will not disrupt active adjacencies and can be accomplished relatively quickly, this is not expected to be a huge problem. Note that after one key change, others routers can boot without causing additional key changes; a flurry of key changes would not be required if several routers reboot near each other.

Another approach would be to separate the protocol master key from the transport keys. For example the transport key used by a given

peer could be a fresh key derived from the protocol master key and nonces announced by that peer. Some mechanism would need to make sure that the peer's announcement of its nonce was fresh; this mechanism would almost certainly involve some form of interaction with the router wishing to guarantee freshness. There are two key advantages of this separation between transport keys and protocol master keys. The first is that the interaction between the MRKMP and routing protocol can be simplified significantly. The second is that even when manually configured protocol master keys are used, replay and adequate DOS protection can be achieved.

2.2. GCKS Election

Before a MRKMP system actually starts working, the routers in the multicast group need to select a GCKS so that they can obtain cryptographic keys to secure subsequent exchanges of routing information. MRKMP specifies an election protocol that dynamically assigns the responsibility of key management to one of the group members. Note that there are already announcer-electing mechanisms provided in some routing protocols (e.g., OSPF and IS-IS). However, much involvement between a MRKMP system and a routing protocol implementation will be introduced if the MRKMP system reuses the announcer-electing mechanism for the election of the GCKS. The state machine of the routing protocol also has to be modified. For instance, in OSPF, after a DR has been elected, routers need to halt their OSPF executions, and carry out the initial exchange to authenticate the DR and collect the keys for subsequent communications. After this step, the routers need to re-start their OSPF state machines so as to exchange routing information. As a consequence of such cases, an individual GCKS electing solution within MRKMP is preferable.

Each router has a GCKS priority. Higher priorities are more preferred GCKSes. As discussed in Section 8, the routing protocol can influence the GCKS election protocol by manipulating the priority so that it is likely that the same router will be the announcer for the routing protocol and the GCKS. Even if two different routers are elected as the announcer and GCKS, then the routing protocol and MRKMP will function correctly.

A key design goal of the election protocol is to maximize the chance that some router permitted to take on the role of GCKS will be elected to that role even when attackers are injecting messages into the election process. The election process can be attacked to cause a router other than the most preferred router to be elected.

2.3. Initial Exchange

The initial exchange is based on IKEv2's IKE_SA_INIT and IKE_SA_AUTH exchanges. During this exchange, an initiating router attempts to authenticate to the router it believes is a GCKS for a group that the initiating router wants to join. Messages are unicast from the initiator to the responding GCKS. Unicast MRKMP P messages form a request/response protocol; the party sending the messages is responsible for retransmissions.

The initial exchange provides capability negotiation, specifically including supported cryptographic suites for the key management protocol. Identification of the initiator and responder is also exchanged. A symmetric key is established to integrity protect and encrypt key management messages. While routing security does not typically require confidentiality, the key management protocol does because keys are exchanged and these must be protected.

Then the identities of each party are cryptographically verified. This can be done using a preshared key or symmetric keys. Other mechanisms may be added as a future extension.

The authentication exchange also provides an opportunity to join a group as part of the initial exchange. In the typical case, a router can obtain the needed key material for a group in two round-trips.

2.4. Group Join Exchange

The primary purpose of the unicast MRKMP messages is to get an initiator the information it needs to join a group and participate in a routing protocol. The initiator indicates what group it wants to join. XXX we need to discuss group naming--if MRKMP is limited to a subnet this may be as simple as saying that initiator wants to join the OSPF group or the IS-IS group.

The responder performs several checks. First, the responder confirms that the responder is currently acting as GCKS for the group in question. Then, the responder confirms that the initiator is permitted to join the group. If these checks pass, then the responder provides a key download payload to the initiator encrypted in the peer key management key. As discussed in Section 2.1.2, the GCKS MUST change the protocol master key if a router was part of the group under the current protocol master key and reboots. In this case, the GCKS SHOULD provide the new and old protocol master key to the initiator, setting the validity times for the old key to permit reception but not transmission. The GCKS MUST use the mechanism in the next section to flood the new key to the rest of the group.

A group association created by this exchange may last beyond the unicast MRKMP association used to create it. Once membership in a group is established, resources are not required to maintain the unicast association with the GCKS.

A member of a group can also use the unicast exchange to request a GCKS to change the protocol master key because that group has exhausted its available sequence space. For protocols where the protocol master key is the same as the transport key, it is critical that no two messages be sent by the same router with the same sequence number and protocol master key. The sequence number space is finite. So if a router is running low on available sequence space it needs to request a new protocol master key be generated.

2.5. Group Key Management

The GCKS shares a KEK with all members of a group. The GCKS can send a multicast message to the group to update the set of protocol master keys, update the KEK, or retire the KEK and request new group join exchanges.

Typically the protocol master key is changed only when needed to provide replay protection or when the KEK changes. The KEK changes whenever a new GCKS is elected or whenever it is administratively desirable to change the keys. For example if an employee leaves an organization it might be desirable to change the KEKs. A KEK is retired whenever forward security is desired: whenever the authorization of who is permitted to be in a group changes and the GCKS needs to make sure that the router is no longer participating. Most authorization changes such as removing a router from service do not require forward security in practical deployments.

3. GKCS Election

The GCKS election process selects a single router to act as GCKS for a group.Similar with other popular announcer electing mechanisms (e.g., VRRP, HSRP), in MRKMP, only GCKSes use multicast to periodically send Advertisement messages. Such advertisements can be used as heart beat packets to indicate the aliveness of GCKSes. In addition, a state machine with six states (Initial, Validate, GCKS, GCKS2, Follower, and Member) is specified for GCKS election. When a router is initially connected to a multicast network, its state is set as Initial. The router then sends a multicast initial advertisement. If a GCKS is working on the network, it will reply to the router with an advertisement. After receiving the advertisement from the GCKS, the router will try to register with the GCKS using the initial exchange. Typically this registration will succeed, and the state of the router is transferred to Member. After a certain period, if the router still does not receive any advertisement from a GCKS or other group members, the router then believes there is no other group member on the network and sets its state as GCKS. If during the period the router does not receive any advertisement from a GCKS but receives advertisements from other more preferred routers on the network, the router believes that the group is involved in a GCKS election process. The router then puts these routers into its candidate list. When the timer to end the Initial state expires, the router tries to authenticate the most preferred router in the candidate list and validate whether it can be a GCKS. If the validation result is possitive, the router then transfer its state to Member, and the router being validated transfers its state to GCKS.

In the absence of attacks, this process functions similar to designated router election protocols in existing routing protocols. Because the election process happens before group keys are established, the initial election process is not integrity-protected. An attacker can inject fake GCKS announcements or initial announcements from fake routers that are more preferred than any router actually in the group. Such attacks can create a denial of service situation. If the election process does not converge within the expected time, or if an authentication attempt fails, then the group is probably under attack. A new state called GCKS2 is introduced. A router permitted to be the GCKS can enter the GCKS2 state after failing to validate a received announcement in the expected time. GCKS2 is used to increase the convergence speed while the system is under attack. If an initial router receives a GCKS2 announcement, the initial router can authenticate and validate the sender, and transfer its own state to Follower, similar to how it would respond to a GCKS announcement. GCKS2 routers attempt to validate each other and to use the resulting security keys to establish a router to act as GCKS. The GCKS2 state does not generate

protocol master keys: until the election result in a GCKS only keying material needed for the election is produced. In the subsequent election, the router will wait for the election results from its GCKS2 router until its GCKS2 end timer expires. In this way, the authenticated entities generate a tree structure and avoid generating large amount of keks and protocol master keys when a adversary keeps sending fake GCKS announcements to distrupt election.

Apart from the initialization of a multicast network, the fail-over of a GCKS can also trigger an election process. For instance, if a router does not receive the heart beat advertisement for a certain period, it will transfer its state to Initial and try to elect a new one. In a GCKS electing process, a router has to stay in the Initial state until a new GCKS is allocated. Particularly, the router first sends its initial advertisement with its priority and waits for a certain period. During the period, if a router receives an initial advertisement which consists of a lower priority, the router then sends the advertisement again with a limited rate. After period, if the router does not find any router with a higher priority, it announces itself as the GCKS. If two routers have the same priority, the one with the lowest IP source address used for messages on the link will be the GCKS. After a router transfers its state to GCKS, it will reply to the initial advertisements from other routers with GCKS advertisements, even when the initial advertisements consist of higher priorities than its priority. This approach guarantees that a GCKS will not be changed frequently after it has been elected. After receiving the GCKS advertisement of the new elected GCKS, other routers transfer their states to Member. However, if a GCKS G1 receives a GCKS advertisement from another router G2 and G2 is a more preferred GCKS, G1 follows the procedure in Section 3.2.

If a node in state member fails to perform an initial exchange with the router it believes to be GCKS, it resets its state to initial but ignores advertisements from that router. This way an attacker cannot disrupt communications indefinitely by masquerading as a GCKS.

If a node transitions to GCKS state, it performs the procedure in Section 3.1.

3.1. A new GCKS is Elected

This section is a detailed description of the election process.

In the following discussion, the packets are identified by all upper case characters.

3.1.1. Parameters, Timers, and Events

Before going into detailed discussion, several parameters are introduced:

- Initial_Anno_Interval, which is the time interval between INITIAL_ANNOUNCEMENTS).
- Initial_End_Interval, which is the time interval to transfer the state of a router from Initial to GCKS/Validate if it does not receive any GCKS or GCKS2 announcement on the link).
- Validate_End_Interval, which is the time interval for a router to transfer its state from Validate to GCKS2 if it does not find any other more preferred router).
- GCKS_Down_Interval, which is the time interval for a Member router to declare a GCKS router is down).
- GCKS2_Down_Interval, which is the time interval for a Follower router to declare a GCKS2 router is down).
- GCKS2_End_Interval, which is the time interval for a router to transfer its state from GCKS2 to GCKS if it does not find any other more preferred router).
- GCKS_Anno_Interval, which is the time interval between GCKS_ANNOUNCEMENTS).
- GCKS2_Anno_Interval, which is the time interval between GCKS2_ANNOUNCEMENTS).

Correspondingly, each router in MRKMP has several timers, Initial_Anno_Timer, Initial_End_Timer, Validate_End_Timer, GCKS_Down_Timer, GCKS2_Down_Timer, GCKS2_End_Timer, GCKS_Anno_Timer, GCKS2_Anno_Timer. Initial_Anno_Timer fires to trigger sending of an INITIAL_ANNOUNCEMENT based on Initial_Announcement_Interval. Initial_End_Timer fires to trigger the transition of a router state from Initial to some other state. Validate_End_Timer fires to trigger the transition of a router state from Validate to GCKS2. GCKS_Down_Timer fires when no GCKS_ANNOUNCEMENT has been heard for GCKS_Down_Interval. GCKS2_Down_Timer fires when no GCKS2_ANNOUNCEMENT has not been heard for GCKS2_Down_Interval. GCKS2_End_Timer fires to trigger the transition of the state of a router from GCKS2 to GCKS. GCKS_Anno_Timer fires to trigger sending of a GCKS_ANNOUNCEMENT based on GCKS_Announcement_Interval. GCKS2_Anno_Timer fires to trigger sending of a GCKS2_ANNOUNCEMENT based on GCKS2_Anno_Interval.

During an election process, a MRKMP router may have to deal with following types of events:

- X_Anno_Received: an X_ANNOUNCEMENT is received.
- Requester_Validated: have authenticated and validated against a some router who believes we should be a GCKS or GCKS2.
- GCKS_Validated: a remote entity has been authenticated and validated to be a GCKS router.
- GCKS2_Validated: a remote entity has been authenticated and validated to be a GCKS2 router.
- Referral_Validated: have authenticated and validated against a candidate who is not a GCKS router but knows one is .
- Referral2_Validated: have authenticated and validated against a candidate who knows a GCKS2 router.
- Authentication/Validation_Failed: the remote entity fails in the authentication or cannot be either a GCKS/GCKS2 or a referral.
- X_Timer_Expired: the timer of type X expired.
- KEK_Expired: we have no valid KEK.

3.1.2. Initial

The timers utilized in this state are Initial_Anno_Timer and Initial_End_Timer.

On entry:

- Send an INITIAL_ANNOUNCEMENT.
- Set the Initial_Anno_Timer with Initial_Anno_Interval.
- Set the Initial_End_Timer with Initial_End_Interval.

Events:

- Initial_Anno_Timer_Expired: send an INITIAL_ANNOUNCEMENT and reset the Initial_Anno_Timer.
- Initial_Anno_Received: if the sender of the announcement is more preferred, add the entity into the candidate list; if less preferred, send an INITIAL_ANNOUNCEMENT with a limited rate.

- GCKS_Anno_Received: add the sender of the announcement to the candidate list; set the the Validate_End_Timer with the remaining period of Initial_End_Interval; transfer to validate.

- GCKS2_Anno_Received: add the sender of the announcement to candidate list; set the Validate_End_Timer with the remaining period of Initial_End_Interval; transfer to validate.

- Requester_Validated: If the requester is looking for a GCKS router and the local policy permits, transfer the state to GCKS2 setting GCKS2_End_Interval to time remaining on Initial_End_timer.

- Initial_End_Timer_Expired: if there are candidates, transfer the state to Validate. If there is no entry in the candidate list, transfer to GCKS.

3.1.3. Validate

The timer utilized in this state is Validate_End_Timer

Entering this state means that we have a router we believe should be GCKS. The purpose of this state is to confirm that e can establish a security association with that router and that router's policy permits it to be a GCKS for this group. The two normal paths through the state machine are Initial leading to GCKS for the most preferred router and Initial leading to Validate leading to Member for other routers.

On entry:

- Authenticate and validate the most preferred entry in the candidate list.

- If Validate_End_timer has more time than Validate_end_Interval, set Validate_End_timer to Validate_End_interval.

Events:

- GCKS_Validated: transfer the state to Member.

- GCKS2_Validated: Transfer the state to Follower.

- Referral_Validated: perform the authentication/validation on the recommended node; move the referring from the candidate list to the black list for Blacklist_Interval.

- Referral2_Validated: perform the authentication/validation on the recommended node; move the referring node from the candidate list

to the black list for Blacklist_Interval.

- Requester_Validated: If the requester is looking for a GCKS/GCKS2 router and the local policy permits, transfer the state to GCKS2

- Validation_Failed: move the router being validated from the candidate list to black list for Blacklist_interval.

- Initial_Anno_Received: if the sender of the announcement is more preferred, add the router into the candidate list; if less preferred, send an INITIAL_ANNOUNCEMENT with a limited rate.

- GCKS_Anno_Received: add the router sending the announcement into the candidate list and perform authentication against that entity.

- GCKS2_Anno_Received: add the router sending the announcement into the candidate list and start the authentication/validation against that entity.

- Validate_End_Timer_Expired: transfer the state to GCKS2.

3.1.4. GCKS2

The timers utilized in this state include GCKS2_Anno_Timer and GCKS2_End_Timer.

This state is not expected to be used in normal operation. This state indicates there has been some authentication/validation problem or another node is behaving in a manner inconsistent with the election state. The purpose of this state is to establish sufficient security keys to integrity protect the election process. It is possible during normal operation to send a brief time in this state if the router being elected GCKS gets an authentication request before Initial_End_timer expires.

On entry:

- Send an GCSK2_ANNOUNCEMENT.

- Set the GCKS2_Anno_Timer with GCKS2_Anno_Interval.

- Set the the GCKS2_End_Timer with GCKS2_End_Interval unless it was set on entry transferring from Initial.

Events:

- GCKS_Anno_Received: add to candidate list; start authentication/ validation.

- GCKS2_Anno_Received: if more preferred, add to candidate list, start authentication/validation. If less preferred, send GCKS2_ANNOUNCEMENT if rate limiting is permitted.

- GCKS_Validated: Transfer to member state; flood KEK to the associated followers.

- GCKS2_Validated: Transfer the state to Follower; flood KEK to the associated followers.

- Referral_Validated: Perform authentication and validation on the recommended node; move the referring node from the candidate list to the black list for Blacklist_Interval.

- Referral2_Validated: if the recommended GCKS2 is more preferred, perform authentication and validation on the recommended node; move the referring from the candidate list to the black list for Blacklist_Interval.

- Requester_Validated: if the requester is looking for a GCKS2, distribute kek.

- Validation_Failed: move the router being validated from the candidate list to black list for Blacklist_interval.

- GCKS2_End_Timer_Expired: transition the state to GCKS.

- GCKS2_Anno_Timer_Expired: send a GCKS2_ANNOUNCEMENT.

3.1.5. GCKS

The timer utilized in this state is GCKS_Anno_Timer.

On entry:

- Senda GCKS_ANNOUNCEMENT.

- Set the GCKS_Anno_Timer with GCKS_Anno_Interval.

- Generate protocol keys; if needed, generate KEK.

Events:

- GCKS_Anno_Timer_Expired: send a GCKS_ANNOUNCEMENT.

- Initial_Anno_Received: send an GCKS_ANNOUNCEMENT immediately if the rate limiting is permitted.

- GCKS2_Anno_Received: send an GCKS_ANNOUNCEMENT immediately if the rate limiting is permitted.

- GCKS_Anno_Received: if the sender is more preferred, add to candidate list and start authentication/validation; Otherwise, send an GCKS_ANNOUNCEMENT immediately if the rate limiting is permitted.

- GCKS_Validated: start network merging operations as what is illustrated in Section 3.2.

- Requester_Validated: If the requester is looking for a GCKS router, distribute kek and protocol master keys; if the requester is another GCKS, start network merging operations as what is illustrated in Section 3.2.

3.1.6. Member

The timer utilized in this state is GCKS_Down_Timer.

On entry:

- Set the GCKS_Down_Timer with GCKS_Down_Interval.

Events:

- GCKS_Down_Timer_Expired: Transfer the state into Initial.

- GCKS_Anno_Received: reset GCKS_Down_Timer.

- Requester_Validated: if the requester is legal, recommend the GCKS router to it.

3.1.7. Follower

The timer utilized in this state is GCKS2_Down_Timer.

On entry:

- Set the GCKS2_Down_Timer with GCKS2_Down_Interval.

Events:

- GCKS2_Down_Timer_Expired: Transfer the state into Initial.

- GCKS2_Anno_Received: reset GCKS2_Down_Timer.

- o GCKS_Anno_Received: Add the announcer to the candidate list and start validation.

- o Requester_Validated: if the requester is legal, recommend the GCKS2 router to it.

- o GCKS_Validated: Transfer the state to member.

3.2. Merging Partitioned Networks

Whenever a GCKS finds that a more preferred router is also acting as a GCKS for the same group, then the group is partitioned. Typically if there is already an active GCKS for a group, even if a more preferred GCKS joins, the GCKS will not change. Two situations can result in multiple GCKSes active for a group. The first is that members of the group do not share common authentication credentials. The second is that the group was previously partitioned so that some nodes could not see election messages from other nodes. After the problem resulting in the partition is fixed, then both active GCKSes will see each others election announcements. The group needs to merge.

The less preferred GCKS performs a unicast mrkmp_merge_sa unicast key management message to the more preferred GCKS. In this message the less preferred GCKS includes its key download payload, so the more preferred GCKS learns the protocol master keys of the less preferred GCKS.

The more preferred GCKS generates a new key download payload including a KEK and the union of all the protocol master keys. The GCKS SHOULD mark the existing protocol master keys as expiring for usage in transmitted packets in a relatively short time. The GCKS SHOULD introduce a new protocol master key. This key download payload is returned to the less preferred GCKS and is sent out in the current KEK using a group key management message.

The less preferred GCKS sends the received key download payload encrypted in its existing KEK. XXX how many retransmits. After all retransmissions of this payload the less preferred GCKS sets its state to member.

As a result of this procedure, members learn the protocol master keys of both GCKSes and converge on a single KEK and GCKS. Changing the protocol master keys during a merge is important for protocols that use the protocol master key as a transport key. The new GCKS does not know which routers have joined the group with the other GCKS. Therefore, it could not correctly detect one of these routers rebooting and change the protocol master key at that point. If the

key is changed as part of the merge, replays are handled.

3.3. Operations on Receiving a Packet

When a router attempts to join an election process, it may have a valid kek. For instance, when a GCKS cannot work properly, the routers on the link need to transfer their state to Initial and raise an election to find a new valid GCKS. If there is Still a valid kek shared by the router, they can use the kek to secure the packets transmitted during the election until a new kek is distributed by the new GCKS. A router holding the valid kek is regarded to be more preferred than a router which doesn't have the key. By using the kek, it is able to prevent an attacker from disturbing the election process by broadcasting fake announcements. Therefore, after an initial router does not find any more preferred router holding the valid key, it then can transfer its state to GCKS directly.

Therefore, the operations on receiving a packet are as follows:

- Check the blacklist. If the sender of the packet is on the blacklist, discard the packet.
- If the state is GCKS, accept the packet and generate an event. GCKS announcements need to be excepted in GCKS state for merges to work.
- If there is a KEK that is not expired, check the packet integrity against any matching KEK.
- If no KEK matches or if the integrity fails to validate, discard the packet.
- If there is no KEK at all or the KEK integrity check passed, process the packet and generate an event.

It is notable this approach limits the scope of the election within the routers managed by the failed GCKS. If there are routers newly accessing the link during the election, no router with a KEK will process their packets. However these routers can process packets from routers with the KEK. In many cases one of the routers with a KEK will be elected GCKS and the other routers can authenticate and join. In the worst case, two independent GCKSes will be elected and then merge.

4. Key Download Payload

What all is actually in the message you get at the end of phase 2 and that is sent out periodically during group key management

For the KEK, this needs to include the key itself, the algorithm (presumably drawn from the IKEv2 symmetric algorithms), key ID, group ID transmit start time, receive start time, and expire time.

The protocol master keys include the key, an algorithm ID, the key ID and thelifetimes.

5. Initial Exchange Details

6. Group Management Unicast Exchanges

6.1. Group Join Exchange

If a router receives a group join exchange for a group for which it is not the GCKS, it MUST return a notification. If it knows the GCKS for the group then it returns MRKMP_WRONG_GCKS including the address of the GCKS or GCKS2 in the notification payload along with an indication of whether the router is a GCKS or GCKS2. The initiator tries the group join exchange (probably with a new initial exchange) with the indicated router. If the responder does not know the GCKS for the group, either because it is not a member of the group or because its GCKS election state is initial, it returns the MRKMP_GCKS_UNKNOWN notification.

7. Group Key Management Operation

7.1. General operation

Periodically the GCKS will send out an update message encrypted in the current KEK including the current group key download payload and parameters. If a new KEK is about to be valid for receiving messages, this is included. Any protocol master keys that are valid for sending or receiving SHOULD be included.

If a previous KEK is still valid for sending, then an update message is sent encrypted in the old KEK. This message MUST include the new KEK. This message SHOULD include the protocol master keys.

7.2. Out of Sequence Space

7.3. Changing the Active GCKS

8. Interface to Routing Protocol

This section describes signaling between MRKMP and the routing protocol. The primary communication between these protocols is that MRKMP populates rows in the key table making protocol master keys available to the routing protocol. However additional signaling is also required from the routing protocol to MRKMP. This section discusses that signaling. All required communication from MRKMP to the routing protocol can be accomplished by manipulating the key table. However an implementation MAY wish to signal MRKMP failures to the routing protocol in order to provide consistent management feedback.

8.1. Joining a Group

When a routing protocol instance wishes to begin communicating on a multicast group, it signals a group join event to MRKMP. This event includes the identity of the group as well as this router's priority for being a GCKS for the group. When MRKMP receives this event, it starts MRKMP for this group and attempts to find a GCKS.

8.2. Priority Adjustment

It is desirable that the GCKS function track the functions within a routing protocol. For example for protocols such as OSPF that designate a router on a link to manage adjacencies for that link, it would be desirable for the GCKS role to be assigned to that router. The routing protocol provides a priority input to the GCKS election process. Initially the routing protocol should map any priority mechanism within the routing protocol to the GCKS election procedure so that routers favored as announcer for a link will also be favored as a GCKS.

However, the routing protocol SHOULD also dynamically manipulate the GCKS election priority based on what happens within the routing protocol. The router actually elected as the announcer SHOULD have a GCKS election priority higher than any other group member. Typically, by the time the routing protocol is able to elect an announcer, a GCKS will already be chosen. However, if a GCKS election is triggered when the routing protocol is already operational, then the election can choose the routing protocol's announcer.

8.3. Leaving a Group

If a routing protocol terminates on an interface, MRKMP needs to be notified that group is no longer joined. MRKMP MUST stop participating in the GCKS election process, stop monitoring for key

management messages and if the current router is a GCKS, stop acting in that role.

9. Security Considerations

An attacker who can suppress packets sent to the group can create a denial of service condition. One attack is to suppress GCKS election packets and cause two routers to believe they are both the GCKS for the group. If the least preferred router never hears the GCKS advertisement from the more preferred router, then the group will remain partitioned. Such an attacker is likely to be able to mount more direct denial of service, for example suppressing the actual routing protocol packets.

The election protocol has been designed to try and resist denial of service conditions. However, the election protocol maintains state in the form of a candidate list and black list. An attacker can consume state by generating fake election announcements. An implementation can discard state if it has insufficient resources. However, if legitimate routers are discarded from the candidate list, the protocol may take longer to converge or may not converge. If entries are removed from the black list, then more resources may be spent on attackers. So the solution has some residual denial of service possibilities. The election protocol requires significant analysis to confirm it meets its design goals.

The security of the election protocol depends on the denial of service resistance of the authentication protocol. It is important that an attacker not be able to cause an authentication to fail by injecting a packet. So, rather than failing an authentication if a bad packet is received, an implementation needs to wait and see if a good packet appears in some timeout.

The security of the system as a whole depends on the pair-wise security between the router currently in the GCKS role and the other routers in the group. Since any router can potentially act as GCKS, the pair-wise security between all members of the group is critical to the security of the system. In practical deployments, information used by the router acting as GCKS to authorize a member joining the group will be configured by some management application. In these deployments, the security of the system depends on the management application correctly maintaining this information on all routers potentially in the group.

10. Acknowledgements

The funding for Sam Hartman's work on this document is provided by Huawei.

XXX add the list of people in the lunch time group unless they are willing to be listed as authors.

11. References

[RFC2627] Wallner, D., Harder, E., and R. Agee, "Key Management for Multicast: Issues and Architectures", RFC 2627, June 1999.

Authors' Addresses

Sam Hartman
Painless Security

Email: hartmans-ietf@mit.edu

Dacheng Zhang
Huawei

Email: zhangdacheng@huawei.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 4, 2011

S. Hartman
Painless Security
D. Zhang
Huawei

March 3, 2011

Analysis of OSPF Security According to KARP Design Guide
draft-ietf-karp-ospf-analysis-00.txt

Abstract

This document analyzes OSPFv2 and OSPFv3 according to the guidelines set forth in section 4.2 of draft-ietf-karp-design-guide.

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 4, 2011.

Copyright Notice

Table of Contents

1. Introduction

This document performs the initial analysis of the current state of OSPFv2 and OSPFv3 according to the requirements of [I-D.ietf-karp-design-guide]. This draft builds on several previous analysis efforts into routing security. The OPSEC working group put together [RFC6039] an analysis of cryptographic issues with routing protocols. Earlier, the RPSEC working group put together [I-D.ietf-rpsec-ospf-vuln] a detailed analysis of OSPF vulnerabilities.

OSPF meets many of the requirements expected from a manually keyed routing protocol. Integrity protection is provided with modern cryptographic algorithms. Algorithm agility is provided: the algorithm can be changed as part of re-keying an interface or peer. Intra-connection re-keying is provided by the specifications, although apparently some implementations have trouble with this in practice. OSPFv2 security does not interfeer with prioritization of packets.

However, some gaps remain between the current state and the requirements for manually keyed routing security expressed in [I-D.ietf-karp-threats-reqs] the requirements. This document explores these gaps and proposes directions for addressing the gaps.

1.1. Requirements to Meet

There are a number of requirements described in section 3 of [I-D.ietf-karp-threats-reqs] thatt OSPF does not currently meet:

Secure Simple PSKs: Today, OSPF directly uses the key as specified. Related key attacks such as those described in section 4.1 of [I-D.hartman-karp-ops-model] are possible.

Replay Protection: OSPFv3 has no replay protection at all. OSPFv2 has most of the mechanisms necessary for intra-connection replay protection. Unfortunately, OSPFv2 does not securely identify the neighbor with whom replay protection state is associated in all cases. This weakness can be used to create significant denial-of-service issues using intra-connection replays. OSPFv2 has no inter-connection replay protection; this creates significant denial-of-service opportunities.

Packet Prioritization: OSPFv3 uses IPsec to process packets. This complicates implementations that wish to process some packets such as hellos and acknowledgements above others. In addition, if IPsec replay mechanisms were used, packets would need to be processed at least by IPsec even if they were low priority.

Neighbor Identification: In some cases, OSPF identifies a neighbor based on the IP address. This is never protected with OSPFv2 and is not typically protected with OSPFv3.

The remainder of this document explains the details of how these requirements fail to be met and proposes mechanisms for addressing them.

1.2. Requirements notation

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].

2. Current State

This section describes the security mechanisms built into OSPFv2 and OSPFv3. There are two goals to this section. First, this section gives a brief explanation of the OSPF security mechanisms to those familiar with connectionless integrity mechanisms but not with OSPF. Second, this section explains the background necessary to understand how OSPF fails to meet some of the requirements proposed for routing security.

2.1. OSPFv2

Appendix D of [RFC2328] describes the basic procedure for cryptographic authentication in OSPFv2. An authentication data field in the OSPF packet header contains a key ID, the length of the authentication data and a sequence number. A message authentication code (MAC) is appended to the OSPF packet. This code protects all fields of the packet including the sequence number but not the IP header.

RFC 2328 defined the use of a keyed-MD5 MAC. While MD5 has not been broken as a MAC, it is not the algorithm of choice for new MACs.

However, RFC 5709 [RFC5709] adds support for the SHA [FIPS180] family of hashes to OSPFv2. The cryptographic authentication described in RFC 5709 meets modern standards for per-packet integrity protection. Thus, OSPFv2 meets the requirement for strong algorithms. Since multiple algorithms are defined and a new algorithm can be selected with each key, OSPFv2 meets the requirement for algorithm agility. In order to provide cryptographic algorithms beleived to have a relatively long useful life, RFC 5709 mandates support for SHA-2 rather than SHA-1.

These security services provide integrity protection on each packet. In addition, limited replay detection is provided. The sequence number is non-decreasing. So, once a router has increased its sequence number, an attacker cannot replay an old packet. Unfortunately, sequence numbers are not required to increase for each packet. For instance, because existing OSPF security solutions do not specify how to set the sequence number, it is possible that some implementation use, e.g., "seconds since reboot" as their sequence numbers. The sequence numbers is thus only increased by every second. Also, no mechanism is provided to deal with the loss of anti-replay state; if sequence numbers are reused when a router reboots, then inter-connection replays are streight forward. Also, because the IP header is not protected, the sequence number may not be associated with the right neighbor; this opens up opportunities for outsiders to perform replay attacks. See Section 3 for analysis

of these attacks.

The mechanism provides good support for key rollover. There is a key ID; in addition mechanisms are described for managing key lifetimes and starting the use of a new key in an orderly manner. Performing orderly key rollover requires that implementations support accepting a new key for received packets before using that key to generate packets. Section D.3 of RFC 2328 requires this support in the form of four configurable lifetimes for each key: two lifetimes control the beginning and ending period for acceptance while two lifetimes control the beginning and ending period for generation. This provides a superset of the functionality in the key table [I-D.ietf-karp-crypto-key-table] regarding lifetime.

The OSPFv2 replay mechanism does not handle packet priorities as described. If packets are processed out-of-order, then if the sequence number increases, packets processed later will be discarded.

2.2. OSPFv3

RFC 4552 [RFC4552] describes how the authentication header and encapsulating security payload mechanism can be used to protect OSPFv3 packets. This mechanism provides per-packet integrity and optional confidentiality using a wide variety of cryptographic algorithms. Because OSPF uses multicast traffic, only manual key management is supported. This mechanism meets requirements related to algorithm selection and agility.

The Security Parameter Index (SPI) provides an identifier for the security association. This along with other IPsec facilities provides a mechanism for moving from one key to another, meeting the key rollover requirements.

Because manual keying is used, no replay protection is provided for OSPFv3. Thus the intra-connection and inter-connection replay requirements are not met.

There is another serious problem with the OSPFv3 security: rather than being integrated into OSPF, it is based on IPsec. In practice, this has lead to deployment problems.

OSPF implementations generally prioritize packets in order to minimize disruption when router resources such as CPU or memory experience contention. When IPsec is used with OSPFv3, the offset of the packet type, which is used to prioritize packets, depends on what integrity transform is used. For this reason, prioritizing packets may be more complex for OSPFv3. One approach is to establish per-SPI filters to find the packet type and act accordingly.

3. Impacts of OSPF Replays

As discussed, neither version of OSPF meets the requirements of inter-connection or intra-connection replay protection. This section discusses the impacts of OSPF replays.

In OSPFv2, two facilities limit the scope of replay attacks. First, when cryptographic authentication is used, each packet includes a sequence number that is non-decreasing. In the current specifications, the sequence number is remembered as part of an adjacency: if an attacker can cause an adjacency to go down, then replay state is lost. Database Description packets also include a per-LSA sequence number that is part of the information that is flooded. Even if a packet is replayed, the per-LSA sequence number will prevent an old LSA from being installed. Unlike the per-packet sequence number, the per-LSA sequence number must increase when an LSA is changed. As a result, replays cannot be used to install old routing information.

While the LSA sequence number provides some defense, there are a number of attacks that are possible because of a per-packet replay. The RPSEC analysis [I-D.ietf-rpsec-ospf-vuln] describes a number of attacks that are possible because of per-packet replays. The most serious appear to be attacks against Hello packets, which may cause an adjacency to fail. Other attacks may cause excessive flooding or excessive use of CPU.

Another serious attack concerns Database Description packets. In addition to the per-packet sequence number that is part of cryptographic authentication for OSPFv2 and the per-LSA sequence numbers, Database Description packets also include a Database Description sequence number. If a Database Description packet with the incorrect sequence number is received, then the database exchange process will be restarted.

The per-packet OSPFv2 sequence number can be used to reduce the window in which a replay is valid. A receiver will harmlessly reject a packet whose per-packet sequence number is older than the one most recently received from a neighbor. Replaying the most recent packet from a neighbor does not appear to create problems. So, if the per-packet sequence number is incremented on every packet sent, then replay attacks should not disrupt OSPFv2. Unfortunately, OSPFv2 does not have a procedure for dealing with sequence numbers reaching the maximum age. It may be possible to figure out a set of rules sufficient to disrupt the damage of packet replays while minimizing the use of the sequence number space.

As mentioned previously, when an adjacency is dropped, replay state

is lost. So, after rebooting or when all adjacencies are lost, a router may allow its sequence number to decrease. An attacker can cause significant damage by replaying a packet captured before the sequence number decrease at a time after the sequence number decrease. If this happens, then the replayed packet will be accepted and the sequence number will be updated. However, the legitimate sender will be using a lower sequence number, so legitimate packets will be rejected. A similar attack is possible in cases where OSPF identifies a neighbor based on source address. An attacker can change the source address of a captured packet and replay it. If the attacker causes a replay from a neighbor with a high sequence number to appear to be from a low sequence number neighbor, then connectivity with that neighbor will be disrupted until the adjacency fails.

OSPFv3 lacks the per-packet sequence number but has the per-LSA sequence number. As such, OSPFv3 has no defense against denial of service attacks that exploit replay.

4. Gap Analysis and Specific Requirements

The design guide requires each design team to enumerate a set of requirements for the routing protocol. The only concerns identified with OSPF are areas where it fails to meet general requirements outlined in the threats and requirements document. This section explains how some of these general requirements map specifically onto the OSPF protocol and enumerates the specific gaps that need to be addressed.

There is a general requirement for inter-connection replay protection. In the context of OSPF, this means that if an adjacency goes down between two neighbors and later is re-established, replaying packets from before the adjacency went down cannot disrupt the adjacency. In the context of OSPF, intra-connection replay protection means that replaying a packet cannot prevent an adjacency from forming or disrupt an adjacency. Meeting the requirements for intra-connection and inter-connection replay protection is a significant gap between the optimal state and where OSPF is today.

Since OSPF uses fields in the IP header, the general requirement to protect the IP header and handle neighbor identification applies. This is another gap that needs to be addressed. Because the replay protection will depend on neighbor identification, the replay protection cannot be adequately adressed without handling this issue as well.

In order to encourage deployment of OSPFv3 security, an authentication option is required that does not have the deployment challenges of IPsec.

In order to support the requirement for simple preshared keys, OSPF needs to make sure that when the same key is used for two different purposes, no problems result.

In order to support packet prioritization, the information needed to prioritize OSPF packets (the packet type) MUST be at a constant location in the packet.

5. Solution Work

A security solution will be developed for OSPFv2 and OSPFv3 based on the OSPFv2 cryptographic authentication option. This solution will have the following improvements over the existing OSPFv2 option:

Detect liveness of neighbors by adding additional information to the Hello exchanges in order to detect inter-connection replay

Add a form of simple key derivation so that if the same preshared key is used for OSPF and other purposes, related key attacks do not result

Support OSPFv3 authentication without use of IPsec

Specify processing rules sufficient to permit replay detection and packet prioritization

Emphasize requirements already present in the OSPF specification sufficient to permit key migration without disrupting adjacencies

Specify the proper use of the key table for OSPF

Protect the source IP address

Require that sequence numbers be incremented on each packet

6. Security Considerations

This memo discusses and compiles vulnerabilities in the existing OSPF cryptographic handling.

In analyzing proposed improvements to OSPF per-packet security, it is desirable to consider how these improvements interact with potential improvements in overall routing security. For example, the impact of replay attacks currently depends on the LSA sequence number mechanism. If cryptographic protections against insider attackers are considered by future work, then that work will need to provide a solution that meets the needs of the per-packet replay defense as well as protection of routing data from insider attack. RFC 2154 [RFC2154] provides an experimental solution for end-to-end protection of routing data in OSPF. It may be beneficial to consider how improvements to the per-packet protections would interact with such a mechanism to future-proof these mechanisms.

7. Acknowledgments

Funding for Sam Hartman's work on this memo is provided by Huawei.

The authors would like to thank Ran Atkinson, Michael Barnes, and Manav Bhatia for valuable comments.

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.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

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

[RFC5709] Bhatia, M., Manral, V., Fanto, M., White, R., Barnes, M., Li, T., and R. Atkinson, "OSPFv2 HMAC-SHA Cryptographic Authentication", RFC 5709, October 2009.

8.2. Informative References

[FIPS180] US National Institute of Standards and Technology, "Secure Hash Standard (SHS)", August 2002.

[I-D.hartman-karp-ops-model] Hartman, S. and D. Zhang, "Operations Model for Router Keying", draft-hartman-karp-ops-model-01 (work in progress), October 2010.

[I-D.ietf-karp-crypto-key-table] Housley, R. and T. Polk, "Database of Long-Lived Symmetric Cryptographic Keys", draft-ietf-karp-crypto-key-table-00 (work in progress), November 2010.

[I-D.ietf-karp-design-guide] Lebovitz, G. and M. Bhatia, "Keying and Authentication for Routing Protocols (KARP) Design Guidelines", draft-ietf-karp-design-guide-00 (work in progress), February 2010.

[I-D.ietf-karp-threats-reqs] Lebovitz, G., Bhatia, M., and R. White, "The Threat Analysis and Requirements for Cryptographic Authentication of Routing Protocols' Transports", draft-ietf-karp-threats-reqs-01 (work in progress), October 2010.

[I-D.ietf-opsec-routing-protocols-crypto-issues] Jaeggli, J., Hares, S., Bhatia, M., Manral, V., and R. White, "Issues with existing Cryptographic Protection Methods for Routing Protocols",

draft-ietf-opsec-routing-protocols-crypto-issues-06 (work in progress), June 2010.

[I-D.ietf-rpsec-ospf-vuln]
Jones, E. and O. Moigne, "OSPF Security Vulnerabilities Analysis", draft-ietf-rpsec-ospf-vuln-02 (work in progress), June 2006.

[RFC2154] Murphy, S., Badger, M., and B. Wellington, "OSPF with Digital Signatures", RFC 2154, June 1997.

[RFC6039] Manral, V., Bhatia, M., Jaeggli, J., and R. White, "Issues with Existing Cryptographic Protection Methods for Routing Protocols", RFC 6039, October 2010.

Authors' Addresses

Sam Hartman
Painless Security

Email: hartmans-ietf@mit.edu

Dacheng Zhang
Huawei

Email: zhangdacheng@huawei.com

KARP Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 8, 2011

X. Liang
ZTE Corporation

March 07, 2011

Negotiation in Keying Management Protocols
draft-liang-karp-negotiation-kmp-00

Abstract

Negotiation is one prominent capability of keying managament protocols (KMPs), especially automated KMPs for Routing Protocols. This document discusses negotiation in KMPs, which includes the reasons for negotiation in KMPs, concerns and possible solutions when using negotiation, and several types of negotiation in KMPs.

Table of Contents

1. Introduction

In RFC2408 [RFC2408] about ISAKMP, the need for negotiation and what can be negotiated were discussed in section 1.2 and section 1.3, respectively. It pointed out that the main reason for security association (SA) negotiation is the diverse security requirements and security services of different networks, and the objective of negotiation is to achieve common supported security functionality for interoperation and cooperation between/among communicating peers.

In the following sections of this draft, more reasons for negotiation in KMPs, concerns and possible solutions when using negotiation, types and technical details of negotiation in KMPs will be discussed.

1.1. 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 RFC2119 [RFC2119].

When used in lower case, these words convey their typical use in common language, and are not to be interpreted as described in RFC2119 [RFC2119].

2. Why Need Negotiation

Before answer the question "Why need negotiation", it is good to make clear what negotiation is in the context of KMPs. Negotiation is one technique by which communicating parties exchange information, which includes settings, conditions, proposals, etc., and achieve some decisions in common to allow subsequent procedures to go on. In KMPs, communicating parties can establish secure channel by negotiating initial SAs, can protect application data by negotiating proper SAs for application being served. Besides the main reason stated in Section 1, there are specific reasons in KMPs especially KMPs for routing protocols, which are listed as follows.

- Algorithm agility: It has three meanings. First, the algorithm can be updated accordingly. For example, router A supports SHA-1, and later step by step, A can be updated and support SHA-1, SHA-256, SHA-384, and SHA-512. Second, alternative algorithm is available when the one in use is broken down. In the case router A supports both SHA-1 and SHA-256, when SHA-1 gets broken down, SHA-256 takes place of SHA-1. Third, when peers don!_t have the identical algorithms deployed, they can agree on using one algorithm in common by negotiation. For example, peer A supports SHA-1 and SHA-256, and peer B supports SHA-256 and SHA-384, then A

and B can communicate using SHA-256 by negotiation. In other words, the routers don!_t need to have the algorithm updating in sync.

- Implementation: Negotiation allows implementors to implement KMP or security mechanims or algorithms in differen ways, but to provide same parameters and/or API functions for interoperation. Because negotiation doesn't care about the implementation, it only care about the key parameters. Attention should be paid to implementation and upgrading properly, since problems are often caused by improper implementation and upgrading, see Section 3, and negotiation makes the problems known, which are not hidden again.

- Configuration: Negotiation reduces the complexity of configuration, reduces the volume of work of configuration. Negotiation releases the rigidity of configuration, and makes configuration flexible. Configuration will not need to be exactly the sync.

Negotiation is one function or ability/capability of KMP, especially automated KMP. Negotiation makes KMP more powerful, and helps KMP to realize the properties such as fresh traffic keys, managing SA lifetimes and easier rekeying.

In summary, negotiation benefits both operators/administrators and users, since negotiation makes configuration of security mechanism more flexible and less complicated, and makes deployment of security mechanism more ease and gradual, hence lowers the cost of operation.

3. Concerns and Possible Solutions When Using Negotiation

Some people argue that negotiation will cause unexpected consequences and make problems far more hidden. Looking closer at those situations, we will find that those problems are caused by improper implementation essentially, not by negotiation. Actually, negotation allows diverse implementations according to Section 2. In the other hand, if improper configurations and improper deployment exist, there will also be problems and unexpected consequences.

In KMPs for routing protocols, what to be negotiated are some parameters, and exactly, the key parameters, not the implementation details. The implementations can be different surely, but the parameters and/or the API functions associated with those parameters should be the same or identical according to some standards or references. If they are implemented with different parameter name, a third party tool could take the role to do the translation or

transform, look at translator or transformer in following paragraph for reference. That!_s not the fault caused by negotiation, since negotiation only involves parameters. In practice, if implementations do according to some rules which at least ask for the same parameters or the same API function, negotiation will not cause problems.

To reduce and solve the above mentioned problems, there may be two possible solutions:

- Translator or transformer: It may be from a third party, and is a tool to do the translation or transforming job for KMP and program. For example, two different implementations impelment the parameters with different name or using different API functions. In this case the translator or transformer can translate the names of the parameters or API functions between KMP and the program.

- Falling-back negotiation mechanism, or re-negotiation mechanism: It seems like backward compatibility capability. When the algorithm with higher security priority doesn't work properly by some reasons, the communicating parties can give up the algorithm, and re-negotiate one with lower security priority that can work. For example, two peers (A and B) negotiate cryptographic authentication, and everything is working before an upgrade. B supports SHA-1 and SHA-256, but A only supports SHA-1. A is upgraded and now supports SHA-256. Unfortunately, B's implementation of the routing authentication using SHA-256 is buggy. The upgrade causes things to break. When using falling-back negotiation mechanism or re-negotiation mechanism, SHA-1 will be used in the case that SHA-256 fails.

4. Negotiation in KMPs

There are several kinds of negotiation in KMPs, which are initial SA negotiation to establish secure channel, peer-to-peer SA negotiation for appplication data such as routing protocols, and group SA negotiation for appplication data such as routing protocols. KMPs such as ISAKMP [RFC2408], IKEv2 [RFC4306] [RFC5996], GDOI [I-D.ietf-msec-gdoi-update], G-IKEv2 [I-D.yeung-g-ikev2], and MRKMP [I-D.hartman-karp-mrkmp], will be discussed or involved in this section.

4.1. Initial SA Negotiation to Establish Secure Channel

In all the KMPs mentioned above, initial SA negotiation to establish secure channel is the first step and must take place, because secret keying materials will be transmitted under the protection of the

secure channel. ISAKMP and IKEv2 have its own exchanges to negotiate initial SA, and these exchanges are called phase 1 (or the first phase) exchange in ISAKMP or initial exchange in IKEv2. GDOI, G-IKEv2 and MRKMP exploit phase 1 exchange in ISAKMP or initial exchange in IKEv2 to fulfill their initial SA negotiation to establish secure channel.

In phase 1 exchange of ISAKMP, there are four types of exchange that are defined to negotiate initial SA and/or have identity authentication, i.e., base exchange, identity protection exchange, authentication only exchange, and aggressive exchange. Any of the four exchange types can be used individually. The initial SA is called ISAKMP SA in ISAKMP.

In initial exchange in IKEv2, there are exchanges (four messages in all) basically to perform initial SA negotiation and identity authentication. The first exchange is IKE_SA_INIT, which includes one pair messages, i.e., IKE_SA_INIT request and IKE_SA_INIT response. IKE_SA_INIT exchange is used to establish secure channel between the initiator and responder. The initial SA is called IKE SA in IKEv2. The second exchange is IKE_AUTH, which includes another pair messages, i.e., IKE_AUTH request and IKE_AUTH response. IKE_AUTH exchange is used to perform identity authentication and negotiate another SA, i.e., the first Child SA, which could be the content of Section 4.2.

4.2. Peer-to-peer SA Negotiation for Routing Protocols

Peer-to-peer SA negotiation for appplication data such as routing protocols happens in ISAKMP and IKEv2, and exactly, in phase 2 (the second phase) exchange of ISAKMP and the IKE_AUTH exchage and the CREATE_CHILD_SA exchange of IKEv2.

The phase 2 negotiation in ISAKMP is used to establish SAs for other security protocols or appplication data to protect many message/data exchanges. For example, SAs for routing protocols can be negotiated in phase 2 of ISAKMP and used to protect routing messages by the routing protocols. The phase 2 negotiation of ISAKMP also use the four types exchanges stated in Section 4.1, and the negotiated SA is protocol SA.

The IKE_AUTH exchage and the CREATE_CHILD_SA exchange of IKEv2 can be used to negotiate SAs for other security protocols or appplication data, and negotiated SAs are called Child SAs.

Both ISAKMP and IKEv2 could be modified to supporting or serving more applications with better security service [I-D.liang-karp-auto-sa-management-rp]. For example, the above

mentioned exchanges can be modified with new payload or extende payload, and new exchanges can be created.

4.3. Group SA Negotiation for Routing Protocols

It is easy to understand initial SA negotiation to establish secure channel and peer-to-peer SA negotiation for other security protocols or appplication data, but it is not easy to understand group SA negotiation for other security protocols or appplication data such as routing protocols in multicast mode or broacast mode, because the group protocols such as the mentioned above GDOI, G-IKEv2, and MRKMP, don't have the group SA negotiation capability. From Section 2, we can see that there are advantages to have negotiation capability, especially in group envirements, otherwise every router asking to join the group have to be configurated in sync.

How to negotiate group SA? One possible approach may be as follows. When GM and GCKS negotiate initial SA to establish secure channel and/or perform identity authentication, GCKS collects security parameters of other security protocols or appplication data such as routing protocols from GM, and those parameters are supported both by GCKS and GM. After a period of time, e.g., two seconds, GCKS have collects parameters from more than one GMs. According to the collected parameters from the legitimate GMs, GCKS chooses the parameters that are supported by all the legitimate GMs or most of the legitimate GMS in the case not all legitimate GMs support, to create the group SA. Under the protection of the initial SA, GCKS sends the created group SA to GMs one by one.

5. Security Considerations

To be completed.

6. IANA Considerations

To be completed.

7. Acknowledgement

To be completed.

8. References

8.1. Normative References

[I-D.ietf-msec-gdoi-update]
Weis, B., Rowles, S., and T. Hardjono, "The Group Domain of Interpretation", draft-ietf-msec-gdoi-update-07 (work in progress), October 2010.

[RFC2408] Maughan, D., Schneider, M., and M. Schertler, "Internet Security Association and Key Management Protocol (ISAKMP)", RFC 2408, November 1998.

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

[RFC4306] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.

[RFC4307] Schiller, J., "Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2)", RFC 4307, December 2005.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

8.2. Informative References

[I-D.hartman-karp-mrkmp]
Hartman, S. and D. Zhang, "Multicast Router Key Management Protocol (MRKMP)", draft-hartman-karp-mrkmp-00 (work in progress), October 2010.

[I-D.ietf-karp-design-guide]
Lebovitz, G. and M. Bhatia, "Keying and Authentication for Routing Protocols (KARP) Design Guidelines", draft-ietf-karp-design-guide-01 (work in progress), September 2010.

[I-D.ietf-karp-framework]
Atwood, W. and G. Lebovitz, "Framework for Cryptographic Authentication of Routing Protocol Packets on the Wire", draft-ietf-karp-framework-00 (work in progress), February 2010.

[I-D.ietf-karp-threats-reqs]
Lebovitz, G., Bhatia, M., and R. White, "The Threat Analysis and Requirements for Cryptographic Authentication of Routing Protocols' Transports",

draft-ietf-karp-threats-reqs-01 (work in progress), October 2010.

[I-D.liang-karp-auto-sa-management-rp] Liang, X., Wang, H., Wei, Y., and C. Wan, "Automated Security Association Management for Routing Protocols", draft-liang-karp-auto-sa-management-rp-00 (work in progress), October 2010.

[I-D.wei-karp-analysis-rp-sa] Wei, Y., Wang, H., Liang, X., and C. Wan, "Analysis of Security Association for Current Routing Protocols", draft-wei-karp-analysis-rp-sa-01 (work in progress), October 2010.

[I-D.yeung-g-ikev2] Rowles, S., Yeung, A., and P. Tran, "Group Key Management using IKEv2", draft-yeung-g-ikev2-01 (work in progress), March 2010.

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC2402] Kent, S. and R. Atkinson, "IP Authentication Header", RFC 2402, November 1998.

[RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

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

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

[RFC4305] Eastlake, D., "Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 4305, December 2005.

[RFC4718] Eronen, P. and P. Hoffman, "IKEv2 Clarifications and Implementation Guidelines", RFC 4718, October 2006.

[RFC4835] Manral, V., "Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 4835, April 2007.

[RFC5709] Bhatia, M., Manral, V., Fanto, M., White, R., Barnes, M., Li, T., and R. Atkinson, "OSPFv2 HMAC-SHA Cryptographic Authentication", RFC 5709, October 2009.

Author's Address

Xiaoping Liang
ZTE Corporation
No. 6, Huashen Avenue, Yuhuatai District
Nanjing, Jiangsu 210012
China

Phone: +86 25 52878217
Email: liang.xiaoping@zte.com.cn

Routing Working Group
Internet-Draft
Intended status: Informational
Expires: August 29, 2011

M. Jethanandani
K. Patel
Cisco Systems, Inc
L. Zheng
Huawei

February 25, 2011

Analysis of BGP, LDP and MSDP Security According to KARP Design Guide
draft-mahesh-bgp-ldp-msdp-analysis-00.txt

Abstract

This document analyzes BGP, LDP and MSDP according to guidelines set forth in section 4.2 of [draft-ietf-karp-design-guide].

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 29, 2011.

Table of Contents

1. Introduction

In March 2006 the Internet Architecture Board (IAB) in its "Unwanted Internet Traffic" workshop described an attack on core routing infrastructure as an ideal attack with the most amount of damage. It called for the tightening the security of the core routing infrastructure.

This document performs the initial analysis of the current state of BGP, LDP and MSDP according to the requirements of [draft-ietf-karp-design-guide]. This draft builds on several previous analysis efforts into routing security. The OPSEC working group put together [draft-ietf-opsec-routing-protocols-crypto-issues] an analysis of cryptographic issues with routing protocols and draft-hartman-ospf-analysis-01 which has a analysis for OSPF.

1.1. Contributing Authors

Anantha Ramaiah, Mach Chen

1.2. Abbreviations

BGP - Border Gateway Protocol

DoS - Denial of Service

KARP - Key and Authentication for Routing Protocols

KDF - Key Derivation Function

KMP - Key Management Protocol

LDP - Label Distribution Protocol

LSR - Label Switch Routers

MAC - Message Authentication Code

MSDP - Multicast Source Distribution Protocol

MD5 - Message Digest algorithm 5

OSPF - OPen Shortest Path First

TCP - Tranmission Control Protocol

UDP - User Datagram Protocol

2. Current State of BGP, LDP and MSDP

This section describes the security mechanisms built into BGP, LDP and MSDP or in the underlying transport protocol.

GTSM [RFC3682] describes a generalized Time to Live (TTL) security mechanism to protect a protocol stack from CPU-utilization based attacks. In addition, most vendors have their TCP based routing protocols do a access list check to permit packets only from known sources. These help preventing DoS attacks from unknown sources.

TCP Robustness [RFC5961] recommends some TCP level mitigations against spoofing attacks targeted towards long lived routing protocol sessions.

Session mode DoS attacks for LDP are the same attacks that TCP is vulnerable to such as SYN attacks. [To be updated]

TCP MD5 [RFC2385] specifies a mechanism to protect BGP and other TCP sessions via the TCP MD5 option. TCP MD5 option provides a way for carrying an MD5 digest in a TCP segment. This digest acts like a signature for that segment, incorporating information known only to the connection end points. The MD5 key used to compute the digest is stored locally on the router. MD5 does not provide a generic mechanism to support Key roll-over. This option is used by routing protocols to provide for session level protection against the introduction of spoofed TCP segments into any existing TCP streams, in particular TCP Reset segments.

However, the Message Authentication Codes (MACs) used by MD5 to compute the signature are considered to be too weak. TCP-AO [RFC5926] specifies a mechamism to protect BGP sessions and its data integrity using cryptographic authentication. In order to accomplish this funtion, it defines two MAC algorithms. It also defines two Key Derivation Functions (KDFs) used to create the traffic keys used by the newly defined and any future specified MACs. Cryptographic research suggests that both these MAC algorithms defined are fairly secured and are not known to be broken in any ways.

In addition, there is no Key Management Protocol (KMP) used to manage the keys that are used for generating the Message Authentication Code (MAC). Most routers are configured with a static key that does not change over the life of the session.

2.1. LDP

Section 5 of LDP [RFC5036] states that LDP is subject to three different types of attacks. It talks about spoofing, protection of

privacy of label distribution and denial of service attacks.

2.1.1. Spoofing attacks

Spoofing attack occur both during the discover phase and during the session communication phase.

2.1.1.1. Discovery exchanges using UDP

Label Switching Routers (LSRs) indicate their willingness to establish and maintain LDP sessions by periodically sending Hello messages. Receipt of a Hello message serves to create a new "Hello adjacency", if one does not already exist, or to refresh an existing one.

Unlike all other LDP messages, the Hello messages are sent using UDP not TCP. This means that they cannot benefit from the security mechanisms available with TCP. LDP [RFC5036] does not provide any security mechanisms for use with Hello messages except to note that some configuration may help protect against bogus discovery events.

Spoofing a Hello packet for an existing adjacency can cause the adjacency to time out and that can result in termination of the associated session. This can occur when the spoofed Hello message specifies a small Hold Time, causing the receiver to expect Hello messages within this interval, while the true neighbor continues sending Hello messages at the lower, previously agreed to, frequency.

Spoofing a Hello packet can also cause the LDP session to be terminated directly. This can occur when the spoofed Hello specifies a different Transport Address from the previously agreed one between neighbors. Spoofed Hello messages are observed and reported as real problem in production networks.

2.1.1.2. Session communication using TCP

LDP like other TCP based routing protocols specifies use of the TCP MD5 Signature Option to provide for the authenticity and integrity of session messages. As stated above, some assert that MD5 authentication is now considered by some to be too weak for this application. A stronger hashing algorithm e.g SHA1, could be deployed to take care of the weakness.

2.1.2. Privacy Issues

LDP provides no mechanism for protecting the privacy of label distribution. The security requirements of label distribution are similar to other routing protocols that need to distribute routing

information.

2.1.3. Denial of Service Attacks

LDP is subject to Denial of Service (DoS) attacks both in its discovery mode as well as during the session mode.

The discovery mode attack is similar to the spoofing attack except that when the spoofed Hello messages are sent with a high enough frequency, they can cause the adjacency to time out.

2.2. MSDP

Similar to BGP and LDP, TCP MD5 [RFC2385] specifies a mechanism to protect TCP sessions via the TCP MD5 option. But with a weak MD5 authentication, TCP MD5 is considered too weak for this application.

MSDP also advocates imposing a limit on number of source address and group addresses (S,G) that can be stored within the protocol and thereby mitigate state explosion due to any denial of service and other attacks.

3. Optimal State for BGP, LDP and MSDP

The ideal state for BGP, LDP and MSDP protocols are when they can withstand any of the known types of attacks.

Additionally, Key Management Protocol (KMP) for the routing sessions should help negotiate unique, pair wise random keys without administrator involvement. It should also negotiate Security Association (SA) parameter required for the session connection, including key life times. It should keep track of those lifetimes and negotiate new keys and parameters before they expire and do so without administrator involvement. In the event of a breach, the keys should be changed immediately.

The DoS attacks for BGP, LDP and MSDP are attacks to the transport protocol, TCP in this case. TCP should be able to withstand any of DoS scenarios by dropping packets that are attack packets in a way that does not impact legitimate packets.

The routing protocols should provide a mechanism to authenticate and validate the routing information carried within the payload.

3.1. LDP

For the spoofing kind of attacks that LDP is vulnerable to during the discovery phase, it should be able to determine the authenticity of the neighbors sending the Hello message.

There is currently no requirement to protect the privacy of label distribution as labels are carried in the clear like other routing information.

4. Gap Analysis for BGP, LDP and MSDP

This section outlines the differences between the current state of the routing protocol and the desired state as outlined in section 4.2 of [draft-ietf-karp-design-guide]. It covers issues that are common to the three protocols leaving protocol specific issues to sub-sections.

The session layer that runs on TCP needs to protect itself by running TCP LISTEN only on interfaces on which its peers have been discovered or that are configured to expect sessions on. Also the use of access list can help protect the edge routers from attacks originating from outside the protected cloud.

Inspite of this BGP, LDP and MDSP sessions are subject to spoofing and man in the middle attacks. While the MD5 option helps somewhat, without a KMP and a stronger MAC, these sessions are still vulnerable to attacks.

TCP-AO [RFC5926]is a step towards correcting both the MAC weakness and KMP. For MAC it specifies two MAC algorithms that MUST be supported. Additional MACs can be added in the future. They are HMAC-SHA-1-96 as specified in HMAC [RFC2104] and AES-128-CMAC-96 as specified in [NIST-SP800-38B]. For KMP it requires that a Key Derivation Function (KDF) MUST be supported. They are KDF_HMAC-SHA1 and KDF_AES_128_CMAC. But this does not address the question of connectionless reset.

[Need to add details about key rollover for manual keys and strategy for automatic keys here]

There is a need to protect authenticity and validity of the routing/label information that is carried in the payload of the sessions. However, we believe that is outside the scope of this document at this time and is being addressed by SIDR WG. Similar mechanisms could be used for intra-domain protocols.

4.1. LDP

As described in LDP [RFC5036], the threat of spoofed Basic Hellos can be reduced by accepting Basic Hellos on interfaces that LSRs trust, and ignoring Basic Hellos not addressed to the "all routers on this subnet" multicast group. Spoofing attacks via Extended Hellos are potentially a more serious threat. An LSR can reduce the threat of spoofed Extended Hellos by filtering them and accepting Hellos from sources permitted by an access list. However, performing the filtering using access lists requires LSR resource, and the LSR is still vulnerable to the IP source address spoofing. Spoofing attacks

can be solved by being able to authenticate the Hello messages, and an LSR can be configured to only accept Hello messages from specific peers when authentication is in use.

5. Security Requirements

This section describes requirements for BGP, LDP and MSFP security that should be met within the routing protocol.

As with all routing protocols, they need protection from both on-path and off-path blind attacks. A better way to protect them would be with per-packet protection using a cryptographic MAC.

Mechanisms are required in order to support key rollover. This should cover both manual and automatic key rollover. Multiple approaches could be used. However since the existing mechanisms provide a protocol field to identify the key as well as management mechanisms to introduce and retire new keys, focusing on the existing mechanism as a starting point is prudent.

Replay protection is required. The replay mechanism needs to be sufficient to prevent an attacker from creating a denial of service or disrupting the integrity of the routing protocol by replaying packets. It is important that an attacker not be able to disrupt service by capturing packets and waiting for replay state to be lost.

6. Acknowledgements

7. References

7.1. Normative References

[RFC2385] Heffernan, A., "Protection of BGP Sessions via the TCP MD5 Signature Option", RFC 2385, August 1998.

[RFC5926] Lebovitz, G. and E. Rescorla, "Cryptographic Algorithms for the TCP Authentication Option (TCP-AO)", RFC 5926, June 2010.

[draft-ietf-karp-design-guide] Lebovitz, G., "KARP Desgin Guidelines", September 2010.

7.2. Informative References

[NIST-SP800-38B] Dworking, "Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication", May 2005.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[RFC3682] Gill, V., Heasley, J., and D. Meyer, "The Generalized TTL Security Mechanism (GTSM)", RFC 3682, February 2004.

[RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.

[RFC5036] Andersson, L., Minei, I., and B. Thomas, "LDP Specification", RFC 5036, October 2007.

[RFC5961] Ramaiah, A., Stewart, R., and M. Dalal, "Improving TCP's Robustness to Blind In-Window Attacks", RFC 5961, August 2010.

[draft-ietf-opsec-routing-protocols-crypto-issues] Manral, "Issues with existing Cryptographic Protection Methods for Routing Protocols", September 2010.

Authors' Addresses

Mahesh Jethanandani
Cisco Systems, Inc
170 Tasman Drive
San Jose, CA 95134
USA

Phone: +1 (408) 527-8230
Email: mahesh@cisco.com

Keyur Patel
Cisco Systems, Inc
170 Tasman Drive
San Jose, CA 95134
USA

Phone: +1 (408) 526-7183
Email: keyupate@cisco.com

Lianshu Zheng
Huawei
No. 3 Xinxi Road
Beijing, 100085
China

Phone: +86 (10) 82882008
Fax:
Email: verozheng@huawei.com
URI:

Network working group
Internet Draft
Intended status: Standards Track
Updates: RFC 5036 (if approved)
Expires: September 2011

L. Zheng
M. Chen
Huawei Technologies

March 14, 2011

LDP Hello Cryptographic Authentication

draft-zheng-mpls-ldp-hello-crypto-auth-01.txt

Status of this Memo

Copyright Notice

Abstract

This document introduces a new Cryptographic Authentication TLV which is used in LDP Hello message as an optional parameter. It enhances the authentication mechanism for LDP by securing the Hello message against spoofing attack.

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

The Label Distribution Protocol (LDP) [RFC 5036] utilizes LDP sessions that run between LDP peers. The peers may be directly connected at the link level or may be remote. A label switching router (LSR) that speaks LDP may be configured with the identity of its peers or may discover them using the LDP Hello message sent encapsulated in UDP that may be addressed to "all routers on this subnet" or to a specific IP address. Periodic Hello messages are

also used to maintain the relationship between LDP peers necessary to keep the LDP session active.

Unlike all other LDP messages, the Hello messages are sent using UDP not TCP. This means that they cannot benefit from the security mechanisms available with TCP. [RFC5036] does not provide any security mechanisms for use with Hello messages except to note that some configuration may help protect against bogus discovery events.

Spoofing a Hello packet for an existing adjacency can cause the valid adjacency to time out and in turn can result in termination of the associated session. This can occur when the spoofed Hello specifies a smaller Hold Time, causing the receiver to expect Hellos within this smaller interval, while the true neighbor continues sending Hellos at the previously agreed lower frequency. Spoofing a Hello packet can also cause the LDP session to be terminated directly, which can occur when the spoofed Hello specifies a different Transport Address, other than the previously agreed one between neighbors. Spoofed Hello messages is observed and reported as real problem in production networks.

As described in [RFC5036], the threat of spoofed Basic Hellos can be reduced by accepting Basic Hellos only on interfaces to which LSRs that can be trusted, and ignoring Basic Hellos not addressed to the "all routers on this subnet" multicast group. Spoofing attacks via Extended Hellos are potentially more serious threat. An LSR can reduce the threat of spoofed Extended Hellos by filtering them and accepting only those originating at sources permitted by an access list. However, performing the filtering using access lists requires LSR resource, and the LSR is still vulnerable to the IP source address spoofing.

This document introduces a new Cryptographic Authentication TLV which is used in LDP Hello message as an optional parameter. It enhances the authentication mechanism for LDP by securing the Hello message against spoofing attack, and an LSR can be configured to only accept Hello messages from specific peers when authentication is in use.

Using this Cryptographic Authentication TLV, one or more secret keys (with corresponding key IDs) are configured in each system. For each LDP Hello packet, the key is used to generate and verify a HMAC Hash that is stored in the LDP Hello packet. For cryptographic hash function, this document proposes to use SHA-1, SHA-256, SHA-384, and SHA-512 defined in US NIST Secure Hash Standard (SHS) [FIPS-180-3]. The HMAC authentication mode defined in NIST FIPS 198 is used [FIPS-

198]. Of the above, implementations MUST include support for at least HMAC-SHA-256 and SHOULD include support for HMAC-SHA-1 and MAY include support for either of HMAC-SHA-384 or HMAC-SHA-512.

2. Cryptographic Authentication TLV

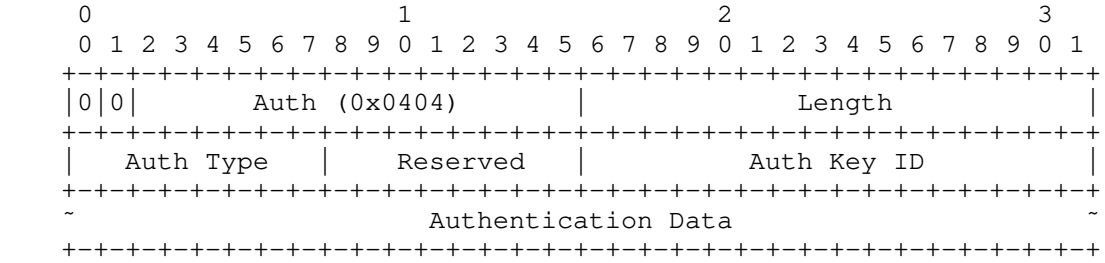
2.1. Optional Parameter for Hello Message

[RFC5036] defines the encoding for the Hello message. Each Hello message contains zero or more Optional Parameters, each encoded as a TLV. Three Optional Parameters are defined by [RFC5036]. This document defines a new Optional Parameter: the Cryptographic Authentication parameter.

Optional Parameter	Type
IPv4 Transport Address	0x0401 (RFC5036)
Configuration Sequence Number	0x0402 (RFC5036)
IPv6 Transport Address	0x0403 (RFC5036)
Cryptographic Authentication	0x0404 (this document, TBD by IANA)

The Cryptographic Authentication TLV Encoding is described in section 2.2.

2.2. Cryptographic Authentication TLV Encoding



- Type: 0x0404 (TBD by IANA), Cryptographic Authentication

- Length: Specifying the length in octets of the value field.

- Auth Type: The authentication type in use

0 - HMAC-SHA-1
1 - HMAC-SHA-256
2 - HMAC-SHA-384
3 - HMAC-SHA-512
4-255 - Reserved for future use
(TBD by IANA)

- Reserved: MUST be set to zero on transmit, and ignored on receipt

- Auth Key ID: The authentication key ID in use for this packet. This allows one or more keys to be active simultaneously.

- Authentication Data:

  This field carries the digest computed by the Cryptographic Authentication algorithm in use. The length of the Authentication Data varies based on the cryptographic algorithm in used, which is shown as below:

Auth type	Length
HMAC-SHA1	20 bytes
HMAC-SHA-256	32 bytes
HMAC-SHA-384	48 bytes
HMAC-SHA-512	64 bytes

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 specified by Auth Type (e.g. SHA-256).

- K is the Authentication Key for the Hello packet.

- Ko is the cryptographic key used with the hash algorithm.

- B is the block size of H, in octets.

  For SHA-1 and SHA-256: B == 64

  For SHA-384 and SHA-512: B == 128

- L is the length of the hash outputs, in octets.

- XOR is the exclusive-or operation.

- Ipad is the byte 0x36 repeated B times.

- Opad is the byte 0x5c repeated B times.

- Apad is the byte 0x878FE1F3 repeated (L/4) times.

3.1. Cryptographic Key

As described in RFC 2104, the authentication key K can be of any length up to B. Applications that use keys longer than B bytes will first hash the key using H and then use the resultant L byte string as the actual key to HMAC.

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 trailing zeros such that Ko is L octets long.

3.2. Hash

First, the Authentication Data field in the Cryptographic Authentication TLV is filled with the value Apad and the Auth Type field is set accordingly per Cryptographic Authentication algorithm in use.

Then, to compute HMAC over the Hello packet it performs:

```
H(Ko XOR Opad || H(Ko XOR Ipad || (Hello Packet)))
```

Hello Packet here is the entire LDP Hello packet including the IP header.

3.3. Result

The resultant Hash becomes the Authentication Data that is sent in the Authentication Data field of the Cryptographic Authentication TLV. The length of the Authentication Data field is always identical to the message digest size of the specific hash function H that is being used.

4. Processing Hello Message Using Cryptographic Authentication

4.1. Transmission Using Cryptographic Authentication

Prior to transmitting Hello message, the Auth Type field is set to indicate the authentication type in use. The Length in the Cryptographic Authentication TLV header is set as per the authentication algorithm that is being used. It is set to 24 for HMAC-SHA-1, 36 for HMAC-SHA-256, 52 for HMAC-SHA-384 and 68 for HMAC-SHA-512.

The Auth Key ID field is set to the ID of the current authentication key. The HMAC Hash is computed as explained in Section 3. The resulting Hash is stored in the Authentication Data field prior to transmission. The authentication key MUST NOT be carried in the packet.

4.2. Receipt Using Cryptographic Authentication

The receiving LSR applies acceptability criteria for received Hellos using cryptographic authentication. If the Cryptographic Authentication TLV is unknown to the receiving LSR, the received packet MUST be discarded according to Section 3.5.1.2.2 of [RFC5036].

If the Cryptographic Authentication TLV in a received Hello packet does not contain a known and acceptable Auth Type value, then the received packet MUST be discarded. If the Auth Key ID field does not match the ID of a configured authentication key, the received packet MUST be discarded.

Before the receiving LSR performs any processing, it needs to save the values of the Authentication Data field. The receiving LSR then replaces the contents of the Authentication Data field with Apad, computes the Hash, using the authentication key specified by the

received Auth Key ID field, as explained in Section 3. If the locally computed Hash is equal to the received value of the Authentication Data field, the received packet is accepted for other normal checks and processing as described in [RFC5036]. Otherwise, the received packet MUST be discarded.

5. Security Considerations

Section 1 of this document describes the security issues arising from the use of unsecured LDP Hello messages. In order to combat those issues, it is RECOMMENDED that all deployments use the Cryptographic Authentication TLV to secure the Hello message.

The quality of the security provided by the Cryptographic Authentication TLV depends completely on the strength of the cryptographic algorithm in use, the strength of the key being used, and the correct implementation of the security mechanism in communicating LDP implementations. Also, the level of security provided by the Cryptographic Authentication TLV varies based on the authentication type used.

6. IANA Considerations

IANA maintains a registry of LDP message parameters with a sub-registry to track LDP TLV Types. This document request IANA to assign a new TLV Types as follows:

TLV	Type
Cryptographic Authentication	0x0404 (TBD)

This document also request IANA to assign a new registry titled "LDP Hello Authentication Type", its recommended values as follows:

Value	LDP Hello Authentication Type Name
0	HMAC-SHA1
1	HMAC-SHA-256
2	HMAC-SHA-384
3	HMAC-SHA-512
4-255 (TBD)	Unassigned

7. Acknowledgments

The authors would like to thank Liu Xuehu for his work on background and motivation for LDP Hello authentication. The authors also would like to thank Adrian Farrel, Thomas Nadeau, So Ning, Eric Rosen, Sam Hartman and Manav Bhatia for their valuable comments.

8. References

8.1. Normative References

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

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

[RFC5036] Andersson, L., Minei, I., and B. Thomas, "LDP Specification", RFC 5036, October 2007.

[FIPS-180-3] National Institute of Standards and 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

[RFC2385] Heffernan, A., "Protection of BGP Sessions via the TCP MD5 Signature Option", RFC 2385, August 1998.

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

[RFC5709] Bhatia, M., Manral, V., Fanto, M., White, R., Barnes, M., Li, T., and R. Atkinson, "OSPFv2 HMAC-SHA Cryptographic Authentication", RFC 5709, October 2009.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection", RFC 5880, June 2010.

Authors' Addresses

Lianshu Zheng
Huawei Technologies Co., Ltd.
Huawei Building, No.3 Xinxi Road,
Hai-Dian District,
Beijing 100085
China

Email: verozheng@huawei.com

Mach(Guoyi) Chen
Huawei Technologies Co., Ltd.
Huawei Building, No.3 Xinxi Road,
Hai-Dian District,
Beijing 100085
China

Email: mach@huawei.com