Internet Engineering Task Force (IETF)
Request for Comments: 6028
Category: Experimental
ISSN: 2070-1721
October 2010

G. Camarillo
A. Keranen
Ericsson

Host Identity Protocol (HIP) Multi-Hop Routing Extension

Abstract

This document specifies two extensions to the Host Identity Protocol (HIP) to implement multi-hop routing. The first extension allows implementing source routing in HIP. That is, a node sending a HIP packet can define a set of nodes that the HIP packet should traverse. The second extension allows a HIP packet to carry and record the list of nodes that forwarded it.

Status of This Memo

Table of Contents

1. Introduction

When the Host Identity Protocol (HIP) [RFC5201] is used in certain contexts, nodes need the ability to perform source routing. That is, a node needs the ability to send a HIP signaling packet that will traverse a set of nodes before reaching its destination. Such features are needed, e.g., in the HIP-Based Overlay Networking Environment (HIP BONE) [HIP-BONE] or if two nodes wish to keep a third, or more, HIP nodes on the signaling path. This document defines an extension that provides HIP with this functionality.

Additionally, when HIP signaling packets are routed through multiple nodes, some of these nodes (e.g., the destination host) need the ability to know the nodes that a particular packet traversed. This document defines another extension that provides HIP with this functionality.

These two extensions enable multi-hop routing in HIP. Before these extensions were specified, there were standardized ways for supporting only a single intermediate node (e.g., a rendezvous server [RFC5204]) between the source of a HIP packet and its destination.

2. Terminology

2.1. 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].

2.2. Definitions

The following terms used in this document are similar to those defined by REsource LOcation And Discovery (RELOAD) [P2PSIP-BASE] but are used here in the context of HIP.

Destination list: A list of Host Identity Tags (HITs) of the nodes that a HIP packet should traverse.

Via list: A list of HITs of the nodes that a HIP packet has traversed.

Symmetric routing: A response to a message is routed back using the same set of intermediary nodes as the original message used, except in reversed order. Also known as symmetric recursive routing.

3. Protocol Definitions

The multi-hop routing extensions may be used in different contexts, and whether a new HIP signaling packet should, for example, include a Via list or have different options enabled can depend on the particular use case, local policies, and different protocols using the extension. This section defines how the new parameters are handled, but when to use these extensions, or how to configure them, is out of scope for this document.

3.1. Creating and Processing Via Lists

When a node sending a HIP packet needs to record the nodes that are on the path that the HIP packet traverses, it includes an empty ROUTE_VIA parameter in the packet.

A node that receives a packet with a ROUTE_VIA parameter SHOULD add its own HIT to the end of the ROUTE_VIA parameter, unless it is the final recipient of the packet. If the node uses a different HIT on the HIP association it used for receiving the packet than for sending it forward, it SHOULD also add the receiving HIT to the route list before the sending HIT.

If the node is the final recipient of the packet, and the received packet generates a response HIP packet, the node checks the SYMMETRIC flag from the ROUTE_VIA parameter. If the SYMMETRIC flag is set, the node MUST create a ROUTE_DST parameter from the ROUTE_VIA parameter, as described in Section 3.2, and include it in the response packet. Also, if an intermediary node generates a new HIP packet (e.g., an error NOTIFY packet) due to a HIP packet that had a ROUTE_VIA parameter with the SYMMETRIC flag set, and the new packet is intended for the sender of the original HIP packet, the node SHOULD construct and add a ROUTE_DST parameter into the new packet as in the previous case.

3.2. Creating Destination Lists

A node that needs to define the other nodes that should be on the path a HIP packet traverses adds a ROUTE_DST parameter to the HIP packet. The node may either decide the path independently, or it may create the path based on a ROUTE_VIA parameter. Only the originator of a signed HIP packet can add a ROUTE_DST parameter to the HIP packet, and none of the nodes on the path can modify it, since the parameter is covered by the signature.

When a node creates a ROUTE_DST parameter due to receiving a packet with a ROUTE_VIA parameter, it copies all the HITs in the ROUTE_VIA parameter to the ROUTE_DST parameter, but in reversed order. This results in the HIP response packet being forwarded using the same path as the packet for which the response was generated. If exactly the same set of nodes should be traversed by the response packet, the MUST_FOLLOW flag (see Table 1) also SHOULD be set in the ROUTE_VIA parameter (and eventually copied to the ROUTE_DST parameter) to prevent the response packet from possibly skipping some nodes on the list.

3.3. Processing Destination Lists

When a node receives a HIP packet that contains a ROUTE_DST parameter, it first looks up its own HIT from the route list. If the node's own HIT is not in the list and the node is not the receiver of the packet, the packet was incorrectly forwarded and MUST be dropped. If the node's HIT is in the list more than once, the list is invalid and the packet MUST be dropped to avoid forwarding loops. The next hop for the packet is the HIT after the node's own HIT in the list. If the node's HIT was the last HIT in the list, the next hop is the receiver's HIT in the HIP header.

If the MUST_FOLLOW flag in the ROUTE_DST parameter is not set, the node SHOULD check whether it has a valid locator for one of the nodes later in the list, or for the receiver of the packet, and it MAY select such a node as the next hop. If the MUST_FOLLOW flag is set, the node MUST NOT skip any nodes in the list.

If the node has a valid locator for the next hop, it MUST forward the HIP packet to the next-hop node. If the node cannot determine a valid locator for the next-hop node, it SHOULD drop the packet and SHOULD send back a NOTIFY error packet with type UNKNOWN_NEXT_HOP (value 90). The Notification Data field for the error notifications SHOULD contain the HIP header of the rejected packet and the ROUTE_DST parameter.

3.4. Fragmentation Considerations

Via and Destination lists with multiple HITs can substantially increase the size of the HIP packets, and thus fragmentation issues (see Section 5.1.3 of [RFC5201]) should be taken into consideration when these extensions are used. Via lists in particular should be used with care, since the final size of the packet is not known unless the maximum possible amount of hops is known beforehand. Both parameters do still have a maximum size based on the maximum number of allowed HITs (see Section 4.1).

4. Packet Formats

This memo defines two new HIP parameters that are used for recording a route via multiple nodes (ROUTE_VIA) and for defining a route that a packet should traverse by the sender of the packet (ROUTE_DST).

The ROUTE_DST parameter is integrity protected with the signature (where present) but ROUTE_VIA is not, so that intermediary nodes can add their own HITs to the list. Both ROUTE_DST and ROUTE_VIA are critical parameters (as defined in Section 5.2.1 of [RFC5201]), since the packet will not be properly routed unless all nodes on the path recognize the parameters.

4.1. Source and Destination Route List Parameters

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |             Type              |             Length            |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |             Flags             |            Reserved           |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                                                               |
     |                            HIT #1                             |
     |                                                               |
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     .                               .                               .
     .                               .                               .
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                                                               |
     |                            HIT #n                             |
     |                                                               |
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

     Type      ROUTE_DST: 4601
               ROUTE_VIA: 64017
     Length    length in octets, excluding Type and Length
               (i.e., number-of-HITs * 16 + 4)
     Flags     bit flags that can be used for requesting special
               handling of the parameter
     Reserved  reserved for future use
     HIT       Host Identity Tag of one of the nodes on the path
```

Figure 1. Format of the ROUTE_VIA and ROUTE_DST Parameters

Figure 1 shows the format of both ROUTE_VIA and ROUTE_DST parameters. The ROUTE_DST parameter, if present, MUST have at least one HIT, but the ROUTE_VIA parameter can also have zero HITs. The ROUTE_DST and ROUTE_VIA parameters SHALL NOT contain more than 32 HITs. The Flags field is used for requesting special handling for Via and Destination lists. The flags defined in this document are shown in Table 1. The Reserved field can be used by future extensions; it MUST be zero when sending and ignored when receiving this parameter.

Pos	Name	Purpose
0	SYMMETRIC	The response packet MUST be sent with a ROUTE_DST list made from the ROUTE_VIA list containing this flag, i.e., using symmetric routing.
1	MUST_FOLLOW	All the nodes in a ROUTE_DST list MUST be traversed, i.e., even if a node would have a valid locator for a node beyond the next hop, it MUST NOT forward the packet there but to the next-hop node.

Table 1. Bit Flags in ROUTE_VIA and ROUTE_DST Parameters

The "Pos" column in Table 1 shows the bit position of the flag (as in Figure 1) in the Flags field, "Name" gives the name of the flag used in this document, and "Purpose" gives a brief description of the meaning of that flag.

The flags apply to both ROUTE_VIA and ROUTE_DST parameters, and when a ROUTE_DST parameter is added to a packet because of a ROUTE_VIA parameter, the same flags MUST be copied to the ROUTE_DST parameter.

5. IANA Considerations

This section is to be interpreted according to [RFC5226].

This document updates the IANA Registry for HIP Parameter Types [RFC5201] by assigning new HIP Parameter Type values for the new HIP Parameters: ROUTE_VIA and ROUTE_DST (defined in Section 4). This document also defines a new Notify Packet Type [RFC5201], UNKNOWN_NEXT_HOP, in Section 3.3.

The ROUTE_DST and ROUTE_VIA parameters utilize bit flags, for which IANA has created and now maintains a new sub-registry entitled "HIP Via Flags" under the "Host Identity Protocol (HIP) Parameters" registry. Initial values for the registry are given in Table 1; future assignments are to be made through IETF Review or IESG Approval [RFC5226]. Assignments consist of the bit position and the name of the flag.

6. Security Considerations

The standard HIP mechanisms (e.g., using signatures, puzzles, and the ENCRYPTED parameter [RFC5201]) provide protection against eavesdropping; replay; message insertion, deletion, and modification; and man-in-the-middle attacks. Yet, the extensions described in this document allow nodes to route HIP messages via other nodes and hence possibly try to mount Denial-of-Service (DoS) attacks against them. The following sections describe possible attacks and means to mitigate them.

6.1. Forged Destination and Via Lists

The Destination list is protected by the HIP signature so that the receiver of the message can check that the list was indeed created by the sender of the message and not modified on the path. Also, the nodes forwarding the message MAY check the signature of the forwarded packets if they have the Host Identity (HI) of the sender (e.g., from an I2 or R1 message [RFC5201]) and drop packets whose signature check fails. With forwarding nodes checking the signature and allowing messages to be forwarded only from nodes for which there is an active HIP association, it is also possible to reliably identify attacking nodes.

The limited amount of HITs allowed in a Destination list limits the impact of attacks using a forged Destination list, and the attacker also needs to know a set of HIP nodes that are able to route the message hop-by-hop for the attack to be effective.

A forged Via list results in a similar attack as with the Destination list and with similar limitations. However, in this attack the Destination list generated from the Via list is validly signed by the responding node. To limit the effect of this kind of attack, a responding node may further decrease the maximum acceptable number of nodes in the Via lists or allow only certain HITs in the lists. However, using these mechanisms requires either good knowledge of the overlay network (i.e., maximum realistic amount of hops) or knowing the HITs of all potential nodes forwarding the messages.

6.2. Forwarding Loops

A malicious node could craft a destination route list that contains the same HIT more than once and thus create a forwarding loop. The check described in Section 3.3 should break such loops, but nodes MAY in addition utilize the OVERLAY_TTL [HIP-BONE] parameter for additional protection against forwarding loops.

7. Acknowledgments

Tom Henderson provided valuable comments and improvement suggestions for this document.

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.

[RFC5201] Moskowitz, R., Nikander, P., Jokela, P., Ed., and T. Henderson, "Host Identity Protocol", RFC 5201, April 2008.

8.2. Informative References

[RFC5204] Laganier, J. and L. Eggert, "Host Identity Protocol (HIP) Rendezvous Extension", RFC 5204, April 2008.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[HIP-BONE] Camarillo, G., Nikander, P., Hautakorpi, J., Keranen, A., and A. Johnston, "HIP BONE: Host Identity Protocol (HIP) Based Overlay Networking Environment", Work in Progress, June 2010.

[P2PSIP-BASE] Jennings, C., Lowekamp, B., Ed., Rescorla, E., Baset, S., and H. Schulzrinne, "REsource LOcation And Discovery (RELOAD) Base Protocol", Work in Progress, March 2010.

Authors' Addresses

Gonzalo Camarillo
Ericsson
Hirsalantie 11
02420 Jorvas
Finland

EMail: Gonzalo.Camarillo@ericsson.com

Ari Keranen
Ericsson
Hirsalantie 11
02420 Jorvas
Finland

EMail: Ari.Keranen@ericsson.com