HIPRG A. de la Oliva
Internet-Draft UC3M
Intended status: Experimental M. Bagnulo
Expires: January 3, 2008 Huawei Labs at UC3M
July 2, 2007

Fault tolerance configurations for HIP multihoming
draft-oliva-hiprg-reap4hip-00

Status of this Memo

This Internet-Draft will expire on January 3, 2008.

Copyright Notice

Abstract

This document considers scenarios for the provision of fault tolerance capabilities in multihomed HIP nodes. In order to support such configurations, this document updates the behaviour for HIP multihoming nodes currently defined and defines the integration of the REAP protocol in HIP.

Table of Contents

1. Introducion

Multihoming support for HIP is defined in draft-ietf-hip-mm [2]. It relies on the usage of UPDATE messages to convey information about the alternative locators available for the HIP nodes. The aforementioned specification defines the basic support for multihoming and covers some basic scenarios but it postpones the analysis of more advanced multihoming scenarios for future study. This document considers additional multihoming scenarios, especially focussing on the provision of fault tolerance capabilities in multihomed HIP nodes. In order to support such fault tolerant configurations, this document updates the behaviour for HIP multihoming nodes defined in [2] and defines the integration of the REAP protocol as defined in draft-ietf-shim6-failure-detection [4] in HIP.

2. Locator management

A multihomed HIP node has multiple locators that can have different reachability status. Some of them can be operational/reachable while other may be not. Fault tolerance is a preferred capability of such configuration. In order to provide basic fault tolerance support, a HIP node should be able to perform the following functions: First, the multihomed HIP nodes must be able to convey the locally available locator set to the peer. Second, the nodes should be able to monitor the communication and detect failures. In case that a failure is detected, they must be able to discover alternative working locator pairs and divert the communication through the alternative locator pair. In this section, we focus in the locator management part, and in the next section we will focus on the failure detection and alternative path exploration part. It should be noted that for the provision of basic fault tolerance capabilities, the locators are managed following the following guidelines:

o All the locators available for each peers that are to be used to provide fault tolerance must be exchanged early in the communication, so they can be used as alternative locators in case of a failure. This is different from the mobility case described in [2] since the peers only know a single locator of the peer at the time.

o However, the locators are only used sequentially and not in parallel. This is so, because as long a locator pair is working, the peers stick to that pair for exchanging data packets and they only change the locator pair used when there is a failure. This is different from the general multihoming scenario considered in [2] since locator pairs are not used in parallel. This particular constraint reduces considerably the possibility of packet reordering and hence the possibility of having problems with the

reply protection window due to reordering of packets that travel through different paths.

In the general multihoming scenario defined in [2], a multihomed node is recommended to create different SAs and use different SPIs for each locator pair available for the communication between two multihomed nodes to avoid problems with the anti-replay protection window resulting from reordering packets when using multiple paths simultaneously. While this is required for the general multihoming scenario, this is an expensive approach, because it requires a high number of SAs to be created and it also requires a significant signaling overhead. Basically in a multihoming scenario where a multihomed node A that has m locators is communicating with another multihomed node B that has n locators, they need to exchange m+n UPDATE messages to convey all the locator information. This is so, because they need to convey SPI information for each of the locator pairs. Node A does so by sending an n UPDATE messages. Each one of these n UPDATE messages contains the m locally available locators with an SPI value for each locator. Each of the n UPDATE messages is addressed to a different locator of the n available at node B. In this way, the information of the m*n SPIs values for the different locator pairs is exchanged between the peers. While all the overhead and complexity is required when using multiple locator pairs in parallel, this is not the case for a fault tolerant configuration, where the locator pairs will be used sequentially. Hence, in the document we define a modified behaviour for HIP multihomed nodes for fault tolerance support.

For fault tolerance, only sequential use of locator pairs is required. This is similar to the mobility case, the difference being that the locator set for each peer is known beforehand. In order to support this configuration, the following behaviour is defined for HIP nodes. Each node will convey the available locator set information to the peer in a single UPDATE message. The Old SPI and the New SPI values of the LOCATOR parameter will be equal and set to the current SPI value for every locator in the message. Each node will use a single SA and a single SPI value for all the locator pairs available for the configuration. Only a single locator pair will be active, and all the traffic will be sent using the preferred (active) locator pair. Upon the reception of one UPDATE message containing multiple locators with a single SPI value for both the OlD SPI and the New SPI for all the locators, the receiver will verify the locators contained in the UPDATE message as defined in [2]. After that, the receiver will identify that it is in the fault tolerance scenario and will create locator pairs using all the received locators and all the locally available locators, irrespectively of the locator to which the UPDATE message was sent. The result is that each of the peers will have all the locator pairs available for use

in case that a failure occurs.

After the locator sets have been exchanged, the peers use the REAP protocol as defined in the next section to detect failure, explore alternative locator pairs and divert the communication through alternative working locators.

3. Failure detection and alternative path exploration for HIP

Multihoming support for the HIP protocol as defined in [2] relies in the usage of the UPDATE message to convey multiple alternative locators for a given HI/HIT that is being used as identifier for ongoing communications between two nodes. By including alternative locators associated to the multihoming configuration, the communication between the two nodes is more reliable, since it is possible to use alternative locator pairs in case the original one should fail. As currently defined, the HIP protocol is capable of exchanging the alternative locators, validate them and use them to exchange packets. However, the capabilities required for detecting failures and exploring alternative working locators are still lacking. The REAP protocol [4] provides such capabilities for the Shim6 protocol [3] and can be used to provide the lacking failure detection and path exploration capabilities in HIP. This section defines how the REAP protocol [4] can be used to detect failures and explore alternative paths between two hosts using HIP multihoming.

3.1. Multihoming scenario

The considered scenario consists of two HIP hosts that are communicating. At least one of them has multiple locators which may have different reachability status. In order to benefit from the enhanced fault tolerance capabilities resulting from multihoming, the decide to exchange the alternative locators available at each end.

The exchange of the locator set of each end host can be performed in two ways:

- Each end point of the communication may send a LOCATOR parameter on R1 and I2 messages of the HIP connection establishment as defined on [2].
- The HIP protocol specified on [1] supports the modification of the locator set currently being used by the exchange of the UPDATE message.

Herein we describe the use of the UPDATE packet to exchange the locator set but a similar scenario would result if the locator sets were exchanged using the R2 and I2 messages.

The exchange of locators in the UPDATE packet is secured by the use of HMAC and HIP_SIGNATURE on the message. A number of combinations of parameters in an UPDATE packet are possible (see [2]). In this scenario we consider the case where one LOCATOR and one ESP_INFO parameter is used in any HIP packet. Other configurations may be possible although are out of the scope of this document. As specified on [2] the LOCATOR parameter should list all the locators that are active on the HIP association, so the UPDATE packet sent to the peer will inform of all the locators which can be used to reach the host on this HIP association.

The locators stored on the LOCATOR parameter may be:

o IPv6 or an IPv4-in-IPv6 format IPv4 adress (for non ESP based usage).

o The concatenation of an ESP SPI (first 32 bits) followed by an IPv6 address or IPv4-in-IPv6 format IPv4 address (128 bits) for ESP use.

On the LOCATOR parameter, there is a bit (P) which express the preferred locator. This bit must be set to one on the active locator for this communication and 0 in all the others to prevent a change of the active locators.

The UPDATE packet may contain a ESP_INFO parameter, rules for processing this parameter are given on [2] and are assumed to be valid on this document. Once an UPDATE message is received, the locators listed on the LOCATOR parameter are processed following the guidelines of [2]. After the processing, the SA will have a list with all the available addresses of the peer. The address in use will be on ACTIVE state and the rest will be on UNVERIFIED state. Note that in [2] is stated that after receiving an UPDATE message with a LOCATOR parameter included, the only valid locator pairs created are between the new locators added and the source address of the UPDATE message. We extend this behaviour on Section 2 to allow the creation of pairs between all the locators of both endpoints.

Note that with this approach a communication between two endhosts (A,B), having A n possible locators and B m, leads to an SA with n*m valid locator pairs. Once finished the locator exchange, we assume each endhost SA will have n*m valid locator pairs.

At this stage, the hosts are ready to benefit from the enhanced fault tolerance capabilities resulting from multihoming, and use the REAP protocol to detect failures in the current locator pair and to explore alternative working locators pairs in case the current one should fail. We describe how this is done in the following section.

3.2. Failure detection and recovery

The REAP protocol [4] defined in the Shim6 architecture [3] provides path failure detection and alternative path exploration capabilities between two multihomed hosts. It relies in two mechanisms, namely, the failure detection mechanism and the path exploration mechanism.

The failure detection mechanism is based on the Forced Bidirectional Detection (FBD) technique, which consists on making sure that whenever there is data traffic in one direction, there is also traffic in the other direction. This is accomplished by injecting additional control messages (called KeepAlives messages) when needed, which guarantee that the frequency of traffic in the reverse direction is above a predetermined threshold. The result is that when there is a ongoing data communication between two REAP peers, both peers can expect an incoming traffic frequency that is above the predetermined threshold defined by REAP. If the incoming traffic frequency is below this threshold, then this implies that a failure has occurred. In other words, after a given period of time no traffic has been received a failure on the path is assumed and the alternative path exploration mechanism is triggered.

The current implementation the REAP protocol relies on two timers, the Keep Alive Timer and the Send Timer, and a control message, namely the Keepalive message. The Keep Alive Timer TKA is started each time a node receives a data packet from its peer, and stopped and reset, each time the node sends a packet to the peer. When the Keep Alive Timer expires, a Keep Alive message is sent to the peer. The Send Timer TSend, defined roughly as three times the Keep Alive Timer plus a deviation to accommodate the Round Trip Time, is started each time the node sends a packet and stopped each time the node receives a packet from the peer. If no answer (either a Keep Alive or data packet) is received in the Send Timer period a failure is assumed and a locator path exploration is started. Consequently, the Send Timer reflects the requirement that when a node sends a payload packet there should be some return traffic within Send Timeout seconds. On the other hand, the Keepalive timer reflects the requirement that when a node receives a payload packet there should a similar response towards the peer within Keepalive seconds (if no traffic is interchanged, there is no Keep Alive signaling).

The path exploration mechanism starts whenever the node has not received any packet during a fixed period of time (Send Timer). A path may become invalid either bacause one of the locators used may became invalid or inoperational, or the pair itself has been declared as inoperational. A full exploration mechanism should check all possible pairs of source/destination locators until at least one working locator pair is found. Instead of using a request/response

approach the first of both sides which detects the failure tries each of the peer's locators sending probes through each of its interfaces. Each probe carries information about the current state of the communication and the probes which have been received so far through the rest of the interfaces. The state of the connection can be one of three possible states: i) Operational, when both peers can see each other, b)Exploring, when one of the peers have detected a problem and has currently not seen any traffic from the peer or c) Inbound_OK, when the node sees traffic from the peer but the peer does not see any traffic from the node. The information related with the rest of the probes received, which is carried on every probe allows the end hosts to be able to know which are the locator pairs working in the outgoing direction, on the case there are multiple probes. The path exploration mechanism ends when both peers have received a probe confirming that the peer can see them. It should be noted that the defined exploration mechanism is capable of discovering locators pairs that are working in only one direction (i.e. unidirectional reachability) thanks to the information about all received probes contained in all the reply probes.

In the current implementation once a node detects a failure, it starts the path exploration mechanism. A Probe message is sent to test the current locator pair, and if no responses are obtained during a period of time called Retransmission Timer (TRTx), the nodes start sending Probes testing the rest of the available address pairs, using all possible source/destination address pairs. Once a probe is received by the node, it sends another probe to the peer indicating that it is seeing traffic from it (Inbound_OK). After receiving this last probe the peer will answer confirming the reception of this last probe and indicating that the new locator pair is in the Operational state.

These Probe messages are used to confirm reachability and can be used as an address verification mechanism to modify the state of the locator being probe to ACTIVE. Note that each end point of the communication explores unidirectional reachability and based on its observations decides the pair of locators to use in a not coordinated way. Therefore the pair of locators selected by each end host may be different.

At the end of the path exploration mechanism, each host will have a pair of ACTIVE locators which can be used to continue the communication.

3.3. Processing of the REAP messages

Figure 1 shows the placement of the REAP module within the HIP Multihoming stack. Once some heuristic has decided that a

communication should be protected by the REAP protocol, an instance of it is created and associated with the SA to protect. The REAP instance is able to communicate with the ESP and HIP layers. The ESP layer should inform of every packet received or sent associated with the SPI used on the protected HIP association. By this way if rekeying is needed due to a change of locator or any other cause, the ESP layer will still be able to inform REAP of the messages received or sent associated to the protected SA.

```
               ---------
               | TCP   |  (sockets bound to HITs)
               ---------
                   |
               ---------
         ----> | ESP   |  {HIT_s, HIT_d} <-> SPI
         |     ---------
        ----       |
       | MH |  ---------
        ---- ->| HIP   |  {HIT_s, HIT_d, SPI} <-> {IP_s, IP_d, SPI}
       |REAP|  ---------
        ----       |
               ---------
               |  IP   |
               ---------
```

Figure 1: Architecture for HIP mobility and multihoming (MH)

The REAP control messages (Probe, KeepAlive) are not protected by ESP and will be indexed by the Sender's and Receiver's HIT pair. Upon reception of a REAP control message, the HIP layer will demultiplex the control packet and forward it to the corresponding REAP instance. It must be taken into account that if several SAs are used to communicate the same HIT pairs only one instanciation of the REAP protocol is needed. On this case all data packets corresponding with the SAs between the HIT pair will be notified to the same instance of REAP.

4. HIP comformant REAP messages

4.1. KeepAlive Message

The format of the keepalive message is as follows:

```
        0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | Next Header   | Header Length |0| Packet Type |  VER. | RES.|1|
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |          Checksum             |           Controls            |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                Sender's Host Identity Tag (HIT)               |
      |                                                               |
      |                                                               |
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |               Receiver's Host Identity Tag (HIT)              |
      |                                                               |
      |                                                               |
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      /                        HIP Parameters                         /
      /                                                               /
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: KeepAlive Message

Fields:

Next Header, Header Length, 0, Version, Reserved, 1, Checksum and HIP Controls:

   These are as specified in Section 5.1 of the HIP protocol description [1].

Packet Type (as specified in [4]):

   This field identifies the KeepAlive message and MUST be set to 66 (KeepAlive)

Sender's Host Identity Tag (HIT):

   As defined in [1]

Receiver's Host Identity Tag (HIT):

   As defined in [1]

HIP parameters:

   This space is reserved for adding HIP parameters. At least the KeepAlive Timeout Option may be added here.

4.2. Probe Message

This message performs REAP exploration. Its format is as follows:

```
        0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | Next Header   | Header Length |0| Packet Type |  VER. | RES.|1|
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |          Checksum             |           Controls            |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                Sender's Host Identity Tag (HIT)               |
      |                                                               |
      |                                                               |
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |               Receiver's Host Identity Tag (HIT)              |
      |                                                               |
      |                                                               |
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      /                        HIP Parameters                         /
      /                                                               /
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | Precvd| Psent |Sta|                 Reserved2                 |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      +                      First probe sent                         +
      |                                                               |
      +                      Source address                           +
      |                                                               |
      +                                                               +
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      +                      First probe sent                         +
      |                                                               |
      +                      Destination address                      +
      |                                                               |
      +                                                               +
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      First probe nonce                        |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      First probe data                         |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
      /                                                               /
      /                      Nth probe sent                           /
      |                                                               |
      +                      Source address                           +
      |                                                               |
      +                                                               +
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      +                      Nth probe sent                           +
      |                                                               |
      +                      Destination address                      +
      |                                                               |
      +                                                               +
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      Nth probe nonce                          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      Nth probe data                           |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      +                      First probe received                     +
      |                                                               |
      +                      Source address                           +
      |                                                               |
      +                                                               +
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      +                      First probe received                     +
      |                                                               |
      +                      Destination address                      +
      |                                                               |
      +                                                               +
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      First probe nonce                        |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      First probe data                         |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      /                                                               /
      /                                                               /
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      +                      Nth probe received                       +
      |                                                               |
      +                      Source address                           +
```

```
      |                                                               |
      +                                                               +
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      +                      Nth probe received                       +
      |                                                               |
      +                      Destination address                      +
      |                                                               |
      +                                                               +
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      Nth probe nonce                          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      Nth probe data                           |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      +                          Options                              +
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      +                          Options                              +
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Probe Message

Fields:

Next Header, Header Length, 0, Version, Reserved, 1, Checksum and HIP Controls:
   These are as specified in Section 5.1 of the HIP protocol description [1].

Packet Type (as specified in [4]):
   This field identifies the Probe message and MUST be set to 67 (Probe)

Sender's Host Identity Tag (HIT):
   As defined in [1].

Receiver's Host Identity Tag (HIT):
   As defined in [1].

HIP parameters:
   This space is reserved for adding HIP parameters. At least the KeepAlive Timeout Option may be added here.

The rest of the parameters on the packet are exactly the same as specified on [4].

Psent

This is a 4-bit field that indicates the number of sent probes included in this probe message. The first set of probe fields pertains to the current message and MUST be present, so the minimum value for this field is 1. Additional sent probe fields are copies of the same fields sent in (recent) earlier probes and may be included or omitted as per any logic employed by the implementation.

Precvd

This is a 4-bit field that indicates the number of received probes included in this probe messsage. Received probe fields are copies of the same fields received in (recent) earlier probes and may be included or omitted as per any logic employed by the implementation.

The fields probe source, probe destination, probe nonce and probe data may be repeated, depending on the value of Psent and Preceived.

Sta (State)

This 2-bit State field is used to inform the peer about the state of the sender. It has three legal values:

0 (Operational) implies that the sender both (a) believes it has no problem communicating and (b) believes that the recipient also has no problem communicating.

1 (Exploring) implies that the sender has a problem communicating with the recipient, e.g., it has not seen any traffic from the recipient even when it expected some.

2 (InboundOk) implies that the sender believes it has no problem communicating, i.e., it at least sees packets from the recipient, but that the recipient either has a problem or has not yet confirmed to the sender that the problem has been solved.

Reserved2

MUST be set to 0 upon transmission and MUST be ignored upon reception.

Probe source

This 128-bit field contains the source IPv6 address used to send the probe.

Probe destination

This 128-bit field contains the destination IPv6 address used to send the probe.

Probe nonce

This is a 32-bit field that is initialized by the sender with a value that allows it to determine which sent probes a received probe correlates with. It is highly recommeded that the nonce field is at least moderately hard to guess so that even on-path attackers can't deduce the next nonce value that will be used. This value SHOULD be generated using a random number generator that is known to have good randomness properties as outlined in RFC 1750 [RFC1750].

Probe data

This is a 32-bit field with no fixed meaning. The probe data field is copied back with no changes. Future flags may define a use for this field.

4.3. Keepalive Timeout Option Format

Either side of a HIP association can notify the peer of the value that it would prefer the peer to use as its Keepalive Timeout value. If the host is using a non-default Send Timeout value, it SHOULD communicate this value as a Keepalive Timeout value to the peer in the below option. This option MAY be sent in the I2, R2, or UPDATE messages. The option SHOULD only need to be sent once in a given HIP association. If a host receives this option it SHOULD update its Keepalive Timeout value for the correspondent.

```
       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 = 10           |         Length  = 4           |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      +           Reserved            |      Keepalive Timeout        |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: KeepAlive Timeout Option Format

Fields:

Type

This field identifies the option and MUST be set to 10 (Keepalive Timeout).

The rest of the fiels on this packet are exactly the same as defined on [4].

Length
This field MUST be set as specified in Section 5.1 of the HIP protocol description [1].

Reserved
16-bit field reserved for future use. Set to zero upon transmit and MUST be ignored upon receipt.

Keepalive Timeout
Value in seconds corresponding to suggested Keepalive Timeout value for the peer.

5. Security Considerations

TBD

6. Acknoledgements

Tom Henderson provided comments and feedback on the contents of this draft.

Antonio de la Oliva is partly funded by OneLab, a research project supported by the European Commission under its Sixth Framework Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the OneLab project or the European Commission.

7. References

[1] Moskowitz, R., Nikander, P., and T. Henderson, "Host Identity Protocol", June 2007.

[2] Henderson, T., "End-Host Mobility and Multihoming with the Host Identity Protocol", March 2007.

[3] Nordmark, E. and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6", May 2007.

[4] Arkko, J. and I. van Beijnum, "Failure Detection and Locator Pair Exploration Protocol for IPv6 Multihoming", June 2007.

Authors' Addresses

Antonio de la Oliva
UC3M

Email: aoliva@it.uc3m.es

Marcelo Bagnulo
Huawei Labs at UC3M

Email: marcelo@it.uc3m.es

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).