HIP (Host Identity Protocol) Immediate Carriage and Conveyance of Upper-
                 layer Protocol Signaling (HICCUPS)
                    draft-nikander-hip-hiccups-04

Status of this Memo

   This Internet-Draft is submitted to IETF in full conformance with the
   provisions of BCP 78 and BCP 79.

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF), its areas, and its working groups. Note that
   other groups may also distribute working documents as Internet-
   Drafts.

   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."

   The list of current Internet-Drafts can be accessed at
   http://www.ietf.org/ietf/1id-abstracts.txt.

   The list of Internet-Draft Shadow Directories can be accessed at
   http://www.ietf.org/shadow.html.

   This Internet-Draft will expire on February 21, 2010.

Copyright Notice

Abstract

   This document defines a new HIP (Host Identity Protocol) packet type
   called DATA. HIP DATA packets are used to securely and reliably
   convey arbitrary protocol messages over the Internet and various
   overlay networks.

Table of Contents

[1](#). Introduction

   Two hosts can use HIP [RFC5201] to establish a Security Association
   (SA) between them in order to exchange arbitrary protocol messages
   over that security association. The establishment of such a security
   association involves a four-way handshake referred to as the HIP base
   exchange. When handling communications between the hosts, HIP
   supports mobility, multihoming, security, and NAT traversal. Some
   applications require these features for their communications but
   cannot accept the overhead involved in establishing a security
   association (i.e., the HIP base exchange) before those communications
   can start.

   In this document, we define the HIP DATA packet, which can be used to
   convey (in a secure and reliable way) protocol messages to a remote
   host without running the HIP base exchange between them. We also
   discuss the trade offs involved in using this packet (i.e., less
   overhead but also less DoS protection) and the situations where it is
   appropriate to use this packet. The HIP_DATA packet is not aimed to
   be a replacement for ESP transport instead it SHOULD only be used to
   exchange few packets between the peers. If a continuous
   communication is required hosts SHOULD run the HIP base exchange to
   set up ESP security association.

2. Background on HIP

The HIP protocol specification [RFC5201] defines a number of messages
and parameters. The parameters are encoded as TLVs, as shown in
Section 2.1.2. Furthermore, the HIP header carries a Next Header
field, allowing other arbitrary packets to be carried within HIP
packets.

2.1. Message formats

2.1.1. HIP fixed header

The HIP packet format consists of a fixed header followed by a
variable number of parameters. The parameter format is described in
Section 2.1.2.

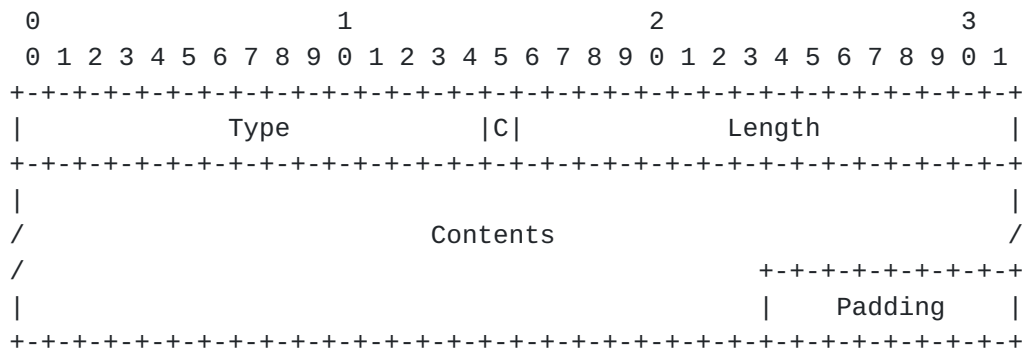The fixed header is defined in Section 5.1 of [RFC5201] and copied
below.

```
 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                         /
/                                                              /
|                                                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The HIP header is logically an IPv6 extension header. The HIP
protocol specification [RFC5201] defines handling only for Next
Header value decimal 59, IPPROTO_NONE, the IPv6 'no next header'
value. This document describes processing for Next Header values
other than decimal 59 which indicates that there is either more
extensions header or data following the HIP header.

2.1.2. HIP parameter format

The HIP parameter format is defined in Section 5.2.1 of [RFC5201],
and copied below.

```
 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             |C|             Length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
/                          Contents                             /
/                                               +-+-+-+-+-+-+-+-+
|                                               |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+


Type          Type code for the parameter
  C           Critical bit, part of the Type.
Length        Length of the parameter, in bytes.
Contents      Parameter specific, defined by Type.
Padding       Padding, 0-7 bytes, added if needed.
```

2.2. HIP Base Exchange, Updates, and State Removal

The HIP base exchange is a four-message half-stateless authentication
and key exchange protocol that creates shared, mutually authenticated
keying material at the communicating parties. These keying
materials, together with associated public keys and IP addresses,
form a HIP Security Association (SA). The details of the protocol
are defined in the HIP base exchange specification [RFC5201].

In addition to creating the HIP SA, the base exchange messages may
carry additional parameters that are used to create additional state.
For example, the HIP ESP specification [RFC5202] defines how HIP can
be used to create end-to-end, host-to-host IPsec ESP Security
Associations, used to carry data packets. However, it is important
to understand that the HIP base exchange is by no means bound to
IPsec; using IPsec ESP to carry data traffic forms just a baseline
and ensures interoperability between initial HIP implementations.

Once there is a HIP SA between two HIP-enabled hosts, they can
exchange further HIP control messages. Typically, UPDATE messages
are used. For example, the HIP mobility and multi-homing
specification [RFC5206] defines how to use UPDATE messages to change
the set of IP addresses associated with a HIP SA.

In addition to the base exchange and updates, the HIP base protocol
specification also defines how one can remove a HIP SA once it is no

   longer needed.

3. Definition of the HIP DATA Packet

   The HIP DATA packet can be used to convey protocol messages to a
   remote host without running the HIP base exchange between them. HIP
   DATA packets are transmitted reliably, as discussed in Section 4.
   The payload of a HIP DATA packet is placed after the HIP header and
   protected by a PAYLOAD_MAC parameter, which is defined in
   Section 3.3. The following is the definition of the HIP DATA packet:

         Header:
           Packet Type = [TBD by IANA: 32]
           SRC HIT = Sender's HIT
           DST HIT = Receiver's HIT

      IP (HIP ([HOST_ID,] SEQ_DATA, PAYLOAD_MAC,
                 HIP_SIGNATURE) PAYLOAD)

      IP (HIP ([HOST_ID,] SEQ_DATA, ACK_DATA, PAYLOAD_MAC,
                 HIP_SIGNATURE) PAYLOAD)

      IP (HIP ([HOST_ID,] ACK_DATA, HIP_SIGNATURE))

   The SEQ_DATA and ACK_DATA parameters are defined in Section 3.1 and
   Section 3.2 respectively. They are used to provide a reliable
   delivery of HIP DATA packets, as discussed in Section 4.

   The HOST_ID parameter is defined in Section 5.2.8 of [RFC5201]. This
   parameter is the sender's Host Identifier that is used to compute the
   HIP DATA packet's signature and to verify it against the received
   signature.

   The PAYLOAD_MAC parameter is defined in Section 3.3. This parameter
   contains the HMAC of the payload carried by the HIP DATA packet. The
   PAYLOAD_MAC contains the checksum of the payload following after the
   HIP DATA. The PAYLOAD_MAC is included in the signed part of the HIP
   DATA packet giving integrity protection also for the payload carried
   after HIP DATA packet.

   The HIP_SIGNATURE parameter is defined in Section 5.2.11. of
   [RFC5201]. It contains a signature over the contents of the HIP DATA
   packet. The calculation and verification of the signature is defined
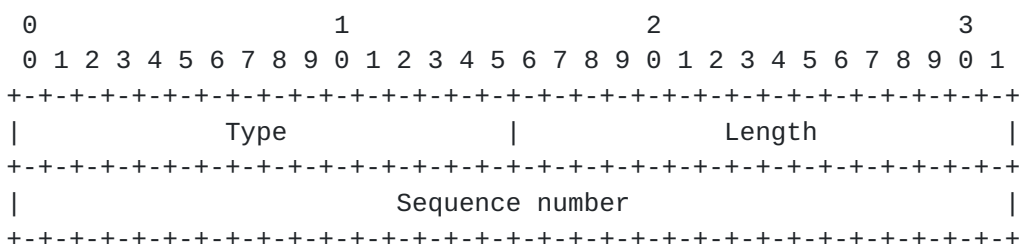   Section 6.4.2. of [RFC5201]

   Section 5.3 of [RFC5201] states the following:

In the future, an OPTIONAL upper-layer payload MAY follow the HIP
header. The Next Header field in the header indicates if there is
additional data following the HIP header.

We have chosen to place the payload after the HIP extension header
and only to place an HMAC of the payload in to the HIP extension
header in a PAYLOAD_MAC parameter because that way the data is
protected by a public key signature with help of HMAC. The payload
that is protected by the PAYLOAD_MAC parameter has been linked to the
appropriate upper-layer protocol by storing the upper-layer protocol
number, 8 bytes of payload data, and by calculating an HMAC over the
data. The HMAC algorithm is same as the algorithm used to generate
the Host Identity Tag.

3.1. Definition of the SEQ_DATA Parameter

The following is the definition of the SEQ_DATA parameter:

```
     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            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        Sequence number                        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+


    Type              [ TBD by IANA:
                        4481 = (2^12 + 2^8 + 2^7 + 1) ]
    Length            4
    Sequence number   32-bit sequence number
```

3.2. Definition of the ACK_DATA Parameter

The following is the definition of the ACK_DATA parameter:
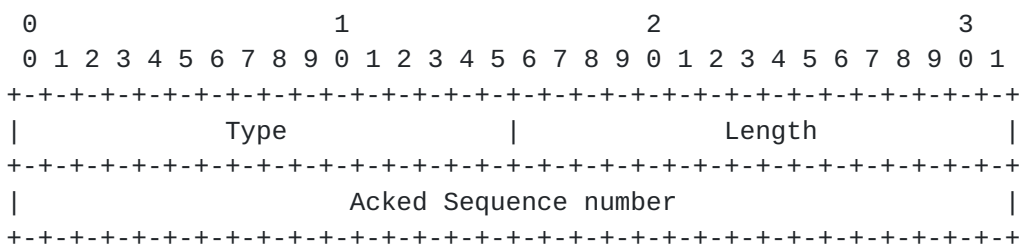
```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |             Type              |             Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                     Acked Sequence number                     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+


    Type                    [ TBD by IANA:
                              4545 = (2^12 + 2^8 + 2^7 + 2^6 + 1) ]
    Length                  4
    Acked Sequence number   32-bit sequence number corresponding to
                            the sequence number being acknowledged
```

3.3. Definition of the PAYLOAD_MAC Parameter

   The following is the definition of the PAYLOAD_MAC parameter:

```
     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             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |   Next header  |                  Reserved                    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        Payload Data                           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                              |
    /                        Payload HMAC                          /
    /                                      +-+-+-+-+-+-+-+-+
    |                                      |    Padding    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
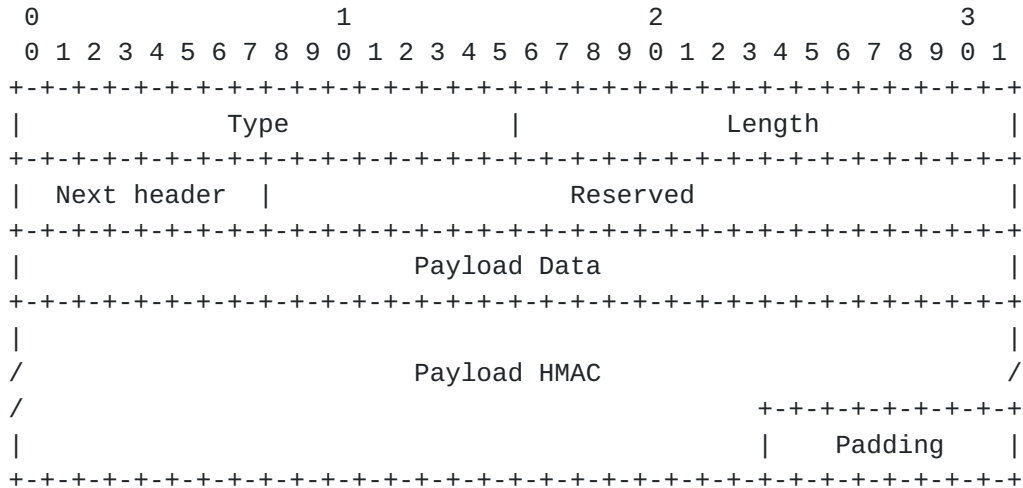
   Type   [TBD by IANA:
                     4577 = (2^12 + 2^8 + 2^7 + 2^6 + 2^5 + 1)]
   Length   length in octets, excluding Type, Length, and
                   Padding
   Next Header   Identifies the data that protected by this HMAC.
                   The values for are defined by IANA "Assigned
                   Numbers".
   Payload Data   8 last bytes of the payload data over which the
                   HMAC is calculated. This field is used to
                   uniquely identify the extension header, in case
                   there are multiple copies of same type.
   Payload HMAC   HMAC computed over the data to which the Next
                   Header and Payload Data points to. The size of
                   the HMAC is the natural size of the computation
                   output depending on the used function.

4. Generation and Reception of HIP DATA Packets

   HIP DATA packets are transmitted reliably. Reliable delivery is
   achieved through the use of retransmissions and of the SEQ_DATA and
   ACK_DATA parameters.

4.1. Handling of SEQ_DATA and ACK_DATA

   A HIP DATA packet contains zero or one SEQ_DATA parameter. The
   presence of a SEQ_DATA parameter indicates that the receiver MUST ACK
   the HIP DATA packet. A HIP DATA packet that does not contain a
   SEQ_DATA parameter is simply an ACK of a previous HIP DATA packet and
   MUST NOT be ACKed.

   A HIP DATA packet contains zero or one ACK_DATA parameters. The ACK
   parameter echoes the SEQ_DATA sequence number of the HIP DATA packet
   packet being ACKed.

   A HIP DATA packet may contain both a SEQ_DATA and an ACK_DATA
   parameter. In this case, the ACK is being piggybacked on an outgoing
   HIP DATA packet. In general, HIP DATA packets carrying SEQ_DATA
   SHOULD be ACKed upon completion of the processing of the HIP DATA
   packet. A host MAY choose to hold the HIP DATA packet carrying ACK
   for a short period of time to allow for the possibility of
   piggybacking the ACK parameter, in a manner similar to TCP delayed
   acknowledgments.

4.2. Generation of a HIP DATA packet

   When a host has upper-layer protocol data to send, it either runs the
   HIP base exchange and sends the data over a SA, or sends the data
   directly using a HIP DATA packet. Section 5 discusses when it is
   appropriate to use each method. This section discusses the case when
   the host chooses to use a HIP DATA packet to send the upper-layer
   protocol data.

   1. The host creates a HIP DATA packet that contains a SEQ_DATA
       parameter. The host is free to choose any value for the SEQ_DATA
       parameter in the first HIP DATA packet it sends to a destination.
       After that first packet, the host MUST choose the value of the
       SEQ_DATA parameter in subsequent HIP DATA packets to the same
       destination so that no SEQ_DATA value is reused before the
       receiver has closed the processing window for the previous packet
       using the same SEQ_DATA value. Practically, giving the values of
       the retransmission timers used with HIP DATA packets, this means
       that hosts must wait the maximum likely lifetime of the packet
       before reusing a given SEQ_DATA value towards a given
       destination. However, it is not required for node to know the

maximum packet lifetime. Rather, it is assumed that the
requirement can be met by maintaining the value as a simple, 32-
bit, "wrap-around" counter, incremented each time a packet is
sent. It is an implementation choice whether to maintain a
single counter for the node or multiple counters (one for each
source HIT, destination HIT combination).

2.   The host creates PAYLOAD_HMAC parameter. The HMAC is calculated
     over the whole PAYLOAD which the Next Header field of
     PAYLOAD_HMAC parameter indicates. The receiver MUST validate
     this HMAC. For calculating the HMAC the host MUST use the same
     hash algorithm as the one that has been used for generating the
     host's HIT as defined in Section 3.2. of [RFC5201].

3.   The host creates HIP_SIGNATURE parameter. The signature is
     calculated over the whole HIP envelope, excluding any parameters
     after the HIP_SIGNATURE, as defined in Section 5.2.11. of
     [RFC5201]. The receiver MUST validate this signature. It MAY
     use either the HI in the packet or the HI acquired by some other
     means.

4.   The hosts sends the created HIP DATA packet and starts a DATA
     timer. The default value for the timer is 2 * RTT estimate. If
     multiple HIP DATA packets are outstanding, multiple timers are in
     effect.

5.   If the DATA timer expires, the HIP DATA packet is resent. The
     HIP DATA packet can be resent DATA_RETRY_MAX times. The DATA
     timer SHOULD be exponentially backed off for subsequent
     retransmissions. If no acknowledgment is received from the peer
     after DATA_RETRY_MAX times, the delivery of the HIP DATA packet
     is considered unsuccessful and the application is notified about
     the error. The DATA timer is canceled upon receiving an ACK from
     the peer that acknowledges receipt of the HIP DATA packet.

4.3. Reception of a HIP DATA packet

A host receiving a HIP DATA packet to decide whether to process it or
not. If the host, following its local policy, suspects that this
packet could be part of a DoS attack. The host MAY respond with an
R1 packet to the HIP DATA packet, if the packet contained SEQ_DATA
and PAYLOAD_HMAC parameter, in order to run the HIP base exchange
with the originator of the HIP DATA packet. If the host chooses to
respond to the HIP DATA with an R1 packet, it creates a new R1 or
selects a precomputed R1 according to the format described in
[RFC5201] Section 5.3.2. The host SHOULD drop the received data
packet if it responded with a R1 packet to the HIP_DATA packet. The
sender of HIP_DATA packet is responsible of retransmission of the

upper-layer protocol data after successful completion of the HIP Base
Exchange.

If the host, following its local policy, decides to process the
incoming HIP DATA packet, it processes it according to the following
rules:

   If the HIP DATA packet contains a SEQ_DATA parameter and no
   ACK_DATA parameter, the HIP DATA packet is processed and replied
   to as described in Section 4.3.1.

   If the HIP DATA packet contains an ACK_DATA parameter and no
   SEQ_DATA parameter, the HIP DATA packet is processed and replied
   to as described in Section 4.3.2.

   If the HIP DATA packet contains both a SEQ_DATA parameter and an
   ACK_DATA parameter, the HIP DATA packet is processed first as
   described in Section 4.3.2 and then the rest of the HIP DATA
   packet is processed and replied to as described in Section 4.3.1.

4.3.1. Handling of SEQ_DATA in a Received HIP DATA packet

   The following steps define the conceptual processing rules for
   handling a SEQ_DATA parameter in a received HIP DATA packet.

   If the value in the received SEQ_DATA corresponds to a HIP DATA
   packet that has recently been processed, the packet is treated as a
   retransmission. The SIGNATURE verification (next step) MUST NOT be
   skipped. (A byte-by-byte comparison of the received and a stored
   packet would be OK, though.) It is recommended that a host cache HIP
   DATA packets sent with ACKs to avoid the cost of generating a new ACK
   packet to respond to a retransmitted HIP DATA packet. The host MUST
   acknowledge, again, such (apparent) HIP DATA packet retransmissions
   but SHOULD also consider rate-limiting such retransmission responses
   to guard against replay attacks.

   The system MUST verify the SIGNATURE in the HIP DATA packet. If the
   verification fails, the packet SHOULD be dropped and an error message
   logged.

   The system MUST verify the PAYLOAD_HMAC by calculating the HMAC over
   the PAYLOAD which the Next Header field indicates. For calculating
   the HMAC the host will use the same hash algorithm that has been used
   to generate the sender's HIT as defined in Section 3.2. of [RFC5201].
   If the verification fails, the packet SHOULD be dropped and an error
   message logged.

   If a new SEQ parameter is being processed, the parameters in the HIP

DATA packet are then processed.

A HIP DATA packet with an ACK_DATA parameter is prepared and sent to
the peer. This ACK_DATA parameter may be included in a separate HIP
DATA packet or piggybacked in a HIP DATA packet with a SEQ_DATA
parameter. The ACK_DATA parameter MAY acknowledge more than one of
the peer's HIP DATA packets.

4.3.2. Handling of ACK_DATA in a Received HIP DATA packet

The following steps define the conceptual processing rules for
handling an ACK_DATA parameter in a received HIP DATA packet.

The sequence number reported in the ACK_DATA must match with an
earlier sent HIP DATA packet that has not already been acknowledged.
If no match is found or if the ACK_DATA does not acknowledge a new
HIP DATA packet, the packet MUST either be dropped if no SEQ_DATA
parameter is present, or the processing steps in Section 4.3.1 are
followed.

The system MUST verify the SIGNATURE in the HIP DATA packet. If the
verification fails, the packet SHOULD be dropped and an error message
logged.

The corresponding DATA timer is stopped so that the now acknowledged
HIP DATA packet is no longer retransmitted. If multiple HIP DATA
packets are newly acknowledged, multiple timers are stopped.

5. Use of the HIP DATA Packet

   HIP currently requires always that the four-message base exchange is
   executed at the first encounter of hosts that have not communicated
   before. This may add additional RTTs (Round Trip Time) to protocols
   based on a single message exchange. However, the four-message
   exchange is essential to preserve the half-stateless DoS protection
   nature of the base exchange. The use of the HIP DATA packet defined
   in this document reduces the initial overhead in the communications
   between two hosts at the expense of decreasing DoS protection.
   Therefore, applications SHOULD NOT use HIP DATA packets in
   environments where DoS attacks are believed to be an issue. For
   example, a HIP-based overlay may have policies in place to control
   which nodes can join the overlay. Any particular node in the overlay
   may want to accept HIP DATA packets from other nodes in the overlay
   given that those other were authorized to join the overlay. However,
   the same node may not want to accept HIP DATA packets from random
   nodes that are not part of the overlay.

   The type of data to be sent is also relevant to whether the use of a
   HIP DATA packet is appropriate. HIP itself does not support
   fragmentation but relies on underlying IP-layer fragmentation. This
   may lead to reliability problems in the case where a message cannot
   be easily split over multiple HIP messages. Therefore, applications
   in environments where fragmentation could be an issue SHOULD NOT
   generate too large HIP DATA packets that may lead to fragmentation.
   The implementation SHOULD check the MTU of the link before sending
   the packet and if the packet size is larger than MTU it SHOULD signal
   to the upper-layer protocol if the packet results in to a ICMP error
   message. Note that there are environments where fragmentation is not
   an issue. For example, in some HIP-based overlays, nodes can
   exchange HIP DATA packets on top of TCP connections that provide
   transport-level fragmentation and, thus, avoid IP-level
   fragmentation.

   HIP currently requires that all messages excluding I1s but including
   HIP DATA packets are digitally signed. This adds to the packet size
   and the processing capacity needed to send packets. However, in
   applications where security is not paramount, it is possible to use
   very short keys, thereby reducing resource consumption.

[6](#). Security considerations

   HIP is designed to provide secure authentication of hosts. HIP also
   attempts to limit the exposure of the host to various denial-of-
   service and man-in-the-middle (MitM) attacks. However, HIP DATA
   packet, which can be sent without running the HIP base exchange
   between hosts has a trade off that it does not provide the denial-of-
   service protection that HIP generally provides. Thus, the host
   should consider always situations where it is appropriate to send or
   receive HIP DATA packet. If the communication consists more than few
   round-trips of data or the data is highly sensitive in nature the
   host SHOULD run the base exchange with the peer host.

7. IANA considerations

   This document updates the IANA Registry for HIP Packet types by
   introducing new packet type for the new HIP_DATA (Section 3) packet.
   This document updates the IANA Registry for HIP Parameter Types by
   introducing new parameter values for the SEQ_DATA (Section 3.1),
   ACK_DATA (Section 3.2), and PAYLOAD_HMAC (Section 3.3) parameters.

8. Acknowledgments

In the usual IETF fashion, a large number of people have contributed
to the actual text or ideas. The list of these people include Miika
Komu, Tobias Heer, Ari Keraenen, Samu Varjonen, Thomas Henderson, and
Jukka Ylitalo. Our apologies to anyone whose name is missing.

9. **Informative references**

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

   [RFC5202]   Jokela, P., Moskowitz, R., and P. Nikander, "Using the
               Encapsulating Security Payload (ESP) Transport Format with
               the Host Identity Protocol (HIP)", RFC 5202, April 2008.

   [RFC5206]   Nikander, P., Henderson, T., Vogt, C., and J. Arkko, "End-
               Host Mobility and Multihoming with the Host Identity
               Protocol", RFC 5206, April 2008.

Authors' Addresses

    Pekka Nikander
    Ericsson
    Hirsalantie 11
    Jorvas 02420
    Finland

    Email: Pekka.Nikander@ericsson.com

    Gonzalo Camarillo
    Ericsson
    Hirsalantie 11
    Jorvas 02420
    Finland

    Email: Gonzalo.Camarillo@ericsson.com

    Jan Melen
    Ericsson
    Hirsalantie 11
    Jorvas 02420
    Finland

    Email: Jan.Melen@ericsson.com