Mobility Extensions (MEXT)   J. Korhonen, Ed.
Internet-Draft   Nokia Siemens Networks
Intended status: Experimental   B. Patil
Expires: April 13, 2012   Nokia
                                                   H. Tschofenig
                                            Nokia Siemens Networks
                                                  D. Kroeselberg
                                                         Siemens
                                                October 11, 2011

Transport Layer Security-based Mobile IPv6 Security Framework for Mobile
                  Node to Home Agent Communication
                    draft-ietf-mext-mip6-tls-02.txt

Abstract

   Mobile IPv6 signaling between a mobile node and its home agent is
   secured using IPsec. The security association between a mobile node
   and the home agent is established using IKEv1 or IKEv2. The security
   model specified for Mobile IPv6, which relies on IKE/IPsec, requires
   interaction between the Mobile IPv6 protocol component and the IKE/
   IPsec module of the IP stack. This document proposes an alternate
   security framework for Mobile IPv6 and Dual-Stack Mobile IPv6, which
   relies on Transport Layer Security for establishing keying material
   and other bootstrapping parameters required to protect Mobile IPv6
   signaling and data traffic between the mobile node and home agent.

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 April 13, 2012.

Copyright Notice

   Copyright (c) 2011 IETF Trust and the persons identified as the

Table of Contents

1. Introduction

   Mobile IPv6 signaling as specified in RFC6275 [RFC6275], and
   optionally user traffic, between a mobile node (MN) and home agent
   (HA) are secured by IPsec [RFC4301]. The current Mobile IPv6
   security architecture is specified in [RFC3776] and [RFC4877]. This
   security model requires a tight coupling between the Mobile IPv6
   protocol part and the IKE(v2)/IPsec part of the IP stack.
   Implementation experience has shown that the use of IKE(v2)/IPsec
   with Mobile IPv6 is fairly complex. The issues with the IKE(v2)/
   IPsec based security architecture are documented in
   [I-D.patil-mext-mip6issueswithipsec].

   This document proposes an alternate security framework for Mobile
   IPv6 and Dual-Stack Mobile IPv6. The objective is to simplify
   implementations as well as make it easy to deploy the protocol
   without compromising on security. Transport Layer Security (TLS)
   [RFC5246] is widely implemented in almost all major operating systems
   and extensively used. Instead of using IKEv2, the security framework
   proposed in this document is based on TLS protected messages to
   exchange keys and bootstrapping parameters between the Mobile Node
   and a new functional entity called as the Home Agent Controller
   (HAC). The Mobile IPv6 signaling between the mobile node and home
   agent is subsequently secured using the resulting keys and negotiated
   cipher suite. The HAC can be co-located with the HA, or can be an
   independent entity. For the latter case, communication between HAC
   and HA is not defined by this document. The Diameter protocol can be
   used between the HA and HAC when the two entities are not collocated.

   The primary advantage of using TLS based establishment of Mobile IP6
   security associations compared to IKEv2 is the ease of implementation
   while providing equivalent level of security. For the protection of
   Mobile IPv6 signaling messages a solution is provided that decouples
   Mobile IPv6 protection from regular IPsec operation to reduce
   complexity in Mobile IP client implementations.

   The security framework proposed in this document is not intended to
   replace the currently specified architecture which relies on IPsec
   and IKEv2. It provides an alternative solution which is more optimal
   for certain deployment scenarios.

2. Terminology and Abbreviations

   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].

Home Agent Controller (HAC):

   The home agent controller is a node responsible for bootstrapping
   Mobile IPv6 security associations between a mobile node and one or
   more home agents. The home agent controller also provides key
   distribution to both mobile nodes and home agents. Mobile IPv6
   bootstrapping is also performed by the HA in addition to the
   security association bootstrapping between the mobile node and
   home agent controller.

Binding Management Messages:

   Mobile IPv6 signaling messages between a mobile node and a home
   agent, correspondent node or mobility access point to manage the
   bindings are referred to as binding management messages. Binding
   Updates and Binding Acknowledgement messages are examples of
   binding management messages.

3. Background

   Work on the design and specification of Mobile IPv6 has been done
   since the late 90s. The security architecture of Mobile IPv6 was
   based on the understanding that IPsec is an inherent and integral
   part of the IPv6 stack and any protocol that needs security should
   use IPsec unless there is a good reason not to. As a result of this
   mindset the Mobile IP6 protocol relied on the use of IPsec for
   security between the MN and HA. While reuse of security components
   that are part of the IP stack is a good objective, in the case of
   Mobile IPv6 implementation complexity increases quite dramatically.
   It should be noted that Mobile IPv4 [RFC5944] for example does not
   use IPsec for security and instead has specified its own security
   solution. Mobile IPv4 has been implemented and deployed on a
   reasonably large scale and the security model has proven itself to be
   sound.

   Mobile IPv6 standardization was completed in 2005 along with the
   security architecture using IKE/IPsec specified in RFC 3776
   [RFC3776]. With the transition to IKEv2 [RFC5996], Mobile IP6
   security has also been updated to rely on the use of IKEv2 [RFC4877].
   Recent implementation exercises of Mobile IPv6 and Dual-stack Mobile
   IPv6 [RFC5555] have identified the complexity of using IPsec and
   IKEv2 in conjunction with Mobile IPv6. At an abstract level it can
   be said that implementing Mobile IPv6 with IPsec and IKEv2 is
   possible only with modifications to the IPsec and IKEv2 components.
   The original design intent was to reuse IPsec without having to
   modify them. The current security model requires an IPsec/IKEv2

implementation that is specific to Mobile IPv6.

This document proposes a security framework based on TLS protected
establishment of Mobile IPv6 security associations with reduced
implementation complexity, while maintaining an equivalent (to IKEv2/
IPsec) level of security.

4. TLS-based Security Establishment

4.1. Overview

The security architecture proposed in this document relies on a
secure TLS session established between the MN and the HAC for
authentication and MN-HA security association bootstrapping.
Authentication of the HAC is done via standard TLS operation where
the HAC uses a TLS server certificate as credentials. MN
authentication is done by the HAC via signaling messages that are
secured by the TLS connection. This can either be MN-only
authentication within the server-authenticated TLS channel, or mutual
authentication between the MN and HAC. Upon successful completion of
authentication, the HAC generates keys which are delivered to the MN
through the secure TLS channel. The same keys are also provided to
the assigned HA. The HAC also provides the MN with MIP6
bootstrapping information such as the address of the HA, the Home
network prefix, and the IPv6 and/or IPv4 HoA.

The MN and HA use security associations based on the keys and SPIs
generated by the HAC and delivered to the MN and HA. Figure 1 below
is an illustration of the process:

```
      MN                              HAC                  HA
      --                              ---                  --
       |                               |                    |
       | /------------------------\ |                    |
       |/                          \|                    |
       |\    TLS session setup     /|                    |
       | \------------------------/ |                    |
       |                               |                    |
       | /------------------------\ |                    |
       |/      MN Authentication    \|                    |
       |\                          /|                    |
       | \------------------------/ |                    |
       |                               |                    |
       | /------------------------\ |                    |
       |/   HAC provisions the MN   \|                    |
       |\  keys, SPI and MIP6 parms /|                    |
       | \------------------------/ |                    |
       |                               |--MNID, keys, SPI->|
       |                               |                    |
       | /--------------------------------------------\ |
       |/     MN-HA SA established; Secures            \ |
       |\     signaling and optionally user traffic   / |
       | \--------------------------------------------/ |
       |                                                  |
       |-----------BU(encrypted)---------------------->|
       |                                                  |
       |<--------BAck(encrypted)-----------------------|
```

                   Figure 1: High level architecture

4.2. Architecture

   The TLS-based security architecture is shown in Figure 2. The
   signaling message exchange between the MN and the HAC is protected by
   TLS. It should be noted that a HAC, a AAA server and a HA are
   logically separate entities and can be collocated in all possible
   combinations. There MUST be a strong trust relationship between the
   HA and the HAC, and the communication between them MUST be both
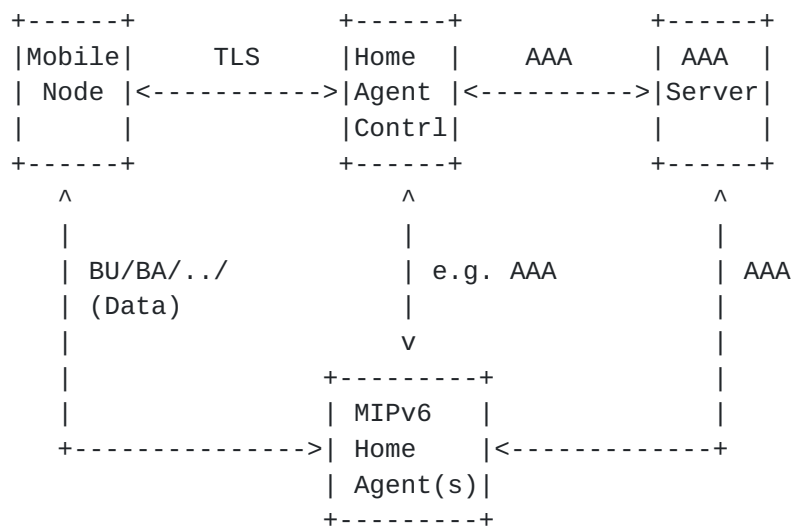   integrity and confidentially protected.

```
   +------+                +------+                +------+
   |Mobile|      TLS       |Home  |      AAA       | AAA  |
   | Node |<----------->|Agent |<---------->|Server|
   |      |                |Contrl|                |      |
   +------+                +------+                +------+
      ^                       ^                       ^
      |                       |                       |
      | BU/BA/../             | e.g. AAA              | AAA
      | (Data)                |                       |
      |                       v                       |
      |                 +---------+                   |
      |                 | MIPv6   |                   |
    +--------------->| Home      |<-------------+
                      | Agent(s)|
                      +---------+
```

            Figure 2: TLS-based Security Architecture Overview

4.3. Security Association Management

   Once the MN has contacted the HAC and mutual authentication has taken
   place between the MN and the HAC, the HAC securely provisions the MN
   with all security related information inside the TLS protected
   tunnel. This security related information constitutes a security
   association (SA) between the MN and the HA. The created SA MUST NOT
   be tied to the Care-of Address (CoA) of the MN.

   The HAC may proactively distribute the SA information to HAs, or the
   HA may query the SA information from the HAC once the MN contacts the
   HA. If the HA requests SA information from the HAC, then the HA MUST
   be able to query/index the SA information from the HAC based on the
   Security Parameter Index (SPI) identifying the correct security
   association between the MN and the HA.

   The HA may want the MN to re-establish the SA even if the existing SA
   is still valid. The HA can indicate this to the MN using a dedicated
   Status Code in a BA (value set to REINIT_SA_WITH_HAC). As a result,
   the MN SHOULD contact the HAC prior to the SA timing out, and the HAC
   would provision the MN and HAs with a new SA to be used subsequently.

   The SA established between MN and HAC SHALL contain at least the
   following information:

   Mobility SPI:

      This parameter is an SPI used by the MN and the HA to index the SA
      between the MN and the HA. The HAC is responsible for assigning
      SPIs to MNs. There is only one SPI for both binding management

messaging and possible user data protection. The same SPI is used
for both directions between the MN and the HA. The SPI values are
assigned by the HAC. The HAC MUST ensure uniqueness of the SPI
values across all MNs controlled by the HAC.

MN-HA keys for ciphering:

A pair of symmetric keys (MN -> HA, HA -> MN) used for ciphering
Mobile IPv6 traffic between the MN and the HA. The HAC is
responsible for generating these keys. The key generation
algorithm is specific to the HAC implementation.

MN-HA shared key for integrity protection:

A pair of symmetric keys (MN -> HA, HA -> MN) used for integrity
protecting Mobile IPv6 traffic between the MN and the HA. This
includes both binding management messages and reverse tunneled
user data traffic between the MN and the HA. The HAC is
responsible for generating these keys. The key generation
algorithm is specific to the HAC implementation. In case of
combined algorithms a separate integrity protection key is not
needed and may be omitted, i.e., the encryption keys SHALL be
used.

Security association validity time:

This parameter represents the validity time for the security
association. The HAC is responsible for defining the lifetime
value based on its policies. The lifetime may be in the order of
hours or weeks. The MN MUST re-contact the HAC before the SA
validity time ends.

Security Association Scope:

This parameter defines whether the security association is applied
to Mobile IPv6 signaling messages only, or to both Mobile IPv6
signaling messages and data traffic.

Selected ciphersuite:

This parameter is the ciphersuite used to protect the traffic
between the MN and the HA. This includes both binding management
messages and reverse tunneled user data traffic between the MN and
the HA. The selected algorithms SHOULD be one of the mutually
supported ciphersuites of the negotiated TLS version between the
MN and the HAC. The HAC is responsible for choosing the mutually
supported ciphersuite that complies with the policy of the HAC.
Obviously, the HAs under HAC's management must have at least one

ciphersuite with the HAC in common and need to be aware of the
implemented ciphersuites. The selected ciphersuite is the same
for both directions (MN -> HA and HA -> MN).

Sequence numbers:

A monotonically increasing unsigned sequence number used in all
protected packets exchanged between the MN and the HA in the same
direction. Sequence numbers are maintained per direction, so each
SA includes two independent sequence numbers (MN -> HA, HA -> MN).
The initial sequence number for each direction MUST always be set
to 0 (zero). Sequence numbers cycle to 0 (zero) when increasing
beyond their maximum defined value.

4.4. Bootstrapping of Additional Mobile IPv6 Parameters

When the MN contacts the HAC to distribute the security related
information, the HAC may also provision the MN with various Mobile
IPv6 related bootstrapping information. Bootstrapping of the
following information SHOULD at least be possible:

Home Agent IP Address:

Concerns both IPv6 and IPv4 home agent addresses.

Mobile IPv6 Service Port Number:

The port number where the HA is listening to UDP [RFC0768]
packets.

Home Address:

Concerns both IPv6 and IPv4 Home Addresses.

Home Link Prefix:

Concerns the IPv6 Home link prefix and the associated prefix
length.
DNS Server Address:

The address of a DNS server that can be reached via the HA. DNS
queries in certain cases cannot be routed to the DNS servers
assigned by the access network to which the MN is attached and
hence an additional DNS server address which is reachable via the
HA needs to be configured.

The Mobile IPv6 related bootstrapping information is delivered from
the HAC to the MN over the same TLS protected tunnel as the security

related information.

4.5. Protecting Traffic Between Mobile Node and Home Agent

The same integrity and confidentiality algorithms MUST be used to
protect both binding management messages and reverse tunneled user
data traffic between the MN and the HA. Generally, all binding
management messages (BUs, BAs and so on) MUST be integrity protected
and SHOULD be confidentially protected. The reverse tunneled user
data traffic SHOULD be equivalently protected. Generally, the
requirements stated in [RFC6275] concerning the protection of the
traffic between the MN and the HA also apply to the mechanisms
defined by this specification.

5. Mobile Node to Home Agent Controller Communication

5.1. Request-response Message Framing over TLS-tunnel

The MN and the HAC communicate with each other using a simple lock-
step request-response protocol that is run inside the protected TLS-
tunnel. A generic message container framing for the request messages
and for the response messages is defined. The message containers are
only meant to be exchanged on top of connection oriented TLS-layer.
Therefore, the end of message exchange is determined by the other end
closing the transport connection (assuming the "application layer"
has also indicated the completion of the message exchange). The peer
initiating the TLS-connection is always sending "Requests" and the
peer accepting the TLS-connection is always sending "Responses". The
format of the message container is shown in Figure 3.

All data inside the Content portion of the message container MUST be
encoded using octets. Fragmentation of message containers is not
supported, which means one request or response at the "application
layer" MUST NOT exceed the maximum size allowed by the message
container format.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Ver |  Rsrvd  | Identifier    | Length                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Content portion..                                            ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
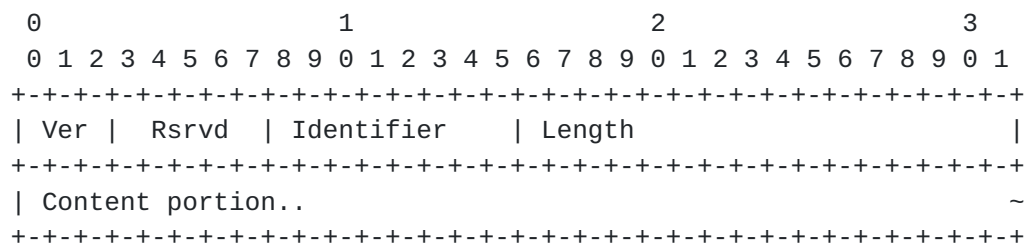
Figure 3: Request-Response Message Container

The three bit Ver field identifies the protocol version. The current

version is 0 i.e. all bits are set to value 0 (zero).

The Rsrvd field MUST be set to value 0 (zero),

The Identifier field is meant for matching requests and responses.
The valid Identifier values are between 1-255. The value 0 MUST NOT
be used. The first request for each communication session between
the MN and the HAC MUST have the Identifier values set to 1.

The Length field tells the length of the Content portion of the
container (i.e. Reserved octet, Identifier octet and Length field
are excluded). The Content portion length MUST always be at least
one octet up to 65535 octets. The value is in network order.

5.2. Request-response Message Content Encoding

The encoding of the message content is similar to HTTP header
encoding, and complies to the augmented Backus-Naur Form (BNF)
defined in Section 2.1 of [RFC2616]. All presented hexadecimal
numbers are in network byte order. From now on, we use TypeValue
header (TV-header) term to refer request-response message content
HTTP-like headers.

5.3. Request-Response Message Exchange

The message exchange between the MN and the HAC is a simple lock-step
request-response type as stated in Section 5.1. A request message
includes monotonically increasing Identifier value that is copied to
the corresponding response message. Each request MUST have a
different Identifier value. Hence, a reliable connection oriented
transport below the message container framing is assumed. The number
of request-response message exchanges MUST NOT exceed 255.

Each new communication session between the MN and the HAC MUST reset
the Identifier value to 1. The MN is also the peer that always sends
only request messages and the HAC only sends response messages. Once
the request-response message exchange completes, the HAC and the MN
MUST close the transport connection and the corresponding TLS-tunnel.

In a case of a HAC side error, the HAC MUST send a response back to a
MN with an appropriate status code and then close the transport
connection.

The first request message - MHAuth-Init - (i.e. the Identifier is 1)
MUST always contain at least the following parameters:

      MN-Identity - See Section 5.5.1.
      Authentication Method - See Section 5.5.2.

   The first response message - MHAuth-Init - (i.e. the Identifier is 1)
   MUST contain at minimum the following parameters:

      Selected authentication Method - See Section 5.5.2.

   The last request message from the MN side - MHAuth-Done - MUST
   contain the following parameters:

      Security Association Scope - See Section 5.6.4.
      Proposed ciphersuites - See Section 5.6.5.
      Message Authenticator - See Section 5.5.5.

   The last response message - MHAuth-Done - that ends the request-
   response message exchange MUST contain the following parameters:

      Status Code - See Section 5.5.4.
      Message Authenticator - See Section 5.5.5.

   And in a case of successful authentication the following additional
   parameters:

      Selected ciphersuite - See Section 5.6.5.
      Security Association Scope - See Section 5.6.4.
      The rest of the security association data - See Section 5.6.

5.4. Home Agent Controller Discovery

   All bootstrapping information, whether for setting up the SA or for
   bootstrapping Mobile IPv6 specific information, is exchanged between
   the MN and the HAC using the framing protocol defined in Section 5.1.
   The IP address of the HAC MAY be statically configured to the MN or
   may be dynamically discovered using DNS. In a case of DNS-based HAC
   discovery, the MN either queries an A/AAAA or a SRV record for the
   HAC IP address. The actual domain name used in queries is up to the
   deployment to decide and out of scope of this specification.

5.5. Generic Request-Response Parameters

5.5.1. Mobile Node Identifier

   An identifier that identifies a MN. The Mobile Node Identifier is in
   form of a Network Access Identifier (NAI) [RFC4282].

```
   mn-id = "mn-id" ":" nai CRLF
   nai = username
       | "@" realm
       | username "@" realm
   ...
```

5.5.2. Authentication Method

The HAC is the peer that mandates the authentication method. The MN
sends its authentication method proposal to the HAC. The HAC, upon
receipt of the MN proposal returns the selected authentication
method. The MN MUST propose at least one authentication method. The
HAC MUST select exactly one authentication method, or return an error
and then close the connection.

```
   auth-method = "auth-method" ":" a-method *("," a-method) CRLF
   a-method =
        "psk" ; Pre-shared key based authentication
      | "eap" ; EAP-based authentication
```

5.5.3. Extensible Authentication Protocol Payload

Each Extensible Authentication Protocol (EAP) [RFC3748] message is an
encoded string of hexadecimal numbers. The "eap-payload" is
completely transparent what EAP-method or EAP message is carried
inside it. The "eap-payload" can appear in both request and response
messages:

```
   eap-payload = "eap-payload" ":" 1*(HEX HEX) CRLF
```

5.5.4. Status Code

The "status-code" MUST only be present in the response message that
ends the request-response message exchange. The "status-code"
follows the principles of HTTP and the definitions found in Section
10 of RFC 2616 also apply for these status codes listed below:

```
   status-code = "status-code" ":" status-value CRLF
   status-value =
        "100" ; Continue
      | "200" ; OK
      | "400" ; Bad Request
      | "401" ; Unauthorized
      | "500" ; Internal Server Error
      | "501" ; Not Implemented
      | "503" ; Service Unavailable
      | "504" ; Gateway Time-out
```

5.5.5. Message Authenticator

   The "auth" header contains data used for authentication purposes. It
   MUST be the last TV-header in the message and calculated over the
   whole message till the start of the "msg-header":

      msg-auth = "auth" ":" 1*(HEX HEX) CRLF

5.5.6. Retry After

      retry-after = "retry-after" ":" rfc1123-date CRLF

5.5.7. End of Message Content

      end-of-message = 2CRLF

5.5.8. Random Values

   Random numbers generated by the MN and the HAC, respectively. The
   length of the random number MUST be 32 octets (before TV-header
   encoding):

      mn-rand = "mn-rand" ":" 32(HEX HEX) CRLF
      hac-rand = "hac-rand" ":" 32(HEX HEX) CRLF

5.6. Security Association Configuration Parameters

   During the Mobile IPv6 bootstrapping, the MN and the HAC negotiate a
   single ciphersuite for protecting the traffic between the MN and the
   HA. The allowed ciphersuites for this specification are a subset of
   those in TLS v1.2 (see Annex A.5 of [RFC5246]) as per Section 5.6.5.
   This might appear as a constraint as the HA and the HAC may have
   implemented different ciphersuites. These two nodes are, however,
   assumed to belong to the same administrative domain. In order to
   avoid exchanging supported MN-HA ciphersuites in the MN-HAC protocol
   and to reuse the TLS ciphersuite negotiation procedure we make this
   simplifying assumption. The selected ciphersuite MUST provide
   integrity and confidentiality protection.

   Section 5.6.5 provides the mapping from the TLS ciphersuites to the
   integrity and encryption algorithms allowed for MN-HA protection.
   This mapping mainly ignores the authentication algorithm part that is
   not required within the context of this specification. For example,
   [RFC5246] defines a number of AES based ciphersuites for TLS
   including 'TLS_RSA_WITH_AES_128_CBC_SHA'. For this specification the
   relevant part is 'AES_128_CBC_SHA'.

   All the parameters described in the following sections apply only to

a request-response protocol response message to the MN. The MN has
no way affecting to the provisioning decision of the HAC.

5.6.1. Security Parameter Index

The 28-bit unsigned SPI number identifies the SA used between the MN
and the HA. The value 0 (zero) is reserved and MUST NOT be used.
Therefore, values ranging from 1 to 268435455 are valid.

The TV-header corresponding to the SPI number is:

    mip6-spi = "mip6-spi" ":" 1*DIGIT CRLF

5.6.2. MN-HA Shared Keys

The MN-HA shared integrity (ikey) and encryption (ekey) keys are used
to protect the traffic between the MN and the HA. The length of
these keys depend on the selected ciphersuite.

The TV-headers that carry these two parameters are:

    mip6-mn-to-ha-ikey = "mip6-mn-to-ha-ikey" ":" 1*(HEX HEX) CRLF
    mip6-ha-to-mn-ikey = "mip6-ha-to-mn-ikey" ":" 1*(HEX HEX) CRLF
    mip6-mn-to-ha-ekey = "mip6-mn-to-ha-ekey" ":" 1*(HEX HEX) CRLF
    mip6-ha-to-mn-ekey = "mip6-ha-to-mn-ekey" ":" 1*(HEX HEX) CRLF

5.6.3. Security Association Validity Time

The end of the SA validity time is encoded using the "rfc1123-date"
format, as defined in Section 3.3.1 of [RFC2616].

The TV-header corresponding to the SA validity time value is:

    mip6-sa-validity-end = "mip6-sa-validity-end" ":" rfc1123-date
    CRLF

5.6.4. Security association scope (SAS)

The SA is applied either to Mobile IPv6 signaling messages only, or
to both Mobile IPv6 signaling messages and data traffic. This policy
MUST be agreed between the MN and HA prior to using the SA.
Otherwise the receiving side would not be aware of whether the SA
applies to data traffic and could not decide how to act when
receiving unprotected packets of PType 1 (see Section 6.4).

    mip6-sas = "mip6-sas" ":" 1DIGIT CRLF

where a value of "0" indicates that the SA does not protect data

traffic and a value of "1" indicates that all data traffic MUST be
protected by the SA. If the mip6-sas value of an SA is set to 1, any
packet received with a PType value that does not match the mip6-sas
value of the SA MUST be silently discarded.

The HAC is the peer that mandates the used security association
scope. The MN sends its proposal to the HAC but eventually the
security association scope returned from the HAC defines the used
scope.

5.6.5. CipherSuites and Ciphersuite-to-Algorithm Mapping

The ciphersuite negotiation between HAC and MN uses a subset of the
TLS 1.2 ciphersuites and follows the TLS 1.2 numeric representation
defined in Annex A.5 of [RFC5246]. The TV-headers corresponding to
the selected ciphersuite and ciphersuite list are:

```
   mip6-ciphersuite = "mip6-ciphersuite" ":" csuite CRLF
   csuite = "{" suite "}"
   suite =
        "00" "," "02" ; CipherSuite NULL_SHA        = {0x00,0x02}
      | "00" "," "3B" ; CipherSuite NULL_SHA256      = {0x00,0x3B}
      | "00" "," "0A" ; CipherSuite 3DES_EDE_CBC_SHA   = {0x00,0x0A}
      | "00" "," "2F" ; CipherSuite AES_128_CBC_SHA    = {0x00,0x2F}
      | "00" "," "3C" ; CipherSuite AES_128_CBC_SHA256 = {0x00,0x3C}
   mip6-suitelist = "mip6-suitelist" ":" csuite *("," csuite) CRLF
```

All other Ciphersuite values are reserved.

The following integrity algorithms MUST be supported by all
implementations:

```
   HMAC-SHA1-96                    [RFC2404]
   AES-XCBC-MAC-96                 [RFC3566]
```

The binding management messages between the MN and HA MUST be
integrity protected. Implementations MUST NOT use a NULL integrity
algorithm.

The following encryption algorithms MUST be supported:

```
   NULL                         [RFC2410]
   TripleDES-CBC                [RFC2451]
   AES-CBC with 128-bit keys    [RFC3602]
```

Traffic between MN and HA MAY be encrypted. Any integrity-only
CipherSuite makes use of the NULL encryption algorithm.

Note: In the present version, this document does not consider
combined algorithms. The following table provides the mapping of
each ciphersuite to a combination of integrity and encryption
algorithms that are part of the negotiated SA between MN and HA.

```
+-------------------+----------------+--------------------------+
|Ciphersuite        |Integ. Algorithm |Encr. Algorithm          |
+-------------------+----------------+--------------------------+
|NULL_SHA           |HMAC-SHA1-96    |NULL                      |
|NULL_SHA256        |AES-XCBC-MAC-96 |NULL                      |
|3DES_EDE_CBC_SHA    |HMAC-SHA1-96    |TripleDES-CBC             |
|AES_128_CBC_SHA     |HMAC-SHA1-96    |AES-CBC with 128-bit keys |
|AES_128_CBC_SHA256 |AES-XCBC-MAC-96 |AES-CBC with 128-bit keys |
+-------------------+----------------+--------------------------+
```

                    Ciphersuite-to-Algorithm Mapping

5.7. Mobile IPv6 Bootstrapping Parameters

In parallel with the SA bootstrapping, the HAC SHOULD provision the
MN with relevant Mobile IPv6 related bootstrapping information.

The following generic BNFs are used to form IP addresses and
prefixes. They are used in subsequent sections.

```
ip6-addr   = 7( word ":" ) word CRLF
word       = 1*4HEX
ip6-prefix = ip6-addr "/" 1*2DIGIT
ip4-addr   = 1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT
ip4-subnet = ip4-addr "/" 1*2DIGIT
```

5.7.1. Home Agent Address

The HAC MAY provision the MN with an IPv4 or an IPv6 address of a HA,
or both.

```
mip6-haa-ip6 = "mip6-haa-ip6" ":" ip6-addr CRLF
mip6-haa-ip4 = "mip6-haa-ip4" ":" ip4-addr CRLF
```

5.7.2. Mobile IPv6 Service Port Number

The HAC SHOULD provision the MN with an UDP port number, where the HA
expects to receive UDP packets. If this parameter is not present,
then the IANA reserved port number (HALTSEC) MUST be used instead.

```
mip6-port = "mip6-port" ":" 1*5DIGIT CRLF
```

5.7.3. Home Addresses and Home Network Prefix

   The HAC MAY provision the MN with an IPv4 or an IPv6 home address, or
   both. The HAC MAY also provision the MN with its home network
   prefix.

```
   mip6-ip6-hoa = "mip6-ip6-hoa" ":" ip6-addr CRLF
   mip6-ip4-hoa = "mip6-ip4-hoa" ":" ip4-addr CRLF
   mip6-ip6-hnp = "mip6-ip6-hnp" ":" ip6-prefix CRLF
   mip6-ip4-hnp = "mip6-ip4-hnp" ":" ip4-subnet CRLF
```

5.7.4. DNS Server

   The HAC may also provide the MN with DNS server configuration
   options. These DNS servers are reachable via the home agent.

```
   dns-ip6 = "dns-ip6" ":" ip6-addr CRLF
   dns-ip4 = "dns-ip4" ":" ip4-addr CRLF
```

5.8. Authentication of the Mobile Node

   This section describes the basic operation required for the MN-HAC
   mutual authentication and the channel binding. The authentication
   protocol described as part of this section is a simple exchange that
   follows the GPSK exchange used by EAP-GPSK [RFC5433]. It is secured
   by the TLS tunnel and is cryptographically bound to the TLS tunnel
   through channel binding based on [RFC5056] and on the channel binding
   type 'tls-server-endpoint' described in [RFC5929]. As a result of
   the channel binding type, this method can only be used with TLS
   ciphersuites that use server certificates and the Certificate
   handshake message. For example, TLS ciphersuites based on PSK or
   anonymous authentication cannot be used.

   The authentication exchange MUST be performed through the encrypted
   TLS tunnel. It performs mutual authentication between the MN and the
   HAC based on a pre-shared key (PSK) or based on an EAP-method (see
   Section 5.9). The PSK protocol is described in this section. It
   consists of the message exchanges (MHAuth-Init, MHAuth-Mid, MHAuth-
   Done) in which both sides exchange nonces and their identities, and
   compute and exchange a message authenticator 'auth' over the
   previously exchanged values, keyed with the pre-shared key. The
   MHAuth-Done messages are used to deal with error situations. Key
   binding with the TLS tunnel is ensured by channel binding of the type
   "tls-server-endpoint" as described by [RFC5929] where the hash of the
   TLS server certificate serves as input to the 'auth' calculation of
   the MHAuth messages.

   Note: The authentication exchange is based on the GPSK exchange used

by EAP-GPSK. In comparison to GPSK, it does not support exchanging
an encrypted container (it always runs through an already protected
TLS tunnel). Furthermore, the initial request of the authentication
exchange (MHAuth-Init) is sent by the MN (client side) and is
comparable to EAP-Response/Identity, which reverses the roles of
request and response messages compared to EAP-GPSK. Figure 4 shows a
successful protocol exchange.


```
 MN                                                          HAC
  |                                                           |
  | Request/MHAuth-Init (...)                                 |
  |---------------------------------------------------------->|
  |                                                           |
  |                                 Response/MHAuth-Init (...) |
  |<----------------------------------------------------------|
  |                                                           |
  | Request/MHAuth-Done (...)                                 |
  |---------------------------------------------------------->|
  |                                                           |
  |                                 Response/MHAuth-Done (...) |
  |<----------------------------------------------------------|
  |                                                           |
```

    Figure 4: Authentication of the Mobile Node Using Shared Secrets

   1) Request/MHAuth-Init: (MN -> HAC)
         mn-id, mn-rand, auth-method=psk

   2) Response/MHAuth-Init: (MN <- HAC)
         [mn-rand, hac-rand, auth-method=psk, [status],] auth

   3) Request/MHAuth-Done: (MN -> HAC)
         mn-rand, hac-rand, sa-scope, ciphersuite-list, auth

   4) Response/MHAuth-Done: (MN <- HAC)
         [sa-scope, sa-data, ciphersuite, bootstrapping-data,] mn-rand,
         hac-rand, status, auth

   Where:

      auth = HMAC-SHA256(PSK, msg-octets | CB-octets)

   The length "mn-rand", "hac-rand" is 32 octets. Note that "|"
   indicates concatenation and optional parameters are shown in square
   brackets [..]. The square brackets can be nested.

   The shared secret PSK can be variable length. 'msg-octets' includes

all payload parameters of the respective message to be signed except
the 'auth' payload. CB-octets is the channel binding input to the
auth calculation that is the "TLS-server-endpoint" channel binding
type. The content and algorithm (only required for the "TLS-server-
endpoint" type) are the same as described in [RFC5929].

The MN starts by selecting a random number 'mn-rand' and choosing a
list of supported authentication methods coded in 'auth-method'. The
MN sends its identity 'mn-id', 'mn-rand' and 'auth-method' to the HAC
in MHAuth-Init. The decision of which authentication method to offer
and which to pick is policy- and implementation-dependent and,
therefore, outside the scope of this document.

In MHAuth-Done, the HAC sends a random number 'hac-rand' and the
selected ciphersuite. The selection MUST be one of the MN-supported
ciphersuites as received in 'ciphersuite-list'. Furthermore, it
repeats the received parameters of the MHAuth-Init message 'mn-rand'.
It computes a message authenticator 'auth' over all the transmitted
parameters except 'auth' itself. The HAC calculates 'auth' over all
parameters and appends it to the message.

The MN verifies the received MAC and the consistency of the
identities, nonces, and ciphersuite parameters transmitted in MHAuth-
Auth. In case of successful verification, the MN computes a MAC over
the session parameter and returns it to the HAC in MHAuth-Done. The
HAC verifies the received MAC and the consistency of the identities,
nonces, and ciphersuite parameters transmitted in MHAuth-Init. If
the verification is successful, MHAuth-Done is prepared and sent by
the HAC to confirm successful completion of the exchange.

5.9. Extensible Authentication Protocol Methods

Basic operation required for the MN-HAC mutual authentication using
EAP-based methods.

```
    MN                                                    HAC
     |                                                     |
     | Request/MHAuth-Init (...)                           |
     |---------------------------------------------------->|
     |                                                     |
     |                             Response/MHAuth-Init (..., |
     |                    eap-payload=EAP-Request/Identity) |
     |<----------------------------------------------------|
     |                                                     |
     | Request/MHAuth-Mid (eap-payload=                    |
     |              EAP-Response/Identity)                 |
     |---------------------------------------------------->|
     |                                                     |
     |       Response/MHAuth-Mid (eap-payload=EAP-Request/...) |
     |<----------------------------------------------------|
     |                                                     |
     :                                                     :
     :         ..EAP-method specific exchanges..           :
     :                                                     :
     |                                                     |
     | Request/MHAuth-Done (eap-payload=EAP-Response/...,  |
     |                   ..., auth)                        |
     |---------------------------------------------------->|
     |                                                     |
     |         Response/MHAuth-Done (eap-payload=EAP-Success, |
     |                           ..., auth)                |
     |<----------------------------------------------------|
     |                                                     |
```
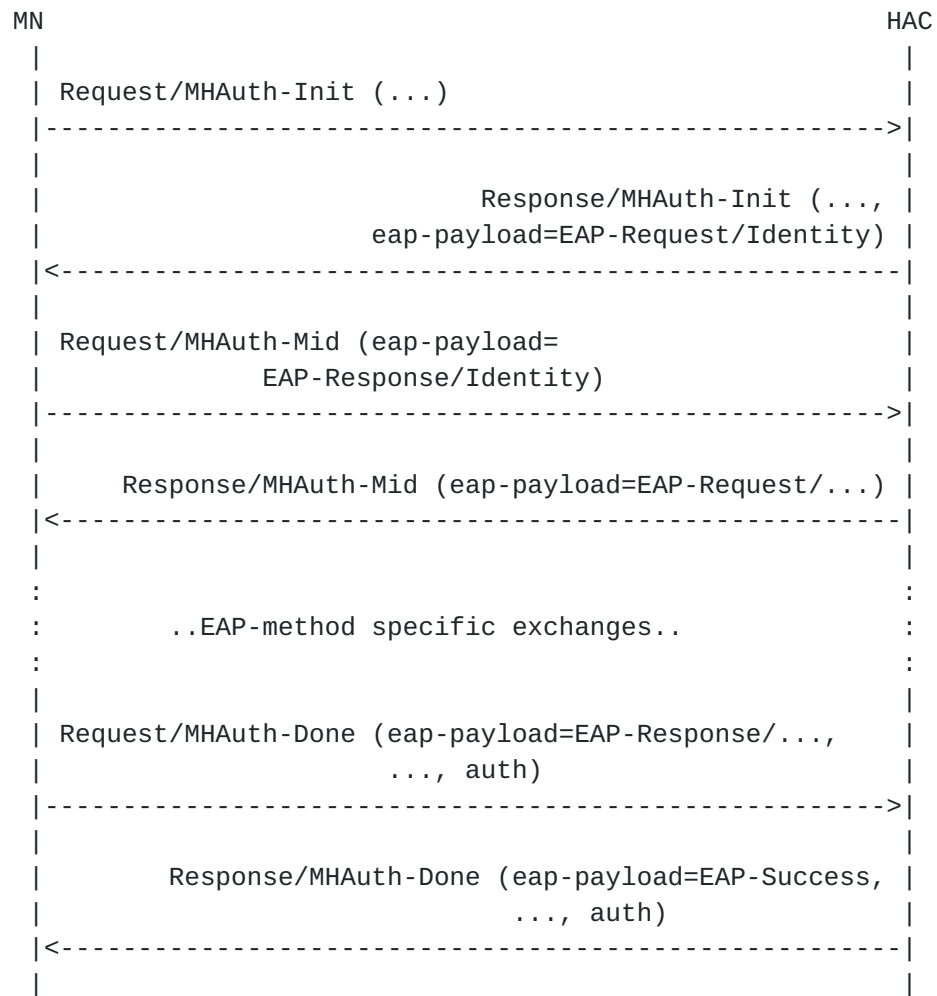
             Figure 5: Authentication of the Mobile Node Using EAP

   1) Request/MHAuth-Init: (MN -> HAC)
          mn-id, mn-rand, auth-method=eap

   2) Response/MHAuth-Init: (MN <- HAC)
          [auth-method=eap, eap, [status,]] auth

   3) Request/MHAuth-Mid: (MN -> HAC)
          eap, auth

   4) Response/MHAuth-Mid: (MN <- HAC)
          eap, auth

       MHAuth-Mid exchange is repeated as many times as needed by the
       used EAP-method.

   5) Request/MHAuth-Done: (MN -> HAC)
          sa-scope, ciphersuite-list, eap, auth

   6) Response/MHAuth-Done: (MN <- HAC)
          [sa-scope, sa-data, ciphersuite, bootstrapping-data,] eap,
          status, auth

   Where:

      auth = HMAC-SHA256(shared-key, msg-octets | CB-octets)

   In MHAuth-Init and MHAuth-Mid messages, shared-key is set to "1". If
   the EAP-method is key-deriving and creates a shared MSK key as a side
   effect of Authentication shared-key MUST be the MSK in all MHAuth-
   Done messages. This MSK MUST NOT be used for any other purpose. In
   case the EAP method does not generate an MSK key, shared-key is set
   to "1".

   'msg-octets' includes all payload parameters of the respective
   message to be signed except the 'auth' payload. CB-octets is the
   channel binding input to the AUTH calculation that is the "TLS-
   server-endpoint" channel binding type. The content and algorithm
   (only required for the "TLS-server-endpoint" type) are the same as
   described in [RFC5929].

6. Mobile Node to Home Agent communication

6.1. General

   The following sections describe the packet formats used for the
   traffic between the MN and the HA. This traffic includes binding
   management messages (for example, BU and BA messages), reverse
   tunneled and encrypted user data, and reverse tunneled plain text
   user data. This specification defines a generic packet format, where
   everything is encapsulated inside UDP. See Section 6.3 and
   Section 6.4 for detailed illustrations of the corresponding packet
   formats.

   The Mobile IPv6 service port number is where the HA expects to
   receive UDP packets. The same port number is used for both binding
   management messages and user data packets. The reason for
   multiplexing data and control messages over the same port number is
   due to the possibility of Network Address and Port Translators
   located along the path between the MN and the HA. The Mobile IPv6
   service MAY use any ephemeral port number as the UDP source port, and
   MUST use the Mobile IPv6 service port number as the UDP destination
   port. The Mobile IPv6 service port is either dynamically assigned to

the MN during the bootstrapping phase (i.e. the mip6-port parameter) or in absence of the bootstrapping parameter the IANA reserved port (HALTSEC) MUST be used.

The encapsulating UDP header is immediately followed by a 4-bit Packet Type (PType) field that defines whether the packet contains an encrypted mobility management message or a, an encrypted user data packet, or a plain text user data packet.

The Packet Type field is followed by a 28-bit SPI value, which identifies the correct SA concerning the encrypted packet. For any packet that is neither integrity protected nor encrypted (i.e. no SA is applied by the originator) the SPI MUST be set to 0 (zero). Mobility management messages MUST always be at least integrity protected. Hence, mobility management messages MUST NOT be sent with a SPI value of 0 (zero).

There is always only one SPI per MN-HA mobility session and the same SPI is used for all types of protected packets independent of the direction.

The SPI value is followed by a 32-bit Sequence Number value that is used to identify retransmissions of encrypted messages. Each endpoint in the security association maintains two "current" Sequence Numbers: the next one to be used for a packet it initiates and the next one it expects to see in a packet from the other end. If the MN and the HA ends initiate very different numbers of messages, the Sequence Numbers in the two directions can be very different. In a case encryption is not used, the Sequence Number MUST be set to 0 (zero). Note that the HA SHOULD initiate a re-establishement of the SA before any of the Sequence Number cycle.

Finally, the Sequence Number field is followed by the data portion, whose content is identified by the Packet Type. The data portion may be protected.

6.2. PType and Security Parameter Index

The PType is a 4-bit field that indicates the Packet Type (PType) of the UDP encapsulated packet. The PType is followed by a a 28-bit SPI value. The PType and the SPI fields are treated as one 32-bit field during the integrity protection calculation.

```
   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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | PType |                      SPI                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
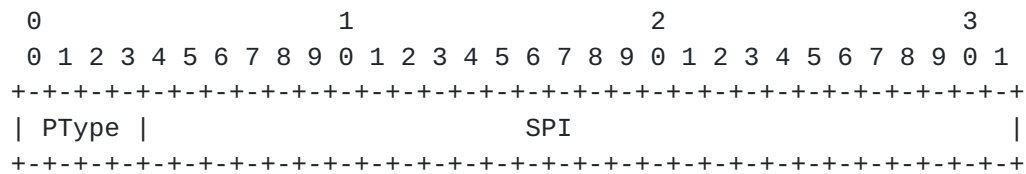
           Figure 6: Security Parameter Index with Packet Type

   A SPI value of 0 (zero) indicates a plaintext packet. If the packet
   is integrity protected or both integrity protected and encrypted, the
   SPI value MUST be different from 0. When the SPI value is set to 0,
   then the PType MUST also be 0.

6.3. Binding Management Message Formats

   The binding management messages that are only meant to be exchanged
   between the MN and the HA MUST be integrity protected and MAY be
   encrypted. They MUST use the packet format shown in Figure 7. All
   packets that are specific to the Mobile IPv6 protocol and contain a
   Mobility Header (as defined in Section 6.1.1. of RFC 6275) SHOULD use
   the packet format shown in Figure 7. (This means that some Mobile
   IPv6 mobility management messages, such as the HoTI message, as
   treated as data packets and using encapsulation described in
   Section 6.4).

```
      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
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      :           IPv4 or IPv6 header (src-addr=Xa, dst-addr=Ya)      :
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      :             UDP header (src-port=Xp,dst-port=Yp)             :
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ------
      |PType=8|                    SPI                        |  ^Int.
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
      |                    Sequence Number                            | |ered
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | ----
      |                  Payload Data* (variable)                     | |   ^
      :                                                               : |   |
      |                                                               | |Conf.
      +               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
      |               |     Padding (0-255 bytes)                      | |ered*
      +-+-+-+-+-+-+-+-+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |   |
      |                               | Pad Length   | Next Header   | v   v
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ------
      |          Integrity Check Value-ICV   (variable)               |
      :                                                               :
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

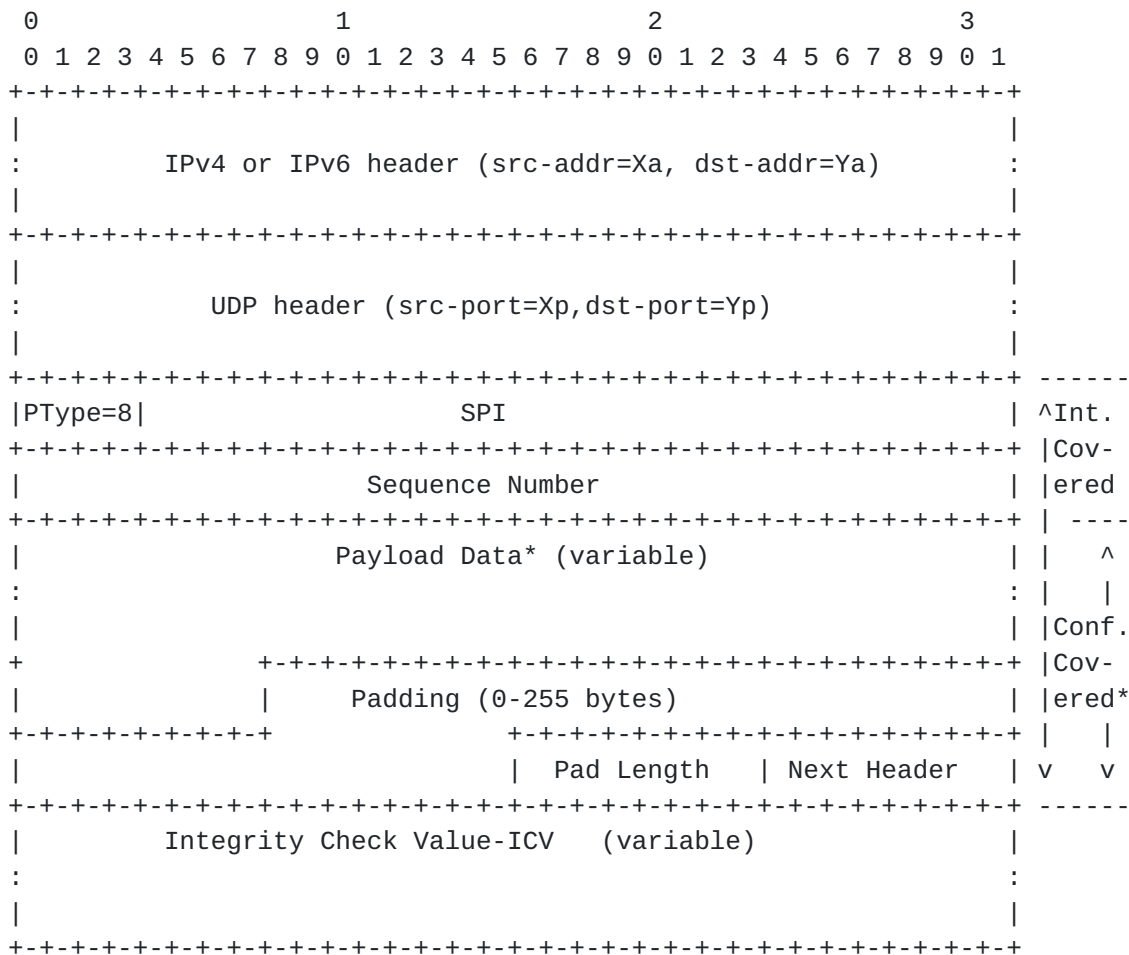   Figure 7: UDP Encapsulated Binding Management Message Format

   The PType value 8 (eight) identifies that the UDP encapsulated packet
   contains a RFC 6275 defined Mobility Header and other relevant IPv6
   extension headers. Note, there is no additional IP header inside the
   encapsulated part. The Next Header field MUST be set to value of the
   first encapsulated header. The encapsulated headers follow the
   natural IPv6 and Mobile IPv6 extension header alignment and
   formatting rules.

   The Padding, Pad Length, Next Header and ICV fields follow the rules
   of Section 2.4 to 2.8 of [RFC4303] unless otherwise stated in this
   document. For a SPI value of 0 (zero) that indicates an unprotected
   packet, the Padding, Pad Length, Next Header and ICV fields MUST NOT
   be present.

   The source and destination IP addresses of the outer IP header (i.e.
   the src-addr and the dst-addr in Figure 7) use the current care-of
   address of the MN and the HA address.

6.4. Reverse Tunneled User Data Packet Formats

   There are two types of reverse tunneled user data packets between the
   MN and the HA. Those that are integrity protected and encrypted and
   those that are plaintext. The MN or the HA decide whether to apply
   integrity protection and encryption to a packet or to send it in
   plaintext based on the mip6-sas value in the SA. If the mip6-sas is
   set to 1 the originator MUST NOT send any plaintext packet, and the
   receiver MUST silently discard any packet with the PType set to 0
   (unprotected). It is RECOMMENDED to apply confidentiality and
   integrity protection of user data traffic. The reverse tunneled IPv4
   or IPv6 user data packets are encapsulated as-is inside the 'Payload
   Data' shown in Figure 8. and Figure 9.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
:          IPv4 or IPv6 header (src-addr=Xa, dst-addr=Ya)       :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
:              UDP header (src-port=Xp,dst-port=Yp)             :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PType=1|                   SPI                         | ^Int.
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
|                    Sequence Number                     | |ered
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | ----
|                    Payload Data* (variable)            | |   ^
:                                                         : |   |
|                                                         | |Conf.
+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
|               |       Padding (0-255 bytes)             | |ered*
+-+-+-+-+-+-+-+-+               +-+-+-+-+-+-+-+-+-+-+-+-+-+ |   |
|                               | Pad Length   | Next Header  | v   v
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ------
|          Integrity Check Value-ICV   (variable)         |
:                                                         :
|                                                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
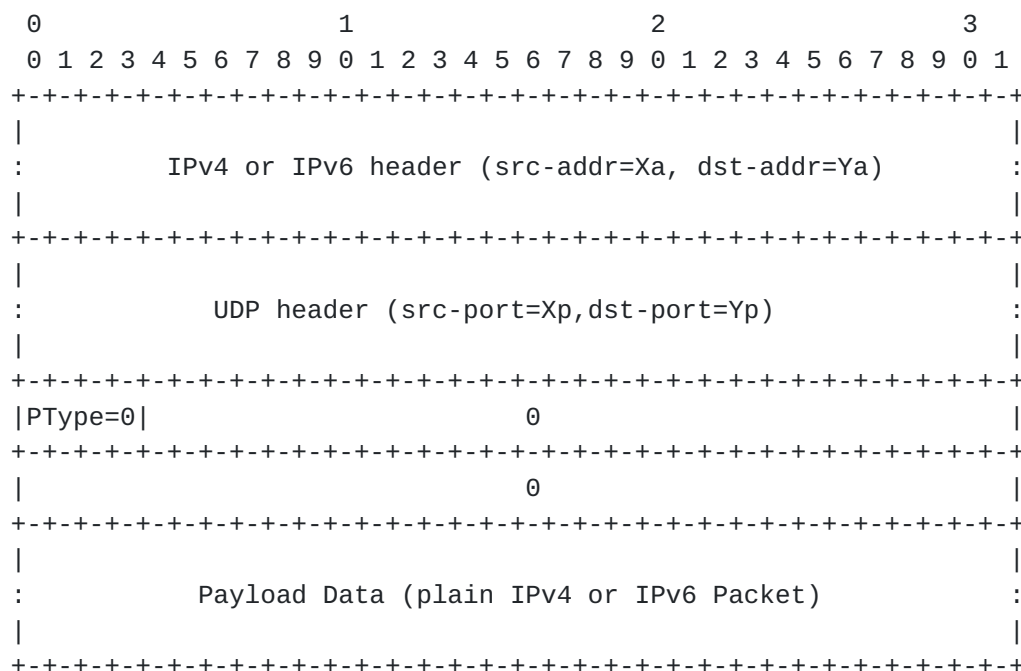
          Figure 8: UDP Encapsulated Protected User Data Packet Format

   The PType value 1 (one) identifies that the UDP encapsulated packet
   contains an encrypted tunneled IPv4/IPv6 user data packet. The Next
   Header field header MUST be set to value corresponding the tunneled
   IP packet (e.g., 41 for IPv6).

The Padding, Pad Length, Next Header and ICV fields follow the rules
of Section 2.4 to 2.8 of [RFC4303] unless otherwise stated in this
document. For a SPI value of 0 (zero) that indicates an unprotected
packet, the Padding, Pad Length, Next Header and ICV fields MUST NOT
be present.

The source and destination IP addresses of the outer IP header (i.e.,
the src-addr and the dst-addr in Figure 8) use the current care-of
address of the MN and the HA address. The ESP protected inner IP
header, which is not shown in Figure 8, uses the home address of the
MN and the correspondent node (CN) address.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
:            IPv4 or IPv6 header (src-addr=Xa, dst-addr=Ya)     :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
:              UDP header (src-port=Xp,dst-port=Yp)             :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PType=0|                     0                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              0                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
:            Payload Data (plain IPv4 or IPv6 Packet)           :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   Figure 9: UDP Encapsulated Non-Protected User Data Packet Format

The PType value 0 (zero) identifies that the UDP encapsulated packet
contains a plaintext tunneled IPv4/IPv6 user data packet. Also the
SPI and the Sequence Number fields MUST be set to 0 (zero).

The source and destination IP addresses of the outer IP header (i.e.,
the src-addr and the dst-addr in Figure 9) use the current care-of
address of the MN and the HA address. The plain text inner IP header
uses the home address of the MN and the CN address.

7. Route Optimization

   The MN-CN route optimization protocol functionality, related messages

and mobility options are out of scope of this specification. A
future specification may add route optimization functionality to the
Transport Layer Security-based Mobile IPv6 protocol.

We discuss few possible route optimization approaches in the
following sections. The route optimization could be done in two
ways: 1) using the return routability procedure described in
[RFC6275] or 2) directly between the MN and the CN i.e. enhancing the
route optimization also for Home Agent-less operation.

Both discussed solution approaches share the same tunneling
considerations between the MN and the CN. When the MN sends UDP
encapsulated packets to the CN, those are destined to CN's CoA. That
implies, the inner tunneled packets would also have the same
destination address as the outer tunneling packets. There should not
be issues regarding the decapsulation and encapsulation as long as
the inner tunneled packet does not have UDP payload with the same
destination port that the CN uses for its MN-CN UDP tunnel.

7.1. Replicating RFC6275 Route Optimization

Obviously [RFC6275] approach has already been verified from the
security considerations and procedures point of view. The existing
protocol proves both the Home Address ownership and verifies the
reachability. However, there are several gaps in scope of this
specification (TLS-based solution). For instance, the binding
management message encapsulation between the MN and the CN must be
specified and how to reach a CN using an IPv4 address. Following the
trend in this specification that would use UDP encapsulated IPv6 and
mobility headers as in Section 6.3. Another gap is the lack of SA
between the MN and the CN, if the communication were to be protected.
In order to enable the protection the CN should actually implement a
HAC function, which would then allow the MN and the CN to negotiate
required ciphersuites.

7.2. Enhanced Route Optimization with Home Agent-less Operation

If the CN were to implement a HAC, then the HAC could also
authenticate the MN. Possible authentication methods include pre-
shared secrets, certificates or using EAP+AAA against MN's home realm
AAA server (assuming the AAA infrastructure is in place). That
approach could actually allow the MN and the CN to setup secured
communication without doing the return routability test and support
Home Agent-less operation of MNs. However, the prerequisite is that
the MN has at some point of time bootstrapped with its HA e.g. to
acquire the HoA.

The "bootstrapping" between the MN and the "CN HAC" would concern a

subset of parameter needed between the MN and the "HA HAC". For
example, there is no need to negotiate Home Addresses and such. The
main use would be establishing the SA and authenticating at least the
MN to the CN.

8. IANA Considerations

8.1. New Registry: Packet Type

IANA is requested to create a new registry for the Packet Type as
described in Section 6.1.

```
Packet Type                      | Value
---------------------------------+---------------------------------
non-encrypted IP packet          | 0
encrypted IP packet              | 1
mobility header                  | 8
```

Following the allocation policies from [RFC5226] new values for the
Packet Type AVP MUST be assigned based on the "RFC Required" policy.

8.2. Status Codes

A new Status Code (to be used in BA messages) is reserved for the
cases where the HA wants to indicate to the MN that it needs to re-
establish the SA information with the HAC. The Result Code is
reserved from the 0-127 code space:

```
    REINIT_SA_WITH_HAC        TBD1
```

8.3. Port Numbers

A new port number (HALTSEC) for UDP packets is reserved from the PORT
NUMBERS registry.

```
    HALTSEC                   TBD2
```


9. Security Considerations

This document describes and uses a number of building blocks that
introduce security mechanisms and need to inter-work in a secure
manner.

The following building blocks are considered from a security point of
view:

1. Discovery of the HAC
2. Authentication and MN-HA SA establishment executed between the MN
    and the HAC (PSK or EAP-based) through a TLS tunnel
3. Protection of MN-HA communication
4. AAA Interworking

9.1. Discovery of the HAC

No dynamic procedure for discovering the HAC by the MN is described
in this document. As such, no specific security considerations apply
to the scope of this document.

9.2. Authentication and Key Exchange executed between the MN and the HAC

This document describes a simple authentication and MN-HA SA
negotiation exchange over TLS. The TLS procedures remain unchanged;
however, channel binding is provided.

Authentication: Server-side certificate based authentication MUST be
    performed using TLS 1.2 [RFC5246].

    The client-side authentication may depend on the specific
    deployment and is therefore not mandated. Note that TLS-PSK
    [RFC4279] cannot be used in conjunction with the methods described
    in section 5.8 and 5.9 of this document due to the limitations of
    the channel binding type used.

    Through the protected TLS tunnel, an additional authentication
    exchange is performed that provides client-side or mutual
    authentication and exchanges SA parameters and optional
    configuration data to be used in the subsequent protection of
    MN-HA communication. The additional authentication exchange can
    either be PSK-based (section 5.8) or EAP-based (section 5.9).
    Both exchanges are always performed within the protected TLS
    tunnel and MUST NOT be used as standalone protocols.

    The simple PSK-based authentication exchange provides mutual
    authentication and follows the GPSK exchange used by EAP-GPSK
    [RFC5433] and has similar properties, although some features of
    GPSK like the exchange of a protected container are not supported.

    The EAP-based authentication exchange simply defines message
    containers to allow carrying the EAP packets between the MN and
    the HAC. In principle, any EAP method can be used. However, it
    is strongly recommended to use only EAP methods that provide
    mutual authentication and that derive keys including an MSK key in
    compliance with [RFC3748].

Both exchanges use channel binding with the TLS tunnel. The
channel binding type 'TLS-server-endpoint' as per [RFC5929] MUST
be used.

Dictionary Attacks: All messages of the authentication exchanges
specified in this document are protected by TLS. However, any
implementation SHOULD assume that the properties of the
authentication exchange are the same as for GPSK [RFC5433] in case
the PSK-based method as per section 5.8. is used, and are the same
as those of the underlying EAP method in case the EAP-based
exchange as per section 5.9 is used.

Replay Protection: The underlying TLS protection provides protection
against replays.

Key Derivation and Key Strength: For TLS, the TLS specific
considerations apply unchanged. For the authentication exchanges
defined in this document, no key derivation step is performed as
the MN-HA keys are generated by the HAC and are distributed to the
MN through the secure TLS connection.

Key Control: No joint key control for MN-HA keys is provided by this
version of the specification.

Lifetime: The TLS-protected authentication exchange between the MN
and the HAC is only to bootstrap keys and other parameters for
usage with MN-HA security. The SAs that contain the keys have an
associated lifetime. The usage of Transport Layer Security (TLS)
Session Resumption without Server-Side State, described in
[RFC5077], provides the ability for the MN to minimize the latency
of future exchanges towards the HA without having to keep state at
the HA itself.

Denial of Service Resistance: The level of resistance against denial
of service attacks SHOULD be considered the same as for common TLS
operation, as TLS is used unchanged. For the PSK-based
authentication exchange, no additional factors are known. For the
EAP-based authentication exchange, any considerations regarding
denial-of-service resistance specific to the chosen EAP method are
expected to be applicable and need to be be taken into account.

Session Independence: Each individual TLS protocol run is
independent from any previous exchange based on the security
properties of the TLS handshake protocol. However, several PSK or
EAP-based authentication exchanges can be performed across the
same TLS connection.

Fragmentation: TLS runs on top of TCP and no fragmentation specific
   considerations apply to the MN-HAC authentication exchanges.

Channel Binding: Both the PSK and the EAP-based exchanges use
   channel binding with the TLS tunnel. The channel binding type
   'TLS-server-endpoint' as per [RFC5929] MUST be used.

Fast Reconnect: This protocol provides session resumption as part of
   TLS and optionally the support for [RFC5077]. No fast reconnect
   is supported for the PSK-based authentication exchange. For the
   EAP-based authentication exchange, availability of fast reconnect
   depends on the EAP method used.

Identity Protection: Based on the security properties of the TLS
   tunnel, passive user identity protection is provided. An attacker
   acting as man-in-the-middle in the TLS connection would be able to
   observe the MN identity value sent in MHAuth-Init messages.

Protected Ciphersuite Negotiation: This protocol provides
   ciphersuite negotiation based on TLS.

Confidentiality: Confidentiality protection of payloads exchanged
   between the MN and the HAC are protected with the TLS Record
   Layer. TLS ciphersuites with confidentiality and integrity
   protection MUST be negotiated and used in order to exchange
   security sensitive material inside the TLS connection.

Cryptographic Binding: No cryptographic bindings are provided by
   this protocol specified in this document.

Perfect Forward Secrecy: Perfect forward secrecy is provided with
   appropriate TLS ciphersuites.

Key confirmation: Key confirmation of the keys established with TLS
   is provided by the TLS Record Layer when the keys are used to
   protect the subsequent TLS exchange.

9.3. Protection of MN and HA Communication

Authentication: Data origin authentication is provided for the
   communication between the MN and the HA. The chosen level of
   security of this authentication depends on the selected
   ciphersuite. Entity authentication is offered by the MN to HAC
   protocol exchange.

Dictionary Attacks: The concept of dictionary attacks is not
   applicable to the MN-HA communication as the keying material used
   for this communication is randomly created by the HAC and its
   length depends on the chosen cryptographic algorithms.

Replay Protection: Replay protection for the communication between
   the MN and the HA is provided based on sequence numbers and
   follows the design of IPsec ESP.

Key Derivation and Key Strength: The strength of the keying material
   established for the communication between the MN and the HA is
   selected based on the negotiated ciphersuite (based on the MN-HAC
   exchange) and the key created by the HAC. The randomness
   requirements for security described in RFC 4086 [RFC4086] are
   applicable to the key generation by the HAC.

Key Control: The keying material established during the MN-HAC
   protocol exchange for subsequent protection of the MN-HA
   communication is created by the HA and therefore no joint key
   control is provided for it.

Key Naming: For the MN-HA communication the security associations
   are indexed with the help of the SPI and additionally based on the
   direction (in-bound communication or out-bound communication).

Lifetime: The lifetime of the MN-HA security associations is based
   on the value in the mip6-sa-validity-end header field exchanged
   during the MN-HAC exchange. The HAC controls the SA lifetime.

Denial of Service Resistance: For the communication between the MN
   and the HA there are no heavy cryptographic operations (such as
   public key computations). As such, there are no DoS concerns.

Session Independence: Sessions are independent from each other when
   new keys are created by via the MN-HAC protocol. A new MN-HAC
   protocol run produces fresh and unique keying material for
   protection of the MN-HA communication.

Fragmentation: There is no additional fragmentation support provided
   beyond what is offered by the network layer.

Channel Binding: Channel binding is not applicable to the MN-HA
   communication.

   Fast Reconnect: The concept of fast reconnect is not applicable to
      the MN-HA communication.

   Identity Protection: User identities SHOULD NOT be exchanged between
      the MN and the HA. In a case binding management messages contain
      the user identity, the messages SHOULD be confidentity protected.

   Protected Ciphersuite Negotiation: The MN-HAC protocol provides
      protected ciphersuite negotiation through a secure TLS connection.

   Confidentiality: Confidentiality protection of payloads exchanged
      between the MN and the HAC (for Mobile IPv6 signaling and
      optionally for the data traffic) is provided utilizing algorithms
      negotiated during the MN-HAC exchange.

   Cryptographic Binding: No cryptographic bindings are provided by
      this protocol specified in this document.

   Perfect Forward Secrecy: Perfect forward secrecy is provided when
      the MN bootstraps new keying material with the help of the MN-HAC
      protocol (assuming that a proper TLS ciphersuite is used).

   Key confirmation: Key confirmation of the MN-HA keying material
      conveyed from the HAC to the MN is provided when the first packets
      are exchanged between the MN and the HA (in both directions as two
      different keys are used).

9.4. AAA Interworking

   The AAA backend infrastructure interworking is not defined in this
   document and therefore out-of-scope.

10. Acknowledgements

   The authors would like to thank Pasi Eronen, Domagoj Premec, Julien
   Laganier and Christian Bauer for their comments.

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.

   [RFC2404] Madson, C. and R. Glenn, "The Use of HMAC-SHA-1-96 within
              ESP and AH", RFC 2404, November 1998.

   [RFC2410] Glenn, R. and S. Kent, "The NULL Encryption Algorithm and
              Its Use With IPsec", RFC 2410, November 1998.

   [RFC2451] Pereira, R. and R. Adams, "The ESP CBC-Mode Cipher
              Algorithms", RFC 2451, November 1998.

   [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H.,
              Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext
              Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

   [RFC3566] Frankel, S. and H. Herbert, "The AES-XCBC-MAC-96 Algorithm
              and Its Use With IPsec", RFC 3566, September 2003.

   [RFC3602] Frankel, S., Glenn, R., and S. Kelly, "The AES-CBC Cipher
              Algorithm and Its Use with IPsec", RFC 3602,
              September 2003.

   [RFC4282] Aboba, B., Beadles, M., Arkko, J., and P. Eronen, "The
              Network Access Identifier", RFC 4282, December 2005.

   [RFC5056] Williams, N., "On the Use of Channel Bindings to Secure
              Channels", RFC 5056, November 2007.

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

   [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security
              (TLS) Protocol Version 1.2", RFC 5246, August 2008.

   [RFC5929] Altman, J., Williams, N., and L. Zhu, "Channel Bindings
              for TLS", RFC 5929, July 2010.

   [RFC6275] Perkins, C., Johnson, D., and J. Arkko, "Mobility Support
              in IPv6", RFC 6275, July 2011.

11.2. Informative References

   [I-D.patil-mext-mip6issueswithipsec]
              Patil, B., Perkins, C., Tschofenig, H., and D. Premec,
              "Problems with the use of IPsec as the security protocol
              for Mobile IPv6", draft-patil-mext-mip6issueswithipsec-04
              (work in progress), October 2011.

   [RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768,
              August 1980.

   [RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H.

Levkowetz, "Extensible Authentication Protocol (EAP)",
RFC 3748, June 2004.

[RFC3776] Arkko, J., Devarapalli, V., and F. Dupont, "Using IPsec to
Protect Mobile IPv6 Signaling Between Mobile Nodes and
Home Agents", RFC 3776, June 2004.

[RFC4086] Eastlake, D., Schiller, J., and S. Crocker, "Randomness
Requirements for Security", BCP 106, RFC 4086, June 2005.

[RFC4279] Eronen, P. and H. Tschofenig, "Pre-Shared Key Ciphersuites
for Transport Layer Security (TLS)", RFC 4279,
December 2005.

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

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

[RFC4877] Devarapalli, V. and F. Dupont, "Mobile IPv6 Operation with
IKEv2 and the Revised IPsec Architecture", RFC 4877,
April 2007.

[RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig,
"Transport Layer Security (TLS) Session Resumption without
Server-Side State", RFC 5077, January 2008.

[RFC5433] Clancy, T. and H. Tschofenig, "Extensible Authentication
Protocol - Generalized Pre-Shared Key (EAP-GPSK) Method",
RFC 5433, February 2009.

[RFC5555] Soliman, H., "Mobile IPv6 Support for Dual Stack Hosts and
Routers", RFC 5555, June 2009.

[RFC5944] Perkins, C., "IP Mobility Support for IPv4, Revised",
RFC 5944, November 2010.

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

Authors' Addresses

    Jouni Korhonen (editor)
    Nokia Siemens Networks
    Linnoitustie 6
    Espoo FIN-02600
    Finland

    Email: jouni.nospam@gmail.com

    Basavaraj Patil
    Nokia
    6021 Connection Drive
    Irving, TX 75039
    USA

    Email: basavaraj.patil@nokia.com

    Hannes Tschofenig
    Nokia Siemens Networks
    Linnoitustie 6
    Espoo 02600
    Finland

    Phone: +358 (50) 4871445
    Email: Hannes.Tschofenig@gmx.net

    Dirk Kroeselberg
    Siemens

    Email: dirk.kroeselberg@siemens.com