          EAP Extensions for EAP Re-authentication Protocol (ERP)

Status of This Memo

Abstract

   The Extensible Authentication Protocol (EAP) is a generic framework
   supporting multiple types of authentication methods. In systems
   where EAP is used for authentication, it is desirable to not repeat
   the entire EAP exchange with another authenticator. This document
   specifies extensions to EAP and the EAP keying hierarchy to support
   an EAP method-independent protocol for efficient re-authentication
   between the peer and an EAP re-authentication server through any
   authenticator. The re-authentication server may be in the home
   network or in the local network to which the peer is connecting.

Table of Contents

1. Introduction

   The Extensible Authentication Protocol (EAP) is a an authentication
   framework that supports multiple authentication methods. The primary
   purpose is network access authentication, and a key-generating method
   is used when the lower layer wants to enforce access control. The
   EAP keying hierarchy defines two keys to be derived by all key-
   generating EAP methods: the Master Session Key (MSK) and the Extended
   MSK (EMSK). In the most common deployment scenario, an EAP peer and
   an EAP server authenticate each other through a third party known as
   the EAP authenticator. The EAP authenticator or an entity controlled
   by the EAP authenticator enforces access control. After successful
   authentication, the EAP server transports the MSK to the EAP
   authenticator; the EAP authenticator and the EAP peer establish
   transient session keys (TSKs) using the MSK as the authentication
   key, key derivation key, or a key transport key, and use the TSK for
   per-packet access enforcement.

   When a peer moves from one authenticator to another, it is desirable
   to avoid a full EAP authentication to support fast handovers. The
   full EAP exchange with another run of the EAP method can take several
   round trips and significant time to complete, causing delays in
   handover times. Some EAP methods specify the use of state from the
   initial authentication to optimize re-authentications by reducing the
   computational overhead, but method-specific re-authentication takes
   at least 2 round trips with the original EAP server in most cases
   (e.g., [6]). It is also important to note that several methods do
   not offer support for re-authentication.

   Key sharing across authenticators is sometimes used as a practical
   solution to lower handover times. In that case, compromise of an
   authenticator results in compromise of keying material established
   via other authenticators. Other solutions for fast re-authentication
   exist in the literature [7] [8].

   In conclusion, to achieve low latency handovers, there is a need for
   a method-independent re-authentication protocol that completes in
   less than 2 round trips, preferably with a local server. The EAP
   re-authentication problem statement is described in detail in [9].

   This document specifies EAP Re-authentication Extensions (ERXs) for
   efficient re-authentication using EAP. The protocol that uses these
   extensions itself is referred to as the EAP Re-authentication
   Protocol (ERP). It supports EAP method-independent re-authentication
   for a peer that has valid, unexpired key material from a previously
   performed EAP authentication. The protocol and the key hierarchy
   required for EAP re-authentication are described in this document.

Note that to support ERP, lower-layer specifications may need to be
revised to allow carrying EAP messages that have a code value higher
than 4 and to accommodate the peer-initiated nature of ERP.
Specifically, the IEEE802.1x specification must be revised and RFC
4306 must be updated to carry ERP messages.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in RFC 2119 [1].

This document uses the basic EAP terminology [2] and EMSK keying
hierarchy terminology [3]. In addition, this document uses the
following terms:

   ER Peer - An EAP peer that supports the EAP Re-authentication
   Protocol. All references to "peer" in this document imply an ER
   peer, unless specifically noted otherwise.

   ER Authenticator - An entity that supports the authenticator
   functionality for EAP re-authentication described in this
   document. All references to "authenticator" in this document
   imply an ER authenticator, unless specifically noted otherwise.

   ER Server - An entity that performs the server portion of ERP
   described here. This entity may or may not be an EAP server. All
   references to "server" in this document imply an ER server, unless
   specifically noted otherwise. An ER server is a logical entity;
   the home ER server is located on the same backend authentication
   server as the EAP server in the home domain. The local ER server
   may not necessarily be a full EAP server.

   ERX - EAP re-authentication extensions.

   ERP - EAP Re-authentication Protocol that uses the
   re-authentication extensions.

   rRK - re-authentication Root Key, derived from the EMSK or DSRK.

   rIK - re-authentication Integrity Key, derived from the rRK.

   rMSK - re-authentication MSK. This is a per-authenticator key,
   derived from the rRK.

keyName-NAI - ERP messages are integrity protected with the rIK or
the DS-rIK. The use of rIK or DS-rIK for integrity protection of
ERP messages is indicated by the EMSKname [3]; the protocol, which
is ERP; and the realm, which indicates the domain name of the ER
server. The EMSKname is copied into the username part of the NAI.

Domain - Refers to a "key management domain" as defined in [3].
For simplicity, it is referred to as "domain" in this document.
The terms "home domain" and "local domain" are used to
differentiate between the originating key management domain that
performs the full EAP exchange with the peer and the local domain
to which a peer may be attached at a given time.

3. ERP Description

ERP allows a peer and server to mutually verify proof of possession
of keying material from an earlier EAP method run and to establish a
security association between the peer and the authenticator. The
authenticator acts as a pass-through entity for the Re-authentication
Protocol in a manner similar to that of an EAP authenticator
described in RFC 3748 [2]. ERP is a single round-trip exchange
between the peer and the server; it is independent of the lower layer
and the EAP method used during the full EAP exchange. The ER server
may be in the home domain or in the same (visited) domain as the peer
and the authenticator.

Figure 2 shows the protocol exchange. The first time the peer
attaches to any network, it performs a full EAP exchange (shown in
Figure 1) with the EAP server; as a result, an MSK is distributed to
the EAP authenticator. The MSK is then used by the authenticator and
the peer to establish TSKs as needed. At the time of the initial EAP
exchange, the peer and the server also derive an EMSK, which is used
to derive a re-authentication Root Key (rRK). More precisely, a
re-authentication Root Key is derived from the EMSK or from a
Domain-Specific Root Key (DSRK), which itself is derived from the
EMSK. The rRK is only available to the peer and the ER server and is
never handed out to any other entity. Further, a re-authentication
Integrity Key (rIK) is derived from the rRK; the peer and the ER
server use the rIK to provide proof of possession while performing an
ERP exchange. The rIK is also never handed out to any entity and is
only available to the peer and server.

When the ER server is in the home domain, the peer and the server use
the rIK and rRK derived from the EMSK; and when the ER server is not
in the home domain, they use the DS-rIK and DS-rRK corresponding to
the local domain. The domain of the ER server is identified by the
realm portion of the keyname-NAI in ERP messages.

3.1. ERP With the Home ER Server

```
  EAP Peer              EAP Authenticator            EAP Server
  =======               =================            ==========

   <--- EAP-Request/ ------
         Identity

   ----- EAP Response/ --->
         Identity          ---AAA(EAP Response/Identity)-->

   <--- EAP Method ------->  <------ AAA(EAP Method -------->
         exchange                      exchange)

                            <----AAA(MSK, EAP-Success)------

   <---EAP-Success---------

                     Figure 1: EAP Authentication


  Peer                  Authenticator                 Server
  ====                  =============                  ======

  [<-- EAP-Initiate/ -----
      Re-auth-Start]
  [<-- EAP-Request/ ------
      Identity]


  ---- EAP-Initiate/ ----> ----AAA(EAP-Initiate/ ---------->
        Re-auth/                   Re-auth/
       [Bootstrap]              [Bootstrap])

   <--- EAP-Finish/ ------> <---AAA(rMSK,EAP-Finish/---------
        Re-auth/                   Re-auth/
       [Bootstrap]              [Bootstrap])

  Note: [] brackets indicate optionality.

                     Figure 2: ERP Exchange
```

   Two new EAP codes, EAP-Initiate and EAP-Finish, are specified in this
   document for the purpose of EAP re-authentication. When the peer
   identifies a target authenticator that supports EAP
   re-authentication, it performs an ERP exchange, as shown in Figure 2;
   the exchange itself may happen when the peer attaches to a new
   authenticator supporting EAP re-authentication, or prior to

attachment. The peer initiates ERP by itself; it may also do so in
response to an EAP-Initiate/Re-auth-Start message from the new
authenticator. The EAP-Initiate/Re-auth-Start message allows the
authenticator to trigger the ERP exchange.

It is plausible that the authenticator does not know whether the peer
supports ERP and whether the peer has performed a full EAP
authentication through another authenticator. The authenticator MAY
initiate the ERP exchange by sending the EAP-Initiate/Re-auth-Start
message, and if there is no response, it will send the EAP-Request/
Identity message. Note that this avoids having two EAP messages in
flight at the same time [2]. The authenticator may send the EAP-
Initiate/Re-auth-Start message and wait for a short, locally
configured amount of time. If the peer does not already know, this
message indicates to the peer that the authenticator supports ERP.
In response to this trigger from the authenticator, the peer can
initiate the ERP exchange by sending an EAP-Initiate/Re-auth message.
If there is no response from the peer after the necessary
retransmissions (see Section 6), the authenticator MUST initiate EAP
by sending an EAP-Request message, typically the EAP-Request/Identity
message. Note that the authenticator may receive an EAP-Initiate/
Re-auth message after it has sent an EAP-Request/Identity message.
If the authenticator supports ERP, it MUST proceed with the ERP
exchange. When the EAP-Request/Identity times out, the authenticator
MUST NOT close the connection if an ERP exchange is in progress or
has already succeeded in establishing a re-authentication MSK.

If the authenticator does not support ERP, it drops EAP-Initiate/
Re-auth messages [2] as the EAP code of those packets is greater than
4. An ERP-capable peer will exhaust the EAP-Initiate/Re-auth message
retransmissions and fall back to EAP authentication by responding to
EAP Request/Identity messages from the authenticator. If the peer
does not support ERP or if it does not have unexpired key material
from a previous EAP authentication, it drops EAP-Initiate/
Re-auth-Start messages. If there is no response to the EAP-Initiate/
Re-auth-Start message, the authenticator SHALL send an EAP Request
message (typically EAP Request/Identity) to start EAP authentication.
From this stage onwards, RFC 3748 rules apply. Note that this may
introduce some delay in starting EAP. In some lower layers, the
delay can be minimized or even avoided by the peer initiating EAP by
sending messages such as EAPoL-Start in the IEEE 802.1X specification
[10].

The peer sends an EAP-Initiate/Re-auth message that contains the
keyName-NAI to identify the ER server's domain and the rIK used to
protect the message, and a sequence number for replay protection.
The EAP-Initiate/Re-auth message is integrity protected with the rIK.
The authenticator uses the realm in the keyName-NAI [4] field to send

the message to the appropriate ER server. The server uses the
keyName to look up the rIK. The server, after verifying proof of
possession of the rIK, and freshness of the message, derives a
re-authentication MSK (rMSK) from the rRK using the sequence number
as an input to the key derivation. The server updates the expected
sequence number to the received sequence number plus one.

In response to the EAP-Initiate/Re-auth message, the server sends an
EAP-Finish/Re-auth message; this message is integrity protected with
the rIK. The server transports the rMSK along with this message to
the authenticator. The rMSK is transported in a manner similar to
that of the MSK along with the EAP-Success message in a full EAP
exchange. Ongoing work in [11] describes an additional key
distribution protocol that can be used to transport the rRK from an
EAP server to one of many different ER servers that share a trust
relationship with the EAP server.

The peer MAY request the server for the rMSK lifetime. If so, the ER
server sends the rMSK lifetime in the EAP-Finish/Re-auth message.

In an ERP bootstrap exchange, the peer MAY request the server for the
rRK lifetime. If so, the ER server sends the rRK lifetime in the
EAP-Finish/Re-auth message.

The peer verifies the replay protection and the integrity of the
message. It then uses the sequence number in the EAP-Finish/Re-auth
message to compute the rMSK. The lower-layer security association
protocol is ready to be triggered after this point.

3.2. ERP with a Local ER Server

The defined ER extensions allow executing the ERP with an ER server
in the local domain (access network). The local ER server may be co-
located with a local AAA server. The peer may learn about the
presence of a local ER server in the network and the local domain
name (or ER server name) either via the lower layer or by means of
ERP bootstrapping. The peer uses the domain name and the EMSK to
compute the DSRK and from that key, the DS-rRK; the peer also uses
the domain name in the realm portion of the keyName-NAI for using ERP
in the local domain. Figure 3 shows the full EAP and subsequent
local ERP exchange; Figure 4 shows it with a local ER server.

```
   Peer          EAP Authenticator    Local ER Server     Home EAP Server
   ====          =================    ===============     ===============

   <-- EAP-Request/ --
       Identity

   -- EAP Response/-->
       Identity        --AAA(EAP Response/-->
                             Identity)        --AAA(EAP Response/ -->
                                                           Identity,
                                                      [DSRK Request,
                                                    domain name])


   <---------------------- EAP Method exchange------------------>

                                              <---AAA(MSK, DSRK, ----
                                                    EMSKname,
                                                  EAP-Success)

                      <---  AAA(MSK,  -----
                            EAP-Success)

   <---EAP-Success-----

           Figure 3: Local ERP Exchange, Initial EAP Exchange


   Peer                 ER Authenticator            Local ER Server
   ====                 ================            ===============

   [<-- EAP-Initiate/ --------
        Re-auth-Start]
   [<-- EAP-Request/ ---------
        Identity]

    ---- EAP-Initiate/ ------->  ----AAA(EAP-Initiate/ -------->
         Re-auth                          Re-auth)

    <--- EAP-Finish/ ----------  <---AAA(rMSK,EAP-Finish/-------
         Re-auth                          Re-auth)

                    Figure 4: Local ERP Exchange
```

As shown in Figure 4, the local ER server may be present in the path
of the full EAP exchange (e.g., this may be one of the AAA entities,
such as AAA proxies, in the path between the authenticator and the
home EAP server of the peer). In that case, the ER server requests
the DSRK by sending the domain name to the EAP server. In response,
the EAP server computes the DSRK by following the procedure specified
in [3] and sends the DSRK and the key name, EMSKname, to the ER
server in the claimed domain. The local domain is responsible for
announcing that same domain name via the lower layer to the peer.

If the peer does not know the domain name (did not receive the domain
name via the lower-layer announcement, due to a missed announcement
or lack of support for domain name announcements in a specific lower
layer), it SHOULD initiate ERP bootstrap exchange with the home ER
server to obtain the domain name. The local ER server SHALL request
the home AAA server for the DSRK by sending the domain name in the
AAA message that carries the EAP-Initiate/Re-auth bootstrap message.
The local ER server MUST be in the path from the peer to the home ER
server. If it is not, it cannot request the DSRK.

After receiving the DSRK and the EMSKname, the local ER server
computes the DS-rRK and the DS-rIK from the DSRK as defined in
Sections 4.1 and 4.3 below. After receiving the domain name, the
peer also derives the DSRK, the DS-rRK, and the DS-rIK. These keys
are referred to by a keyName-NAI formed as follows: the username part
of the NAI is the EMSKname, the realm portion of the NAI is the
domain name. Both parties also maintain a sequence number
(initialized to zero) corresponding to the specific keyName-NAI.

Subsequently, when the peer attaches to an authenticator within the
local domain, it may perform an ERP exchange with the local ER server
to obtain an rMSK for the new authenticator.

4. ER Key Hierarchy

Each time the peer re-authenticates to the network, the peer and the
authenticator establish an rMSK. The rMSK serves the same purposes
that an MSK, which is the result of full EAP authentication, serves.
To prove possession of the rRK, we specify the derivation of another
key, the rIK. These keys are derived from the rRK. Together they
constitute the ER key hierarchy.

The rRK is derived from either the EMSK or a DSRK as specified in
Section 4.1. For the purpose of rRK derivation, this document
specifies derivation of a Usage-Specific Root Key (USRK) or a Domain-
Specific USRK (DSUSRK) in accordance with [3] for re-authentication.
The USRK designated for re-authentication is the re-authentication
root key (rRK). A DSUSRK designated for re-authentication is the DS-

rRK available to a local ER server in a particular domain. For
simplicity, the keys are referred to without the DS label in the rest
of the document. However, the scope of the various keys is limited
to just the respective domains they are derived for, in the case of
the domain specific keys. Based on the ER server with which the peer
performs the ERP exchange, it knows the corresponding keys that must
be used.

The rRK is used to derive an rIK, and rMSKs for one or more
authenticators. The figure below shows the key hierarchy with the
rRK, rIK, and rMSKs.

```
                    rRK
                     |
          +--------+--------+
          |        |        |
         rIK     rMSK1 ...rMSKn
```
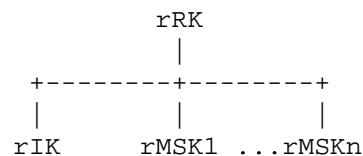
Figure 5: Re-authentication Key Hierarchy

The derivations in this document are according to [3]. Key
derivations and field encodings, where unspecified, default to that
document.

4.1. rRK Derivation

The rRK may be derived from the EMSK or DSRK. This section provides
the relevant key derivations for that purpose.

The rRK is derived as specified in [3].

rRK = KDF (K, S), where

   K = EMSK or K = DSRK and

   S = rRK Label | "\0" | length

The rRK Label is an IANA-assigned 8-bit ASCII string:

   EAP Re-authentication Root Key@ietf.org

assigned from the "USRK key labels" name space in accordance with
[3].

The KDF and algorithm agility for the KDF are as defined in [3].

An rRK derived from the DSRK is referred to as a DS-rRK in the rest
of the document. All the key derivation and properties specified in
this section remain the same.

4.2. rRK Properties

The rRK has the following properties. These properties apply to the
rRK regardless of the parent key used to derive it.

o The length of the rRK MUST be equal to the length of the parent
   key used to derive it.

o The rRK is to be used only as a root key for re-authentication and
   never used to directly protect any data.

o The rRK is only used for derivation of rIK and rMSK as specified
   in this document.

o The rRK MUST remain on the peer and the server that derived it and
   MUST NOT be transported to any other entity.

o The lifetime of the rRK is never greater than that of its parent
   key. The rRK is expired when the parent key expires and MUST be
   removed from use at that time.

4.3. rIK Derivation

The re-authentication Integrity Key (rIK) is used for integrity
protecting the ERP exchange. This serves as the proof of possession
of valid keying material from a previous full EAP exchange by the
peer to the server.

The rIK is derived as follows.

rIK = KDF (K, S), where

   K = rRK and

   S = rIK Label | "\0" | cryptosuite | length

The rIK Label is the 8-bit ASCII string:

   Re-authentication Integrity Key@ietf.org

The length field refers to the length of the rIK in octets encoded as
specified in [3].

The cryptosuite and length of the rIK are part of the input to the
key derivation function to ensure cryptographic separation of keys if
different rIKs of different lengths for use with different Message
Authentication Code (MAC) algorithms are derived from the same rRK.
The cryptosuite is encoded as an 8-bit number; see Section 5.3.2 for
the cryptosuite specification.

The rIK is referred to by EMSKname-NAI within the context of ERP
messages. The username part of EMSKname-NAI is the EMSKname; the
realm is the domain name of the ER server. In case of ERP with the
home ER server, the peer uses the realm from its original NAI; in
case of a local ER server, the peer uses the domain name received at
the lower layer or through an ERP bootstrapping exchange.

An rIK derived from a DS-rRK is referred to as a DS-rIK in the rest
of the document. All the key derivation and properties specified in
this section remain the same.

4.4. rIK Properties

The rIK has the following properties.

o The length of the rIK MUST be equal to the length of the rRK.

o The rIK is only used for authentication of the ERP exchange as
   specified in this document.

o The rIK MUST NOT be used to derive any other keys.

o The rIK must remain on the peer and the server and MUST NOT be
   transported to any other entity.

o The rIK is cryptographically separate from any other keys derived
   from the rRK.

o The lifetime of the rIK is never greater than that of its parent
   key. The rIK MUST be expired when the EMSK expires and MUST be
   removed from use at that time.

4.5. rIK Usage

The rIK is the key whose possession is demonstrated by the peer and
the ERP server to the other party. The peer demonstrates possession
of the rIK by computing the integrity checksum over the EAP-Initiate/
Re-auth message. When the peer uses the rIK for the first time, it
can choose the integrity algorithm to use with the rIK. The peer and
the server MUST use the same integrity algorithm with a given rIK for

all ERP messages protected with that key. The peer and the server
store the algorithm information after the first use, and they employ
the same algorithm for all subsequent uses of that rIK.

If the server's policy does not allow the use of the cryptosuite
selected by the peer, the server SHALL reject the EAP-Initiate/
Re-auth message and SHOULD send a list of acceptable cryptosuites in
the EAP-Finish/Re-auth message.

The rIK length may be different from the key length required by an
integrity algorithm. In case of hash-based MAC algorithms, the key
is first hashed to the required key length as specified in [5]. In
case of cipher-based MAC algorithms, if the required key length is
less than 32 octets, the rIK is hashed using HMAC-SHA256 and the
first k octets of the output are used, where k is the key length
required by the algorithm. If the required key length is more than
32 octets, the first k octets of the rIK are used by the cipher-based
MAC algorithm.

4.6. rMSK Derivation

The rMSK is derived at the peer and server and delivered to the
authenticator. The rMSK is derived following an EAP Re-auth Protocol
exchange.

The rMSK is derived as follows.

rMSK = KDF (K, S), where

    K = rRK and

    S = rMSK label | "\0" | SEQ | length

The rMSK label is the 8-bit ASCII string:

    Re-authentication Master Session Key@ietf.org

The length field refers to the length of the rMSK in octets. The
length field is encoded as specified in [3].

SEQ is the sequence number sent by the peer in the EAP-Initiate/
Re-auth message. This field is encoded as a 16-bit number in network
byte order (see Section 5.3.2).

An rMSK derived from a DS-rRK is referred to as a DS-rIK in the rest
of the document. All the key derivation and properties specified in
this section remain the same.

4.7. rMSK Properties

   The rMSK has the following properties:

   o The length of the rMSK MUST be equal to the length of the rRK.

   o The rMSK is delivered to the authenticator and is used for the
      same purposes that an MSK is used at an authenticator.

   o The rMSK is cryptographically separate from any other keys derived
      from the rRK.

   o The lifetime of the rMSK is less than or equal to that of the rRK.
      It MUST NOT be greater than the lifetime of the rRK.

   o If a new rRK is derived, subsequent rMSKs MUST be derived from the
      new rRK. Previously delivered rMSKs MAY still be used until the
      expiry of the lifetime.

   o A given rMSK MUST NOT be shared by multiple authenticators.

5. Protocol Details

5.1. ERP Bootstrapping

   We identify two types of bootstrapping for ERP: explicit and implicit
   bootstrapping. In implicit bootstrapping, the local ER server SHOULD
   include its domain name and SHOULD request the DSRK from the home AAA
   server during the initial EAP exchange, in the AAA message
   encapsulating the first EAP Response message sent by the peer. If
   the EAP exchange is successful, the server sends the DSRK for the
   local ER server (derived using the EMSK and the domain name as
   specified in [3]), EMSKname, and DSRK lifetime along with the EAP-
   Success message. The local ER server MUST extract the DSRK,
   EMSKname, and DSRK lifetime (if present) before forwarding the EAP-
   Success message to the peer, as specified in [12]. Note that the MSK
   (also present along with the EAP Success message) is extracted by the
   EAP authenticator as usual. The peer learns the domain name through
   the EAP-Initiate/Re-auth-Start message or via lower-layer
   announcements. When the domain name is available to the peer during
   or after the full EAP authentication, it attempts to use ERP when it
   associates with a new authenticator.

   If the peer does not know the domain name (did not receive the domain
   name via the lower-layer announcement, due to a missed announcement
   or lack of support for domain name announcements in a specific lower
   layer), it SHOULD initiate ERP bootstrap exchange (ERP exchange with
   the bootstrap flag turned on) with the home ER server to obtain the

domain name. The local ER server behavior is the same as described
above. The peer MAY also initiate bootstrapping to fetch information
such as the rRK lifetime from the AAA server.

The following steps describe the ERP explicit bootstrapping process:

o The peer sends the EAP-Initiate/Re-auth message with the
   bootstrapping flag turned on. The bootstrap message is always
   sent to the home AAA server, and the keyname-NAI attribute in the
   bootstrap message is constructed as follows: the username portion
   of the NAI contains the EMSKname, and the realm portion contains
   the home domain name.

o In addition, the message MUST contain a sequence number for replay
   protection, a cryptosuite, and an integrity checksum. The
   cryptosuite indicates the authentication algorithm. The integrity
   checksum indicates that the message originated at the claimed
   entity, the peer indicated by the Peer-ID, or the rIKname.

o The peer MAY additionally set the lifetime flag to request the key
   lifetimes.

o When an ERP-capable authenticator receives the EAP-Initiate/
   Re-auth message from a peer, it copies the contents of the
   keyName-NAI into the User-Name attribute of RADIUS [13]. The rest
   of the process is similar to that described in [14] and [12].

o If a local ER server is present, the local ER server MUST include
   the DSRK request and its domain name in the AAA message
   encapsulating the EAP-Initiate/Re-auth message sent by the peer.

o Upon receipt of an EAP-Initiate/Re-auth message, the server
   verifies whether the message is fresh or is a replay by evaluating
   whether the received sequence number is equal to or greater than
   the expected sequence number for that rIK. The server then
   verifies to ensure that the cryptosuite used by the peer is
   acceptable. Next, it verifies the origin authentication of the
   message by looking up the rIK. If any of the checks fail, the
   server sends an EAP-Finish/Re-auth message with the Result flag
   set to '1'. Please refer to Section 5.2.2 for details on failure
   handling. This error MUST NOT have any correlation to any EAP-
   Success message that may have been received by the EAP
   authenticator and the peer earlier. If the EAP-Initiate/Re-auth
   message is well-formed and valid, the server prepares the EAP-
   Finish/Re-auth message. The bootstrap flag MUST be set to
   indicate that this is a bootstrapping exchange. The message
   contains the following fields:

* A sequence number for replay protection.

* The same keyName-NAI as in the EAP-Initiate/Re-auth message.

* If the lifetime flag was set in the EAP-Initiate/Re-auth
  message, the ER server SHOULD include the rRK lifetime and the
  rMSK lifetime in the EAP-Finish/Re-auth message. The server
  may have a local policy for the network to maintain and enforce
  lifetime unilaterally. In such cases, the server need not
  respond to the peer's request for the lifetime.

* If the bootstrap flag is set and a DSRK request is received,
  the server MUST include the domain name to which the DSRK is
  being sent.

* If the home ER server verifies the authorization of a local
  domain server, it MAY include the Authorization Indication TLV
  to indicate to the peer that the server (that received the DSRK
  and that is advertising the domain included in the domain name
  TLV) is authorized.

* An authentication tag MUST be included to prove that the EAP-
  Finish/Re-auth message originates at a server that possesses
  the rIK corresponding to the EMSKname-NAI.

o If the ERP exchange is successful, and the local ER server sent a
   DSRK request, the home ER server MUST include the DSRK for the
   local ER server (derived using the EMSK and the domain name as
   specified in [3]), EMSKname, and DSRK lifetime along with the EAP-
   Finish/Re-auth message.

o In addition, the rMSK is sent along with the EAP-Finish/Re-auth
   message, in a AAA attribute [12].

o The local ER server MUST extract the DSRK, EMSKname, and DSRK
   lifetime (if present), before forwarding the EAP-Finish/Re-auth
   message to the peer, as specified in [12].

o The authenticator receives the rMSK.

o When the peer receives an EAP-Finish/Re-auth message with the
   bootstrap flag set, if a local domain name is present, it MUST use
   that to derive the appropriate DSRK, DS-rRK, DS-rIK, and keyName-
   NAI, and initialize the replay counter for the DS-rIK. If not,
   the peer SHOULD derive the domain-specific keys using the domain
   name it learned via the lower layer or from the EAP-Initiate/
   Re-auth-Start message. If the peer does not know the domain name,
   it must assume that there is no local ER server available.

   o The peer MAY also verify the Authorization Indication TLV.

   o The procedures for encapsulating the ERP and obtaining relevant
      keys using RADIUS and Diameter are specified in [12] and [15],
      respectively.

   Since the ER bootstrapping exchange is typically done immediately
   following the full EAP exchange, it is feasible that the process is
   completed through the same entity that served as the EAP
   authenticator for the full EAP exchange. In this case, the lower
   layer may already have established TSKs based on the MSK received
   earlier. The lower layer may then choose to ignore the rMSK that was
   received with the ER bootstrapping exchange. Alternatively, the
   lower layer may choose to establish a new TSK using the rMSK. In
   either case, the authenticator and the peer know which key is used
   based on whether or not a TSK establishment exchange is initiated.
   The bootstrapping exchange may also be carried out via a new
   authenticator, in which case, the rMSK received SHOULD trigger a
   lower layer TSK establishment exchange.

5.2. Steps in ERP

   When a peer that has an active rRK and rIK associates with a new
   authenticator that supports ERP, it may perform an ERP exchange with
   that authenticator. ERP is typically a peer-initiated exchange,
   consisting of an EAP-Initiate/Re-auth and an EAP-Finish/Re-auth
   message. The ERP exchange may be performed with a local ER server
   (when one is present) or with the original EAP server.

   It is plausible for the network to trigger the EAP re-authentication
   process, however. An ERP-capable authenticator SHOULD send an EAP-
   Initiate/Re-auth-Start message to indicate support for ERP. The peer
   may or may not wait for these messages to arrive to initiate the EAP-
   Initiate/Re-auth message.

   The EAP-Initiate/Re-auth-Start message SHOULD be sent by an ERP-
   capable authenticator. The authenticator may retransmit it a few
   times until it receives an EAP-Initiate/Re-auth message in response
   from the peer. The EAP-Initiate/Re-auth message from the peer may
   have originated before the peer receives either an EAP-Request/
   Identity or an EAP-Initiate/Re-auth-Start message from the
   authenticator. Hence, the Identifier value in the EAP-Initiate/
   Re-auth message is independent of the Identifier value in the EAP-
   Initiate/Re-auth-Start or the EAP-Request/Identity messages.

Operational Considerations at the Peer:

ERP requires that the peer maintain retransmission timers for
reliable transport of EAP re-authentication messages. The
reliability considerations of Section 4.3 of RFC 3748 apply with the
peer as the retransmitting entity.

The EAP Re-auth Protocol has the following steps:

   The peer sends an EAP-Initiate/Re-auth message. At a minimum, the
   message SHALL include the following fields:

      a 16-bit sequence number for replay protection

      keyName-NAI as a TLV attribute to identify the rIK used to
      integrity protect the message.

      cryptosuite to indicate the authentication algorithm used to
      compute the integrity checksum.

      authentication tag over the message.

   When the peer is performing ERP with a local ER server, it MUST
   use the corresponding DS-rIK it shares with the local ER server.
   The peer SHOULD set the lifetime flag to request the key lifetimes
   from the server. The peer can use the rRK lifetime to know when
   to trigger an EAP method exchange and the rMSK lifetime to know
   when to trigger another ERP exchange.

   The authenticator copies the contents of the value field of the
   keyName-NAI TLV into the User-Name RADIUS attribute in the AAA
   message to the ER server.

   The server uses the keyName-NAI to look up the rIK. It MUST first
   verify whether the sequence number is equal to or greater than the
   expected sequence number. If the server supports a sequence
   number window size greater than 1, it MUST verify whether the
   sequence number falls within the window and has not been received
   before. The server MUST then verify to ensure that the
   cryptosuite used by the peer is acceptable. The server then
   proceeds to verify the integrity of the message using the rIK,
   thereby verifying proof of possession of that key by the peer. If
   any of these verifications fail, the server MUST send an EAP-
   Finish/Re-auth message with the Result flag set to '1' (Failure).
   Please refer to Section 5.2.2 for details on failure handling.
   Otherwise, it MUST compute an rMSK from the rRK using the sequence
   number as the additional input to the key derivation.

In response to a well-formed EAP Re-auth/Initiate message, the
server MUST send an EAP-Finish/Re-auth message with the following
considerations:

a 16-bit sequence number for replay protection, which MUST be
the same as the received sequence number. The local copy of
the sequence number MUST be incremented by 1. If the server
supports multiple simultaneous ERP exchanges, it MUST instead
update the sequence number window.

keyName-NAI as a TLV attribute to identify the rIK used to
integrity protect the message.

cryptosuite to indicate the authentication algorithm used to
compute the integrity checksum.

authentication tag over the message.

If the lifetime flag was set in the EAP-Initiate/Re-auth
message, the ER server SHOULD include the rRK lifetime and the
rMSK lifetime.

The server transports the rMSK along with this message to the
authenticator. The rMSK is transported in a manner similar to the
MSK transport along with the EAP-Success message in a regular EAP
exchange.

The peer looks up the sequence number to verify whether it is
expecting an EAP-Finish/Re-auth message with that sequence number
protected by the keyName-NAI. It then verifies the integrity of
the message. If the verifications fail, the peer logs an error
and stops the process; otherwise, it proceeds to the next step.

The peer uses the sequence number to compute the rMSK.

The lower-layer security association protocol can be triggered at
this point.

5.2.1. Multiple Simultaneous Runs of ERP

When a peer is within the range of multiple authenticators, it may
choose to run ERP via all of them simultaneously to the same ER
server. In that case, it is plausible that the ERP messages may
arrive out of order, resulting in the ER server rejecting legitimate
EAP-Initiate/Re-auth messages.

To facilitate such operation, an ER server MAY allow multiple
simultaneous ERP exchanges by accepting all EAP-Initiate/Re-auth
messages with SEQ number values within a window of allowed values.
Recall that the SEQ number allows replay protection. Replay window
maintenance mechanisms are a local matter.

5.2.2. ERP Failure Handling

If the processing of the EAP-Initiate/Re-auth message results in a
failure, the ER server MUST send an EAP-Finish Re-auth message with
the Result flag set to '1'. If the server has a valid rIK for the
peer, it MUST integrity protect the EAP-Finish/Re-auth failure
message. If the failure is due to an unacceptable cryptosuite, the
server SHOULD send a list of acceptable cryptosuites (in a TLV of
Type 5) along with the EAP-Finish/Re-auth message. In this case, the
server MUST indicate the cryptosuite used to protect the EAP-Finish/
Re-auth message in the cryptosuite. The rIK used with the EAP-
Finish/Re-auth message in this case MUST be computed as specified in
Section 4.3 using the new cryptosuite. If the server does not have a
valid rIK for the peer, the EAP-Finish/Re-auth message indicating a
failure will be unauthenticated; the server MAY include a list of
acceptable cryptosuites in the message.

The peer, upon receiving an EAP-Finish/Re-auth message with the
Result flag set to '1', MUST verify the sequence number and the
Authentication Tag to determine the validity of the message. If the
peer supports the cryptosuite, it MUST verify the integrity of the
received EAP-Finish/Re-auth message. If the EAP-Finish message
contains a TLV of Type 5, the peer SHOULD retry the ERP exchange with
a cryptosuite picked from the list included by the server. The peer
MUST use the appropriate rIK for the subsequent ERP exchange, by
computing it with the corresponding cryptosuite, as specified in
Section 4.3. If the PRF in the chosen cryptosuite is different from
the PRF originally used by the peer, it MUST derive a new DSRK (if
required), rRK, and rIK before proceeding with the subsequent ERP
exchange.

If the peer cannot verify the integrity of the received message, it
MAY choose to retry the ERP exchange with one of the cryptosuites in
the TLV of Type 5, after a failure has been clearly determined
following the procedure in the next paragraph.

If the replay or integrity checks fail, the failure message may have
been sent by an attacker. It may also imply that the server and peer
do not support the same cryptosuites; however, the peer cannot
determine if that is the case. Hence, the peer SHOULD continue the
ERP exchange per the retransmission timers before declaring a
failure.

When the peer runs explicit bootstrapping (ERP with the bootstrapping
flag on), there may not be a local ER server available to send a DSRK
Request and the domain name. In that case, the server cannot send
the DSRK and MUST NOT include the domain name TLV. When the peer
receives a response in the bootstrapping exchange without a domain
name TLV, it assumes that there is no local ER server. The home ER
server sends an rMSK to the ER authenticator, however, and the peer
SHALL run the TSK establishment protocol as usual.

5.3. New EAP Packets

Two new EAP Codes are defined for the purpose of ERP: EAP-Initiate
and EAP-Finish. The packet format for these messages follows the EAP
packet format defined in RFC 3748 [2].

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Code      |   Identifier  |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |  Type-Data ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
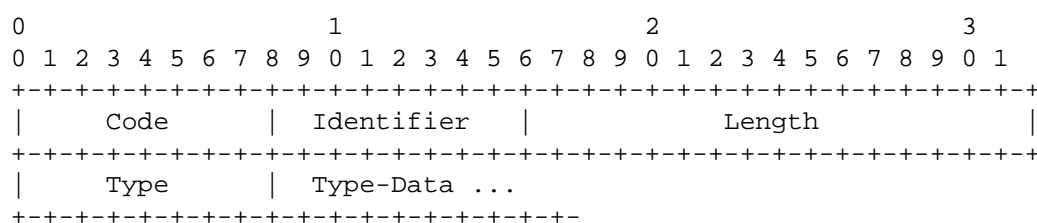
Figure 6: EAP Packet

Code

   5 Initiate

   6 Finish

   Two new code values are defined for the purpose of ERP.

Identifier

   The Identifier field is one octet. The Identifier field MUST
   be the same if an EAP-Initiate packet is retransmitted due to a
   timeout while waiting for a Finish message. Any new
   (non-retransmission) Initiate message MUST use a new Identifier
   field.

   The Identifier field of the Finish message MUST match that of
   the currently outstanding Initiate message. A peer or
   authenticator receiving a Finish message whose Identifier value
   does not match that of the currently outstanding Initiate
   message MUST silently discard the packet.

In order to avoid confusion between new EAP-Initiate messages
and retransmissions, the peer must choose an Identifier value
that is different from the previous EAP-Initiate message,
especially if that exchange has not finished. It is
RECOMMENDED that the authenticator clear EAP Re-auth state
after 300 seconds.

Type

This field indicates that this is an ERP exchange. Two type
values are defined in this document for this purpose --
Re-auth-Start (assigned Type 1) and Re-auth (assigned Type 2).

Type-Data

The Type-Data field varies with the Type of re-authentication
packet.

5.3.1. EAP-Initiate/Re-auth-Start Packet

The EAP-Initiate/Re-auth-Start packet contains the parameters shown
in Figure 7.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Code      |  Identifier   |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |   Reserved    |     1 or more TVs or TLVs     ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
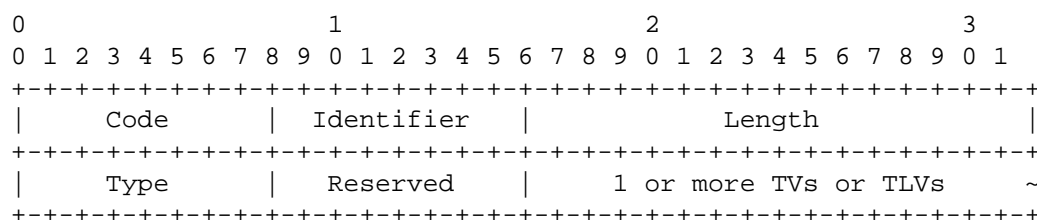
Figure 7: EAP-Initiate/Re-auth-Start Packet

Type = 1.

Reserved, MUST be zero. Set to zero on transmission and ignored
on reception.

One or more TVs or TLVs are used to convey information to the
peer; for instance, the authenticator may send the domain name to
the peer.

TVs or TLVs: In the TV payloads, there is a 1-octet type payload
and a value with type-specific length. In the TLV payloads, there
is a 1-octet type payload and a 1-octet length payload. The
length field indicates the length of the value expressed in number
of octets.

Domain-Name: This is a TLV payload. The Type is 4. The domain
name is to be used as the realm in an NAI [4]. The Domain-Name
attribute SHOULD be present in an EAP-Initiate/Re-auth-Start
message.

In addition, channel binding information MAY be included; see
Section 5.5 for discussion. See Figure 11 for parameter
specification.

5.3.1.1. Authenticator Operation

The authenticator MAY send the EAP-Initiate/Re-auth-Start message to
indicate support for ERP to the peer and to initiate ERP if the peer
has already performed full EAP authentication and has unexpired key
material. The authenticator SHOULD include the domain name TLV to
allow the peer to learn it without lower-layer support or the ERP
bootstrapping exchange.

The authenticator MAY include channel binding information so that the
peer can send the information to the server in the EAP-Initiate/
Re-auth message so that the server can verify whether the
authenticator is claiming the same identity to both parties.

The authenticator MAY re-transmit the EAP-Initiate/Re-auth-Start
message a few times for reliable transport.

5.3.1.2. Peer Operation

The peer SHOULD send the EAP-Initiate/Re-auth message in response to
the EAP-Initiate/Re-auth-Start message from the authenticator. If
the peer does not recognize the Initiate code value, it silently
discards the message. If the peer has already sent the EAP-Initiate/
Re-auth message to begin the ERP exchange, it silently discards the
message.

If the EAP-Initiate/Re-auth-Start message contains the domain name,
and if the peer does not already have the domain information, the
peer SHOULD use the domain name to compute the DSRK and use the
corresponding DS-rIK to send an EAP-Initiate/Re-auth message to start
an ERP exchange with the local ER server. If the peer has already
initiated an ERP exchange with the home ER server, it MAY choose to
not start an ERP exchange with the local ER server.

5.3.2. EAP-Initiate/Re-auth Packet

   The EAP-Initiate/Re-auth packet contains the parameters shown in
   Figure 8.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Code      |  Identifier   |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |R|B|L| Reserved|              SEQ              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   1 or more TVs or TLVs                      ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| cryptosuite   |           Authentication Tag                ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

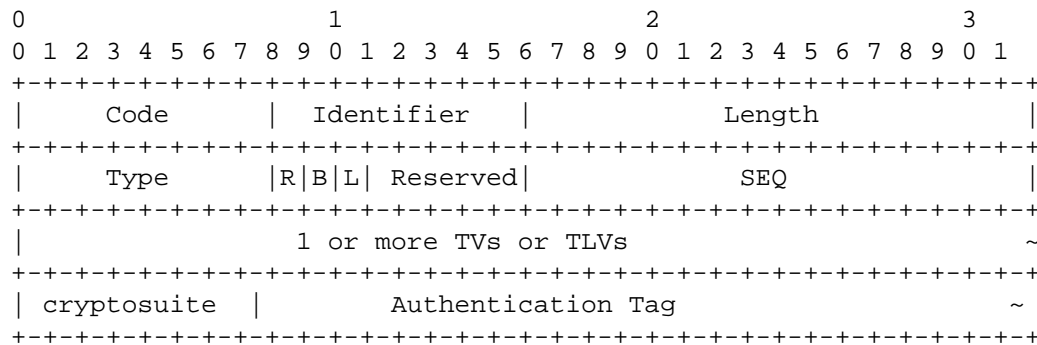   Figure 8: EAP-Initiate/Re-auth Packet

      Type = 2.

      Flags

         'R' - The R flag is set to 0 and ignored upon reception.

         'B' - The B flag is used as the bootstrapping flag. If the
         flag is turned on, the message is a bootstrap message.

         'L' - The L flag is used to request the key lifetimes from the
         server.

         The rest of the 5 bits are set to 0 and ignored on reception.

      SEQ: A 16-bit sequence number is used for replay protection. The
      SEQ number field is initialized to 0 every time a new rRK is
      derived.

      TVs or TLVs: In the TV payloads, there is a 1-octet type payload
      and a value with type-specific length. In the TLV payloads, there
      is a 1-octet type payload and a 1-octet length payload. The
      length field indicates the length of the value expressed in number
      of octets.

         keyName-NAI: This is carried in a TLV payload. The Type is 1.
         The NAI is variable in length, not exceeding 253 octets. The
         EMSKname is in the username part of the NAI and is encoded in
         hexadecimal values. The EMSKname is 64 bits in length and so
         the username portion takes up 128 octets. If the rIK is

derived from the EMSK, the realm part of the NAI is the home
domain name, and if the rIK is derived from a DSRK, the realm
part of the NAI is the domain name used in the derivation of
the DSRK. The NAI syntax follows [4]. Exactly one keyName-NAI
attribute SHALL be present in an EAP-Initiate/Re-auth packet.

In addition, channel binding information MAY be included; see
Section 5.5 for discussion. See Figure 11 for parameter
specification.

Cryptosuite: This field indicates the integrity algorithm used for
ERP. Key lengths and output lengths are either indicated or are
obvious from the cryptosuite name. We specify some cryptosuites
below:

*   0 RESERVED

*   1 HMAC-SHA256-64

*   2 HMAC-SHA256-128

*   3 HMAC-SHA256-256

HMAC-SHA256-128 is mandatory to implement and should be enabled in
the default configuration.

Authentication Tag: This field contains the integrity checksum
over the ERP packet, excluding the authentication tag field
itself. The length of the field is indicated by the Cryptosuite.

5.3.3. EAP-Finish/Re-auth Packet

The EAP-Finish/Re-auth packet contains the parameters shown in
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Code      |  Identifier   |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |R|B|L| Reserved |            SEQ               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    1 or more TVs or TLVs                      ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| cryptosuite   |        Authentication Tag                    ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
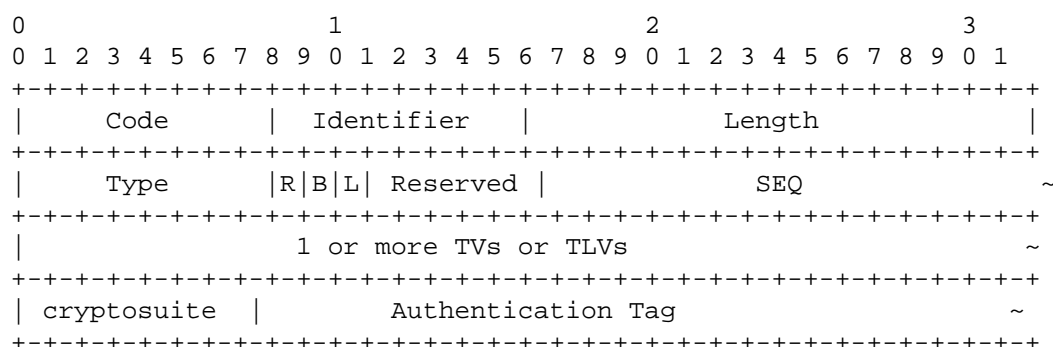
Figure 9: EAP-Finish/Re-auth Packet

Type = 2.

Flags

    'R' - The R flag is used as the Result flag. When set to 0, it
    indicates success, and when set to '1', it indicates a failure.

    'B' - The B flag is used as the bootstrapping flag. If the
    flag is turned on, the message is a bootstrap message.

    'L' - The L flag is used to indicate the presence of the rRK
    lifetime TLV.

    The rest of the 5 bits are set to 0 and ignored on reception.

SEQ: A 16-bit sequence number is used for replay protection. The
SEQ number field is initialized to 0 every time a new rRK is
derived.

TVs or TLVs: In the TV payloads, there is a 1-octet type payload
and a value with type-specific length. In the TLV payloads, there
is a 1-octet type payload and a 1-octet length payload. The
length field indicates the length of the value expressed in number
of octets.

    keyName-NAI: This is carried in a TLV payload. The Type is 1.
    The NAI is variable in length, not exceeding 253 octets.
    EMSKname is in the username part of the NAI and is encoded in
    hexadecimal values. The EMSKname is 64 bits in length and so
    the username portion takes up 16 octets. If the rIK is derived
    from the EMSK, the realm part of the NAI is the home domain
    name, and if the rIK is derived from a DSRK, the realm part of
    the NAI is the domain name used in the derivation of the DSRK.
    The NAI syntax follows [4]. Exactly one instance of the
    keyName-NAI attribute SHALL be present in an EAP-Finish/Re-auth
    message.

    rRK Lifetime: This is a TV payload. The Type is 2. The value
    field is a 32-bit field and contains the lifetime of the rRK in
    seconds. If the 'L' flag is set, the rRK Lifetime attribute
    SHOULD be present.

    rMSK Lifetime: This is a TV payload. The Type is 3. The value
    field is a 32-bit field and contains the lifetime of the rMSK
    in seconds. If the 'L' flag is set, the rMSK Lifetime
    attribute SHOULD be present.

Domain-Name: This is a TLV payload. The Type is 4. The domain
name is to be used as the realm in an NAI [4]. Domain-Name
attribute MUST be present in an EAP-Finish/Re-auth message if
the bootstrapping flag is set and if the local ER server sent a
DSRK request.

List of cryptosuites: This is a TLV payload. The Type is 5.
The value field contains a list of cryptosuites, each of size 1
octet. The cryptosuite values are as specified in Figure 8.
The server SHOULD include this attribute if the cryptosuite
used in the EAP-Initiate/Re-auth message was not acceptable and
the message is being rejected. The server MAY include this
attribute in other cases. The server MAY use this attribute to
signal to the peer about its cryptographic algorithm
capabilities.

Authorization Indication: This is a TLV payload. The Type is
6. This attribute MAY be included in the EAP-Finish/Re-auth
message when a DSRK is delivered to a local ER server and if
the home ER server can verify the authorization of the local ER
server to advertise the domain name included in the domain TLV
in the same message. The value field in the TLV contains an
authentication tag computed over the entire packet, starting
from the first bit of the code field to the last bit of the
cryptosuite field, with the value field of the Authorization
Indication TLV filled with all 0s for the computation. The key
used for the computation MUST be derived from the EMSK with key
label "DSRK Delivery Authorized Key@ietf.org" and optional data
containing an ASCII string representing the key management
domain that the DSRK is being derived for.

In addition, channel binding information MAY be included: see
Section 5.5 for discussion. See Figure 11 for parameter
specification. The server sends this information so that the
peer can verify the information seen at the lower layer, if
channel binding is to be supported.

Cryptosuite: This field indicates the integrity algorithm and the
PRF used for ERP. Key lengths and output lengths are either
indicated or are obvious from the cryptosuite name.

Authentication Tag: This field contains the integrity checksum
over the ERP packet, excluding the authentication tag field
itself. The length of the field is indicated by the Cryptosuite.

5.3.4. TV and TLV Attributes

   The TV attributes that may be present in the EAP-Initiate or EAP-
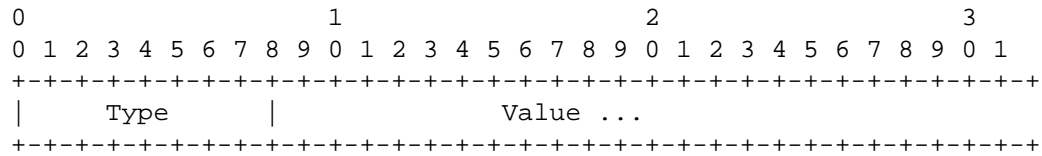   Finish messages are of the following format:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |               Value ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
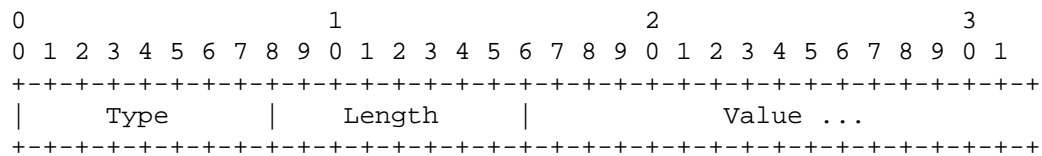
                    Figure 10: TV Attribute Format

   The TLV attributes that may be present in the EAP-Initiate or EAP-
   Finish messages are of the following 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |             Value ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                    Figure 11: TLV Attribute Format

   The following Types are defined in this document:

      '1' - keyName-NAI: This is a TLV payload.

      '2' - rRK Lifetime: This is a TV payload.

      '3' - rMSK Lifetime: This is a TV payload.

      '4' - domain name: This is a TLV payload.

      '5' - cryptosuite list: This is a TLV payload.

      '6' - Authorization Indication: This is a TLV payload.

   The TLV type range of 128-191 is reserved to carry channel binding
   information in the EAP-Initiate and Finish/Re-auth messages.
   Below are the current assignments (all of them are TLVs):

      '128' - Called-Station-Id [13]

      '129' - Calling-Station-Id [13]

      '130' - NAS-Identifier [13]

        '131' - NAS-IP-Address [13]

        '132' - NAS-IPv6-Address [16]

   The length field indicates the length of the value part of the
   attribute in octets.

5.4. Replay Protection

   For replay protection, ERP uses sequence numbers. The sequence
   number is maintained per rIK and is initialized to zero in both
   directions. In the first EAP-Initiate/Re-auth message, the peer uses
   the sequence number zero or higher. Note that the when the sequence
   number rotates, the rIK MUST be changed by running EAP
   authentication. The server expects a sequence number of zero or
   higher. When the server receives an EAP-Initiate/Re-auth message, it
   uses the same sequence number in the EAP-Finish/Re-auth message. The
   server then sets the expected sequence number to the received
   sequence number plus 1. The server accepts sequence numbers greater
   than or equal to the expected sequence number.

   If the peer sends an EAP-Initiate/Re-auth message, but does not
   receive a response, it retransmits the request (with no changes to
   the message itself) a pre-configured number of times before giving
   up. However, it is plausible that the server itself may have
   responded to the message and it was lost in transit. Thus, the peer
   MUST increment the sequence number and use the new sequence number to
   send subsequent EAP re-authentication messages. The peer SHOULD
   increment the sequence number by 1; however, it may choose to
   increment by a larger number. When the sequence number rotates, the
   peer MUST run full EAP authentication.

5.5. Channel Binding

   ERP provides a protected facility to carry channel binding (CB)
   information, according to the guidelines in Section 7.15 of [2]. The
   TLV type range of 128-191 is reserved to carry CB information in the
   EAP-Initiate/Re-auth and EAP-Finish/Re-auth messages. Called-
   Station-Id, Calling-Station-Id, NAS-Identifier, NAS-IP-Address, and
   NAS-IPv6-Address are some examples of channel binding information
   listed in RFC 3748, and they are assigned values 128-132. Additional
   values are IANA managed based on IETF Consensus [17].

   The authenticator MAY provide CB information to the peer via the EAP-
   Initiate/Re-auth-Start message. The peer sends the information to
   the server in the EAP-Initiate/Re-auth message; the server verifies
   whether the authenticator identity available via AAA attributes is
   the same as the identity provided to the peer.

If the peer does not include the CB information in the EAP-Initiate/
Re-auth message, and if the local ER server's policy requires channel
binding support, it SHALL send the CB attributes for the peer's
verification. The peer attempts to verify the CB information if the
authenticator has sent the CB parameters, and it proceeds with the
lower-layer security association establishment if the attributes
match. Otherwise, the peer SHALL NOT proceed with the lower-layer
security association establishment.

6. Lower-Layer Considerations

The authenticator is responsible for retransmission of EAP-Initiate/
Re-auth-Start messages. The authenticator MAY retransmit the message
a few times or until it receives an EAP-Initiate/Re-auth message from
the peer. The authenticator may not know whether the peer supports
ERP; in those cases, the peer may be silently dropping the EAP-
Initiate/Re-auth-Start packets. Thus, retransmission of these
packets should be kept to a minimum. The exact number is up to each
lower layer.

The Identifier value in the EAP-Initiate/Re-auth packet is
independent of the Identifier value in the EAP-Initiate/Re-auth-Start
packet.

The peer is responsible for retransmission of EAP-Initiate/Re-auth
messages.

Retransmitted packets MUST be sent with the same Identifier value in
order to distinguish them from new packets. By default, where the
EAP-Initiate message is sent over an unreliable lower layer, the
retransmission timer SHOULD be dynamically estimated. A maximum of
3-5 retransmissions is suggested (this is based on the recommendation
of [2]). Where the EAP-Initiate message is sent over a reliable
lower layer, the retransmission timer SHOULD be set to an infinite
value, so that retransmissions do not occur at the EAP layer. Please
refer to RFC 3748 [2] for additional guidance on setting timers.

The Identifier value in the EAP-Finish/Re-auth packet is the same as
the Identifier value in the EAP-Initiate/Re-auth packet.

If an authenticator receives a valid duplicate EAP-Initiate/Re-auth
message for which it has already sent an EAP-Finish/Re-auth message,
it MUST resend the EAP-Finish/Re-auth message without reprocessing
the EAP-Initiate/Re-auth message. To facilitate this, the
authenticator SHALL store a copy of the EAP-Finish/Re-auth message
for a finite amount of time. The actual value of time is a local
matter; this specification recommends a value of 100 milliseconds.

The lower layer may provide facilities for exchanging information
between the peer and the authenticator about support for ERP, for the
authenticator to send the domain name information and channel binding
information to the peer

Note that to support ERP, lower-layer specifications may need to be
revised. Specifically, the IEEE802.1x specification must be revised
to allow carrying EAP messages of the new codes defined in this
document in order to support ERP. Similarly, RFC 4306 must be
updated to include EAP code values higher than 4 in order to use ERP
with Internet Key Exchange Protocol version 2 (IKEv2). IKEv2 may
also be updated to support peer-initiated ERP for optimized
operation. Other lower layers may need similar revisions.

Our analysis indicates that some EAP implementations are not RFC 3748
compliant in that instead of silently dropping EAP packets with code
values higher than 4, they may consider it an error. To accommodate
such non-compliant EAP implementations, additional guidance has been
provided below. Furthermore, it may not be easy to upgrade all the
peers in some cases. In such cases, authenticators may be configured
to not send EAP-Initiate/Re-auth-Start; peers may learn whether an
authenticator supports ERP via configuration, from advertisements at
the lower layer.

In order to accommodate implementations that are not compliant to RFC
3748, such lower layers SHOULD ensure that both parties support ERP;
this is trivial for an instance when using a lower layer that is
known to always support ERP. For lower layers where ERP support is
not guaranteed, ERP support may be indicated through signaling (e.g.,
piggy-backed on a beacon) or through negotiation. Alternatively,
clients may recognize environments where ERP is available based on
pre-configuration. Other similar mechanisms may also be used. When
ERP support cannot be verified, lower layers may mandate falling back
to full EAP authentication to accommodate EAP implementations that
are not compliant to RFC 3748.

7. Transport of ERP Messages

AAA Transport of ERP messages is specified in [11] and [12].

8. Security Considerations

   This section provides an analysis of the protocol in accordance with
   the AAA key management requirements specified in [18].

      Cryptographic algorithm independence

         The EAP Re-auth Protocol satisfies this requirement. The
         algorithm chosen by the peer for the MAC generation is
         indicated in the EAP-Initiate/Re-auth message. If the chosen
         algorithm is unacceptable, the EAP server returns an EAP-
         Finish/Re-auth message with Failure indication. Algorithm
         agility for the KDF is specified in [3]. Only when the
         algorithms used are acceptable, the server proceeds with
         derivation of keys and verification of the proof of possession
         of relevant keying material by the peer. A full-blown
         negotiation of algorithms cannot be provided in a single round
         trip protocol. Hence, while the protocol provides algorithm
         agility, it does not provide true negotiation.

      Strong, fresh session keys

         ERP results in the derivation of strong, fresh keys that are
         unique for the given session. An rMSK is always derived
         on-demand when the peer requires a key with a new
         authenticator. The derivation ensures that the compromise of
         one rMSK does not result in the compromise of a different rMSK
         at any time.

      Limit key scope

         The scope of all the keys derived by ERP is well defined. The
         rRK and rIK are never shared with any entity and always remain
         on the peer and the server. The rMSK is provided only to the
         authenticator through which the peer performs the ERP exchange.
         No other authenticator is authorized to use that rMSK.

      Replay detection mechanism

         For replay protection of ERP messages, a sequence number
         associated with the rIK is used. The sequence number is
         maintained by the peer and the server, and initialized to zero
         when the rIK is generated. The peer increments the sequence
         number by one after it sends an ERP message. The server sets
         the expected sequence number to the received sequence number
         plus one after verifying the validity of the received message
         and responds to the message.

Authenticate all parties

   The EAP Re-auth Protocol provides mutual authentication of the
   peer and the server. Both parties need to possess the keying
   material that resulted from a previous EAP exchange in order to
   successfully derive the required keys. Also, both the EAP
   re-authentication Response and the EAP re-authentication
   Information messages are integrity protected so that the peer
   and the server can verify each other. When the ERP exchange is
   executed with a local ER server, the peer and the local server
   mutually authenticate each other via that exchange in the same
   manner. The peer and the authenticator authenticate each other
   in the secure association protocol executed by the lower layer,
   just as in the case of a regular EAP exchange.

Peer and authenticator authorization

   The peer and authenticator demonstrate possession of the same
   key material without disclosing it, as part of the lower-layer
   secure association protocol. Channel binding with ERP may be
   used to verify consistency of the identities exchanged, when
   the identities used in the lower layer differ from that
   exchanged within the AAA protocol.

Keying material confidentiality

   The peer and the server derive the keys independently using
   parameters known to each entity. The AAA server sends the DSRK
   of a domain to the corresponding local ER server via the AAA
   protocol. Likewise, the ER server sends the rMSK to the
   authenticator via the AAA protocol.

   Note that compromise of the DSRK results in compromise of all
   keys derived from it. Moreover, there is no forward secrecy
   within ERP. Thus, compromise of an DSRK retroactively
   compromises all ERP keys.

   It is RECOMMENDED that the AAA protocol be protected using
   IPsec or TLS so that the keys are protected in transit. Note,
   however, that keys may be exposed to AAA proxies along the way
   and compromise of any of those proxies may result in compromise
   of keys being transported through them.

   The home ER server MUST NOT hand out a given DSRK to a local
   domain server more than once, unless it can verify that the
   entity receiving the DSRK after the first time is the same as
   that received the DSRK originally. If the home ER server
   verifies authorization of a local domain server, it MAY hand

      out the DSRK to that domain more than once. In this case, the
      home ER server includes the Authorization Indication TLV to
      assure the peer that DSRK delivery is secure.

   Confirm cryptosuite selection

      Crypto algorithms for integrity and key derivation in the
      context of ERP MAY be the same as that used by the EAP method.
      In that case, the EAP method is responsible for confirming the
      cryptosuite selection. Furthermore, the cryptosuite is
      included in the ERP exchange by the peer and confirmed by the
      server. The protocol allows the server to reject the
      cryptosuite selected by the peer and provide alternatives.
      When a suitable rIK is not available for the peer, the
      alternatives may be sent in an unprotected fashion. The peer
      is allowed to retry the exchange using one of the allowed
      cryptosuites. However, in this case, any en route
      modifications to the list sent by the server will go
      undetected. If the server does have an rIK available for the
      peer, the list will be provided in a protected manner and this
      issue does not apply.

   Uniquely named keys

      All keys produced within the ERP context can be referred to
      uniquely as specified in this document. Also, the key names do
      not reveal any part of the keying material.

   Prevent the domino effect

      The compromise of one peer does not result in the compromise of
      keying material held by any other peer in the system. Also,
      the rMSK is meant for a single authenticator and is not shared
      with any other authenticator. Hence, the compromise of one
      authenticator does not lead to the compromise of sessions or
      keys held by any other authenticator in the system. Hence, the
      EAP Re-auth Protocol allows prevention of the domino effect by
      appropriately defining key scope.

      However, if keys are transported using hop-by-hop protection,
      compromise of a proxy may result in compromise of key material,
      i.e., the DSRK being sent to a local ER server.

Bind key to its context

All the keys derived for ERP are bound to the appropriate
context using appropriate key labels. Lifetime of a child key
is less than or equal to that of its parent key as specified in
RFC 4962 [18]. The key usage, lifetime and the parties that
have access to the keys are specified.

Confidentiality of identity

Deployments where privacy is a concern may find the use of
rIKname-NAI to route ERP messages serves their privacy
requirements. Note that it is plausible to associate multiple
runs of ERP messages since the rIKname is not changed as part
of the ERP protocol. There was no consensus for that
requirement at the time of development of this specification.
If the rIKname is not used and the Peer-ID is used instead, the
ERP exchange will reveal the Peer-ID over the wire.

Authorization restriction

All the keys derived are limited in lifetime by that of the
parent key or by server policy. Any domain-specific keys are
further restricted for use only in the domain for which the
keys are derived. All the keys specified in this document are
meant for use in ERP only. Any other restrictions of session
keys may be imposed by the specific lower layer and are out of
scope for this specification.

A denial-of-service (DoS) attack on the peer may be possible when
using the EAP Initiate/Re-auth message. An attacker may send a bogus
EAP-Initiate/Re-auth message, which may be carried by the
authenticator in a RADIUS-Access-Request to the server; in response,
the server may send an EAP-Finish/Re-auth with Failure indication in
a RADIUS Access-Reject message. Note that such attacks may be
plausible with the EAPoL-Start capability of IEEE 802.11 and other
similar facilities in other link layers and where the peer can
initiate EAP authentication. An attacker may use such messages to
start an EAP method run, which fails and may result in the server
sending a RADIUS Access-Reject message, thus resulting in the link-
layer connections being terminated.

To prevent such DoS attacks, an ERP failure should not result in
deletion of any authorization state established by a full EAP
exchange. Alternatively, the lower layers and AAA protocols may
define mechanisms to allow two link-layer security associations (SAs)
derived from different EAP keying materials for the same peer to
exist so that smooth migration from the current link layer SA to the

new one is possible during rekey. These mechanisms prevent the link
layer connections from being terminated when a re-authentication
procedure fails due to the bogus EAP-Initiate/Re-auth message.

When a DSRK is sent from a home ER server to a local domain server or
when a rMSK is sent from an ER server to an authenticator, in the
absence of end-to-end security between the entity that is sending the
key and the entity receiving the key, it is plausible for other
entities to get access to keys being sent to an ER server in another
domain. This mode of key transport is similar to that of MSK
transport in the context of EAP authentication. We further observe
that ERP is for access authentication and does not support end-to-end
data security. In typical implementations, the traffic is in the
clear beyond the access control enforcement point (the authenticator
or an entity delegated by the authenticator for access control
enforcement). The model works as long as entities in the middle of
the network do not use keys intended for other parties to steal
service from an access network. If that is not achievable, key
delivery must be protected in an end-to-end manner.

9. IANA Considerations

This document specifies IANA registration of two new 'Packet Codes'
from the EAP registry:

o 5 (Initiate)

o 6 (Finish)

These values are in accordance with [2].

This document also specifies creation of a new table, Message Types,
in the EAP registry with the following assigned numbers:

o 0 Reserved

o 1 (Re-auth-Start, applies to Initiate Code only)

o 2 (Re-auth, applies to Initiate and Finish Codes)

o 3-191 IANA managed and assigned based on IETF Consensus [17]

o 192-255 Private use

Next, we specify creation of a new table, EAP Initiate and Finish
Attributes, associated with EAP Initiate and Finish messages in the
EAP registry with the following assigned numbers.

o 0: Reserved

o keyName-NAI: This is a TLV payload. The Type is 1.

o rRK Lifetime: This is a TV payload. The Type is 2.

o rMSK Lifetime: This is a TV payload. The Type is 3.

o Domain name: This is a TLV payload. The Type is 4.

o Cryptosuite list: This is a TLV payload. The Type is 5.

o Authorization Indication: This is a TLV payload. The Type is 6.

o 7-127: Used to carry other non-channel-binding-related attributes.
   IANA managed and assigned based on IETF Consensus [17].

o The TLV type range of 128-191 is reserved to carry CB information
   in the EAP-Initiate/Re-auth and EAP-Finish/Re-auth messages.
   Below are the current assignments (all of them are TLVs):

   * Called-Station-Id: 128

   * Calling-Station-Id: 129

   * NAS-Identifier: 130

   * NAS-IP-Address: 131

   * NAS-IPv6-Address: 132

   133-191: Used to carry other channel-binding-related attributes.
   IANA managed and assigned based on IETF Consensus [17].

o 192-255: Reserved for Private use.

   We specify creation of another registry, 'Re-authentication
   Cryptosuites', with the following assigned numbers:

o 0: Reserved

o 1: HMAC-SHA256-64

o 2: HMAC-SHA256-128

o 3: HMAC-SHA256-256

o 4-191: IANA managed and assigned based on IETF consensus [17]

   o 192-255: Reserved for Private use.

   Further, this document registers a Re-auth usage label from the "USRK
   Key Labels" name space with a value

      EAP Re-authentication Root Key@ietf.org

   and DSRK-authorized delivery key from the "USRK Key Labels" name
   space

      DSRK Delivery Authorized Key@ietf.org

   in accordance with [3].

10. Acknowledgments

   In writing this document, we benefited from discussing the problem
   space and the protocol itself with a number of folks including
   Bernard Aboba, Jari Arkko, Sam Hartman, Russ Housley, Joe Salowey,
   Jesse Walker, Charles Clancy, Michaela Vanderveen, Kedar Gaonkar,
   Parag Agashe, Dinesh Dharmaraju, Pasi Eronen, Dan Harkins, Yoshi
   Ohba, Glen Zorn, Alan DeKok, Katrin Hoeper, and other participants of
   the HOKEY working group. The credit for the idea to use EAP-
   Initiate/Re-auth-Start goes to Charles Clancy, and the multiple link-
   layer SAs idea to mitigate the DoS attack goes to Yoshi Ohba. Katrin
   Hoeper suggested the use of the windowing technique to handle
   multiple simultaneous ER exchanges. Many thanks to Pasi Eronen for
   the suggestion to use hexadecimal encoding for rIKname when sent as
   part of keyName-NAI field. Thanks to Bernard Aboba for suggestions
   in clarifying the EAP lock-step operation, and Joe Salowey and Glen
   Zorn for help in specifying AAA transport of ERP messages. Thanks to
   Sam Hartman for the DSRK Authorization Indication mechanism.

11. References

11.1. Normative References

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

   [2]   Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H.
         Levkowetz, "Extensible Authentication Protocol (EAP)",
         RFC 3748, June 2004.

   [3]   Salowey, J., Dondeti, L., Narayanan, V., and M. Nakhjiri,
         "Specification for the Derivation of Root Keys from an Extended
         Master Session Key (EMSK)", RFC 5295, August 2008.

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

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

11.2. Informative References

   [6]   Arkko, J. and H. Haverinen, "Extensible Authentication Protocol
         Method for 3rd Generation Authentication and Key Agreement
         (EAP-AKA)", RFC 4187, January 2006.

   [7]   Lopez, R., Skarmeta, A., Bournelle, J., Laurent-Maknavicus, M.,
         and J. Combes, "Improved EAP keying framework for a secure
         mobility access service", IWCMC '06, Proceedings of the 2006
         International Conference on Wireless Communications and Mobile
         Computing, New York, NY, USA, 2006.

   [8]   Arbaugh, W. and B. Aboba, "Handoff Extension to RADIUS", Work
         in Progress, October 2003.

   [9]   Clancy, T., Nakhjiri, M., Narayanan, V., and L. Dondeti,
         "Handover Key Management and Re-Authentication Problem
         Statement", RFC 5169, March 2008.

   [10] Institute of Electrical and Electronics Engineers, "IEEE
         Standards for Local and Metropolitan Area Networks: Port based
         Network Access Control, IEEE Std 802.1X-2004", December 2004.

   [11] Nakhjiri, M. and Y. Ohba, "Derivation, delivery and management
         of EAP based keys for handover and re-authentication", Work
         in Progress, February 2008.

   [12] Gaonkar, K., Dondeti, L., Narayanan, V., and G. Zorn, "RADIUS
         Support for EAP Re-authentication Protocol", Work in Progress,
         February 2008.

   [13] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote
         Authentication Dial In User Service (RADIUS)", RFC 2865,
         June 2000.

   [14] Aboba, B. and P. Calhoun, "RADIUS (Remote Authentication Dial
         In User Service) Support For Extensible Authentication Protocol
         (EAP)", RFC 3579, September 2003.

   [15] Dondeti, L. and H. Tschofenig, "Diameter Support for EAP Re-
         authentication Protocol", Work in Progress, November 2007.

   [16] Aboba, B., Zorn, G., and D. Mitton, "RADIUS and IPv6",
         RFC 3162, August 2001.

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

   [18] Housley, R. and B. Aboba, "Guidance for Authentication,
         Authorization, and Accounting (AAA) Key Management", BCP 132,
         RFC 4962, July 2007.

Appendix A. Example ERP Exchange

    0. Authenticator --> Peer: [EAP-Initiate/Re-auth-Start]

    1. Peer --> Authenticator: EAP Initiate/Re-auth(SEQ, keyName-NAI,
                                 cryptosuite,Auth-tag*)

    1a. Authenticator --> Re-auth-Server: AAA-Request{Authenticator-Id,
                                 EAP Initiate/Re-auth(SEQ,keyName-NAI,
                                 cryptosuite,Auth-tag*)

    2. ER-Server --> Authenticator: AAA-Response{rMSK,
                                 EAP-Finish/Re-auth(SEQ,keyName-NAI,
                                 cryptosuite,[CB-Info],Auth-tag*)

    2b. Authenticator --> Peer: EAP-Finish/Re-auth(SEQ,keyName-NAI,
                                 cryptosuite,[CB-Info],Auth-tag*)

    * Auth-tag computation is over the entire EAP Initiate/Finish
      message; the code values for Initiate and Finish are different and
      thus reflection attacks are mitigated.

Authors' Addresses

    Vidya Narayanan
    Qualcomm, Inc.
    5775 Morehouse Dr.
    San Diego, CA 92121
    USA

    Phone: +1 858-845-2483
    EMail: vidyan@qualcomm.com

    Lakshminath Dondeti
    Qualcomm, Inc.
    5775 Morehouse Dr.
    San Diego, CA 92121
    USA

    Phone: +1 858-845-1267
    EMail: ldondeti@qualcomm.com

Full Copyright Statement

Intellectual Property