            Extensible Authentication Protocol Method for
            Global System for Mobile Communications (GSM)
                 Subscriber Identity Modules (EAP-SIM)

Status of This Memo

   This memo provides information for the Internet community. It does
   not specify an Internet standard of any kind. Distribution of this
   memo is unlimited.

Copyright Notice

IESG Note

   The EAP-SIM protocol was developed by 3GPP. The documentation of
   EAP-SIM is provided as information to the Internet community. While
   the EAP WG has verified that EAP-SIM is compatible with EAP, as
   defined in RFC 3748, no other review has been done, including
   validation of the security claims. The IETF has also not reviewed
   the security of the cryptographic algorithms.

Abstract

   This document specifies an Extensible Authentication Protocol (EAP)
   mechanism for authentication and session key distribution using the
   Global System for Mobile Communications (GSM) Subscriber Identity
   Module (SIM). GSM is a second generation mobile network standard.
   The EAP-SIM mechanism specifies enhancements to GSM authentication
   and key agreement whereby multiple authentication triplets can be
   combined to create authentication responses and session keys of
   greater strength than the individual GSM triplets. The mechanism
   also includes network authentication, user anonymity support, result
   indications, and a fast re-authentication procedure.

Table of Contents

1. Introduction

   This document specifies an Extensible Authentication Protocol (EAP)
   [RFC3748] mechanism for authentication and session key distribution
   using the Global System for Mobile Communications (GSM) Subscriber
   Identity Module (SIM).

   GSM is a second generation mobile network standard. Second
   generation mobile networks and third generation mobile networks use
   different authentication and key agreement mechanisms. EAP-AKA
   [EAP-AKA] specifies an EAP method that is based on the Authentication
   and Key Agreement (AKA) mechanism used in 3rd generation mobile
   networks.

   GSM authentication is based on a challenge-response mechanism. The
   A3/A8 authentication and key derivation algorithms that run on the
   SIM can be given a 128-bit random number (RAND) as a challenge. The
   SIM runs operator-specific algorithms, which take the RAND and a
   secret key Ki (stored on the SIM) as input, and produce a 32-bit
   response (SRES) and a 64-bit long key Kc as output. The Kc key is
   originally intended to be used as an encryption key over the air
   interface, but in this protocol, it is used for deriving keying
   material and is not directly used. Hence, the secrecy of Kc is
   critical to the security of this protocol. For more information
   about GSM authentication, see [GSM-03.20]. See Section 12.1 for more
   discussion about the GSM algorithms used in EAP-SIM.

   The lack of mutual authentication is a weakness in GSM
   authentication. The derived 64-bit cipher key (Kc) is not strong
   enough for data networks in which stronger and longer keys are
   required. Hence, in EAP-SIM, several RAND challenges are used for
   generating several 64-bit Kc keys, which are combined to constitute
   stronger keying material. In EAP-SIM, the client issues a random
   number NONCE_MT to the network in order to contribute to key
   derivation, and to prevent replays of EAP-SIM requests from previous

exchanges. The NONCE_MT can be conceived as the client's challenge
to the network. EAP-SIM also extends the combined RAND challenges
and other messages with a message authentication code in order to
provide message integrity protection along with mutual
authentication.

EAP-SIM specifies optional support for protecting the privacy of
subscriber identity using the same concept as the GSM, which uses
pseudonyms/temporary identifiers. It also specifies an optional fast
re-authentication procedure.

The security of EAP-SIM builds on underlying GSM mechanisms. The
security properties of EAP-SIM are documented in Section 11 of this
document. Implementers and users of EAP-SIM are advised to carefully
study the security considerations in Section 11 in order to determine
whether the security properties are sufficient for the environment in
question, especially as the secrecy of Kc keys is essential to the
security of EAP-SIM. In brief, EAP-SIM is in no sense weaker than
the GSM mechanisms. In some cases EAP-SIM provides better security
properties than the underlying GSM mechanisms, particularly if the
SIM credentials are only used for EAP-SIM and are not re-used from
GSM/GPRS. Many of the security features of EAP-SIM rely upon the
secrecy of the Kc values in the SIM triplets, so protecting these
values is key to the security of the EAP-SIM protocol.

The 3rd Generation Partnership Project (3GPP) has specified an
enhanced Authentication and Key Agreement (AKA) architecture for the
Universal Mobile Telecommunications System (UMTS). The 3rd
generation AKA mechanism includes mutual authentication, replay
protection, and derivation of longer session keys. EAP-AKA [EAP-AKA]
specifies an EAP method that is based on the 3rd generation AKA.
EAP-AKA, which is a more secure protocol, may be used instead of
EAP-SIM, if 3rd generation identity modules and 3G network
infrastructures are available.

2. Terms

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].

The terms and abbreviations "authenticator", "backend authentication
server", "EAP server", "peer", "Silently Discard", "Master Session
Key (MSK)", and "Extended Master Session Key (EMSK)" in this document
are to be interpreted as described in [RFC3748].

This document frequently uses the following terms and abbreviations:

AAA protocol

        Authentication, Authorization, and Accounting protocol

AuC

        Authentication Centre. The GSM network element that provides
        the authentication triplets for authenticating
        the subscriber.

Authentication vector

        GSM triplets can be alternatively called authentication
        vectors.

EAP

        Extensible Authentication Protocol

Fast re-authentication

        An EAP-SIM authentication exchange that is based on keys
        derived upon a preceding full authentication exchange.
        The GSM authentication and key exchange algorithms are not
        used in the fast re-authentication procedure.

Fast Re-authentication Identity

        A fast re-authentication identity of the peer, including an NAI
        realm portion in environments where a realm is used. Used on
        fast re-authentication only.

Fast Re-authentication Username

        The username portion of fast re-authentication identity,
        i.e., not including any realm portions.

Full authentication

        An EAP-SIM authentication exchange based on the GSM
        authentication and key agreement algorithms.

GSM

        Global System for Mobile communications.

GSM Triplet

> The tuple formed by the three GSM authentication values RAND,
> Kc, and SRES.

IMSI

> International Mobile Subscriber Identifier, used in GSM to
> identify subscribers.

MAC

> Message Authentication Code

NAI

> Network Access Identifier

Nonce

> A value that is used at most once or that is never repeated
> within the same cryptographic context. In general, a nonce can
> be predictable (e.g., a counter) or unpredictable (e.g., a
> random value). Since some cryptographic properties may depend
> on the randomness of the nonce, attention should be paid to
> whether a nonce is required to be random or not. In this
> document, the term nonce is only used to denote random nonces,
> and it is not used to denote counters.

Permanent Identity

> The permanent identity of the peer, including an NAI realm
> portion in environments where a realm is used. The permanent
> identity is usually based on the IMSI. Used on full
> authentication only.

Permanent Username

> The username portion of permanent identity, i.e., not including
> any realm portions.

Pseudonym Identity

> A pseudonym identity of the peer, including an NAI realm
> portion in environments where a realm is used. Used on
> full authentication only.

   Pseudonym Username

      The username portion of pseudonym identity, i.e., not including
      any realm portions.

   SIM

      Subscriber Identity Module. The SIM is traditionally a smart
      card distributed by a GSM operator.

3. Overview

   Figure 1 shows an overview of the EAP-SIM full authentication
   procedure, wherein optional protected success indications are not
   used. The authenticator typically communicates with an EAP server
   that is located on a backend authentication server using an AAA
   protocol. The authenticator shown in the figure is often simply
   relaying EAP messages to and from the EAP server, but these backend
   AAA communications are not shown.

```
    Peer                                              Authenticator
     |                                    EAP-Request/Identity      |
     |<-----------------------------------------------------------|
     |                                                             |
     | EAP-Response/Identity                                       |
     |----------------------------------------------------------->|
     |                                                             |
     |                         EAP-Request/SIM/Start (AT_VERSION_LIST) |
     |<-----------------------------------------------------------|
     |                                                             |
     | EAP-Response/SIM/Start (AT_NONCE_MT, AT_SELECTED_VERSION)|
     |----------------------------------------------------------->|
     |                                                             |
     |              EAP-Request/SIM/Challenge (AT_RAND, AT_MAC)    |
     |<-----------------------------------------------------------|
    +------------------------------------+                        |
    | Peer runs GSM algorithms, verifies |                        |
    | AT_MAC and derives session keys    |                        |
    +------------------------------------+                        |
     |   EAP-Response/SIM/Challenge (AT_MAC)                       |
     |----------------------------------------------------------->|
     |                                                             |
     |                                            EAP-Success      |
     |<-----------------------------------------------------------|
     |                                                             |
```
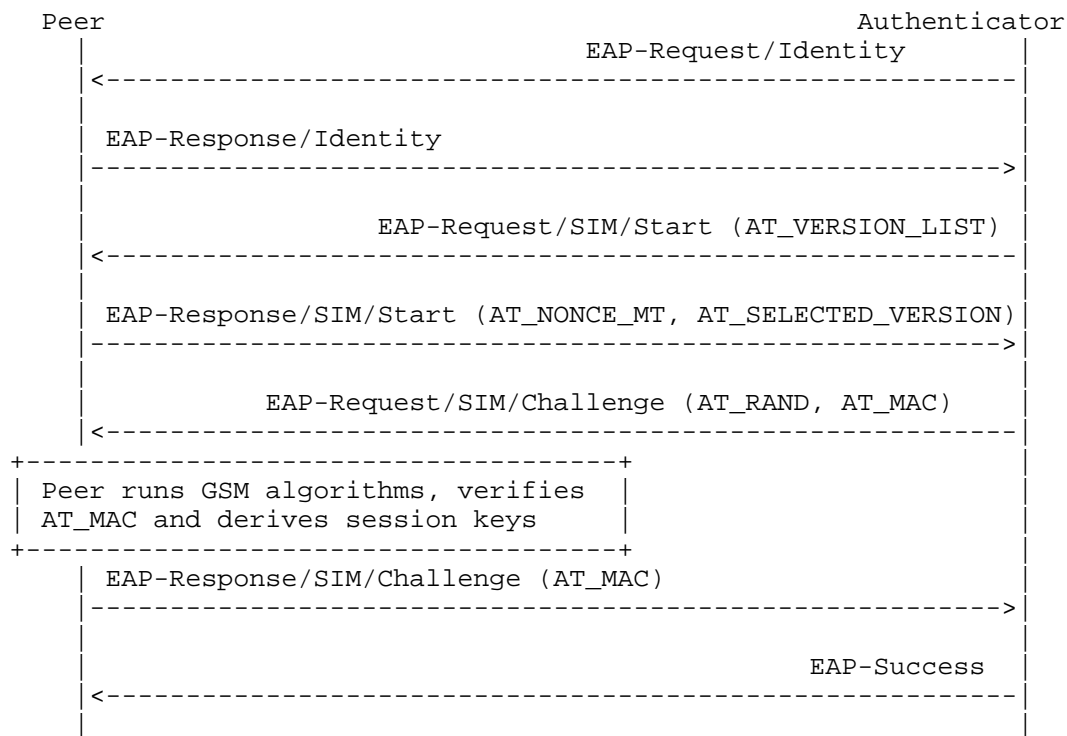
           Figure 1: EAP-SIM full authentication procedure

The first EAP Request issued by the authenticator is
EAP-Request/Identity. On full authentication, the peer's response
includes either the user's International Mobile Subscriber Identity
(IMSI) or a temporary identity (pseudonym) if identity privacy is in
effect, as specified in Section 4.2.

Following the peer's EAP-Response/Identity packet, the peer receives
EAP Requests of Type 18 (SIM) from the EAP server and sends the
corresponding EAP Responses. The EAP packets that are of the Type
SIM also have a Subtype field. On full authentication, the first
EAP-Request/SIM packet is of the Subtype 10 (Start). EAP-SIM packets
encapsulate parameters in attributes, encoded in a Type, Length,
Value format. The packet format and the use of attributes are
specified in Section 8.

The EAP-Request/SIM/Start packet contains the list of EAP-SIM
versions supported by the EAP server in the AT_VERSION_LIST
attribute. This packet may also include attributes for requesting
the subscriber identity, as specified in Section 4.2.

The peer responds to a EAP-Request/SIM/Start with the
EAP-Response/SIM/Start packet, which includes the AT_NONCE_MT
attribute that contains a random number NONCE_MT, chosen by the peer,
and the AT_SELECTED_VERSION attribute that contains the version
number selected by the peer. The version negotiation is protected by
including the version list and the selected version in the
calculation of keying material (Section 7).

After receiving the EAP Response/SIM/Start, the EAP server obtains n
GSM triplets for use in authenticating the subscriber, where n = 2 or
n = 3. From the triplets, the EAP server derives the keying
material, as specified in Section 7. The triplets may be obtained by
contacting an Authentication Centre (AuC) on the GSM network; per GSM
specifications, between 1 and 5 triplets may be obtained at a time.
Triplets may be stored in the EAP server for use at a later time, but
triplets MUST NOT be re-used, except in some error cases that are
specified in Section 10.9.

The next EAP Request the EAP Server issues is of the type SIM and
subtype Challenge (11). It contains the RAND challenges and a
message authentication code attribute AT_MAC to cover the challenges.
The AT_MAC attribute is a general message authentication code
attribute that is used in many EAP-SIM messages.

On receipt of the EAP-Request/SIM/Challenge message, the peer runs
the GSM authentication algorithm and calculates a copy of the message
authentication code. The peer then verifies that the calculated MAC
equals the received MAC. If the MAC's do not match, then the peer

sends the EAP-Response/SIM/Client-Error packet and the authentication
exchange terminates.

Since the RANDs given to a peer are accompanied by the message
authentication code AT_MAC, and since the peer's NONCE_MT value
contributes to AT_MAC, the peer is able to verify that the EAP-SIM
message is fresh (i.e., not a replay) and that the sender possesses
valid GSM triplets for the subscriber.

If all checks out, the peer responds with the
EAP-Response/SIM/Challenge, containing the AT_MAC attribute that
covers the peer's SRES response values (Section 9.4). The EAP server
verifies that the MAC is correct. Because protected success
indications are not used in this example, the EAP server sends the
EAP-Success packet, indicating that the authentication was
successful. (Protected success indications are discussed in
Section 6.2.) The EAP server may also include derived keying
material in the message it sends to the authenticator. The peer has
derived the same keying material, so the authenticator does not
forward the keying material to the peer along with EAP-Success.

EAP-SIM also includes a separate fast re-authentication procedure
that does not make use of the A3/A8 algorithms or the GSM
infrastructure. Fast re-authentication is based on keys derived on
full authentication. If the peer has maintained state information
for fast re-authentication and wants to use fast re-authentication,
then the peer indicates this by using a specific fast
re-authentication identity instead of the permanent identity or a
pseudonym identity. The fast re-authentication procedure is
described in Section 5.

4. Operation

4.1. Version Negotiation

EAP-SIM includes version negotiation so as to allow future
developments in the protocol. The version negotiation is performed
on full authentication and it uses two attributes, AT_VERSION_LIST,
which the server always includes in EAP-Request/SIM/Start, and
AT_SELECTED_VERSION, which the peer includes in
EAP-Response/SIM/Start on full authentication.

AT_VERSION_LIST includes the EAP-SIM versions supported by the
server. If AT_VERSION_LIST does not include a version that is
implemented by the peer and allowed in the peer's security policy,
then the peer MUST send the EAP-Response/SIM/Client-Error packet
(Section 9.7) to the server with the error code "unsupported
version". If a suitable version is included, then the peer includes

the AT_SELECTED_VERSION attribute, containing the selected version in
the EAP-Response/SIM/Start packet. The peer MUST only indicate a
version that is included in the AT_VERSION_LIST. If several versions
are acceptable, then the peer SHOULD choose the version that occurs
first in the version list.

The version number list of AT_VERSION_LIST and the selected version
of AT_SELECTED_VERSION are included in the key derivation procedure
(Section 7). If an attacker modifies either one of these attributes,
then the peer and the server derive different keying material.
Because K_aut keys are different, the server and peer calculate
different AT_MAC values. Hence, the peer detects that AT_MAC,
included in EAP-Request/SIM/Challenge, is incorrect and sends the
EAP-Response/SIM/Client-Error packet. The authentication procedure
terminates.

4.2. Identity Management

4.2.1. Format, Generation and Usage of Peer Identities

4.2.1.1. General

In the beginning of EAP authentication, the Authenticator or the EAP
server usually issues the EAP-Request/Identity packet to the peer.
The peer responds with the EAP-Response/Identity, which contains the
user's identity. The formats of these packets are specified in
[RFC3748].

GSM subscribers are identified with the International Mobile
Subscriber Identity (IMSI) [GSM-03.03]. The IMSI is a string of not
more than 15 digits. It is composed of a three digit Mobile Country
Code (MCC), a two or three digit Mobile Network Code (MNC), and a
Mobile Subscriber Identification Number (MSIN) of no more than 10
digits. MCC and MNC uniquely identify the GSM operator and help
identify the AuC from which the authentication vectors need to be
retrieved for this subscriber.

Internet AAA protocols identify users with the Network Access
Identifier (NAI) [RFC4282]. When used in a roaming environment, the
NAI is composed of a username and a realm, separated with "@"
(username@realm). The username portion identifies the subscriber
within the realm.

This section specifies the peer identity format used in EAP-SIM. In
this document, the term "identity" or "peer identity" refers to the
whole identity string that is used to identify the peer. The peer

identity may include a realm portion. "Username" refers to the
portion of the peer identity that identifies the user, i.e., the
username does not include the realm portion.

4.2.1.2. Identity Privacy Support

EAP-SIM includes optional identity privacy (anonymity) support that
can be used to hide the cleartext permanent identity and thereby make
the subscriber's EAP exchanges untraceable to eavesdroppers. Because
the permanent identity never changes, revealing it would help
observers to track the user. The permanent identity is usually based
on the IMSI, which may further help the tracking, because the same
identifier may be used in other contexts as well. Identity privacy
is based on temporary identities, or pseudonyms, which are equivalent
to but separate from the Temporary Mobile Subscriber Identities
(TMSI) that are used on cellular networks. Please see Section 12.2
for security considerations regarding identity privacy.

4.2.1.3. Username Types in EAP-SIM identities

There are three types of usernames in EAP-SIM peer identities:

(1) Permanent usernames. For example,
1123456789098765@myoperator.com might be a valid permanent identity.
In this example, 1123456789098765 is the permanent username.

(2) Pseudonym usernames. For example, 3s7ah6n9q@myoperator.com might
be a valid pseudonym identity. In this example, 3s7ah6n9q is the
pseudonym username.

(3) Fast re-authentication usernames. For example,
53953754@myoperator.com might be a valid fast re-authentication
identity. In this case, 53953754 is the fast re-authentication
username. Unlike permanent usernames and pseudonym usernames, fast
re-authentication usernames are one-time identifiers, which are not
re-used across EAP exchanges.

The first two types of identities are used only on full
authentication and the last one only on fast re-authentication. When
the optional identity privacy support is not used, the non-pseudonym
permanent identity is used on full authentication. The fast
re-authentication exchange is specified in Section 5.

4.2.1.4. Username Decoration

In some environments, the peer may need to decorate the identity by
prepending or appending the username with a string, in order to
indicate supplementary AAA routing information in addition to the NAI

realm. (The usage of an NAI realm portion is not considered
decoration.) Username decoration is out of the scope of this
document. However, it should be noted that username decoration might
prevent the server from recognizing a valid username. Hence,
although the peer MAY use username decoration in the identities that
the peer includes in EAP-Response/Identity, and although the EAP
server MAY accept a decorated peer username in this message, the peer
or the EAP server MUST NOT decorate any other peer identities that
are used in various EAP-SIM attributes. Only the identity used in
the EAP-Response/Identity may be decorated.

4.2.1.5. NAI Realm Portion

The peer MAY include a realm portion in the peer identity, as per the
NAI format. The use of a realm portion is not mandatory.

If a realm is used, the realm MAY be chosen by the subscriber's home
operator and it MAY be a configurable parameter in the EAP-SIM peer
implementation. In this case, the peer is typically configured with
the NAI realm of the home operator. Operators MAY reserve a specific
realm name for EAP-SIM users. This convention makes it easy to
recognize that the NAI identifies a GSM subscriber. Such a reserved
NAI realm may be a useful hint as to the first authentication method
to use during method negotiation. When the peer is using a pseudonym
username instead of the permanent username, the peer selects the
realm name portion similarly as it select the realm portion when
using the permanent username.

If no configured realm name is available, the peer MAY derive the
realm name from the MCC and MNC portions of the IMSI. A RECOMMENDED
way to derive the realm from the IMSI using the realm 3gppnetwork.org
is specified in [3GPP-TS-23.003].

Some old implementations derive the realm name from the IMSI by
concatenating "mnc", the MNC digits of IMSI, ".mcc", the MCC digits
of IMSI, and ".owlan.org". For example, if the IMSI is
123456789098765, and the MNC is three digits long, then the derived
realm name is "mnc456.mcc123.owlan.org". As there are no DNS servers
running at owlan.org, these realm names can only be used with
manually configured AAA routing. New implementations SHOULD use the
mechanism specified in [3GPP-TS-23.003] instead of owlan.org.

The IMSI is a string of digits without any explicit structure, so the
peer may not be able to determine the length of the MNC portion. If
the peer is not able to determine whether the MNC is two or three
digits long, the peer MAY use a 3-digit MNC. If the correct length
of the MNC is two, then the MNC used in the realm name includes the
first digit of the MSIN. Hence, when configuring AAA networks for

operators that have 2-digit MNCs, the network SHOULD also be prepared
for realm names with incorrect, 3-digit MNCs.

4.2.1.6. Format of the Permanent Username

The non-pseudonym permanent username SHOULD be derived from the IMSI.
In this case, the permanent username MUST be of the format "1" |
IMSI, where the character "|" denotes concatenation. In other words,
the first character of the username is the digit one (ASCII value 31
hexadecimal), followed by the IMSI. The IMSI is encoded as an ASCII
string that consists of not more than 15 decimal digits (ASCII values
between 30 and 39 hexadecimal), one character per IMSI digit, in the
order specified in [GSM-03.03]. For example, a permanent username
derived from the IMSI 295023820005424 would be encoded as the ASCII
string "1295023820005424" (byte values in hexadecimal notation: 31 32
39 35 30 32 33 38 32 30 30 30 35 34 32 34).

The EAP server MAY use the leading "1" as a hint to try EAP-SIM as
the first authentication method during method negotiation, rather
than, for example EAP/AKA. The EAP-SIM server MAY propose EAP-SIM,
even if the leading character was not "1".

Alternatively, an implementation MAY choose a permanent username that
is not based on the IMSI. In this case, the selection of the
username, its format, and its processing is out of the scope of this
document. In this case, the peer implementation MUST NOT prepend any
leading characters to the username.

4.2.1.7. Generating Pseudonyms and Fast Re-authentication Identities by
          the Server

Pseudonym usernames and fast re-authentication identities are
generated by the EAP server. The EAP server produces pseudonym
usernames and fast re-authentication identities in an
implementation-dependent manner. Only the EAP server needs to be
able to map the pseudonym username to the permanent identity, or to
recognize a fast re-authentication identity.

EAP-SIM includes no provisions to ensure that the same EAP server
that generated a pseudonym username will be used on the
authentication exchange when the pseudonym username is used. It is
recommended that the EAP servers implement some centralized mechanism
to allow all EAP servers of the home operator to map pseudonyms
generated by other severs to the permanent identity. If no such
mechanism is available, then the EAP server failing to understand a
pseudonym issued by another server can request the that peer send the
permanent identity.

When issuing a fast re-authentication identity, the EAP server may
include a realm name in the identity to make the fast
re-authentication request be forwarded to the same EAP server.

When generating fast re-authentication identities, the server SHOULD
choose a fresh, new fast re-authentication identity that is different
from the previous ones that were used after the same full
authentication exchange. A full authentication exchange and the
associated fast re-authentication exchanges are referred to here as
the same "full authentication context". The fast re-authentication
identity SHOULD include a random component. This random component
works as a full authentication context identifier. A
context-specific fast re-authentication identity can help the server
to detect whether its fast re-authentication state information
matches that of its peer (in other words, whether the state
information is from the same full authentication exchange). The
random component also makes the fast re-authentication identities
unpredictable, so an attacker cannot initiate a fast
re-authentication exchange to get the server's EAP-Request/SIM/
Re-authentication packet.

Transmitting pseudonyms and fast re-authentication identities from
the server to the peer is discussed in Section 4.2.1.8. The
pseudonym is transmitted as a username, without an NAI realm, and the
fast re-authentication identity is transmitted as a complete NAI,
including a realm portion if a realm is required. The realm is
included in the fast re-authentication identity to allow the server
to include a server-specific realm.

Regardless of the construction method, the pseudonym username MUST
conform to the grammar specified for the username portion of an NAI.
The fast re-authentication identity also MUST conform to the NAI
grammar. The EAP servers that the subscribers of an operator can use
MUST ensure that the pseudonym usernames and the username portions
used in fast re-authentication identities they generate are unique.

In any case, it is necessary that permanent usernames, pseudonym
usernames, and fast re-authentication usernames are separate and
recognizable from each other. It is also desirable that EAP-SIM and
EAP-AKA [EAP-AKA] usernames be distinguishable from each other as an
aid for the server on which method to offer.

In general, it is the task of the EAP server and the policies of its
administrator to ensure sufficient separation of the usernames.
Pseudonym usernames and fast re-authentication usernames are both
produced and used by the EAP server. The EAP server MUST compose
pseudonym usernames and fast re-authentication usernames so that it
can determine if an NAI username is an EAP-SIM pseudonym username or

an EAP-SIM fast re-authentication username. For instance, when the
usernames have been derived from the IMSI, the server could use
different leading characters in the pseudonym usernames and fast
re-authentication usernames (e.g., the pseudonym could begin with a
leading "3" character). When mapping a fast re-authentication
identity to a permanent identity, the server SHOULD only examine the
username portion of the fast re-authentication identity and ignore
the realm portion of the identity.

Because the peer may fail to save a pseudonym username sent in an
EAP-Request/SIM/Challenge, for example due to malfunction, the EAP
server SHOULD maintain at least the most recently used pseudonym
username in addition to the most recently issued pseudonym username.
If the authentication exchange is not completed successfully, then
the server SHOULD NOT overwrite the pseudonym username that was
issued during the most recent successful authentication exchange.

4.2.1.8. Transmitting Pseudonyms and Fast Re-authentication Identities
          to the Peer

   The server transmits pseudonym usernames and fast re-authentication
   identities to the peer in cipher, using the AT_ENCR_DATA attribute.

   The EAP-Request/SIM/Challenge message MAY include an encrypted
   pseudonym username and/or an encrypted fast re-authentication
   identity in the value field of the AT_ENCR_DATA attribute. Because
   identity privacy support and fast re-authentication are optional
   implementations, the peer MAY ignore the AT_ENCR_DATA attribute and
   always use the permanent identity. On fast re-authentication
   (discussed in Section 5), the server MAY include a new, encrypted
   fast re-authentication identity in the
   EAP-Request/SIM/Re-authentication message.

   On receipt of the EAP-Request/SIM/Challenge, the peer MAY decrypt the
   encrypted data in AT_ENCR_DATA. If the authentication exchange is
   successful, and the encrypted data includes a pseudonym username,
   then the peer may use the obtained pseudonym username on the next
   full authentication. If a fast re-authentication identity is
   included, then the peer MAY save it together with other fast
   re-authentication state information, as discussed in Section 5, for
   the next fast re-authentication. If the authentication exchange does
   not complete successfully, the peer MUST ignore the received
   pseudonym username and the fast re-authentication identity.

   If the peer does not receive a new pseudonym username in the
   EAP-Request/SIM/Challenge message, the peer MAY use an old pseudonym
   username instead of the permanent username on the next full
   authentication. The username portions of fast re-authentication

identities are one-time usernames, which the peer MUST NOT re-use.
When the peer uses a fast re-authentication identity in an EAP
exchange, the peer MUST discard the fast re-authentication identity
and not re-use it in another EAP authentication exchange, even if the
authentication exchange was not completed.

4.2.1.9. Usage of the Pseudonym by the Peer

   When the optional identity privacy support is used on full
   authentication, the peer MAY use a pseudonym username received as
   part of a previous full authentication sequence as the username
   portion of the NAI. The peer MUST NOT modify the pseudonym username
   received in AT_NEXT_PSEUDONYM. However, as discussed above, the peer
   MAY need to decorate the username in some environments by appending
   or prepending the username with a string that indicates supplementary
   AAA routing information.

   When using a pseudonym username in an environment where a realm
   portion is used, the peer concatenates the received pseudonym
   username with the "@" character and an NAI realm portion. The
   selection of the NAI realm is discussed above. The peer can select
   the realm portion similarly, regardless of whether it uses the
   permanent username or a pseudonym username.

4.2.1.10. Usage of the Fast Re-authentication Identity by the Peer

   On fast re-authentication, the peer uses the fast re-authentication
   identity that was received as part of the previous authentication
   sequence. A new re-authentication identity may be delivered as part
   of both full authentication and fast re-authentication. The peer
   MUST NOT modify the username part of the fast re-authentication
   identity received in AT_NEXT_REAUTH_ID, except in cases when username
   decoration is required. Even in these cases, the "root" fast
   re-authentication username must not be modified, but it may be
   appended or prepended with another string.

4.2.2. Communicating the Peer Identity to the Server

4.2.2.1. General

   The peer identity MAY be communicated to the server with the
   EAP-Response/Identity message. This message MAY contain the
   permanent identity, a pseudonym identity, or a fast re-authentication
   identity. If the peer uses the permanent identity or a pseudonym
   identity, which the server is able to map to the permanent identity,
   then the authentication proceeds as discussed in the overview of
   Section 3. If the peer uses a fast re-authentication identity, and
   if the fast re-authentication identity matches with a valid fast

re-authentication identity maintained by the server, and if the
server agrees to use fast re-authentication, then a fast
re-authentication exchange is performed, as described in Section 5.

The peer identity can also be transmitted from the peer to the server
using EAP-SIM messages instead of the EAP-Response/Identity. In this
case, the server includes an identity-requesting attribute
(AT_ANY_ID_REQ, AT_FULLAUTH_ID_REQ or AT_PERMANENT_ID_REQ) in the
EAP-Request/SIM/Start message, and the peer includes the AT_IDENTITY
attribute, which contains the peer's identity, in the
EAP-Response/SIM/Start message. The AT_ANY_ID_REQ attribute is a
general identity-requesting attribute, which the server uses if it
does not specify which kind of an identity the peer should return in
AT_IDENTITY. The server uses the AT_FULLAUTH_ID_REQ attribute to
request either the permanent identity or a pseudonym identity. The
server uses the AT_PERMANENT_ID_REQ attribute to request that the
peer send its permanent identity.

The identity format in the AT_IDENTITY attribute is the same as in
the EAP-Response/Identity packet (except that identity decoration is
not allowed). The AT_IDENTITY attribute contains a permanent
identity, a pseudonym identity, or a fast re-authentication identity.

Please note that the EAP-SIM peer and the EAP-SIM server only process
the AT_IDENTITY attribute; entities that only pass through EAP
packets do not process this attribute. Hence, the authenticator and
other intermediate AAA elements (such as possible AAA proxy servers)
will continue to refer to the peer with the original identity from
the EAP-Response/Identity packet unless the identity authenticated in
the AT_IDENTITY attribute is communicated to them in another way
within the AAA protocol.

4.2.2.2. Relying on EAP-Response/Identity Discouraged

The EAP-Response/Identity packet is not method-specific, so in many
implementations it may be handled by an EAP Framework. This
introduces an additional layer of processing between the EAP peer and
EAP server. The extra layer of processing may cache identity
responses or add decorations to the identity. A modification of the
identity response will cause the EAP peer and EAP server to use
different identities in the key derivation, which will cause the
protocol to fail.

For this reason, it is RECOMMENDED that the EAP peer and server use
the method-specific identity attributes in EAP-SIM, and the server is
strongly discouraged from relying upon the EAP-Response/Identity.

In particular, if the EAP server receives a decorated identity in
EAP-Response/Identity, then the EAP server MUST use the
identity-requesting attributes to request that the peer send an
unmodified and undecorated copy of the identity in AT_IDENTITY.

4.2.3. Choice of Identity for the EAP-Response/Identity

If EAP-SIM peer is started upon receiving an EAP-Request/Identity
message, then the peer MAY use an EAP-SIM identity in the EAP-
Response/Identity packet. In this case, the peer performs the
following steps.

If the peer has maintained fast re-authentication state information
and wants to use fast re-authentication, then the peer transmits the
fast re-authentication identity in EAP-Response/Identity.

Else, if the peer has a pseudonym username available, then the peer
transmits the pseudonym identity in EAP-Response/Identity.

In other cases, the peer transmits the permanent identity in
EAP-Response/Identity.

4.2.4. Server Operation in the Beginning of EAP-SIM Exchange

As discussed in Section 4.2.2.2, the server SHOULD NOT rely on an
identity string received in EAP-Response/Identity. Therefore, the
RECOMMENDED way to start an EAP-SIM exchange is to ignore any
received identity strings. The server SHOULD begin the EAP-SIM
exchange by issuing the EAP-Request/SIM/Start packet with an
identity-requesting attribute to indicate that the server wants the
peer to include an identity in the AT_IDENTITY attribute of the EAP-
Response/SIM/Start message. Three methods to request an identity
from the peer are discussed below.

If the server chooses not to ignore the contents of EAP-
Response/Identity, then the server may have already received an EAP-
SIM identity in this packet. However, if the EAP server has not
received any EAP-SIM peer identity (permanent identity, pseudonym
identity, or fast re-authentication identity) from the peer when
sending the first EAP-SIM request, or if the EAP server has received
an EAP-Response/Identity packet but the contents do not appear to be
a valid permanent identity, pseudonym identity or a re-authentication
identity, then the server MUST request an identity from the peer
using one of the methods below.

The server sends the EAP-Request/SIM/Start message with the
AT_PERMANENT_ID_REQ attribute to indicate that the server wants the
peer to include the permanent identity in the AT_IDENTITY attribute

of the EAP-Response/SIM/Start message. This is done in the following
cases:

o The server does not support fast re-authentication or identity
   privacy.

o The server decided to process a received identity, and the server
   recognizes the received identity as a pseudonym identity but the
   server is not able to map the pseudonym identity to a permanent
   identity.

The server issues the EAP-Request/SIM/Start packet with the
AT_FULLAUTH_ID_REQ attribute to indicate that the server wants the
peer to include a full authentication identity (pseudonym identity or
permanent identity) in the AT_IDENTITY attribute of the
EAP-Response/SIM/Start message. This is done in the following cases:

o The server does not support fast re-authentication and the server
   supports identity privacy.

o The server decided to process a received identity, and the server
   recognizes the received identity as a re-authentication identity
   but the server is not able to map the re-authentication identity
   to a permanent identity.

The server issues the EAP-Request/SIM/Start packet with the
AT_ANY_ID_REQ attribute to indicate that the server wants the peer to
include an identity in the AT_IDENTITY attribute of the
EAP-Response/SIM/Start message, and the server does not indicate any
preferred type for the identity. This is done in other cases, such
as when the server ignores a received EAP-Response/Identity, the
server does not have any identity, or the server does not recognize
the format of a received identity.

4.2.5. Processing of EAP-Request/SIM/Start by the Peer

   Upon receipt of an EAP-Request/SIM/Start message, the peer MUST
   perform the following steps.

   If the EAP-Request/SIM/Start does not include an identity request
   attribute, then the peer responds with EAP-Response/SIM/Start without
   AT_IDENTITY. The peer includes the AT_SELECTED_VERSION and
   AT_NONCE_MT attributes, because the exchange is a full authentication
   exchange.

   If the EAP-Request/SIM/Start includes AT_PERMANENT_ID_REQ, and if the
   peer does not have a pseudonym available, then the peer MUST respond
   with EAP-Response/SIM/Start and include the permanent identity in

AT_IDENTITY. If the peer has a pseudonym available, then the peer
MAY refuse to send the permanent identity; hence, in this case the
peer MUST either respond with EAP-Response/SIM/Start and include the
permanent identity in AT_IDENTITY or respond with EAP-Response/SIM/
Client-Error packet with the code "unable to process packet".

If the EAP-Request/SIM/Start includes AT_FULL_AUTH_ID_REQ, and if the
peer has a pseudonym available, then the peer SHOULD respond with
EAP-Response/SIM/Start and include the pseudonym identity in
AT_IDENTITY. If the peer does not have a pseudonym when it receives
this message, then the peer MUST respond with EAP-Response/SIM/Start
and include the permanent identity in AT_IDENTITY. The Peer MUST NOT
use a re-authentication identity in the AT_IDENTITY attribute.

If the EAP-Request/SIM/Start includes AT_ANY_ID_REQ, and if the peer
has maintained fast re-authentication state information and the peer
wants to use fast re-authentication, then the peer responds with
EAP-Response/SIM/Start and includes the fast re-authentication
identity in AT_IDENTITY. Else, if the peer has a pseudonym identity
available, then the peer responds with EAP-Response/SIM/Start and
includes the pseudonym identity in AT_IDENTITY. Else, the peer
responds with EAP-Response/SIM/Start and includes the permanent
identity in AT_IDENTITY.

An EAP-SIM exchange may include several EAP/SIM/Start rounds. The
server may issue a second EAP-Request/SIM/Start if it was not able to
recognize the identity that the peer used in the previous AT_IDENTITY
attribute. At most, three EAP/SIM/Start rounds can be used, so the
peer MUST NOT respond to more than three EAP-Request/SIM/Start
messages within an EAP exchange. The peer MUST verify that the
sequence of EAP-Request/SIM/Start packets that the peer receives
comply with the sequencing rules defined in this document. That is,
AT_ANY_ID_REQ can only be used in the first EAP-Request/SIM/Start; in
other words, AT_ANY_ID_REQ MUST NOT be used in the second or third
EAP-Request/SIM/Start. AT_FULLAUTH_ID_REQ MUST NOT be used if the
previous EAP-Request/SIM/Start included AT_PERMANENT_ID_REQ. The
peer operation, in cases when it receives an unexpected attribute or
an unexpected message, is specified in Section 6.3.1.

4.2.6. Attacks Against Identity Privacy

The section above specifies two possible ways the peer can operate
upon receipt of AT_PERMANENT_ID_REQ. This is because a received
AT_PERMANENT_ID_REQ does not necessarily originate from the valid
network, but an active attacker may transmit an EAP-Request/SIM/
Start packet with an AT_PERMANENT_ID_REQ attribute to the peer, in an
effort to find out the true identity of the user. If the peer does
not want to reveal its permanent identity, then the peer sends the

EAP-Response/SIM/Client-Error packet with the error code "unable to
process packet", and the authentication exchange terminates.

Basically, there are two different policies that the peer can employ
with regard to AT_PERMANENT_ID_REQ. A "conservative" peer assumes
that the network is able to maintain pseudonyms robustly. Therefore,
if a conservative peer has a pseudonym username, the peer responds
with EAP-Response/SIM/Client-Error to the EAP packet with
AT_PERMANENT_ID_REQ, because the peer believes that the valid network
is able to map the pseudonym identity to the peer's permanent
identity. (Alternatively, the conservative peer may accept
AT_PERMANENT_ID_REQ in certain circumstances, for example, if the
pseudonym was received a long time ago.) The benefit of this policy
is that it protects the peer against active attacks on anonymity. On
the other hand, a "liberal" peer always accepts the
AT_PERMANENT_ID_REQ and responds with the permanent identity. The
benefit of this policy is that it works even if the valid network
sometimes loses pseudonyms and is not able to map them to the
permanent identity.

4.2.7. Processing of AT_IDENTITY by the Server

When the server receives an EAP-Response/SIM/Start message with the
AT_IDENTITY (in response to the server's identity requesting
attribute), the server MUST operate as follows.

If the server used AT_PERMANENT_ID_REQ, and if the AT_IDENTITY does
not contain a valid permanent identity, then the server sends
EAP-Request/SIM/Notification with AT_NOTIFICATION code "General
failure" (16384), and the EAP exchange terminates. If the server
recognizes the permanent identity and is able to continue, then the
server proceeds with full authentication by sending EAP-Request/SIM/
Challenge.

If the server used AT_FULLAUTH_ID_REQ, and if AT_IDENTITY contains a
valid permanent identity or a pseudonym identity that the server can
map to a valid permanent identity, then the server proceeds with full
authentication by sending EAP-Request/SIM/Challenge. If AT_IDENTITY
contains a pseudonym identity that the server is not able to map to a
valid permanent identity, or an identity that the server is not able
to recognize or classify, then the server sends EAP-Request/SIM/Start
with AT_PERMANENT_ID_REQ.

If the server used AT_ANY_ID_REQ, and if the AT_IDENTITY contains a
valid permanent identity or a pseudonym identity that the server can
map to a valid permanent identity, then the server proceeds with full
authentication by sending EAP-Request/SIM/Challenge.

   If the server used AT_ANY_ID_REQ, and if AT_IDENTITY contains a valid
   fast re-authentication identity and the server agrees on using
   re-authentication, then the server proceeds with fast
   re-authentication by sending EAP-Request/SIM/Re-authentication
   (Section 5).

   If the server used AT_ANY_ID_REQ, and if the peer sent an
   EAP-Response/SIM/Start with only AT_IDENTITY (indicating
   re-authentication), but the server is not able to map the identity to
   a permanent identity, then the server sends EAP-Request/SIM/Start
   with AT_FULLAUTH_ID_REQ.

   If the server used AT_ANY_ID_REQ, and if AT_IDENTITY contains a valid
   fast re-authentication identity that the server is able to map to a
   permanent identity, and if the server does not want to use fast
   re-authentication, then the server sends EAP-Request/SIM/Start
   without any identity requesting attributes.

   If the server used AT_ANY_ID_REQ, and AT_IDENTITY contains an
   identity that the server recognizes as a pseudonym identity but the
   server is not able to map the pseudonym identity to a permanent
   identity, then the server sends EAP-Request/SIM/Start with
   AT_PERMANENT_ID_REQ.

   If the server used AT_ANY_ID_REQ, and AT_IDENTITY contains an
   identity that the server is not able to recognize or classify, then
   the server sends EAP-Request/SIM/Start with AT_FULLAUTH_ID_REQ.

4.3. Message Sequence Examples (Informative)

   This section contains non-normative message sequence examples to
   illustrate how the peer identity can be communicated to the server.

4.3.1. Full Authentication

   This case for full authentication is illustrated below in Figure 2.
   In this case, AT_IDENTITY contains either the permanent identity or a
   pseudonym identity. The same sequence is also used in case the
   server uses the AT_FULLAUTH_ID_REQ in EAP-Request/SIM/Start.

```
     Peer                                            Authenticator
       |                                                  |
       |                        +-----------------------------+
       |                        | Server does not have a      |
       |                        | Subscriber identity available|
       |                        | When starting EAP-SIM       |
       |                        +-----------------------------+
       |                                                  |
       |            EAP-Request/SIM/Start                 |
       |            (AT_ANY_ID_REQ, AT_VERSION_LIST)      |
       |<-------------------------------------------------|
       |                                                  |
       |                                                  |
       | EAP-Response/SIM/Start                           |
       | (AT_IDENTITY, AT_NONCE_MT,                       |
       |  AT_SELECTED_VERSION)                            |
       |------------------------------------------------->|
       |                                                  |
```
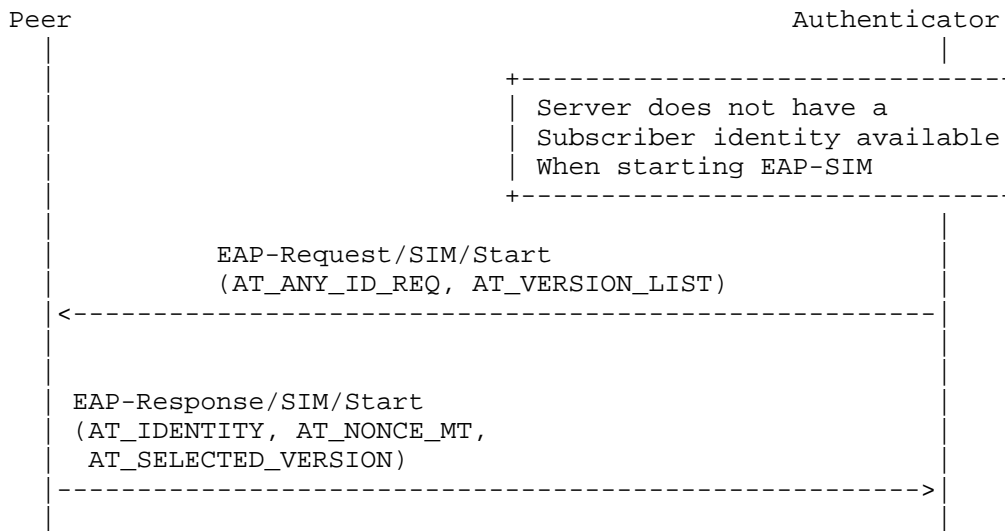
          Figure 2: Requesting any identity, full authentication

   If the peer uses its full authentication identity and the AT_IDENTITY
   attribute contains a valid permanent identity or a valid pseudonym
   identity that the EAP server is able to map to the permanent
   identity, then the full authentication sequence proceeds as usual
   with the EAP Server issuing the EAP-Request/SIM/Challenge message.

4.3.2. Fast Re-authentication

   The case when the server uses the AT_ANY_ID_REQ and the peer wants to
   perform fast re-authentication is illustrated below in Figure 3.

```
     Peer                                              Authenticator
      |                                                      |
      |                              +------------------------------+
      |                              | Server does not have a       |
      |                              | Subscriber identity available|
      |                              | When starting EAP-SIM        |
      |                              +------------------------------+
      |                                                      |
      |           EAP-Request/SIM/Start                      |
      |           (AT_ANY_ID_REQ, AT_VERSION_LIST)           |
      |<-----------------------------------------------------|
      |                                                      |
      |                                                      |
      | EAP-Response/SIM/Start                               |
      | (AT_IDENTITY containing a fast re-auth. identity)    |
      |----------------------------------------------------->|
      |                                                      |
```
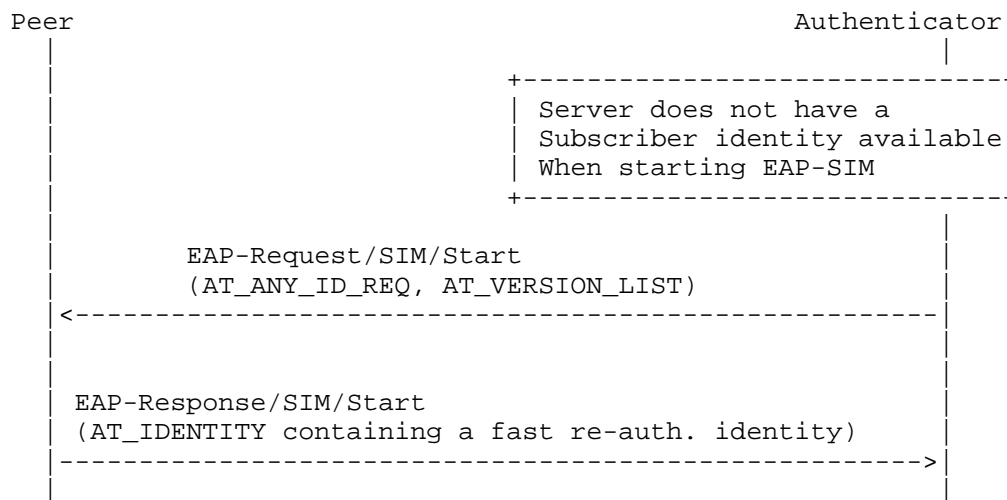
            Figure 3: Requesting any identity, fast re-authentication

   On fast re-authentication, if the AT_IDENTITY attribute contains a
   valid fast re-authentication identity and the server agrees on using
   fast re-authentication, then the server proceeds with the fast
   re-authentication sequence and issues the EAP-Request/SIM/
   Re-authentication packet, as specified in Section 5.

4.3.3. Fall Back to Full Authentication

   Figure 4 illustrates cases in which the server does not recognize the
   fast re-authentication identity the peer used in AT_IDENTITY, and
   issues a second EAP-Request/SIM/Start message.

```
        Peer                                               Authenticator
         |                                                        |
         |                        +------------------------------+
         |                        | Server does not have a       |
         |                        | Subscriber identity available|
         |                        | When starting EAP-SIM        |
         |                        +------------------------------+
         |                                                        |
         |          EAP-Request/SIM/Start                         |
         |          (AT_ANY_ID_REQ, AT_VERSION_LIST)              |
         |<-------------------------------------------------------|
         |                                                        |
         |                                                        |
         | EAP-Response/SIM/Start                                 |
         | (AT_IDENTITY containing a fast re-auth. identity)      |
         |------------------------------------------------------->|
         |                                                        |
         |                        +------------------------------+
         |                        | Server does not recognize    |
         |                        | The fast re-auth.            |
         |                        | Identity                     |
         |                        +------------------------------+
         |                                                        |
         |        EAP-Request/SIM/Start                           |
         |        (AT_FULLAUTH_ID_REQ, AT_VERSION_LIST)           |
         |<-------------------------------------------------------|
         |                                                        |
         |                                                        |
         | EAP-Response/SIM/Start                                 |
         | (AT_IDENTITY with a full-auth. identity, AT_NONCE_MT,  |
         |  AT_SELECTED_VERSION)                                  |
         |------------------------------------------------------->|
         |                                                        |
```

              Figure 4: Fall back to full authentication

4.3.4. Requesting the Permanent Identity 1

   Figure 5 illustrates the case in which the EAP server fails to map
   the pseudonym identity included in the EAP-Response/Identity packet
   to a valid permanent identity.

```
      Peer                                              Authenticator
        |                                                    |
        |                            EAP-Request/Identity    |
        |<---------------------------------------------------|
        |                                                    |
        | EAP-Response/Identity                              |
        | (Includes a pseudonym)                             |
        |--------------------------------------------------->|
        |                                                    |
        |                        +-------------------------------+
        |                        | Server fails to map the       |
        |                        | Pseudonym to a permanent id.  |
        |                        +-------------------------------+
        |  EAP-Request/SIM/Start                             |
        |  (AT_PERMANENT_ID_REQ, AT_VERSION_LIST)            |
        |<---------------------------------------------------|
        |                                                    |
        | EAP-Response/SIM/Start                             |
        | (AT_IDENTITY with permanent identity, AT_NONCE_MT, |
        |  AT_SELECTED_VERSION)                              |
        |--------------------------------------------------->|
        |                                                    |
```
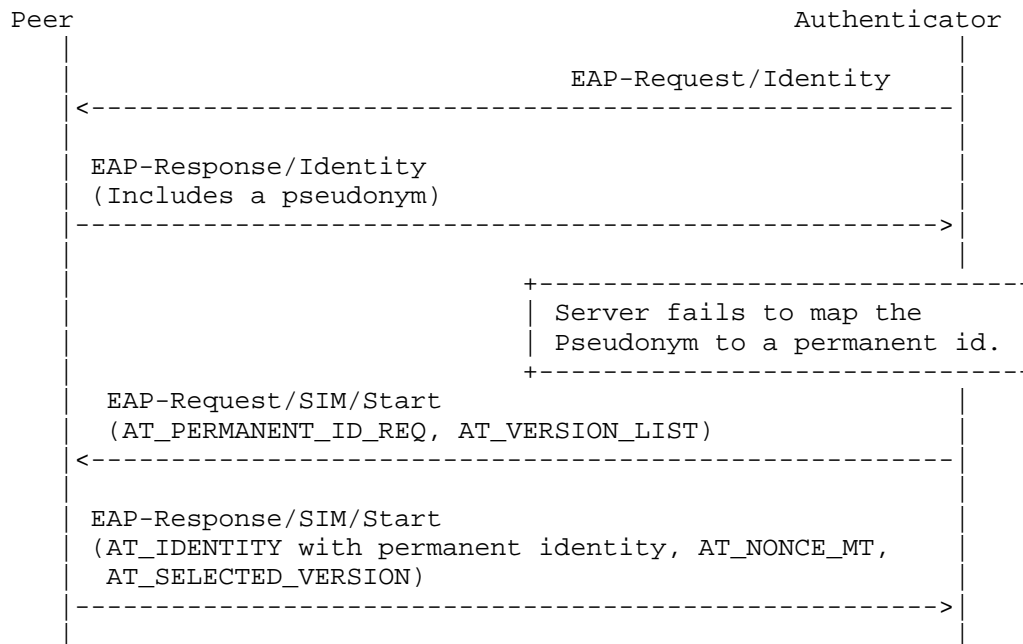
            Figure 5: Requesting the permanent identity

   If the server recognizes the permanent identity, then the
   authentication sequence proceeds as usual with the EAP Server issuing
   the EAP-Request/SIM/Challenge message.

4.3.5. Requesting the Permanent Identity 2

   Figure 6 illustrates the case in which the EAP server fails to map
   the pseudonym included in the AT_IDENTITY attribute to a valid
   permanent identity.

```
        Peer                                              Authenticator
          |                                                   |
          |                              +------------------------------+
          |                              | Server does not have a       |
          |                              | Subscriber identity available|
          |                              | When starting EAP-SIM        |
          |                              +------------------------------+
          |            EAP-Request/SIM/Start                  |
          |            (AT_ANY_ID_REQ, AT_VERSION_LIST)        |
          |<--------------------------------------------------|
          |                                                   |
          |EAP-Response/SIM/Start                             |
          |(AT_IDENTITY with a pseudonym identity, AT_NONCE_MT,|
          | AT_SELECTED_VERSION)                              |
          |-------------------------------------------------->|
          |                              +------------------------------+
          |                              | Server fails to map the      |
          |                              | Pseudonym in AT_IDENTITY      |
          |                              | to a valid permanent identity |
          |                              +------------------------------+
          |                                                   |
          |            EAP-Request/SIM/Start                  |
          |            (AT_PERMANENT_ID_REQ, AT_VERSION_LIST) |
          |<--------------------------------------------------|
          |                                                   |
          | EAP-Response/SIM/Start                            |
          | (AT_IDENTITY with permanent identity,            |
          |  AT_NONCE_MT, AT_SELECTED_VERSION)               |
          |-------------------------------------------------->|
          |                                                   |
```

   Figure 6: Requesting a permanent identity (two EAP-SIM Start rounds)

4.3.6. Three EAP-SIM/Start Roundtrips

   In the worst case, there are three EAP/SIM/Start round trips before
   the server obtains an acceptable identity. This case is illustrated
   in Figure 7.

```
   Peer                                               Authenticator
    |                                                      |
    |                      +------------------------------+
    |                      | Server does not have a       |
    |                      | Subscriber identity available|
    |                      | When starting EAP-SIM        |
    |                      +------------------------------+
    |         EAP-Request/SIM/Start                        |
    |         (Includes AT_ANY_ID_REQ, AT_VERSION_LIST)    |
    |<-----------------------------------------------------|
    |                                                      |
    | EAP-Response/SIM/Start                               |
    | (AT_IDENTITY with fast re-auth. identity)            |
    |----------------------------------------------------->|
    |                                                      |
    |                      +------------------------------+
    |                      | Server does not accept       |
    |                      | The fast re-auth.            |
    |                      | Identity                     |
    |                      +------------------------------+
    |      EAP-Request/SIM/Start                           |
    |      (AT_FULLAUTH_ID_REQ, AT_VERSION_LIST)           |
    |<-----------------------------------------------------|
    |                                                      |
    :                                                      :
    :                                                      :
    :                                                      :
    :                                                      :
    |EAP-Response/SIM/Start                                |
    |(AT_IDENTITY with a pseudonym identity, AT_NONCE_MT,  |
    | AT_SELECTED_VERSION)                                 |
    |----------------------------------------------------->|
    |                                                      |
    |                      +------------------------------+
    |                      | Server fails to map the      |
    |                      | Pseudonym in AT_IDENTITY      |
    |                      | to a valid permanent identity |
    |                      +------------------------------+
    |          EAP-Request/SIM/Start                       |
    |          (AT_PERMANENT_ID_REQ, AT_VERSION_LIST)      |
    |<-----------------------------------------------------|
    |                                                      |
    | EAP-Response/SIM/Start                               |
    | (AT_IDENTITY with permanent identity, AT_NONCE_MT,   |
    |  AT_SELECTED_VERSION)                                |
    |----------------------------------------------------->|
    |                                                      |
```
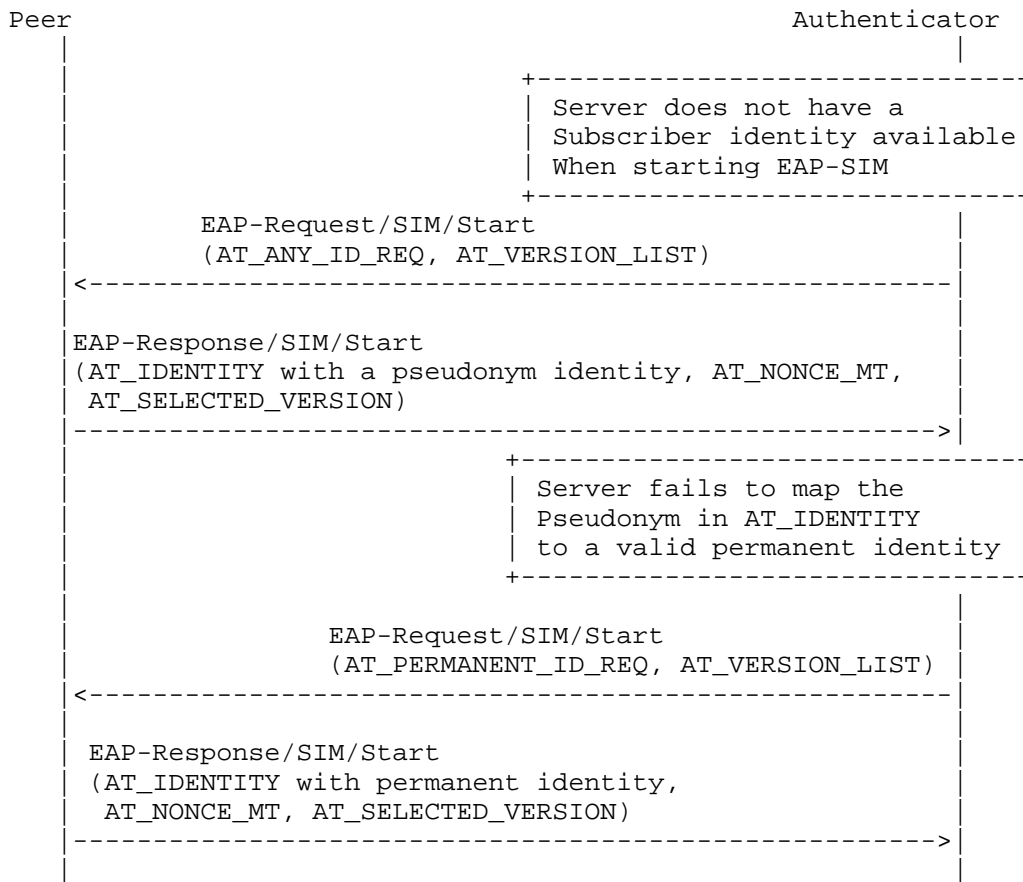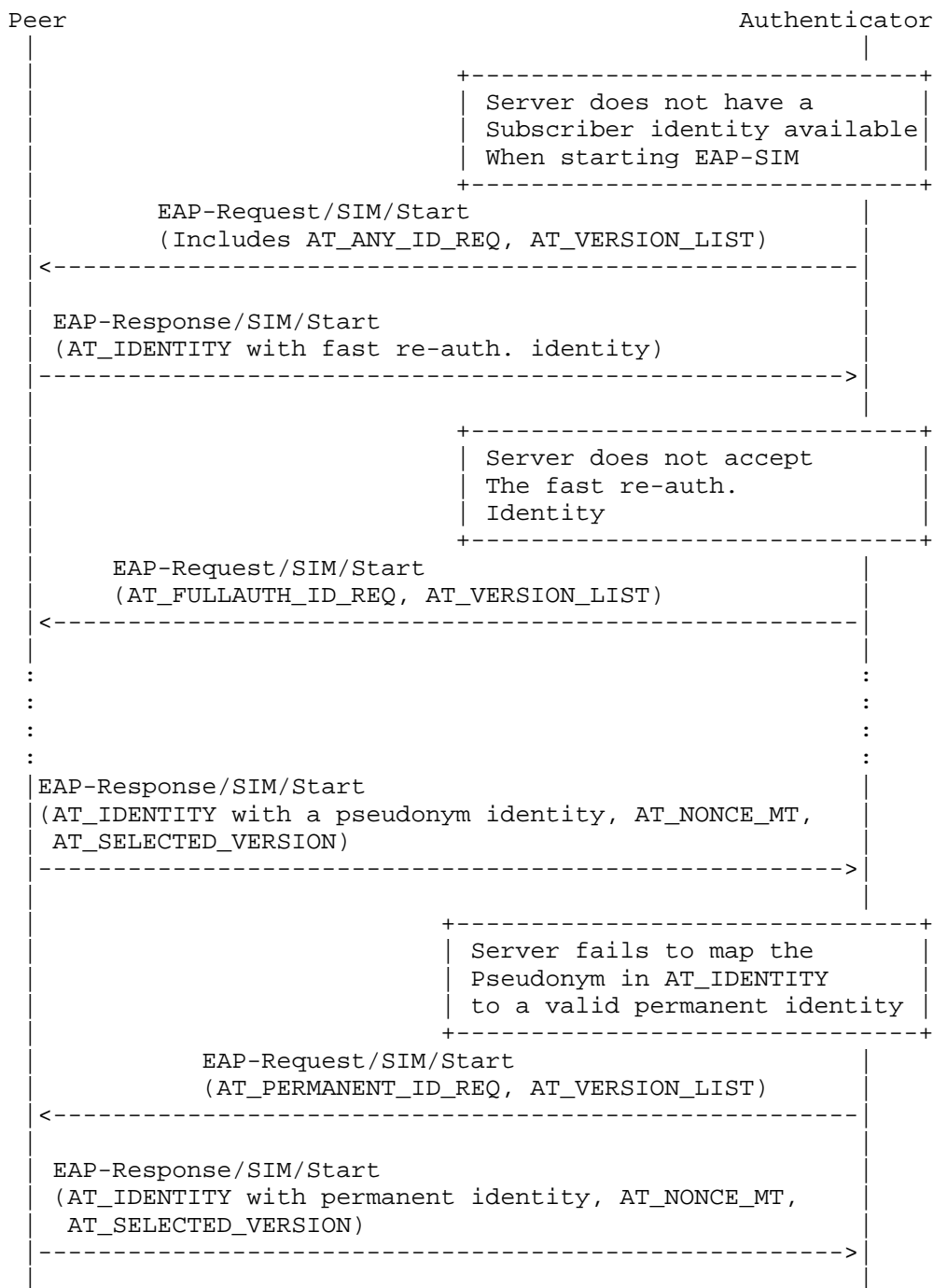
                Figure 7: Three EAP-SIM Start rounds

   After the last EAP-Response/SIM/Start message, the full
   authentication sequence proceeds as usual. If the EAP Server
   recognizes the permanent identity and is able to proceed, the server
   issues the EAP-Request/SIM/Challenge message.

5. Fast Re-Authentication

5.1. General

   In some environments, EAP authentication may be performed frequently.
   Because the EAP-SIM full authentication procedure makes use of the
   GSM SIM A3/A8 algorithms, and therefore requires 2 or 3 fresh
   triplets from the Authentication Centre, the full authentication
   procedure is not very well suited for frequent use. Therefore,
   EAP-SIM includes a more inexpensive fast re-authentication procedure
   that does not make use of the SIM A3/A8 algorithms and does not need
   new triplets from the Authentication Centre. Re-authentication can
   be performed in fewer roundtrips than the full authentication.

   Fast re-authentication is optional to implement for both the EAP-SIM
   server and peer. On each EAP authentication, either one of the
   entities may also fall back on full authentication if it does not
   want to use fast re-authentication.

   Fast re-authentication is based on the keys derived on the preceding
   full authentication. The same K_aut and K_encr keys that were used
   in full authentication are used to protect EAP-SIM packets and
   attributes, and the original Master Key from full authentication is
   used to generate a fresh Master Session Key, as specified in Section
   7.

   The fast re-authentication exchange makes use of an unsigned 16-bit
   counter, included in the AT_COUNTER attribute. The counter has three
   goals: 1) it can be used to limit the number of successive
   reauthentication exchanges without full authentication 2) it
   contributes to the keying material, and 3) it protects the peer and
   the server from replays. On full authentication, both the server and
   the peer initialize the counter to one. The counter value of at
   least one is used on the first fast re-authentication. On subsequent
   fast re-authentications, the counter MUST be greater than on any of
   the previous re-authentications. For example, on the second fast
   re-authentication, the counter value is two or greater. The
   AT_COUNTER attribute is encrypted.

   Both the peer and the EAP server maintain a copy of the counter. The
   EAP server sends its counter value to the peer in the fast
   re-authentication request. The peer MUST verify that its counter
   value is less than or equal to the value sent by the EAP server.

   The server includes an encrypted server random nonce (AT_NONCE_S) in
   the fast re-authentication request. The AT_MAC attribute in the
   peer's response is calculated over NONCE_S to provide a
   challenge/response authentication scheme. The NONCE_S also
   contributes to the new Master Session Key.

   Both the peer and the server SHOULD have an upper limit for the
   number of subsequent fast re-authentications allowed before a full
   authentication needs to be performed. Because a 16-bit counter is
   used in fast re-authentication, the theoretical maximum number of
   re-authentications is reached when the counter value reaches FFFF
   hexadecimal.

   In order to use fast re-authentication, the peer and the EAP server
   need to store the following values: Master Key, latest counter value
   and the next fast re-authentication identity. K_aut, K_encr may
   either be stored or derived again from MK. The server may also need
   to store the permanent identity of the user.

5.2. Comparison to UMTS AKA

   When analyzing the fast re-authentication exchange, it may be helpful
   to compare it with the UMTS Authentication and Key Agreement (AKA)
   exchange, which it resembles closely. The counter corresponds to the
   UMTS AKA sequence number, NONCE_S corresponds to RAND, AT_MAC in
   EAP-Request/SIM/Re-authentication corresponds to AUTN, the AT_MAC in
   EAP-Response/SIM/Re-authentication corresponds to RES,
   AT_COUNTER_TOO_SMALL corresponds to AUTS, and encrypting the counter
   corresponds to the usage of the Anonymity Key. Also, the key
   generation on fast re-authentication, with regard to random or fresh
   material, is similar to UMTS AKA -- the server generates the NONCE_S
   and counter values, and the peer only verifies that the counter value
   is fresh.

   It should also be noted that encrypting the AT_NONCE_S, AT_COUNTER,
   or AT_COUNTER_TOO_SMALL attributes is not important to the security
   of the fast re-authentication exchange.

5.3. Fast Re-authentication Identity

   The fast re-authentication procedure makes use of separate
   re-authentication user identities. Pseudonyms and the permanent
   identity are reserved for full authentication only. If a
   re-authentication identity is lost and the network does not recognize
   it, the EAP server can fall back on full authentication.

If the EAP server supports fast re-authentication, it MAY include the
skippable AT_NEXT_REAUTH_ID attribute in the encrypted data of
EAP-Request/SIM/Challenge message (Section 9.3). This attribute
contains a new fast re-authentication identity for the next fast
re-authentication. The attribute also works as a capability flag
that, indicating that the server supports fast re-authentication, and
that the server wants to continue using fast re-authentication within
the current context. The peer MAY ignore this attribute, in which
case it MUST use full authentication next time. If the peer wants to
use re-authentication, it uses this fast re-authentication identity
on next authentication. Even if the peer has a fast
re-authentication identity, the peer MAY discard the fast
re-authentication identity and use a pseudonym or the permanent
identity instead, in which case full authentication MUST be
performed. If the EAP server does not include the AT_NEXT_REAUTH_ID
in the encrypted data of EAP-Request/SIM/Challenge or
EAP-Request/SIM/ Re-authentication, then the peer MUST discard its
current fast re-authentication state information and perform a full
authentication next time.

In environments where a realm portion is needed in the peer identity,
the fast re-authentication identity received in AT_NEXT_REAUTH_ID
MUST contain both a username portion and a realm portion, as per the
NAI format. The EAP Server can choose an appropriate realm part in
order to have the AAA infrastructure route subsequent fast
re-authentication related requests to the same AAA server. For
example, the realm part MAY include a portion that is specific to the
AAA server. Hence, it is sufficient to store the context required
for fast re-authentication in the AAA server that performed the full
authentication.

The peer MAY use the fast re-authentication identity in the
EAP-Response/Identity packet or, in response to the server's
AT_ANY_ID_REQ attribute, the peer MAY use the fast re-authentication
identity in the AT_IDENTITY attribute of the EAP-Response/SIM/Start
packet.

The peer MUST NOT modify the username portion of the fast
re-authentication identity, but the peer MAY modify the realm portion
or replace it with another realm portion. The peer might need to
modify the realm in order to influence the AAA routing, for example,
to make sure that the correct server is reached. It should be noted
that sharing the same fast re-authentication key among several
servers may have security risks, so changing the realm portion of the
NAI in order to change the EAP server is not desirable.

Even if the peer uses a fast re-authentication identity, the server
may want to fall back on full authentication, for example because the
server does not recognize the fast re-authentication identity or does
not want to use fast re-authentication. In this case, the server
starts the full authentication procedure by issuing an
EAP-Request/SIM/Start packet. This packet always starts a full
authentication sequence if it does not include the AT_ANY_ID_REQ
attribute. If the server was not able to recover the peer's identity
from the fast re-authentication identity, the server includes either
the AT_FULLAUTH_ID_REQ or the AT_PERMANENT_ID_REQ attribute in this
EAP request.

5.4. Fast Re-authentication Procedure

Figure 8 illustrates the fast re-authentication procedure. In this
example, the optional protected success indication is not used.
Encrypted attributes are denoted with '*'. The peer uses its
re-authentication identity in the EAP-Response/Identity packet. As
discussed above, an alternative way to communicate the
re-authentication identity to the server is for the peer to use the
AT_IDENTITY attribute in the EAP-Response/SIM/Start message. This
latter case is not illustrated in the figure below, and it is only
possible when the server requests that the peer send its identity by
including the AT_ANY_ID_REQ attribute in the EAP-Request/SIM/Start
packet.

If the server recognizes the identity as a valid fast
re-authentication identity, and if the server agrees to use fast
re-authentication, then the server sends the EAP-Request/SIM/
Re-authentication packet to the peer. This packet MUST include the
encrypted AT_COUNTER attribute, with a fresh counter value, the
encrypted AT_NONCE_S attribute that contains a random number chosen
by the server, the AT_ENCR_DATA and the AT_IV attributes used for
encryption, and the AT_MAC attribute that contains a message
authentication code over the packet. The packet MAY also include an
encrypted AT_NEXT_REAUTH_ID attribute that contains the next fast
re-authentication identity.

Fast re-authentication identities are one-time identities. If the
peer does not receive a new fast re-authentication identity, it MUST
use either the permanent identity or a pseudonym identity on the next
authentication to initiate full authentication.

The peer verifies that AT_MAC is correct, and that the counter value
is fresh (greater than any previously used value). The peer MAY save
the next fast re-authentication identity from the encrypted
AT_NEXT_REAUTH_ID for next time. If all checks are successful, the
peer responds with the EAP-Response/SIM/Re-authentication packet,

including the AT_COUNTER attribute with the same counter value and
AT_MAC attribute.

The server verifies the AT_MAC attribute and also verifies that the
counter value is the same that it used in the EAP-Request/SIM/
Re-authentication packet. If these checks are successful, the
re-authentication has succeeded and the server sends the EAP-Success
packet to the peer.

If protected success indications (Section 6.2) were used, the
EAP-Success packet would be preceded by an EAP-SIM notification
round.

```
            Peer                                      Authenticator
              |                                            |
              |                        EAP-Request/Identity |
              |<-------------------------------------------------------|
              |                                            |
              | EAP-Response/Identity                      |
              | (Includes a fast re-authentication identity) |
              |------------------------------------------------------->|
              |                                            |
              |                  +-------------------------------+
              |                  | Server recognizes the identity |
              |                  | and agrees to use fast         |
              |                  | re-authentication              |
              |                  +-------------------------------+
              |                                            |
              :                                            :
              :                                            :
              :                                            :
              :                                            :
              |  EAP-Request/SIM/Re-authentication         |
              | (AT_IV, AT_ENCR_DATA, *AT_COUNTER,         |
              |  *AT_NONCE_S, *AT_NEXT_REAUTH_ID, AT_MAC)  |
              |<-------------------------------------------------------|
              |                                            |
     +-------------------------------------------------+    |
     | Peer verifies AT_MAC and the freshness of       |    |
     | the counter. Peer MAY store the new fast re-    |    |
     | authentication identity for next re-auth.       |    |
     +-------------------------------------------------+    |
              |                                            |
              | EAP-Response/SIM/Re-authentication         |
              | (AT_IV, AT_ENCR_DATA, *AT_COUNTER with same value, |
              |  AT_MAC)                                   |
              |------------------------------------------------------->|
              |                  +-------------------------------+
              |                  | Server verifies AT_MAC and    |
              |                  | the counter                   |
              |                  +-------------------------------+
              |                                            |
              |                             EAP-Success    |
              |<-------------------------------------------------------|
              |                                            |
```
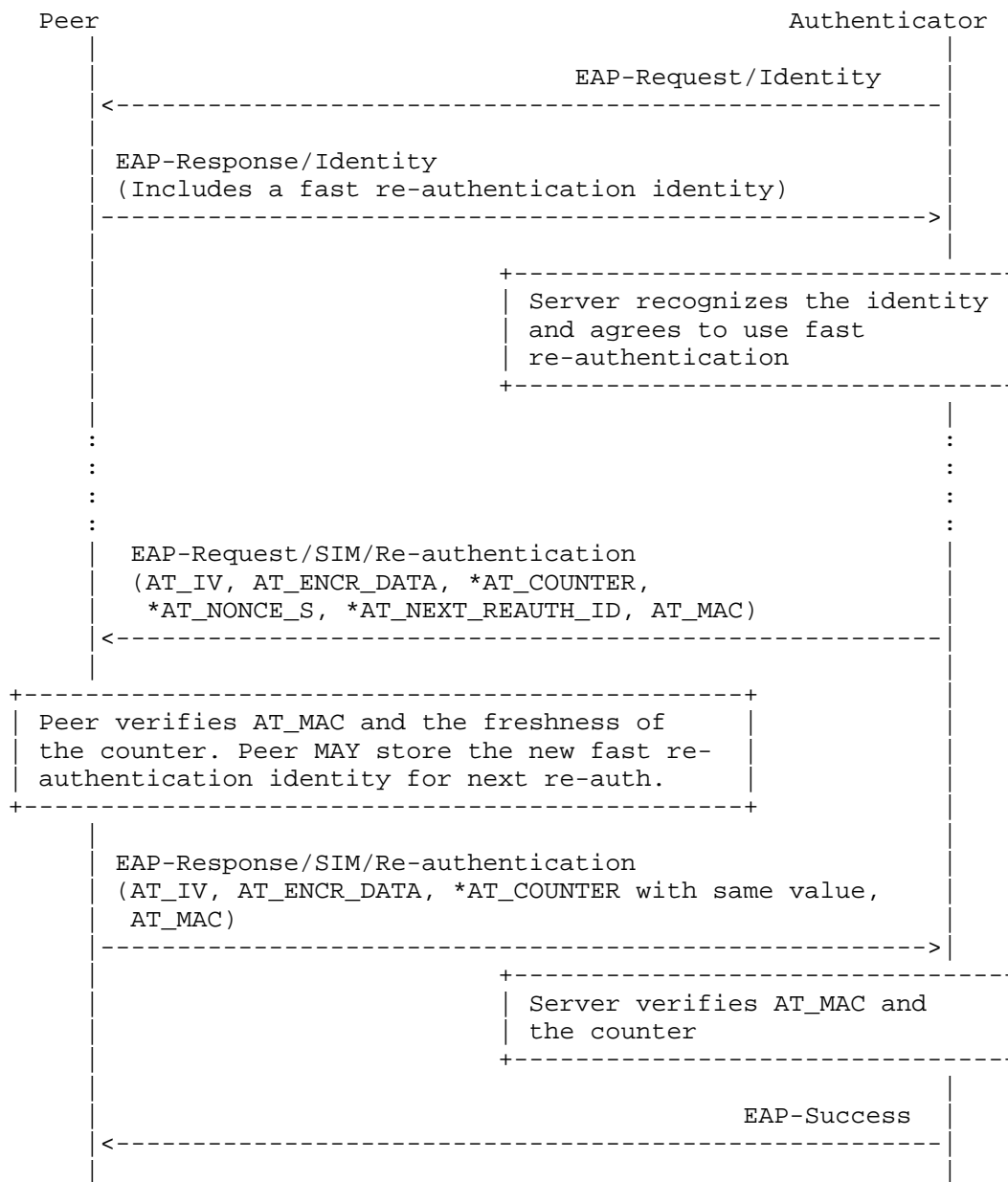
                   Figure 8: Fast Re-authentication

5.5. Fast Re-authentication Procedure when Counter Is Too Small

   If the peer does not accept the counter value of EAP-Request/SIM/
   Re-authentication, it indicates the counter synchronization problem
   by including the encrypted AT_COUNTER_TOO_SMALL in EAP-Response/SIM/
   Re-authentication. The server responds with EAP-Request/SIM/Start to
   initiate a normal full authentication procedure. This is illustrated
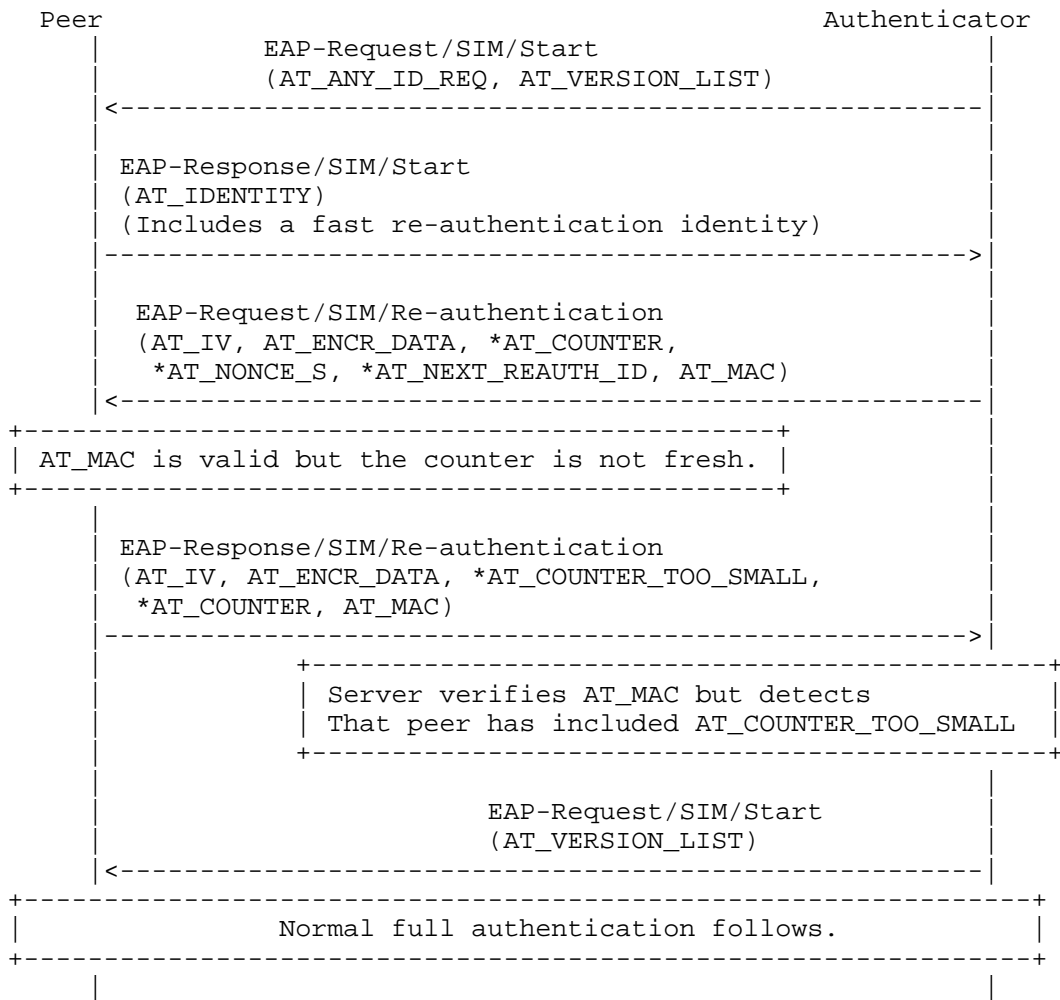   in Figure 9. Encrypted attributes are denoted with '*'.

```
        Peer                                              Authenticator
          |                EAP-Request/SIM/Start                |
          |                (AT_ANY_ID_REQ, AT_VERSION_LIST)     |
          |<----------------------------------------------------|
          |                                                     |
          | EAP-Response/SIM/Start                              |
          | (AT_IDENTITY)                                       |
          | (Includes a fast re-authentication identity)        |
          |---------------------------------------------------->|
          |                                                     |
          |  EAP-Request/SIM/Re-authentication                  |
          |  (AT_IV, AT_ENCR_DATA, *AT_COUNTER,                 |
          |   *AT_NONCE_S, *AT_NEXT_REAUTH_ID, AT_MAC)          |
          |<----------------------------------------------------|
      +------------------------------------------------+        |
      | AT_MAC is valid but the counter is not fresh.  |        |
      +------------------------------------------------+        |
          |                                                     |
          | EAP-Response/SIM/Re-authentication                  |
          | (AT_IV, AT_ENCR_DATA, *AT_COUNTER_TOO_SMALL,        |
          |  *AT_COUNTER, AT_MAC)                               |
          |---------------------------------------------------->|
          |                    +--------------------------------------------+
          |                    | Server verifies AT_MAC but detects         |
          |                    | That peer has included AT_COUNTER_TOO_SMALL |
          |                    +--------------------------------------------+
          |                                                     |
          |                    EAP-Request/SIM/Start            |
          |                    (AT_VERSION_LIST)                |
          |<----------------------------------------------------|
      +-------------------------------------------------------------+
      |              Normal full authentication follows.            |
      +-------------------------------------------------------------+
          |                                                     |
```

           Figure 9: Fast Re-authentication, counter is not fresh

In the figure above, the first three messages are similar to the
basic fast re-authentication case. When the peer detects that the
counter value is not fresh, it includes the AT_COUNTER_TOO_SMALL
attribute in EAP-Response/SIM/Re-authentication. This attribute
doesn't contain any data, but it is a request for the server to
initiate full authentication. In this case, the peer MUST ignore the
contents of the server's AT_NEXT_REAUTH_ID attribute.

On receipt of AT_COUNTER_TOO_SMALL, the server verifies AT_MAC and
verifies that AT_COUNTER contains the same counter value as in the
EAP-Request/SIM/Re-authentication packet. If not, the server
terminates the authentication exchange by sending the
EAP-Request/SIM/Notification with AT_NOTIFICATION code "General
failure" (16384). If all checks on the packet are successful, the
server transmits a new EAP-Request/SIM/Start packet and the full
authentication procedure is performed as usual. Since the server
already knows the subscriber identity, it MUST NOT include
AT_ANY_ID_REQ, AT_FULLAUTH_ID_REQ, or AT_PERMANENT_ID_REQ in the
EAP-Request/SIM/Start.

It should be noted that in this case, peer identity is only
transmitted in the AT_IDENTITY attribute at the beginning of the
whole EAP exchange. The fast re-authentication identity used in this
AT_IDENTITY attribute will be used in key derivation (see Section 7).

6. EAP-SIM Notifications

6.1. General

EAP-SIM does not prohibit the use of the EAP Notifications as
specified in [RFC3748]. EAP Notifications can be used at any time in
the EAP-SIM exchange. It should be noted that EAP-SIM does not
protect EAP Notifications. EAP-SIM also specifies method-specific
EAP-SIM notifications that are protected in some cases.

The EAP server can use EAP-SIM notifications to convey notifications
and result indications (Section 6.2) to the peer.

The server MUST use notifications in cases discussed in
Section 6.3.2. When the EAP server issues an
EAP-Request/SIM/Notification packet to the peer, the peer MUST
process the notification packet. The peer MAY show a notification
message to the user and the peer MUST respond to the EAP server with
an EAP-Response/SIM/Notification packet, even if the peer did not
recognize the notification code.

An EAP-SIM full authentication exchange or a fast re-authentication
exchange MUST NOT include more than one EAP-SIM notification round.

The notification code is a 16-bit number. The most significant bit
is called the Success bit (S bit). The S bit specifies whether the
notification implies failure. The code values with the S bit set to
zero (code values 0...32767) are used on unsuccessful cases. The
receipt of a notification code from this range implies a failed EAP
exchange, so the peer can use the notification as a failure
indication. After receiving the EAP-Response/SIM/Notification for
these notification codes, the server MUST send the EAP-Failure
packet.

The receipt of a notification code with the S bit set to one (values
32768...65536) does not imply failure. Notification code "Success"
(32768) has been reserved as a general notification code to indicate
successful authentication.

The second most significant bit of the notification code is called
the Phase bit (P bit). It specifies at which phase of the EAP-SIM
exchange the notification can be used. If the P bit is set to zero,
the notification can only be used after a successful
EAP/SIM/Challenge round in full authentication or a successful
EAP/SIM/Re-authentication round in reauthentication. A
re-authentication round is considered successful only if the peer has
successfully verified AT_MAC and AT_COUNTER attributes, and does not
include the AT_COUNTER_TOO_SMALL attribute in
EAP-Response/SIM/Re-authentication.

If the P bit is set to one, the notification can only by used before
the EAP/SIM/Challenge round in full authentication, or before the
EAP/SIM/Re-authentication round in reauthentication. These
notifications can only be used to indicate various failure cases. In
other words, if the P bit is set to one, then the S bit MUST be set
to zero.

Section 9.8 and Section 9.9 specify what other attributes must be
included in the notification packets.

Some of the notification codes are authorization related and, hence,
are not usually considered part of the responsibility of an EAP
method. However, they are included as part of EAP-SIM because there
are currently no other ways to convey this information to the user in
a localizable way, and the information is potentially useful for the
user. An EAP-SIM server implementation may decide never to send
these EAP-SIM notifications.

6.2. Result Indications

   As discussed in Section 6.3, the server and the peer use explicit
   error messages in all error cases. If the server detects an error
   after successful authentication, the server uses an EAP-SIM
   notification to indicate failure to the peer. In this case, the
   result indication is integrity and replay protected.

   By sending an EAP-Response/SIM/Challenge packet or an
   EAP-Response/SIM/Re-authentication packet (without
   AT_COUNTER_TOO_SMALL), the peer indicates that it has successfully
   authenticated the server and that the peer's local policy accepts the
   EAP exchange. In other words, these packets are implicit success
   indications from the peer to the server.

   EAP-SIM also supports optional protected success indications from the
   server to the peer. If the EAP server wants to use protected success
   indications, it includes the AT_RESULT_IND attribute in the
   EAP-Request/SIM/Challenge or the EAP-Request/SIM/Re-authentication
   packet. This attribute indicates that the EAP server would like to
   use result indications in both successful and unsuccessful cases. If
   the peer also wants this, the peer includes AT_RESULT_IND in
   EAP-Response/SIM/Challenge or EAP-Response/SIM/Re-authentication.
   The peer MUST NOT include AT_RESULT_IND if it did not receive
   AT_RESULT_IND from the server. If both the peer and the server used
   AT_RESULT_IND, then the EAP exchange is not complete yet, but an
   EAP-SIM notification round will follow. The following EAP-SIM
   notification may indicate either failure or success.

   Success indications with the AT_NOTIFICATION code "Success" (32768)
   can only be used if both the server and the peer indicate they want
   to use them with AT_RESULT_IND. If the server did not include
   AT_RESULT_IND in the EAP-Request/SIM/Challenge or
   EAP-Request/SIM/Re-authentication packet, or if the peer did not
   include AT_RESULT_IND in the corresponding response packet, then the
   server MUST NOT use protected success indications.

   Because the server uses the AT_NOTIFICATION code "Success" (32768) to
   indicate that the EAP exchange has completed successfully, the EAP
   exchange cannot fail when the server processes the EAP-SIM response
   to this notification. Hence, the server MUST ignore the contents of
   the EAP-SIM response it receives from the
   EAP-Request/SIM/Notification with this code. Regardless of the
   contents of the EAP-SIM response, the server MUST send EAP-Success as
   the next packet.

6.3. Error Cases

   This section specifies the operation of the peer and the server in
   error cases. The subsections below require the EAP-SIM peer and
   server to send an error packet (EAP-Response/SIM/Client-Error from
   the peer or EAP-Request/SIM/Notification from the server) in error
   cases. However, implementations SHOULD NOT rely upon the correct
   error reporting behavior of the peer, authenticator, or the server.
   It is possible for error and other messages to be lost in transit or
   for a malicious participant to attempt to consume resources by not
   issuing error messages. Both the peer and the EAP server SHOULD have
   a mechanism to clean up state, even if an error message or
   EAP-Success is not received after a timeout period.

6.3.1. Peer Operation

   In general, if an EAP-SIM peer detects an error in a received EAP-SIM
   packet, the EAP-SIM implementation responds with the
   EAP-Response/SIM/Client-Error packet. In response to the
   EAP-Response/SIM/Client-Error, the EAP server MUST issue the
   EAP-Failure packet and the authentication exchange terminates.

   By default, the peer uses the client error code 0, "unable to process
   packet". This error code is used in the following cases:

   o EAP exchange is not acceptable according to the peer's local
      policy.

   o the peer is not able to parse the EAP request, i.e., the EAP
      request is malformed.

   o the peer encountered a malformed attribute.

   o wrong attribute types or duplicate attributes have been included
      in the EAP request.

   o a mandatory attribute is missing.

   o unrecognized, non-skippable attribute.

   o unrecognized or unexpected EAP-SIM Subtype in the EAP request.

   o A RAND challenge repeated in AT_RAND.

   o invalid AT_MAC. The peer SHOULD log this event.

   o invalid pad bytes in AT_PADDING.

   o the peer does not want to process AT_PERMANENT_ID_REQ.

   Separate error codes have been defined for the following error cases
   in Section 10.19:

   As specified in Section 4.1, when processing the AT_VERSION_LIST
   attribute, which lists the EAP-SIM versions supported by the server,
   if the attribute does not include a version that is implemented by
   the peer and allowed in the peer's security policy, then the peer
   MUST send the EAP-Response/SIM/Client-Error packet with the error
   code "unsupported version".

   If the number of RAND challenges is smaller than what is required by
   peer's local policy when processing the AT_RAND attribute, the peer
   MUST send the EAP-Response/SIM/Client-Error packet with the error
   code "insufficient number of challenges".

   If the peer believes that the RAND challenges included in AT_RAND are
   not fresh e.g., because it is capable of remembering some previously
   used RANDs, the peer MUST send the EAP-Response/SIM/Client-Error
   packet with the error code "RANDs are not fresh".

6.3.2. Server Operation

   If an EAP-SIM server detects an error in a received EAP-SIM response,
   the server MUST issue the EAP-Request/SIM/Notification packet with an
   AT_NOTIFICATION code that implies failure. By default, the server
   uses one of the general failure codes ("General failure after
   authentication" (0), or "General failure" (16384)). The choice
   between these two codes depends on the phase of the EAP-SIM exchange,
   see Section 6. When the server issues an EAP-
   Request/SIM/Notification that implies failure, the error cases
   include the following:

   o the server is not able to parse the peer's EAP response

   o the server encounters a malformed attribute, a non-recognized
      non-skippable attribute, or a duplicate attribute

   o a mandatory attribute is missing or an invalid attribute was
      included

   o unrecognized or unexpected EAP-SIM Subtype in the EAP Response

   o invalid AT_MAC. The server SHOULD log this event.

   o invalid AT_COUNTER

6.3.3. EAP-Failure

   The EAP-SIM server sends EAP-Failure in two cases:

   1) In response to an EAP-Response/SIM/Client-Error packet the server
      has received from the peer, or

   2) Following an EAP-SIM notification round, when the AT_NOTIFICATION
      code implies failure.

   The EAP-SIM server MUST NOT send EAP-Failure in cases other than
   these two. However, it should be noted that even though the EAP-SIM
   server would not send an EAP-Failure, an authorization decision that
   happens outside EAP-SIM, such as in the AAA server or in an
   intermediate AAA proxy, may result in a failed exchange.

   The peer MUST accept the EAP-Failure packet in case 1) and case 2),
   above. The peer SHOULD silently discard the EAP-Failure packet in
   other cases.

6.3.4. EAP-Success

   On full authentication, the server can only send EAP-Success after
   the EAP/SIM/Challenge round. The peer MUST silently discard any
   EAP-Success packets if they are received before the peer has
   successfully authenticated the server and sent the
   EAP-Response/SIM/Challenge packet.

   If the peer did not indicate that it wants to use protected success
   indications with AT_RESULT_IND (as discussed in Section 6.2) on full
   authentication, then the peer MUST accept EAP-Success after a
   successful EAP/SIM/Challenge round.

   If the peer indicated that it wants to use protected success
   indications with AT_RESULT_IND (as discussed in Section 6.2), then
   the peer MUST NOT accept EAP-Success after a successful
   EAP/SIM/Challenge round. In this case, the peer MUST only accept
   EAP-Success after receiving an EAP-SIM Notification with the
   AT_NOTIFICATION code "Success" (32768).

   On fast re-authentication, EAP-Success can only be sent after the
   EAP/SIM/Re-authentication round. The peer MUST silently discard any
   EAP-Success packets if they are received before the peer has
   successfully authenticated the server and sent the
   EAP-Response/SIM/Re-authentication packet.

   If the peer did not indicate that it wants to use protected success
   indications with AT_RESULT_IND (as discussed in Section 6.2) on fast

re-authentication, then the peer MUST accept EAP-Success after a
successful EAP/SIM/Re-authentication round.

If the peer indicated that it wants to use protected success
indications with AT_RESULT_IND (as discussed in Section 6.2), then
the peer MUST NOT accept EAP-Success after a successful EAP/SIM/Re-
authentication round. In this case, the peer MUST only accept
EAP-Success after receiving an EAP-SIM Notification with the
AT_NOTIFICATION code "Success" (32768).

If the peer receives an EAP-SIM notification (Section 6) that
indicates failure, then the peer MUST no longer accept the
EAP-Success packet, even if the server authentication was
successfully completed.

7. Key Generation

This section specifies how keying material is generated.

On EAP-SIM full authentication, a Master Key (MK) is derived from the
underlying GSM authentication values (Kc keys), the NONCE_MT, and
other relevant context as follows.

MK = SHA1(Identity|n*Kc| NONCE_MT| Version List| Selected Version)

In the formula above, the "|" character denotes concatenation.
"Identity" denotes the peer identity string without any terminating
null characters. It is the identity from the last AT_IDENTITY
attribute sent by the peer in this exchange, or, if AT_IDENTITY was
not used, it is the identity from the EAP-Response/Identity packet.
The identity string is included as-is, without any changes. As
discussed in Section 4.2.2.2, relying on EAP-Response/Identity for
conveying the EAP-SIM peer identity is discouraged, and the server
SHOULD use the EAP-SIM method-specific identity attributes.

The notation n*Kc in the formula above denotes the n Kc values
concatenated. The Kc keys are used in the same order as the RAND
challenges in AT_RAND attribute. NONCE_MT denotes the NONCE_MT value
(not the AT_NONCE_MT attribute, but only the nonce value). The
Version List includes the 2-byte-supported version numbers from
AT_VERSION_LIST, in the same order as in the attribute. The Selected
Version is the 2-byte selected version from AT_SELECTED_VERSION.
Network byte order is used, just as in the attributes. The hash
function SHA-1 is specified in [SHA-1]. If several EAP/SIM/Start
roundtrips are used in an EAP-SIM exchange, then the NONCE_MT,
Version List and Selected version from the last EAP/SIM/Start round
are used, and the previous EAP/SIM/Start rounds are ignored.

The Master Key is fed into a Pseudo-Random number Function (PRF)
which generates separate Transient EAP Keys (TEKs) for protecting
EAP-SIM packets, as well as a Master Session Key (MSK) for link layer
security, and an Extended Master Session Key (EMSK) for other
purposes. On fast re-authentication, the same TEKs MUST be used for
protecting EAP packets, but a new MSK and a new EMSK MUST be derived
from the original MK and from new values exchanged in the fast
re-authentication.

EAP-SIM requires two TEKs for its own purposes; the authentication
key K_aut is to be used with the AT_MAC attribute, and the encryption
key K_encr is to be used with the AT_ENCR_DATA attribute. The same
K_aut and K_encr keys are used in full authentication and subsequent
fast re-authentications.

Key derivation is based on the random number generation specified in
NIST Federal Information Processing Standards (FIPS) Publication
186-2 [PRF]. The pseudo-random number generator is specified in the
change notice 1 (2001 October 5) of [PRF] (Algorithm 1). As
specified in the change notice (page 74), when Algorithm 1 is used as
a general-purpose pseudo-random number generator, the "mod q" term in
step 3.3 is omitted. The function G used in the algorithm is
constructed via the Secure Hash Standard, as specified in Appendix
3.3 of the standard. It should be noted that the function G is very
similar to SHA-1, but the message padding is different. Please refer
to [PRF] for full details. For convenience, the random number
algorithm with the correct modification is cited in Appendix B.

160-bit XKEY and XVAL values are used, so b = 160. On each full
authentication, the Master Key is used as the initial secret seed-key
XKEY. The optional user input values (XSEED_j) in step 3.1 are set
to zero.

On full authentication, the resulting 320-bit random numbers (x_0,
x_1, ..., x_{m-1}) are concatenated and partitioned into suitable-sized
chunks and used as keys in the following order: K_encr (128 bits),
K_aut (128 bits), Master Session Key (64 bytes), Extended Master
Session Key (64 bytes).

On fast re-authentication, the same pseudo-random number generator
can be used to generate a new Master Session Key and a new Extended
Master Session Key. The seed value XKEY' is calculated as follows:

XKEY' = SHA1(Identity|counter|NONCE_S| MK)

In the formula above, the Identity denotes the fast re-authentication
identity, without any terminating null characters, from the
AT_IDENTITY attribute of the EAP-Response/SIM/Start packet, or, if

EAP-Response/SIM/Start was not used on fast re-authentication, it
denotes the identity string from the EAP-Response/Identity packet.
The counter denotes the counter value from the AT_COUNTER attribute
used in the EAP-Response/SIM/Re-authentication packet. The counter
is used in network byte order. NONCE_S denotes the 16-byte NONCE_S
value from the AT_NONCE_S attribute used in the
EAP-Request/SIM/Re-authentication packet. The MK is the Master Key
derived on the preceding full authentication.

On fast re-authentication, the pseudo-random number generator is run
with the new seed value XKEY', and the resulting 320-bit random
numbers (x_0, x_1, ..., x_m-1) are concatenated and partitioned into
two 64-byte chunks and used as the new 64-byte Master Session Key and
the new 64-byte Extended Master Session Key. Note that because
K_encr and K_aut are not derived on fast re-authentication, the
Master Session Key and the Extended Master Session key are obtained
from the beginning of the key stream (x_0, x_1, ...).

The first 32 bytes of the MSK can be used as the Pairwise Master Key
(PMK) for IEEE 802.11i.

When the RADIUS attributes specified in [RFC2548] are used to
transport keying material, then the first 32 bytes of the MSK
correspond to MS-MPPE-RECV-KEY and the second 32 bytes to
MS-MPPE-SEND-KEY. In this case, only 64 bytes of keying material
(the MSK) are used.

When generating the initial Master Key, the hash function is used as
a mixing function to combine several session keys (Kc's) generated by
the GSM authentication procedure and the random number NONCE_MT into
a single session key. There are several reasons for this. The
current GSM session keys are, at most, 64 bits, so two or more of
them are needed to generate a longer key. By using a one-way
function to combine the keys, we are assured that, even if an
attacker managed to learn one of the EAP-SIM session keys, it
wouldn't help him in learning the original GSM Kc's. In addition,
since we include the random number NONCE_MT in the calculation, the
peer is able to verify that the EAP-SIM packets it receives from the
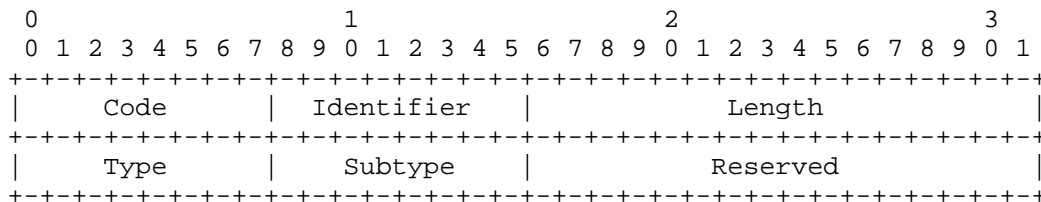network are fresh and not replays (also see Section 11).

8. Message Format and Protocol Extensibility
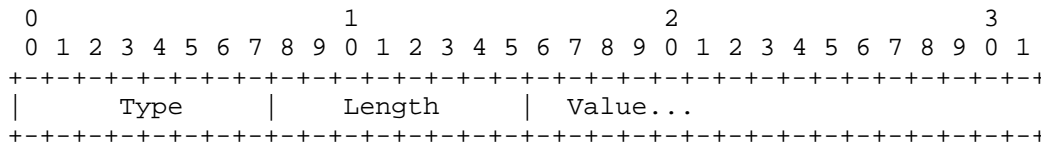
8.1. Message Format

As specified in [RFC3748], EAP packets begin with the Code,
Identifiers, Length, and Type fields, which are followed by EAP-
method-specific Type-Data. The Code field in the EAP header is set
to 1 for EAP requests, and to 2 for EAP Responses. The usage of the

RFC 4186   EAP-SIM Authentication   January 2006

   Length and Identifier fields in the EAP header are also specified in
   [RFC3748]. In EAP-SIM, the Type field is set to 18.

   In EAP-SIM, the Type-Data begins with an EAP-SIM header that consists
   of a 1-octet Subtype field and a 2-octet reserved field. The Subtype
   values used in EAP-SIM are defined in the IANA considerations section
   of the EAP-AKA specification [EAP-AKA]. The formats of the EAP
   header and the EAP-SIM header are shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Code      |  Identifier   |            Length             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type      |    Subtype    |           Reserved            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   The rest of the Type-Data that immediately follows the EAP-SIM header
   consists of attributes that are encoded in Type, Length, Value
   format. The figure below shows the generic format of an attribute.

```
    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...
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   Attribute Type

         Indicates the particular type of attribute. The attribute type
         values are listed in the IANA considerations section of the
         EAP-AKA specification [EAP-AKA].

   Length

         Indicates the length of this attribute in multiples of four
         bytes. The maximum length of an attribute is 1024 bytes. The
         length includes the Attribute Type and Length bytes.

   Value

         The particular data associated with this attribute. This field
         is always included and it may be two or more bytes in length.
         The type and length fields determine the format and length
         of the value field.

Haverinen & Salowey   Informational   [Page 46]

   Attributes numbered within the range 0 through 127 are called
   non-skippable attributes. When an EAP-SIM peer encounters a
   non-skippable attribute that the peer does not recognize, the peer
   MUST send the EAP-Response/SIM/Client-Error packet, which terminates
   the authentication exchange. If an EAP-SIM server encounters a
   non-skippable attribute that the server does not recognize, then the
   server sends the EAP-Request/SIM/Notification packet with an
   AT_NOTIFICATION code, which implies general failure ("General failure
   after authentication" (0), or "General failure" (16384), depending on
   the phase of the exchange), which terminates the authentication
   exchange.

   Attributes within the range of 128 through 255 are called skippable
   attributes. When a skippable attribute is encountered and is not
   recognized, it is ignored. The rest of the attributes and message
   data MUST still be processed. The Length field of the attribute is
   used to skip the attribute value in searching for the next attribute.

   Unless otherwise specified, the order of the attributes in an EAP-SIM
   message is insignificant and an EAP-SIM implementation should not
   assume a certain order to be used.

   Attributes can be encapsulated within other attributes. In other
   words, the value field of an attribute type can be specified to
   contain other attributes.

8.2. Protocol Extensibility

   EAP-SIM can be extended by specifying new attribute types. If
   skippable attributes are used, it is possible to extend the protocol
   without breaking old implementations.

   However, any new attributes added to the EAP-Request/SIM/Start or
   EAP-Response/SIM/Start packets would not be integrity-protected.
   Therefore, these messages MUST NOT be extended in the current version
   of EAP-SIM. If the list of supported EAP-SIM versions in the
   AT_VERSION_LIST does not include versions other than 1, then the
   server MUST NOT include attributes other than those specified in this
   document in the EAP-Request/SIM/Start message. Note that future
   versions of this protocol might specify new attributes for
   EAP-Request/SIM/Start and still support version 1 of the protocol.
   In this case, the server might send an EAP-Request/SIM/Start message
   that includes new attributes and indicates support for protocol
   version 1 and other versions in the AT_VERSION_LIST attribute. If
   the peer selects version 1, then the peer MUST ignore any other
   attributes included in EAP-Request/SIM/Start, other than those
   specified in this document. If the selected EAP-SIM version in
   peer's AT_SELECTED_VERSION is 1, then the peer MUST NOT include other

attributes aside from those specified in this document in the
EAP-Response/SIM/Start message.

When specifying new attributes, it should be noted that EAP-SIM does
not support message fragmentation. Hence, the sizes of the new
extensions MUST be limited so that the maximum transfer unit (MTU) of
the underlying lower layer is not exceeded. According to [RFC3748],
lower layers must provide an EAP MTU of 1020 bytes or greater, so any
extensions to EAP-SIM SHOULD NOT exceed the EAP MTU of 1020 bytes.

Because EAP-SIM supports version negotiation, new versions of the
protocol can also be specified by using a new version number.

9. Messages

   This section specifies the messages used in EAP-SIM. It specifies
   when a message may be transmitted or accepted, which attributes are
   allowed in a message, which attributes are required in a message, and
   other message-specific details. The general message format is
   specified in Section 8.1.

9.1. EAP-Request/SIM/Start

   In full authentication the first SIM-specific EAP Request is
   EAP-Request/SIM/Start. The EAP/SIM/Start roundtrip is used for two
   purposes. In full authentication this packet is used to request the
   peer to send the AT_NONCE_MT attribute to the server. In addition,
   as specified in Section 4.2, the Start round trip may be used by the
   server for obtaining the peer identity. As discussed in Section 4.2,
   several Start rounds may be required to obtain a valid peer identity.

   The server MUST always include the AT_VERSION_LIST attribute.

   The server MAY include one of the following identity-requesting
   attributes: AT_PERMANENT_ID_REQ, AT_FULLAUTH_ID_REQ, or
   AT_ANY_ID_REQ. These three attributes are mutually exclusive, so the
   server MUST NOT include more than one of the attributes.

   If the server has received a response from the peer, it MUST NOT
   issue a new EAP-Request/SIM/Start packet if it has previously issued
   an EAP-Request/SIM/Start message either without any identity
   requesting attributes or with the AT_PERMANENT_ID_REQ attribute.

   If the server has received a response from the peer, it MUST NOT
   issue a new EAP-Request/SIM/Start packet with the AT_ANY_ID_REQ or
   AT_FULLAUTH_ID_REQ attributes if it has previously issued an
   EAP-Request/SIM/Start message with the AT_FULLAUTH_ID_REQ attribute.

      If the server has received a response from the peer, it MUST NOT
      issue a new EAP-Request/SIM/Start packet with the AT_ANY_ID_REQ
      attribute if the server has previously issued an
      EAP-Request/SIM/Start message with the AT_ANY_ID_REQ attribute.

      This message MUST NOT include AT_MAC, AT_IV, or AT_ENCR_DATA.

9.2. EAP-Response/SIM/Start

      The peer sends EAP-Response/SIM/Start in response to a valid
      EAP-Request/SIM/Start from the server.

      If and only if the server's EAP-Request/SIM/Start includes one of the
      identity-requesting attributes, then the peer MUST include the
      AT_IDENTITY attribute. The usage of AT_IDENTITY is defined in
      Section 4.2.

      The AT_NONCE_MT attribute MUST NOT be included if the AT_IDENTITY
      with a fast re-authentication identity is present for fast
      re-authentication. AT_NONCE_MT MUST be included in all other cases
      (full authentication).

      The AT_SELECTED_VERSION attribute MUST NOT be included if the
      AT_IDENTITY attribute with a fast re-authentication identity is
      present for fast re-authentication. In all other cases,
      AT_SELECTED_VERSION MUST be included (full authentication). This
      attribute is used in version negotiation, as specified in
      Section 4.1.

      This message MUST NOT include AT_MAC, AT_IV, or AT_ENCR_DATA.

9.3. EAP-Request/SIM/Challenge

      The server sends the EAP-Request/SIM/Challenge after receiving a
      valid EAP-Response/SIM/Start that contains AT_NONCE_MT and
      AT_SELECTED_VERSION, and after successfully obtaining the subscriber
      identity.

      The AT_RAND attribute MUST be included.

      The AT_RESULT_IND attribute MAY be included. The usage of this
      attribute is discussed in Section 6.2.

      The AT_MAC attribute MUST be included. For
      EAP-Request/SIM/Challenge, the MAC code is calculated over the
      following data:

      EAP packet| NONCE_MT

The EAP packet is represented as specified in Section 8.1. It is
followed by the 16-byte NONCE_MT value from the peer's AT_NONCE_MT
attribute.

The EAP-Request/SIM/Challenge packet MAY include encrypted attributes
for identity privacy and for communicating the next fast
re-authentication identity. In this case, the AT_IV and AT_ENCR_DATA
attributes are included (Section 10.12).

The plaintext of the AT_ENCR_DATA value field consists of nested
attributes. The nested attributes MAY include AT_PADDING (as
specified in Section 10.12). If the server supports identity privacy
and wants to communicate a pseudonym to the peer for the next full
authentication, then the nested encrypted attributes include the
AT_NEXT_PSEUDONYM attribute. If the server supports
re-authentication and wants to communicate a fast re-authentication
identity to the peer, then the nested encrypted attributes include
the AT_NEXT_REAUTH_ID attribute.

When processing this message, the peer MUST process AT_RAND before
processing other attributes. Only if AT_RAND is verified to be
valid, the peer derives keys and verifies AT_MAC. The operation in
case an error occurs is specified in Section 6.3.1.

9.4. EAP-Response/SIM/Challenge

The peer sends EAP-Response/SIM/Challenge in response to a valid
EAP-Request/SIM/Challenge.

Sending this packet indicates that the peer has successfully
authenticated the server and that the EAP exchange will be accepted
by the peer's local policy. Hence, if these conditions are not met,
then the peer MUST NOT send EAP-Response/SIM/Challenge, but the peer
MUST send EAP-Response/SIM/Client-Error.

The AT_MAC attribute MUST be included. For EAP-
Response/SIM/Challenge, the MAC code is calculated over the following
data:

EAP packet| n*SRES

The EAP packet is represented as specified in Section 8.1. The EAP
packet bytes are immediately followed by the two or three SRES values
concatenated, denoted above with the notation n*SRES. The SRES
values are used in the same order as the corresponding RAND
challenges in the server's AT_RAND attribute.

   The AT_RESULT_IND attribute MAY be included if it was included in
   EAP-Request/SIM/Challenge. The usage of this attribute is discussed
   in Section 6.2.

   Later versions of this protocol MAY make use of the AT_ENCR_DATA and
   AT_IV attributes in this message to include encrypted (skippable)
   attributes. The EAP server MUST process EAP-Response/SIM/Challenge
   messages that include these attributes even if the server did not
   implement these optional attributes.

9.5. EAP-Request/SIM/Re-authentication

   The server sends the EAP-Request/SIM/Re-authentication message if it
   wants to use fast re-authentication, and if it has received a valid
   fast re-authentication identity in EAP-Response/Identity or
   EAP-Response/SIM/Start.

   AT_MAC MUST be included. No message-specific data is included in the
   MAC calculation. See Section 10.14.

   The AT_RESULT_IND attribute MAY be included. The usage of this
   attribute is discussed in Section 6.2.

   The AT_IV and AT_ENCR_DATA attributes MUST be included. The
   plaintext consists of the following nested encrypted attributes,
   which MUST be included: AT_COUNTER and AT_NONCE_S. In addition, the
   nested encrypted attributes MAY include the following attributes:
   AT_NEXT_REAUTH_ID and AT_PADDING.

9.6. EAP-Response/SIM/Re-authentication

   The client sends the EAP-Response/SIM/Re-authentication packet in
   response to a valid EAP-Request/SIM/Re-authentication.

   The AT_MAC attribute MUST be included. For
   EAP-Response/SIM/Re-authentication, the MAC code is calculated over
   the following data:

   EAP packet| NONCE_S

   The EAP packet is represented as specified in Section 8.1. It is
   followed by the 16-byte NONCE_S value from the server's AT_NONCE_S
   attribute.

   The AT_IV and AT_ENCR_DATA attributes MUST be included. The nested
   encrypted attributes MUST include the AT_COUNTER attribute. The
   AT_COUNTER_TOO_SMALL attribute MAY be included in the nested

encrypted attributes, and it is included in cases specified in
Section 5. The AT_PADDING attribute MAY be included.

The AT_RESULT_IND attribute MAY be included if it was included in
EAP-Request/SIM/Re-authentication. The usage of this attribute is
discussed in Section 6.2.

Sending this packet without AT_COUNTER_TOO_SMALL indicates that the
peer has successfully authenticated the server and that the EAP
exchange will be accepted by the peer's local policy. Hence, if
these conditions are not met, then the peer MUST NOT send
EAP-Response/SIM/Re-authentication, but the peer MUST send
EAP-Response/SIM/Client-Error.

9.7. EAP-Response/SIM/Client-Error

The peer sends EAP-Response/SIM/Client-Error in error cases, as
specified in Section 6.3.1.

The AT_CLIENT_ERROR_CODE attribute MUST be included.

The AT_MAC, AT_IV, or AT_ENCR_DATA attributes MUST NOT be used with
this packet.

9.8. EAP-Request/SIM/Notification

The usage of this message is specified in Section 6. The
AT_NOTIFICATION attribute MUST be included.

The AT_MAC attribute MUST be included if the P bit of the
notification code in AT_NOTIFICATION is set to zero, and MUST NOT be
included in cases when the P bit is set to one. The P bit is
discussed in Section 6.

No message-specific data is included in the MAC calculation. See
Section 10.14.

If EAP-Request/SIM/Notification is used on a fast re-authentication
exchange, and if the P bit in AT_NOTIFICATION is set to zero, then
AT_COUNTER is used for replay protection. In this case, the
AT_ENCR_DATA and AT_IV attributes MUST be included, and the
encapsulated plaintext attributes MUST include the AT_COUNTER
attribute. The counter value included in AT_COUNTER MUST be the same
as in the EAP-Request/SIM/Re-authentication packet on the same fast
re-authentication exchange.

9.9. EAP-Response/SIM/Notification

   The usage of this message is specified in Section 6. This packet is
   an acknowledgement of EAP-Request/SIM/Notification.

   The AT_MAC attribute MUST be included in cases when the P bit of the
   notification code in AT_NOTIFICATION of EAP-Request/SIM/Notification
   is set to zero, and MUST NOT be included in cases when the P bit is
   set to one. The P bit is discussed in Section 6.

   No message-specific data is included in the MAC calculation, see
   Section 10.14.

   If EAP-Request/SIM/Notification is used on a fast re-authentication
   exchange, and if the P bit in AT_NOTIFICATION is set to zero, then
   AT_COUNTER is used for replay protection. In this case, the
   AT_ENCR_DATA and AT_IV attributes MUST be included, and the
   encapsulated plaintext attributes MUST include the AT_COUNTER
   attribute. The counter value included in AT_COUNTER MUST be the same
   as in the EAP-Request/SIM/Re-authentication packet on the same fast
   re-authentication exchange.

10. Attributes

   This section specifies the format of message attributes. The
   attribute type numbers are specified in the IANA considerations
   section of the EAP-AKA specification [EAP-AKA].

10.1. Table of Attributes

   The following table provides a guide to which attributes may be found
   in which kinds of messages, and in what quantity. Messages are
   denoted with numbers in parentheses as follows: (1)
   EAP-Request/SIM/Start, (2) EAP-Response/SIM/Start, (3)
   EAP-Request/SIM/Challenge, (4) EAP-Response/SIM/Challenge, (5)
   EAP-Request/SIM/Notification, (6) EAP-Response/SIM/Notification, (7)
   EAP-Response/SIM/Client-Error, (8) EAP-Request/SIM/Re-authentication,
   and (9) EAP-Response/SIM/Re-authentication. The column denoted with
   "Encr" indicates whether the attribute is a nested attribute that
   MUST be included within AT_ENCR_DATA, and the column denoted with
   "Skip" indicates whether the attribute is a skippable attribute.
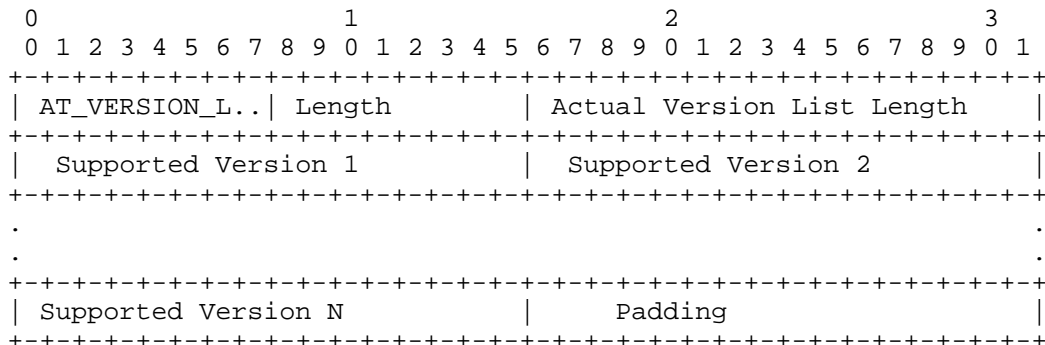
   "0" indicates that the attribute MUST NOT be included in the message,
   "1" indicates that the attribute MUST be included in the message,
   "0-1" indicates that the attribute is sometimes included in the
   message, and "0*" indicates that the attribute is not included in the
   message in cases specified in this document, but MAY be included in
   future versions of the protocol.

```
             Attribute (1) (2) (3) (4) (5) (6) (7) (8) (9)   Encr Skip
       AT_VERSION_LIST   1   0   0   0   0   0   0   0   0     N    N
   AT_SELECTED_VERSION   0  0-1  0   0   0   0   0   0   0     N    N
          AT_NONCE_MT    0  0-1  0   0   0   0   0   0   0     N    N
    AT_PERMANENT_ID_REQ 0-1  0   0   0   0   0   0   0   0     N    N
         AT_ANY_ID_REQ  0-1  0   0   0   0   0   0   0   0     N    N
    AT_FULLAUTH_ID_REQ  0-1  0   0   0   0   0   0   0   0     N    N
          AT_IDENTITY    0  0-1  0   0   0   0   0   0   0     N    N
              AT_RAND    0   0   1   0   0   0   0   0   0     N    N
    AT_NEXT_PSEUDONYM    0   0  0-1  0   0   0   0   0   0     Y    Y
    AT_NEXT_REAUTH_ID    0   0  0-1  0   0   0   0  0-1  0     Y    Y
                AT_IV    0   0  0-1  0* 0-1 0-1  0   1   1     N    Y
          AT_ENCR_DATA   0   0  0-1  0* 0-1 0-1  0   1   1     N    Y
            AT_PADDING   0   0  0-1  0* 0-1 0-1  0  0-1 0-1    Y    N
         AT_RESULT_IND   0   0  0-1 0-1  0   0   0  0-1 0-1    N    Y
               AT_MAC    0   0   1   1  0-1 0-1  0   1   1     N    N
           AT_COUNTER    0   0   0   0  0-1 0-1  0   1   1     Y    N
   AT_COUNTER_TOO_SMALL  0   0   0   0   0   0   0   0  0-1    Y    N
           AT_NONCE_S    0   0   0   0   0   0   0   1   0     Y    N
        AT_NOTIFICATION  0   0   0   0   1   0   0   0   0     N    N
   AT_CLIENT_ERROR_CODE  0   0   0   0   0   0   1   0   0     N    N
```

   It should be noted that attributes AT_PERMANENT_ID_REQ,
   AT_ANY_ID_REQ, and AT_FULLAUTH_ID_REQ are mutually exclusive; only
   one of them can be included at the same time. If one of the
   attributes AT_IV and AT_ENCR_DATA is included, then both of the
   attributes MUST be included.

10.2. AT_VERSION_LIST

   The format of the AT_VERSION_LIST attribute is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | AT_VERSION_L..| Length        | Actual Version List Length    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Supported Version 1          | Supported Version 2           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   .                                                               .
   .                                                               .
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Supported Version N          |       Padding                 |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
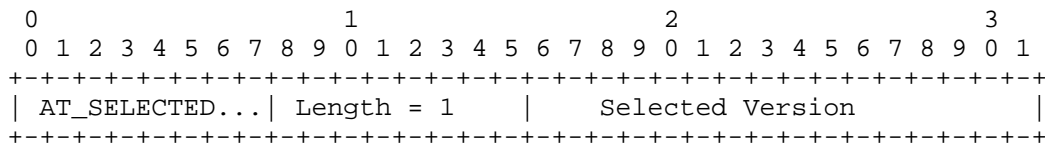
   This attribute is used in version negotiation, as specified in
   Section 4.1. The attribute contains the version numbers supported by
   the EAP-SIM server. The server MUST only include versions that it

implements and that are allowed in its security policy. The server
SHOULD list the versions in the order of preference, with the most
preferred versions listed first. At least one version number MUST be
included. The version number for the protocol described in this
document is one (0001 hexadecimal).

The value field of this attribute begins with 2-byte Actual Version
List Length, which specifies the length of the Version List in bytes,
not including the Actual Version List Length attribute length. This
field is followed by the list of the versions supported by the
server, which each have a length of 2 bytes. For example, if there
is only one supported version, then the Actual Version List Length is
2. Because the length of the attribute must be a multiple of 4
bytes, the sender pads the value field with zero bytes when
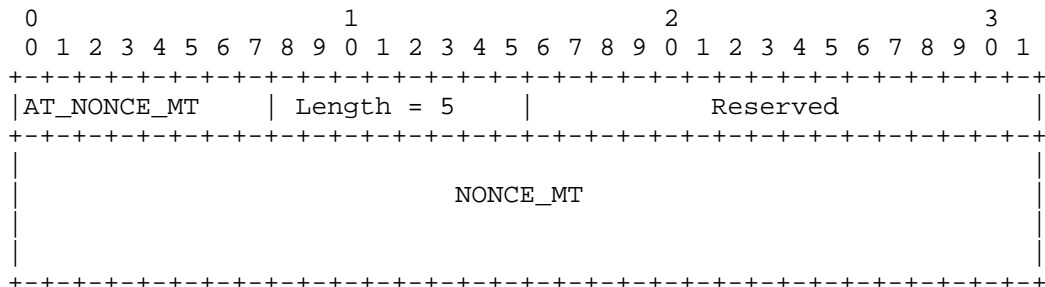necessary.

10.3. AT_SELECTED_VERSION

The format of the AT_SELECTED_VERSION attribute is shown below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| AT_SELECTED...| Length = 1    |      Selected Version         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

This attribute is used in version negotiation, as specified in
Section 4.1. The value field of this attribute contains a two-byte
version number, which indicates the EAP-SIM version that the peer
wants to use.

10.4. AT_NONCE_MT

The format of the AT_NONCE_MT attribute is shown below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|AT_NONCE_MT    | Length = 5    |            Reserved           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                          NONCE_MT                             |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
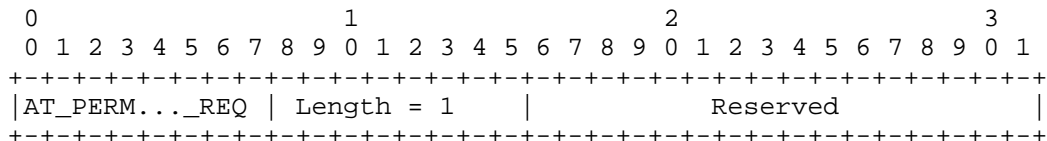
   The value field of the NONCE_MT attribute contains two reserved bytes
   followed by a random number freshly generated by the peer (16 bytes
   long) for this EAP-SIM authentication exchange. The random number is
   used as a seed value for the new keying material. The reserved bytes
   are set to zero upon sending and ignored upon reception.

   The peer MUST NOT re-use the NONCE_MT value from a previous EAP-SIM
   authentication exchange. If an EAP-SIM exchange includes several
   EAP/SIM/Start rounds, then the peer SHOULD use the same NONCE_MT
   value in all EAP-Response/SIM/Start packets. The peer SHOULD use a
   good source of randomness to generate NONCE_MT. Please see [RFC4086]
   for more information about generating random numbers for security
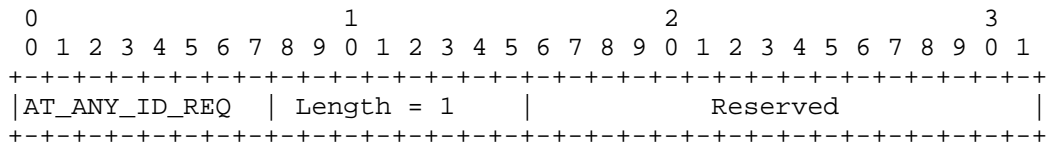   applications.

10.5. AT_PERMANENT_ID_REQ

   The format of the AT_PERMANENT_ID_REQ attribute is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |AT_PERM..._REQ | Length = 1    |           Reserved            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   The use of the AT_PERMANENT_ID_REQ is defined in Section 4.2. The
   value field contains only two reserved bytes, which are set to zero
   on sending and ignored on reception.
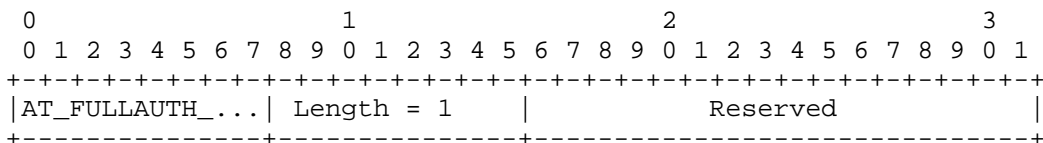
10.6. AT_ANY_ID_REQ

   The format of the AT_ANY_ID_REQ attribute is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |AT_ANY_ID_REQ  | Length = 1    |           Reserved            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   The use of the AT_ANY_ID_REQ is defined in Section 4.2. The value
   field contains only two reserved bytes, which are set to zero on
   sending and ignored on reception.
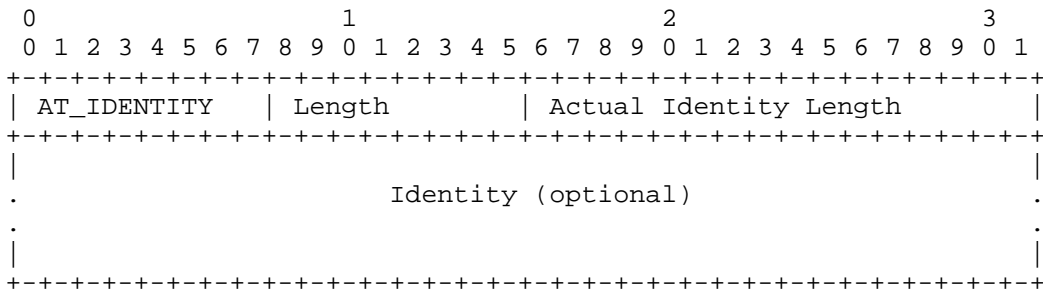
10.7. AT_FULLAUTH_ID_REQ

   The format of the AT_FULLAUTH_ID_REQ attribute is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |AT_FULLAUTH_...| Length = 1    |             Reserved          |
   +--------------+--------------+------------------------------+
```

   The use of the AT_FULLAUTH_ID_REQ is defined in Section 4.2. The
   value field contains only two reserved bytes, which are set to zero
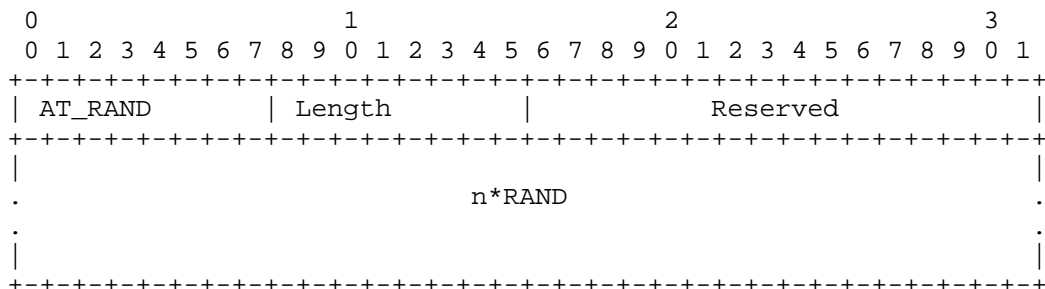   on sending and ignored on reception.

10.8. AT_IDENTITY

   The format of the AT_IDENTITY attribute is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | AT_IDENTITY   | Length        | Actual Identity Length        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   .                    Identity (optional)                        .
   .                                                               .
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   The use of the AT_IDENTITY is defined in Section 4.2. The value
   field of this attribute begins with a 2-byte actual identity length,
   which specifies the length of the identity in bytes. This field is
   followed by the subscriber identity of the indicated actual length.
   The identity is the permanent identity, a pseudonym identity, or a
   fast re-authentication identity. The identity format is specified in
   Section 4.2.1. The same identity format is used in the AT_IDENTITY
   attribute and the EAP-Response/Identity packet, with the exception
   that the peer MUST NOT decorate the identity it includes in
   AT_IDENTITY. The identity does not include any terminating null
   characters. Because the length of the attribute must be a multiple
   of 4 bytes, the sender pads the identity with zero bytes when
   necessary.

10.9. AT_RAND

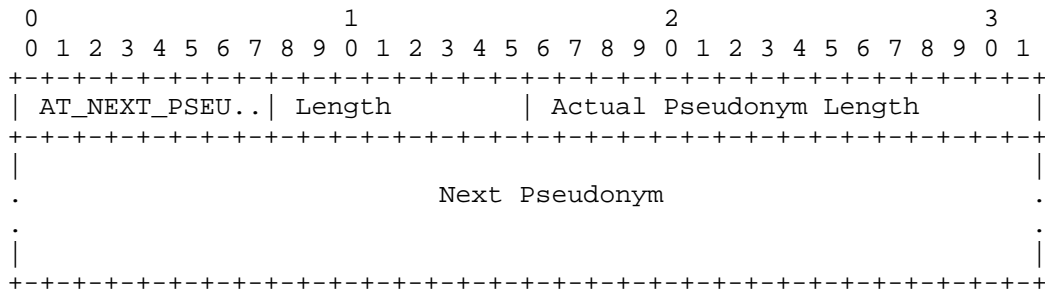   The format of the AT_RAND attribute is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | AT_RAND       | Length        |            Reserved           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   .                            n*RAND                             .
   .                                                               .
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   The value field of this attribute contains two reserved bytes
   followed by n GSM RANDs, each 16 bytes long. The value of n can be
   determined by the attribute length. The reserved bytes are set to
   zero upon sending and ignored upon reception.

   The number of RAND challenges (n) MUST be two or three. The peer
   MUST verify that the number of RAND challenges is sufficient
   according to the peer's policy. The server MUST use different RAND
   values. In other words, a RAND value can only be included once in
   AT_RAND. When processing the AT_RAND attribute, the peer MUST check
   that the RANDs are different.

   The EAP server MUST obtain fresh RANDs for each EAP-SIM full
   authentication exchange. More specifically, the server MUST consider
   RANDs it included in AT_RAND to be consumed if the server receives an
   EAP-Response/SIM/Challenge packet with a valid AT_MAC, or an
   EAP-Response/SIM/Client-Error with the code "insufficient number of
   challenges" or "RANDs are not fresh". However, in other cases (if
   the server does not receive a response to its
   EAP-Request/SIM/Challenge packet, or if the server receives a
   response other than the cases listed above), the server does not need
   to consider the RANDs to be consumed, and the server MAY re-use the
   RANDs in the AT_RAND attribute of the next full authentication
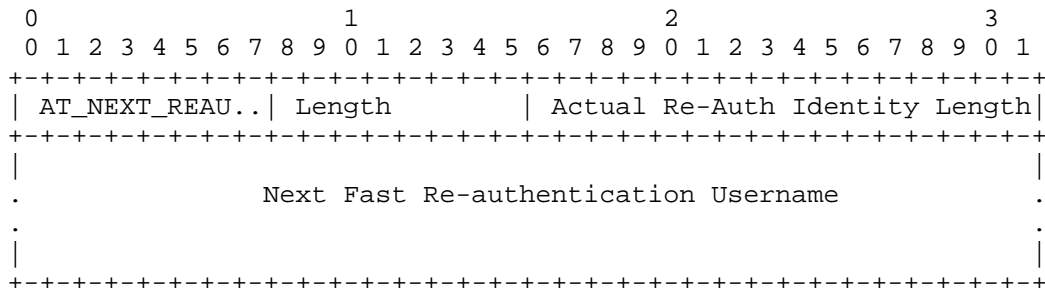   attempt.

10.10. AT_NEXT_PSEUDONYM

   The format of the AT_NEXT_PSEUDONYM attribute is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | AT_NEXT_PSEU..| Length        | Actual Pseudonym Length       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   .                       Next Pseudonym                          .
   .                                                               .
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   The value field of this attribute begins with the 2-byte actual
   pseudonym length, which specifies the length of the following
   pseudonym in bytes. This field is followed by a pseudonym username
   that the peer can use in the next authentication. The username MUST
   NOT include any realm portion. The username does not include any
   terminating null characters. Because the length of the attribute
   must be a multiple of 4 bytes, the sender pads the pseudonym with
   zero bytes when necessary. The username encoding MUST follow the
   UTF-8 transformation format [RFC3629]. This attribute MUST always be
   encrypted by encapsulating it within the AT_ENCR_DATA attribute.

10.11. AT_NEXT_REAUTH_ID

   The format of the AT_NEXT_REAUTH_ID attribute is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | AT_NEXT_REAU..| Length        | Actual Re-Auth Identity Length|
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   .            Next Fast Re-authentication Username               .
   .                                                               .
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   The value field of this attribute begins with the 2-byte actual
   re-authentication identity length which specifies the length of the
   following fast re-authentication identity in bytes. This field is
   followed by a fast re-authentication identity that the peer can use
   in the next fast re-authentication, as described in Section 5. In
   environments where a realm portion is required, the fast
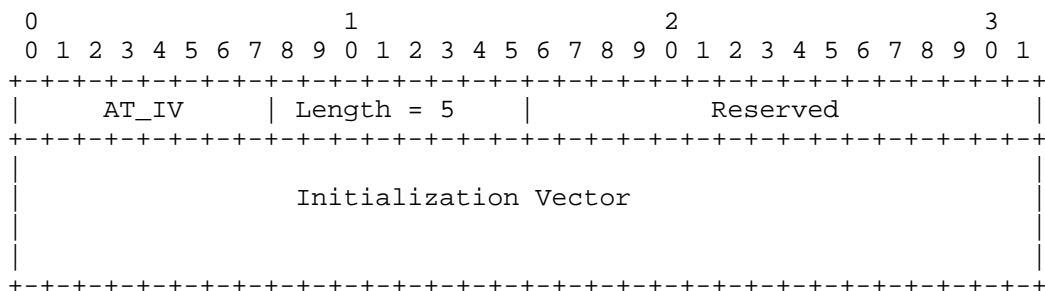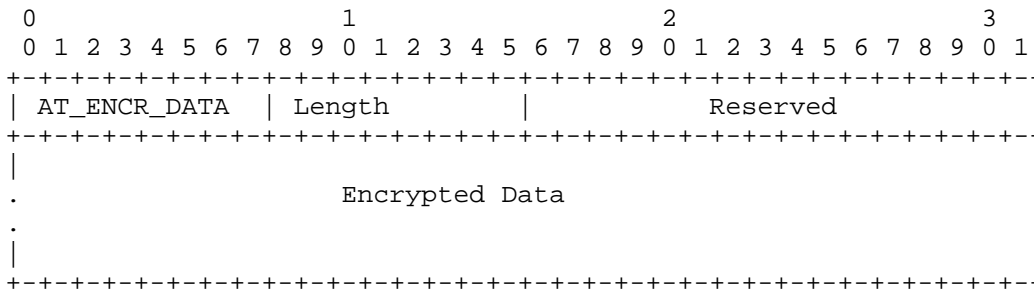   re-authentication identity includes both a username portion and a

realm name portion. The fast re-authentication identity does not
include any terminating null characters. Because the length of the
attribute must be a multiple of 4 bytes, the sender pads the fast
re-authentication identity with zero bytes when necessary. The
identity encoding MUST follow the UTF-8 transformation format
[RFC3629]. This attribute MUST always be encrypted by encapsulating
it within the AT_ENCR_DATA attribute.

10.12. AT_IV, AT_ENCR_DATA, and AT_PADDING

   AT_IV and AT_ENCR_DATA attributes can be used to transmit encrypted
   information between the EAP-SIM peer and server.

   The value field of AT_IV contains two reserved bytes followed by a
   16-byte initialization vector required by the AT_ENCR_DATA attribute.
   The reserved bytes are set to zero when sending and ignored on
   reception. The AT_IV attribute MUST be included if and only if the
   AT_ENCR_DATA is included. Section 6.3 specifies the operation if a
   packet that does not meet this condition is encountered.

   The sender of the AT_IV attribute chooses the initialization vector
   at random. The sender MUST NOT re-use the initialization vector
   value from previous EAP-SIM packets. The sender SHOULD use a good
   source of randomness to generate the initialization vector. Please
   see [RFC4086] for more information about generating random numbers
   for security applications. The format of AT_IV is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     AT_IV     | Length = 5    |           Reserved            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                                                               |
   |                     Initialization Vector                     |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
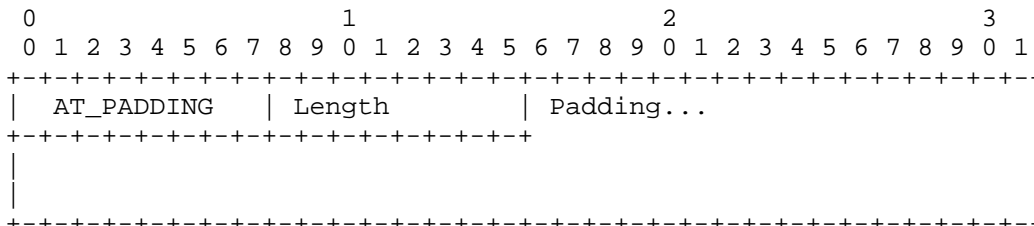
   The value field of the AT_ENCR_DATA attribute consists of two
   reserved bytes followed by cipher text bytes encrypted using the
   Advanced Encryption Standard (AES) [AES] with a 128-bit key in the
   Cipher Block Chaining (CBC) mode of operation using the
   initialization vector from the AT_IV attribute. The reserved bytes
   are set to zero when sending and ignored on reception. Please see
   [CBC] for a description of the CBC mode. The format of the
   AT_ENCR_DATA attribute is shown below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| AT_ENCR_DATA  | Length        |           Reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
.                     Encrypted Data                            .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The derivation of the encryption key (K_encr) is specified in Section
7.

The plaintext consists of nested EAP-SIM attributes.

The encryption algorithm requires the length of the plaintext to be a
multiple of 16 bytes. The sender may need to include the AT_PADDING
attribute as the last attribute within AT_ENCR_DATA. The AT_PADDING
attribute is not included if the total length of other nested
attributes within the AT_ENCR_DATA attribute is a multiple of 16
bytes. As usual, the Length of the Padding attribute includes the
Attribute Type and Attribute Length fields. The length of the
Padding attribute is 4, 8, or 12 bytes. It is chosen so that the
length of the value field of the AT_ENCR_DATA attribute becomes a
multiple of 16 bytes. The actual pad bytes in the value field are
set to zero (00 hexadecimal) on sending. The recipient of the
message MUST verify that the pad bytes are set to zero. If this
verification fails on the peer, then it MUST send the
EAP-Response/SIM/Client-Error packet with the error code "unable to
process packet" to terminate the authentication exchange. If this
verification fails on the server, then the server sends the peer the
EAP-Request/SIM/Notification packet with an AT_NOTIFICATION code that
implies failure to terminate the authentication exchange. The format
of the AT_PADDING attribute is shown below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  AT_PADDING   | Length        | Padding...                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                                |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

10.13. AT_RESULT_IND

   The format of the AT_RESULT_IND attribute is shown below.
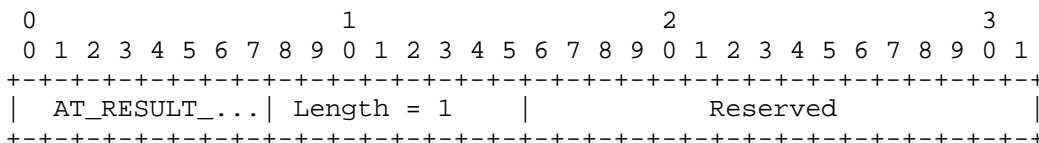
```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  AT_RESULT_...| Length = 1    |             Reserved          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   The value field of this attribute consists of two reserved bytes,
   which are set to zero upon sending and ignored upon reception. This
   attribute is always sent unencrypted, so it MUST NOT be encapsulated
   within the AT_ENCR_DATA attribute.

10.14. AT_MAC

   The AT_MAC attribute is used for EAP-SIM message authentication.
   Section 8 specifies in which messages AT_MAC MUST be included.

   The value field of the AT_MAC attribute contains two reserved bytes
   followed by a keyed message authentication code (MAC). The MAC is
   calculated over the whole EAP packet and concatenated with optional
   message-specific data, with the exception that the value field of the
   MAC attribute is set to zero when calculating the MAC. The EAP
   packet includes the EAP header that begins with the Code field, the
   EAP-SIM header that begins with the Subtype field, and all the
   attributes, as specified in Section 8.1. The reserved bytes in
   AT_MAC are set to zero when sending and ignored on reception. The
   contents of the message-specific data that may be included in the MAC
   calculation are specified separately for each EAP-SIM message in
   Section 9.

   The format of the AT_MAC attribute is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     AT_MAC     | Length = 5    |           Reserved            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                           MAC                                 |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
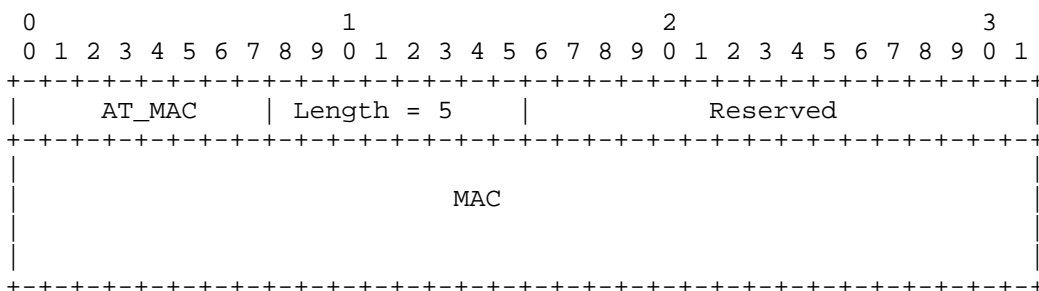
The MAC algorithm is an HMAC-SHA1-128 [RFC2104] keyed hash value.
(The HMAC-SHA1-128 value is obtained from the 20-byte HMAC-SHA1 value
by truncating the output to the first 16 bytes. Hence, the length of
the MAC is 16 bytes. The derivation of the authentication key
(K_aut) used in the calculation of the MAC is specified in Section 7.

When the AT_MAC attribute is included in an EAP-SIM message, the
recipient MUST process the AT_MAC attribute before looking at any
other attributes, except when processing EAP-Request/SIM/Challenge.
The processing of EAP-Request/SIM/Challenge is specified in Section
9.3. If the message authentication code is invalid, then the
recipient MUST ignore all other attributes in the message and operate
as specified in Section 6.3.

10.15. AT_COUNTER

The format of the AT_COUNTER attribute is shown below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  AT_COUNTER   | Length = 1    |            Counter            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
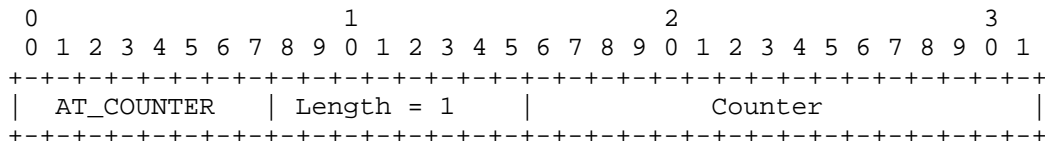
The value field of the AT_COUNTER attribute consists of a 16-bit
unsigned integer counter value, represented in network byte order.
This attribute MUST always be encrypted by encapsulating it within
the AT_ENCR_DATA attribute.

10.16. AT_COUNTER_TOO_SMALL

The format of the AT_COUNTER_TOO_SMALL attribute is shown below.
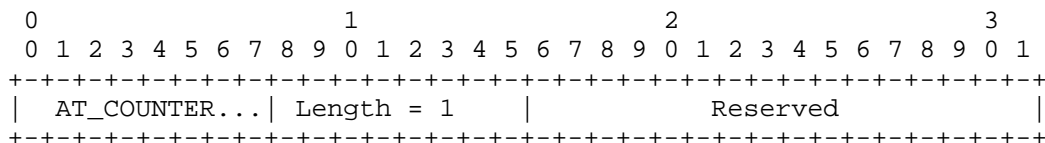
```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| AT_COUNTER...| Length = 1     |            Reserved           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The value field of this attribute consists of two reserved bytes,
which are set to zero upon sending and ignored upon reception. This
attribute MUST always be encrypted by encapsulating it within the
AT_ENCR_DATA attribute.

10.17. AT_NONCE_S

   The format of the AT_NONCE_S attribute is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | AT_NONCE_S    | Length = 5    |            Reserved           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                                                               |
   |                                                               |
   |                           NONCE_S                             |
   |                                                               |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
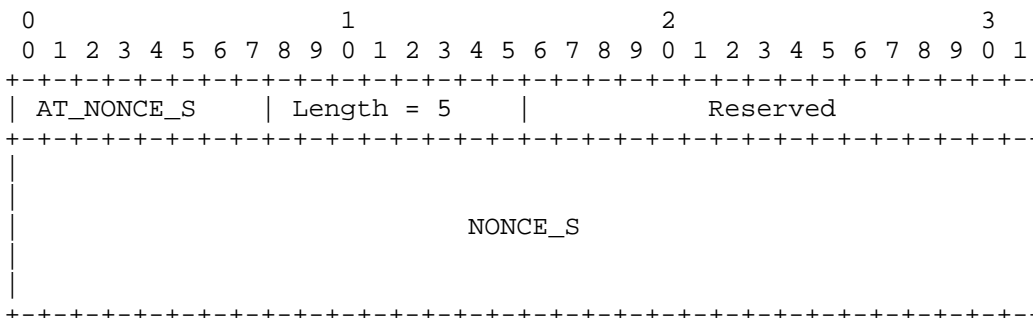
   The value field of the AT_NONCE_S attribute contains two reserved
   bytes followed by a random number freshly generated by the server (16
   bytes) for this EAP-SIM fast re-authentication. The random number is
   used as a challenge for the peer and also as a seed value for the new
   keying material. The reserved bytes are set to zero upon sending and
   ignored upon reception. This attribute MUST always be encrypted by
   encapsulating it within the AT_ENCR_DATA attribute.

   The server MUST NOT re-use the NONCE_S value from any previous
   EAP-SIM fast re-authentication exchange. The server SHOULD use a
   good source of randomness to generate NONCE_S. Please see [RFC4086]
   for more information about generating random numbers for security
   applications.

10.18. AT_NOTIFICATION

   The format of the AT_NOTIFICATION attribute is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |AT_NOTIFICATION| Length = 1    |S|P|   Notification Code       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
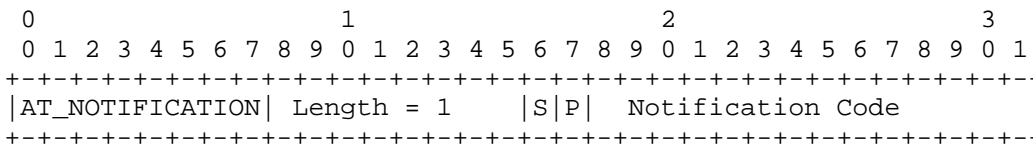
   The value field of this attribute contains a two-byte notification
   code. The first and second bit (S and P) of the notification code
   are interpreted as described in Section 6.

   The notification code values listed below have been reserved. The
   descriptions below illustrate the semantics of the notifications.

The peer implementation MAY use different wordings when presenting
the notifications to the user. The "requested service" depends on
the environment where EAP-SIM is applied.

0 - General failure after authentication. (Implies failure, used
after successful authentication.)

16384 - General failure. (Implies failure, used before
authentication.)

32768 - Success. User has been successfully authenticated. (Does
not imply failure, used after successful authentication). The usage
of this code is discussed in Section 6.2.

1026 - User has been temporarily denied access to the requested
service. (Implies failure, used after successful authentication.)

1031 - User has not subscribed to the requested service. (Implies
failure, used after successful authentication.)

10.19. AT_CLIENT_ERROR_CODE

The format of the AT_CLIENT_ERROR_CODE attribute is shown below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|AT_CLIENT_ERR..| Length = 1    |      Client Error Code        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
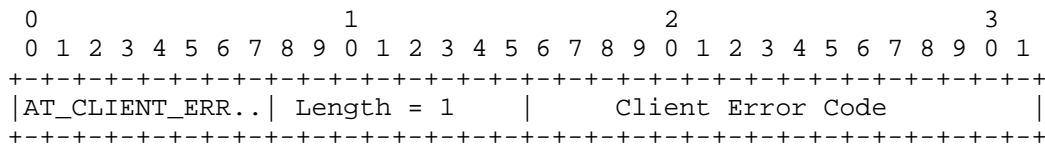
The value field of this attribute contains a two-byte client error
code. The following error code values have been reserved.

   0   "unable to process packet": a general error code

   1   "unsupported version": the peer does not support any of
         the versions listed in AT_VERSION_LIST

   2   "insufficient number of challenges": the peer's policy
         requires more triplets than the server included in AT_RAND

   3   "RANDs are not fresh": the peer believes that the RAND
         challenges included in AT_RAND were not fresh

11. IANA Considerations

   IANA has assigned the EAP type number 18 for this protocol.

   EAP-SIM shares most of the protocol design, such as attributes and
   message Subtypes, with EAP-AKA [EAP-AKA]. EAP-SIM protocol numbers
   should be administered in the same IANA registry as EAP-AKA. The
   initial values are listed in [EAP-AKA] for both protocols, so this
   document does not require any new registries or parameter allocation.
   As a common registry is used for EAP-SIM and EAP-AKA, the protocol
   number allocation policy for both protocols is specified in
   [EAP-AKA].

12. Security Considerations

   The EAP specification [RFC3748] describes the security
   vulnerabilities of EAP, which does not include its own security
   mechanisms. This section discusses the claimed security properties
   of EAP-SIM, as well as vulnerabilities and security recommendations.

12.1. A3 and A8 Algorithms

   The GSM A3 and A8 algorithms are used in EAP-SIM. [GSM-03.20]
   specifies the general GSM authentication procedure and the external
   interface (inputs and outputs) of the A3 and A8 algorithms. The
   operation of these functions falls completely within the domain of an
   individual operator, and therefore, the functions are specified by
   each operator rather than being fully standardised. The GSM-MILENAGE
   algorithm, specified publicly in [3GPP-TS-55.205], is an example
   algorithm set for A3 and A8 algorithms.

   The security of the A3 and A8 algorithms is important to the security
   of EAP-SIM. Some A3/A8 algorithms have been compromised; see [GSM-
   Cloning] for discussion about the security of COMP-128 version 1.
   Note that several revised versions of the COMP-128 A3/A8 algorithm
   have been devised after the publication of these weaknesses and that
   the publicly specified GSM-MILENAGE algorithm is not vulnerable to
   any known attacks.

12.2. Identity Protection

   EAP-SIM includes optional identity privacy support that protects the
   privacy of the subscriber identity against passive eavesdropping.
   This document only specifies a mechanism to deliver pseudonyms from
   the server to the peer as part of an EAP-SIM exchange. Hence, a peer
   that has not yet performed any EAP-SIM exchanges does not typically
   have a pseudonym available. If the peer does not have a pseudonym
   available, then the privacy mechanism cannot be used, but the

permanent identity will have to be sent in the clear. The terminal
SHOULD store the pseudonym in a non-volatile memory so that it can be
maintained across reboots. An active attacker that impersonates the
network may use the AT_PERMANENT_ID_REQ attribute to attempt to learn
the subscriber's permanent identity. However, as discussed in
Section 4.2.2, the terminal can refuse to send the cleartext
permanent identity if it believes that the network should be able to
recognize the pseudonym.

If the peer and server cannot guarantee that the pseudonym will be
maintained reliably, and identity privacy is required, then
additional protection from an external security mechanism (such as
Protected Extensible Authentication Protocol (PEAP) [PEAP]) may be
used. If an external security mechanism is in use, the identity
privacy features of EAP-SIM may not be useful. The security
considerations of using an external security mechanism with EAP-SIM
are beyond the scope of this document.

12.3. Mutual Authentication and Triplet Exposure

EAP-SIM provides mutual authentication. The peer believes that the
network is authentic because the network can calculate a correct
AT_MAC value in the EAP-Request/SIM/Challenge packet. To calculate
AT_MAC it is sufficient to know the RAND and Kc values from the GSM
triplets (RAND, SRES, Kc) used in the authentication. Because the
network selects the RAND challenges and the triplets, an attacker
that knows n (2 or 3) GSM triplets for the subscriber is able to
impersonate a valid network to the peer. (Some peers MAY employ an
implementation-specific counter-measure against impersonating a valid
network by re-using a previously used RAND; see below.) In other
words, the security of EAP-SIM is based on the secrecy of Kc keys,
which are considered secret intermediate results in the EAP-SIM
cryptographic calculations.

Given physical access to the SIM card, it is easy to obtain any
number of GSM triplets.

Another way to obtain triplets is to mount an attack on the peer
platform via a virus or other malicious piece of software. The peer
SHOULD be protected against triplet querying attacks by malicious
software. Care should be taken not to expose Kc keys to attackers
when they are stored or handled by the peer, or transmitted between
subsystems of the peer. Steps should be taken to limit the
transport, storage, and handling of these values outside a protected
environment within the peer. However, the virus protection of the
peer and the security capabilities of the peer's operating system are
outside the scope of this document.

The EAP-SIM server typically obtains the triplets from the Home
Location Register (HLR). An attacker might try to obtain triplets by
attacking against the network used between the EAP-SIM server and the
HLR. Care should be taken not to expose Kc keys to attackers when
they are stored or handled by the EAP-SIM server, or transmitted
between the EAP server and the HLR. Steps should be taken to limit
the transport, storage, and handling of these values outside a
protected environment. However, the protection of the communications
between the EAP-SIM server and the HLR is outside the scope of this
document.

If the same SIM credentials are also used for GSM traffic, the
triplets could be revealed in the GSM network; see Section 12.8.

In GSM, the network is allowed to re-use the RAND challenge in
consecutive authentication exchanges. This is not allowed in
EAP-SIM. The EAP-SIM server is mandated to use fresh triplets (RAND
challenges) in consecutive authentication exchanges, as specified in
Section 3. EAP-SIM does not mandate any means for the peer to check
if the RANDs are fresh, so the security of the scheme leans on the
secrecy of the triplets. However, the peer MAY employ
implementation-specific mechanisms to remember some of the previously
used RANDs, and the peer MAY check the freshness of the server's
RANDs. The operation in cases when the peer detects that the RANDs
are not fresh is specified in Section 6.3.1.

Preventing the re-use of authentication vectors has been taken into
account in the design of the UMTS Authentication and Key Agreement
(AKA), which is used in EAP-AKA [EAP-AKA]. In cases when the triplet
re-use properties of EAP-SIM are not considered sufficient, it is
advised to use EAP-AKA.

Note that EAP-SIM mutual authentication is done with the EAP server.
In general, EAP methods do not authenticate the identity or services
provided by the EAP authenticator (if distinct from the EAP server)
unless they provide the so-called channel bindings property. The
vulnerabilities related to this have been discussed in [RFC3748],
[EAP-Keying], [Service-Identity].

EAP-SIM does not provide the channel bindings property, so it only
authenticates the EAP server. However, ongoing work such as
[Service-Identity] may provide such support as an extension to
popular EAP methods such as EAP-TLS, EAP-SIM, or EAP-AKA.

12.4. Flooding the Authentication Centre

   The EAP-SIM server typically obtains authentication vectors from the
   Authentication Centre (AuC). EAP-SIM introduces a new usage for the
   AuC. The protocols between the EAP-SIM server and the AuC are out of
   the scope of this document. However, it should be noted that a
   malicious EAP-SIM peer may generate a lot of protocol requests to
   mount a denial of service attack. The EAP-SIM server implementation
   SHOULD take this into account and SHOULD take steps to limit the
   traffic that it generates towards the AuC, preventing the attacker
   from flooding the AuC and from extending the denial of service attack
   from EAP-SIM to other users of the AuC.

12.5. Key Derivation

   EAP-SIM supports key derivation. The key hierarchy is specified in
   Section 7. EAP-SIM combines several GSM triplets in order to
   generate stronger keying material and stronger AT_MAC values. The
   actual strength of the resulting keys depends, among other things, on
   operator-specific parameters including authentication algorithms, the
   strength of the Ki key, and the quality of the RAND challenges. For
   example, some SIM cards generate Kc keys with 10 bits set to zero.
   Such restrictions may prevent the concatenation technique from
   yielding strong session keys. Because the strength of the Ki key is
   128 bits, the ultimate strength of any derived secret key material is
   never more than 128 bits.

   It should also be noted that a security policy that allows n=2 to be
   used may compromise the security of a future policy that requires
   three triplets, because adversaries may be able to exploit the
   messages exchanged when the weaker policy is applied.

   There is no known way to obtain complete GSM triplets by mounting an
   attack against EAP-SIM. A passive eavesdropper can learn n*RAND and
   AT_MAC and may be able to link this information to the subscriber
   identity. An active attacker that impersonates a GSM subscriber can
   easily obtain n*RAND and AT_MAC values from the EAP server for any
   given subscriber identity. However, calculating the Kc and SRES
   values from AT_MAC would require the attacker to reverse the keyed
   message authentication code function HMAC-SHA1-128.

   As EAP-SIM does not expose any values calculated from an individual
   GSM Kc keys, it is not possible to mount a brute force attack on only
   one of the Kc keys in EAP-SIM. Therefore, when considering brute
   force attacks on the values exposed in EAP-SIM, the effective length
   of EAP-SIM session keys is not compromised by the fact that they are

combined from several shorter keys, i.e., the effective length of 128
bits may be achieved. For additional considerations, see Section
12.8.

12.6. Cryptographic Separation of Keys and Session Independence

The EAP Transient Keys used to protect EAP-SIM packets (K_encr,
K_aut), the Master Session Key, and the Extended Master Session Key
are cryptographically separate in EAP-SIM. An attacker cannot derive
any non-trivial information about any of these keys based on the
other keys. An attacker also cannot calculate the pre-shared secret
(Ki) from the GSM Kc keys, from EAP-SIM K_encr, from EAP-SIM K_aut,
from the Master Session Key, or from the Extended Master Session Key.

Each EAP-SIM exchange generates fresh keying material, and the keying
material exported from the method upon separate EAP-SIM exchanges is
cryptographically separate. The EAP-SIM peer contributes to the
keying material with the NONCE_MT parameter, which must be chosen
freshly for each full authentication exchange. The EAP server is
mandated to choose the RAND challenges freshly for each full
authentication exchange. If either the server or the peer chooses
its random value (NONCE_MT or RAND challenges) freshly, even if the
other entity re-used its value from a previous exchange, then the EAP
Transient Keys, the Master Session Key, and the Extended Master
Session Key will be different and cryptographically separate from the
corresponding values derived upon the previous full authentication
exchange.

On fast re-authentication, freshness of the Master Session Key and
the Extended Master Session Key is provided with a counter
(AT_COUNTER). The same EAP Transient Keys (K_encr, K_aut) that were
used in the full authentication exchange are used to protect the EAP
negotiation. However, replay and integrity protection across all the
fast re-authentication exchanges that use the same EAP Transient Keys
is provided with AT_COUNTER.

[RFC3748] defines session independence as the "demonstration that
passive attacks (such as capture of the EAP conversation) or active
attacks (including compromise of the MSK or EMSK) do not enable
compromise of subsequent or prior MSKs or EMSKs". Because the MSKs
and EMSKs are separate between EAP exchanges, EAP-SIM supports this
security claim.

It should be noted that [Patel-2003], which predates [RFC3748], uses
a slightly different meaning for session independence. The EAP-SIM
protocol does not allow the peer to ensure that different Kc key
values would be used in different exchanges. Only the server is able
to ensure that fresh RANDs, and therefore, fresh Kc keys are used.

Hence, the peer cannot guarantee EAP-SIM sessions to be independent
with regard to the internal Kc values. However, in EAP-SIM, the Kc
keys are considered to be secret intermediate results, which are not
exported outside the method. See Section 12.3 for more information
about RAND re-use.

12.7. Dictionary Attacks

   Because EAP-SIM is not a password protocol, it is not vulnerable to
   dictionary attacks. (The pre-shared symmetric secret stored on the
   SIM card is not a passphrase, nor is it derived from a passphrase.)

12.8. Credentials Re-use

   EAP-SIM cannot prevent attacks over the GSM or GPRS radio networks.
   If the same SIM credentials are also used in GSM or GPRS, it is
   possible to mount attacks over the cellular interface.

   A passive attacker can eavesdrop GSM or GPRS traffic and obtain RAND,
   SRES pairs. He can then use a brute force attack or other
   cryptanalysis techniques to obtain the 64-bit Kc keys used to encrypt
   the GSM or GPRS data. This makes it possible to attack each 64-bit
   key separately.

   An active attacker can mount a "rogue GSM/GPRS base station attack",
   replaying previously seen RAND challenges to obtain SRES values. He
   can then use a brute force attack to obtain the Kc keys. If
   successful, the attacker can impersonate a valid network or decrypt
   previously seen traffic, because EAP-SIM does not provide perfect
   forward secrecy (PFS).

   Due to several weaknesses in the GSM encryption algorithms, the
   effective key strength of the Kc keys is much less than the expected
   64 bits (no more than 40 bits if the A5/1 GSM encryption algorithm is
   used; as documented in [Barkan-2003], an active attacker can force
   the peer to use the weaker A5/2 algorithm that can be broken in less
   than a second).

   Because the A5 encryption algorithm is not used in EAP-SIM, and
   because EAP-SIM does not expose any values calculated from individual
   Kc keys, it should be noted that these attacks are not possible if
   the SIM credentials used in EAP-SIM are not shared in GSM/GPRS.

   At the time this document was written, the 3rd Generation Partnership
   Project (3GPP) has started to work on fixes to these A5
   vulnerabilities. One of the solution proposals discussed in 3GPP is
   integrity-protected A5 version negotiation, which would require the
   base station to prove knowledge of the Kc key before the terminal

sends any values calculated from the Kc to the network. Another
proposal is so-called special RANDs, where some bits of the RAND
challenge would be used for cryptographic separation by indicating
the allowed use of the triplet, such as the allowed A5 algorithm in
GSM or the fact that the triplet is intended for EAP-SIM. This is
currently a work in progress, and the mechanisms have not been
selected yet.

12.9. Integrity and Replay Protection, and Confidentiality

   AT_MAC, AT_IV, AT_ENCR_DATA, and AT_COUNTER attributes are used to
   provide integrity, replay and confidentiality protection for EAP-SIM
   requests and responses. Integrity protection with AT_MAC includes
   the EAP header. These attributes cannot be used during the
   EAP/SIM/Start roundtrip. However, the protocol values (user identity
   string, NONCE_MT, and version negotiation parameters) are
   (implicitly) protected by later EAP-SIM messages by including them in
   key derivation.

   Integrity protection (AT_MAC) is based on a keyed message
   authentication code. Confidentiality (AT_ENCR_DATA and AT_IV) is
   based on a block cipher.

   Confidentiality protection is applied only to a part of the protocol
   fields. The table of attributes in Section 10.1 summarizes which
   fields are confidentiality-protected. It should be noted that the
   error and notification code attributes AT_CLIENT_ERROR_CODE and
   AT_NOTIFICATION are not confidential, but they are transmitted in the
   clear. Identity protection is discussed in Section 12.2.

   On full authentication, replay protection of the EAP exchange is
   provided by the RAND values from the underlying GSM authentication
   scheme and the use of the NONCE_MT value. Protection against replays
   of EAP-SIM messages is also based on the fact that messages that can
   include AT_MAC can only be sent once with a certain EAP-SIM Subtype,
   and on the fact that a different K_aut key will be used for
   calculating AT_MAC in each full authentication exchange.

   On fast re-authentication, a counter included in AT_COUNTER and a
   server random nonce is used to provide replay protection. The
   AT_COUNTER attribute is also included in EAP-SIM notifications if it
   is used after successful authentication in order to provide replay
   protection between re-authentication exchanges.

   Because EAP-SIM is not a tunneling method, EAP-Request/Notification,
   EAP-Response/Notification, EAP-Success, or EAP-Failure packets are
   not confidential, integrity-protected, or replay-protected in
   EAP-SIM. On physically insecure networks, this may enable an

attacker to send false notifications to the peer and to mount denial
of service attacks by spoofing these packets. As discussed in
Section 6.3, the peer will only accept EAP-Success after the peer
successfully authenticates the server. Hence, the attacker cannot
force the peer to believe successful mutual authentication has
occurred until the peer successfully authenticates the server or
after the peer fails to authenticate the server.

The security considerations of EAP-SIM result indications are covered
in Section 12.11

An eavesdropper will see the EAP-Request/Notification,
EAP-Response/Notification, EAP-Success, and EAP-Failure packets sent
in the clear. With EAP-SIM, confidential information MUST NOT be
transmitted in EAP Notification packets.

12.10. Negotiation Attacks

EAP-SIM does not protect the EAP-Response/Nak packet. Because
EAP-SIM does not protect the EAP method negotiation, EAP method
downgrading attacks may be possible, especially if the user uses the
same identity with EAP-SIM and other EAP methods.

EAP-SIM includes a version negotiation procedure. In EAP-SIM the
keying material derivation includes the version list and selected
version to ensure that the protocol cannot be downgraded and that the
peer and server use the same version of EAP-SIM.

EAP-SIM does not support ciphersuite negotiation.

12.11. Protected Result Indications

EAP-SIM supports optional protected success indications and
acknowledged failure indications. If a failure occurs after
successful authentication, then the EAP-SIM failure indication is
integrity- and replay-protected.

Even if an EAP-Failure packet is lost when using EAP-SIM over an
unreliable medium, then the EAP-SIM failure indications will help
ensure that the peer and EAP server will know the other party's
authentication decision. If protected success indications are used,
then the loss of Success packet will also be addressed by the
acknowledged, integrity- and replay-protected EAP-SIM success
indication. If the optional success indications are not used, then
the peer may end up believing that the server succeeded
authentication, when it actually failed. Since access will not be

granted in this case, protected result indications are not needed
unless the client is not able to realize it does not have access for
an extended period of time.

12.12. Man-in-the-Middle Attacks

In order to avoid man-in-the-middle attacks and session hijacking,
user data SHOULD be integrity-protected on physically insecure
networks. The EAP-SIM Master Session Key, or keys derived from it,
MAY be used as the integrity protection keys, or, if an external
security mechanism such as PEAP is used, then the link integrity
protection keys MAY be derived by the external security mechanism.

There are man-in-the-middle attacks associated with the use of any
EAP method within a tunneled protocol. For instance, an early
version of PEAP [PEAP-02] was vulnerable to this attack. This
specification does not address these attacks. If EAP-SIM is used
with a tunneling protocol, there should be cryptographic binding
provided between the protocol and EAP-SIM to prevent
man-in-the-middle attacks through rogue authenticators being able to
setup one-way authenticated tunnels. For example, newer versions of
PEAP include such cryptographic binding. The EAP-SIM Master Session
Key MAY be used to provide the cryptographic binding. However, the
mechanism by which the binding is provided depends on the tunneling
protocol and is beyond the scope of this document.

12.13. Generating Random Numbers

An EAP-SIM implementation SHOULD use a good source of randomness to
generate the random numbers required in the protocol. Please see
[RFC4086] for more information on generating random numbers for
security applications.

13. Security Claims

This section provides the security claims required by [RFC3748].

Auth. mechanism: EAP-SIM is based on the GSM SIM mechanism, which is
a challenge/response authentication and key agreement mechanism based
on a symmetric 128-bit pre-shared secret. EAP-SIM also makes use of
a peer challenge to provide mutual authentication.

Ciphersuite negotiation: No

Mutual authentication: Yes (Section 12.3)

Integrity protection: Yes (Section 12.9)

   Replay protection: Yes (Section 12.9)

   Confidentiality: Yes, except method-specific success and failure
   indications (Section 12.2, Section 12.9)

   Key derivation: Yes

   Key strength: EAP-SIM supports key derivation with 128-bit effective
   key strength (Section 12.5). However, as discussed in Section 11, if
   the same credentials are used in GSM/GPRS and in EAP-SIM, then the
   key strength may be reduced considerably, basically to the same level
   as in GSM, by mounting attacks over GSM/GPRS. For example an active
   attack using a false GSM/GPRS base station reduces the effective key
   strength to almost zero.

   Description of key hierarchy: Please see Section 7.

   Dictionary attack protection: N/A (Section 12.7)

   Fast reconnect: Yes

   Cryptographic binding: N/A

   Session independence: Yes (Section 12.6)

   Fragmentation: No

   Channel binding: No

   Indication of vulnerabilities: Vulnerabilities are discussed in
   Section 12.

14. Acknowledgements and Contributions

14.1. Contributors

   In addition to the editors, Nora Dabbous, Jose Puthenkulam, and
   Prasanna Satarasinghe were significant contributors to this document.

   Pasi Eronen and Jukka-Pekka Honkanen contributed Appendix A.

14.2. Acknowledgements

   Juha Ala-Laurila, N. Asokan, Jan-Erik Ekberg, Patrik Flykt,
   Jukka-Pekka Honkanen, Antti Kuikka, Jukka Latva, Lassi Lehtinen, Jyri
   Rinnemaa, Timo Takamaki, and Raimo Vuonnala contributed many original
   ideas and concepts to this protocol.

N. Asokan, Pasi Eronen, and Jukka-Pekka Honkanen contributed and
helped in innumerable ways during the development of the protocol.

Valtteri Niemi and Kaisa Nyberg contributed substantially to the
design of the key derivation and the fast re-authentication
procedure, and have also provided their cryptographic expertise in
many discussions related to this protocol.

Simon Blake-Wilson provided very helpful comments on key derivation
and version negotiation.

Thanks to Greg Rose for his very valuable comments to an early
version of this specification [S3-020125], and for reviewing and
providing very useful comments on version 12.

Thanks to Bernard Aboba, Vladimir Alperovich, Florent Bersani,
Jacques Caron, Gopal Dommety, Augustin Farrugia, Mark Grayson, Max de
Groot, Prakash Iyer, Nishi Kant, Victor Lortz, Jouni Malinen, Sarvar
Patel, Tom Porcher, Michael Richardson, Stefan Schroeder, Uma
Shankar, Jesse Walker, and Thomas Wieland for their contributions and
critiques. Special thanks to Max for proposing improvements to the
MAC calculation.

Thanks to Glen Zorn for reviewing this document and for providing
very useful comments on the protocol.

Thanks to Sarvar Patel for his review of the protocol [Patel-2003].

Thanks to Bernard Aboba for reviewing this document for RFC 3748
compliance.

The identity privacy support is based on the identity privacy support
of [EAP-SRP]. The attribute format is based on the extension format
of Mobile IPv4 [RFC3344].

This protocol has been partly developed in parallel with EAP-AKA
[EAP-AKA], and hence this specification incorporates many ideas from
Jari Arkko.

14.2.1. Contributors' Addresses

   Nora Dabbous
   Gemplus
   34 rue Guynemer
   92447 Issy les Moulineaux
   France

   Phone: +33 1 4648 2000
   EMail: nora.dabbous@gemplus.com

   Jose Puthenkulam
   Intel Corporation
   2111 NE 25th Avenue, JF2-58
   Hillsboro, OR 97124
   USA

   Phone: +1 503 264 6121
   EMail: jose.p.puthenkulam@intel.com

   Prasanna Satarasinghe
   Transat Technologies
   180 State Street, Suite 240
   Southlake, TX 76092
   USA

   Phone: + 1 817 4814412
   EMail: prasannas@transat-tech.com

15. References

15.1. Normative References

   [GSM-03.20]   European Telecommunications Standards Institute,
                         "GSM Technical Specification GSM 03.20 (ETS 300
                         534): "Digital cellular telecommunication system
                         (Phase 2); Security related network functions"",
                         August 1997.

   [RFC2119]   Bradner, S., "Key words for use in RFCs to
                         Indicate Requirement Levels", BCP 14, RFC 2119,
                         March 1997.

   [GSM-03.03]   European Telecommunications Standards Institute,
                         "GSM Technical Specification GSM 03.03 (ETS 300
                         523): "Digital cellular telecommunication system
                         (Phase 2); Numbering, addressing and
                         identification"", April 1997.

   [RFC2104]   Krawczyk, H., Bellare, M., and R. Canetti, "HMAC:
                         Keyed-Hashing for Message Authentication", RFC
                         2104, February 1997.

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

   [AES]   National Institute of Standards and Technology,
                         "Federal Information Processing Standards (FIPS)
                         Publication 197, "Advanced Encryption Standard
                         (AES)"", November 2001.
                         http://csrc.nist.gov/publications/fips/fips197/
                         fips-197.pdf

   [CBC]   National Institute of Standards and Technology,
                         "NIST Special Publication 800-38A, "Recommendation
                         for Block Cipher Modes of Operation - Methods and
                         Techniques"", December 2001.
                         http://csrc.nist.gov/publications/nistpubs/
                         800-38a/sp800-38a.pdf

   [SHA-1]   National Institute of Standards and Technology,
                         U.S. Department of Commerce, "Federal Information
                         Processing Standard (FIPS) Publication 180-1,
                         "Secure Hash Standard"", April 1995.

    [PRF]   National Institute of Standards and Technology,
                           "Federal Information Processing Standards (FIPS)
                           Publication 186-2 (with change notice); Digital
                           Signature Standard (DSS)", January 2000.
                           Available on-line at:
                           http://csrc.nist.gov/publications/
                           fips/fips186-2/fips186-2-change1.pdf

    [RFC3629]   Yergeau, F., "UTF-8, a transformation format of
                           ISO 10646", STD 63, RFC 3629, November 2003.

    [RFC3748]   Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J.,
                           and H. Levkowetz, "Extensible Authentication
                           Protocol (EAP)", RFC 3748, June 2004.

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

15.2. Informative References

    [3GPP-TS-23.003]   3rd Generation Partnership Project, "3GPP
                           Technical Specification 3GPP TS 23.003 V6.8.0:
                           "3rd Generation Parnership Project; Technical
                           Specification Group Core Network; Numbering,
                           addressing and identification (Release 6)"",
                           December 2005.

    [3GPP-TS-55.205]   3rd Generation Partnership Project, "3GPP
                           Technical Specification 3GPP TS 55.205 V 6.0.0:
                           "3rd Generation Partnership Project; Technical
                           Specification Group Services and System Aspects;
                           Specification of the GSM-MILENAGE Algorithms: An
                           example algorithm set for the GSM Authentication
                           and Key Generation functions A3 and A8 (Release
                           6)"", December 2002.

    [PEAP]   Palekar, A., Simon, D., Zorn, G., Salowey, J.,
                           Zhou, H., and S. Josefsson, "Protected EAP
                           Protocol (PEAP) Version 2", Work in Progress,
                           October 2004.

    [PEAP-02]   Anderson, H., Josefsson, S., Zorn, G., Simon, D.,
                           and A. Palekar, "Protected EAP Protocol (PEAP)",
                           Work in Progress, February 2002.

   [EAP-Keying]   Aboba, B., Simon, D., Arkko, J., Eronen, P., and
                         H. Levkowetz, "Extensible Authentication Protocol
                         (EAP) Key Management Framework", Work in Progress,
                         October 2005.

   [Service-Identity] Arkko, J. and P. Eronen, "Authenticated Service
                         Information for the Extensible Authentication
                         Protocol (EAP)", Work in Progress, October 2004.

   [RFC4086]   Eastlake, D., 3rd, Schiller, J., and S. Crocker,
                         "Randomness Requirements for Security", BCP 106,
                         RFC 4086, June 2005.

   [S3-020125]   Qualcomm, "Comments on draft EAP/SIM, 3rd
                         Generation Partnership Project document 3GPP TSG
                         SA WG3 Security S3#22, S3-020125", February 2002.

   [RFC3344]   Perkins, C., "IP Mobility Support for IPv4", RFC
                         3344, August 2002.

   [RFC2548]   Zorn, G., "Microsoft Vendor-specific RADIUS
                         Attributes ", RFC 2548, March 1999.

   [EAP-SRP]   Carlson, J., Aboba, B., and H. Haverinen, "EAP
                         SRP-SHA1 Authentication Protocol", Work in
                         Progress, July 2001.

   [GSM-Cloning]   Wagner, D., "GSM Cloning". Web page about
                         COMP-128 version 1 vulnerabilities, available at
                         http://www.isaac.cs.berkeley.edu/isaac/gsm.html

   [Barkan-2003]   Barkan, E., Biham, E., and N. Keller, "Instant
                         Ciphertext-Only Cryptanalysis of GSM Encrypted
                         Communications". available on-line at
                         http://cryptome.org/gsm-crack-bbk.pdf

   [Patel-2003]   Patel, S., "Analysis of EAP-SIM Session Key
                         Agreement". Posted to the EAP mailing list 29
                         May,2003. http://
                         mail.frascone.com/pipermail/public/eap/2003-May/
                         001267.html

Appendix A.   Test Vectors

   Test vectors for the NIST FIPS 186-2 pseudo-random number generator
   [PRF] are available at the following URL:
   http://csrc.nist.gov/encryption/dss/Examples-1024bit.pdf

   The following examples show the contents of EAP-SIM packets on full
   authentication and fast re-authentication.

A.1.   EAP-Request/Identity

   The first packet is a plain Identity Request:

        01   ; Code: Request
        00   ; Identifier: 0
        00 05   ; Length: 5 octets
        01   ; Type: Identity

A.2.   EAP-Response/Identity

   The client's identity is "1244070100000001@eapsim.foo", so it
   responds with the following packet:

        02   ; Code: Response
        00   ; Identifier: 0
        00 20   ; Length: 32 octets
        01   ; Type: Identity
           31 32 34 34   ; "1244070100000001@eapsim.foo"
           30 37 30 31
           30 30 30 30
           30 30 30 31
           40 65 61 70
           73 69 6d 2e
           66 6f 6f

A.3. EAP-Request/SIM/Start

   The server's first packet looks like this:

```
        01                      ; Code: Request
        01                      ; Identifier: 1
        00 10                   ; Length: 16 octets
        12                      ; Type: EAP-SIM
           0a                   ; EAP-SIM subtype: Start
           00 00                ; (reserved)
           0f                   ; Attribute type: AT_VERSION_LIST
              02                ; Attribute length: 8 octets (2*4)
              00 02             ; Actual version list length: 2 octets
              00 01             ; Version: 1
              00 00             ; (attribute padding)
```

A.4. EAP-Response/SIM/Start

   The client selects a nonce and responds with the following packet:

```
        02                      ; Code: Response
        01                      ; Identifier: 1
        00 20                   ; Length: 32 octets
        12                      ; Type: EAP-SIM
           0a                   ; EAP-SIM subtype: Start
           00 00                ; (reserved)
           07                   ; Attribute type: AT_NONCE_MT
              05                ; Attribute length: 20 octets (5*4)
              00 00             ; (reserved)
              01 23 45 67       ; NONCE_MT value
              89 ab cd ef
              fe dc ba 98
              76 54 32 10
           10                   ; Attribute type: AT_SELECTED_VERSION
              01                ; Attribute length: 4 octets (1*4)
              00 01             ; Version: 1
```

A.5. EAP-Request/SIM/Challenge

   Next, the server selects three authentication triplets

           (RAND1,SRES1,Kc1) = (10111213 14151617 18191a1b 1c1d1e1f,
                                 d1d2d3d4,
                                 a0a1a2a3 a4a5a6a7)
           (RAND2,SRES2,Kc2) = (20212223 24252627 28292a2b 2c2d2e2f,
                                 e1e2e3e4,
                                 b0b1b2b3 b4b5b6b7)
           (RAND3,SRES3,Kc3) = (30313233 34353637 38393a3b 3c3d3e3f,
                                 f1f2f3f4,
                                 c0c1c2c3 c4c5c6c7)

   Next, the MK is calculated as specified in Section 7*.

   MK = e576d5ca 332e9930 018bf1ba ee2763c7 95b3c712

   And the other keys are derived using the PRNG:

           K_encr = 536e5ebc 4465582a a6a8ec99 86ebb620
           K_aut = 25af1942 efcbf4bc 72b39434 21f2a974
           MSK =   39d45aea f4e30601 983e972b 6cfd46d1
                    c3637733 65690d09 cd44976b 525f47d3
                    a60a985e 955c53b0 90b2e4b7 3719196a
                    40254296 8fd14a88 8f46b9a7 886e4488
           EMSK =   5949eab0 fff69d52 315c6c63 4fd14a7f
                    0d52023d 56f79698 fa6596ab eed4f93f
                    bb48eb53 4d985414 ceed0d9a 8ed33c38
                    7c9dfdab 92ffbdf2 40fcecf6 5a2c93b9

   Next, the server selects a pseudonym and a fast re-authentication
   identity (in this case, "w8w49PexCazWJ&xCIARmxuMKht5S1sxR
   DqXSEFBEg3DcZP9cIxTe5J4OyIwNGVzxeJOU1G" and
   "Y24fNSrz8BP274jOJaF17WfxI8YO7QX0
   0pMXk9XMMVOw7broaNhTczuFq53aEpOkk3L0dm@eapsim.foo", respectively).

   The following plaintext will be encrypted and stored in the
   AT_ENCR_DATA attribute:

```
     84                    ; Attribute type: AT_NEXT_PSEUDONYM
        13                 ; Attribute length: 76 octets (19*4)
        00 46              ; Actual pseudonym length: 70 octets
        77 38 77 34 39 50 65 78 43 61 7a 57 4a 26 78 43
        49 41 52 6d 78 75 4d 4b 68 74 35 53 31 73 78 52
        44 71 58 53 45 46 42 45 67 33 44 63 5a 50 39 63
        49 78 54 65 35 4a 34 4f 79 49 77 4e 47 56 7a 78
        65 4a 4f 55 31 47
        00 00              ; (attribute padding)
     85                    ; Attribute type: AT_NEXT_REAUTH_ID
        16                 ; Attribute length: 88 octets (22*4)
        00 51              ; Actual re-auth identity length: 81 octets
        59 32 34 66 4e 53 72 7a 38 42 50 32 37 34 6a 4f
        4a 61 46 31 37 57 66 78 49 38 59 4f 37 51 58 30
        30 70 4d 58 6b 39 58 4d 4d 56 4f 77 37 62 72 6f
        61 4e 68 54 63 7a 75 46 71 35 33 61 45 70 4f 6b
        6b 33 4c 30 64 6d 40 65 61 70 73 69 6d 2e 66 6f
        6f
        00 00 00           ; (attribute padding)
     06                    ; Attribute type: AT_PADDING
        03                 ; Attribute length: 12 octets (3*4)
        00 00 00 00
        00 00 00 00
        00 00
```

   The EAP packet looks like this:

```
     01                    ; Code: Request
     02                    ; Identifier: 2
     01 18                 ; Length: 280 octets
     12                    ; Type: EAP-SIM
        0b                 ; EAP-SIM subtype: Challenge
        00 00              ; (reserved)
        01                 ; Attribute type: AT_RAND
           0d              ; Attribute length: 52 octets (13*4)
           00 00           ; (reserved)
           10 11 12 13     ; first RAND
           14 15 16 17
           18 19 1a 1b
           1c 1d 1e 1f
           20 21 22 23     ; second RAND
           24 25 26 27
           28 29 2a 2b
           2c 2d 2e 2f
```

```
        30 31 32 33     ; third RAND
        34 35 36 37
        38 39 3a 3b
        3c 3d 3e 3f
     81                 ; Attribute type: AT_IV
        05              ; Attribute length: 20 octets (5*4)
        00 00           ; (reserved)
        9e 18 b0 c2     ; IV value
        9a 65 22 63
        c0 6e fb 54
        dd 00 a8 95
     82                 ; Attribute type: AT_ENCR_DATA
        2d              ; Attribute length: 180 octets (45*4)
        00 00           ; (reserved)
        55 f2 93 9b bd b1 b1 9e a1 b4 7f c0 b3 e0 be 4c
        ab 2c f7 37 2d 98 e3 02 3c 6b b9 24 15 72 3d 58
        ba d6 6c e0 84 e1 01 b6 0f 53 58 35 4b d4 21 82
        78 ae a7 bf 2c ba ce 33 10 6a ed dc 62 5b 0c 1d
        5a a6 7a 41 73 9a e5 b5 79 50 97 3f c7 ff 83 01
        07 3c 6f 95 31 50 fc 30 3e a1 52 d1 e1 0a 2d 1f
        4f 52 26 da a1 ee 90 05 47 22 52 bd b3 b7 1d 6f
        0c 3a 34 90 31 6c 46 92 98 71 bd 45 cd fd bc a6
        11 2f 07 f8 be 71 79 90 d2 5f 6d d7 f2 b7 b3 20
        bf 4d 5a 99 2e 88 03 31 d7 29 94 5a ec 75 ae 5d
        43 c8 ed a5 fe 62 33 fc ac 49 4e e6 7a 0d 50 4d
     0b                 ; Attribute type: AT_MAC
        05              ; Attribute length: 20 octets (5*4)
        00 00           ; (reserved)
        fe f3 24 ac     ; MAC value
        39 62 b5 9f
        3b d7 82 53
        ae 4d cb 6a
```

   The MAC is calculated over the EAP packet above (with MAC value set
   to zero), followed by the NONCE_MT value (a total of 296 bytes).

A.6. EAP-Response/SIM/Challenge

   The client's response looks like this:

```
   02                      ; Code: Response
   02                      ; Identifier: 2
   00 1c                   ; Length: 28 octets
   12                      ; Type: EAP-SIM
      0b                   ; EAP-SIM subtype: Challenge
      00 00                ; (reserved)
      0b                   ; Attribute type: AT_MAC
         05                ; Attribute length: 20 octets (5*4)
         00 00             ; (reserved)
         f5 6d 64 33       ; MAC value
         e6 8e d2 97
         6a c1 19 37
         fc 3d 11 54
```

   The MAC is calculated over the EAP packet above (with MAC value set
   to zero), followed by the SRES values (a total of 40 bytes).

A.7. EAP-Success

   The last packet is an EAP-Success:

```
   03                      ; Code: Success
   02                      ; Identifier: 2
   00 04                   ; Length: 4 octets
```

A.8. Fast Re-authentication

   When performing fast re-authentication, the EAP-Request/Identity
   packet is the same as usual. The EAP-Response/Identity contains the
   fast re-authentication identity (from AT_ENCR_DATA attribute above):

```
   02                      ; Code: Response
   00                      ; Identifier: 0
   00 56                   ; Length: 86 octets
   01                      ; Type: Identity
      59 32 34 66 4e 53 72 7a 38 42 50 32 37 34 6a 4f
      4a 61 46 31 37 57 66 78 49 38 59 4f 37 51 58 30
      30 70 4d 58 6b 39 58 4d 4d 56 4f 77 37 62 72 6f
      61 4e 68 54 63 7a 75 46 71 35 33 61 45 70 4f 6b
      6b 33 4c 30 64 6d 40 65 61 70 73 69 6d 2e 66 6f
      6f
```

A.9. EAP-Request/SIM/Re-authentication

   The server recognizes the reauthentication identity, so it will
   respond with EAP-Request/SIM/Re-authentication. It retrieves the
   associated counter value, generates a nonce, and picks a new
   reauthentication identity (in this case,
   "uta0M0iyIsMwWp5TTdSdnOLvg2XDVf21OYt1vnfiMcs5dnIDHOIFVavIRzMR
   yzW6vFzdHW@eapsim.foo").

   The following plaintext will be encrypted and stored in the
   AT_ENCR_DATA attribute. Note that AT_PADDING is not used because the
   length of the plaintext is a multiple of 16 bytes.

```
        13                   ; Attribute type: AT_COUNTER
           01                ; Attribute length: 4 octets (1*4)
           00 01             ; Counter value
        15                   ; Attribute type: AT_NONCE_S
           05                ; Attribute length: 20 octets (5*4)
           00 00             ; (reserved)
           01 23 45 67       ; NONCE_S value
           89 ab cd ef
           fe dc ba 98
           76 54 32 10
        85                   ; Attribute type: AT_NEXT_REAUTH_ID
           16                ; Attribute length: 88 octets (22*4)
           00 51             ; Actual re-auth identity length: 81 octets
           75 74 61 30 4d 30 69 79 49 73 4d 77 57 70 35 54
           54 64 53 64 6e 4f 4c 76 67 32 58 44 56 66 32 31
           4f 59 74 31 76 6e 66 69 4d 63 73 35 64 6e 49 44
           48 4f 49 46 56 61 76 49 52 7a 4d 52 79 7a 57 36
           76 46 7a 64 48 57 40 65 61 70 73 69 6d 2e 66 6f
           6f
           00 00 00          ; (attribute padding)
```

The EAP packet looks like this:

```
01                      ; Code: Request
01                      ; Identifier: 1
00 a4                   ; Length: 164 octets
12                      ; Type: EAP-SIM
   0d                   ; EAP-SIM subtype: Re-authentication
   00 00                ; (reserved)
   81                   ; Attribute type: AT_IV
      05                ; Attribute length: 20 octets (5*4)
      00 00             ; (reserved)
      d5 85 ac 77       ; IV value
      86 b9 03 36
      65 7c 77 b4
      65 75 b9 c4
   82                   ; Attribute type: AT_ENCR_DATA
      1d                ; Attribute length: 116 octets (29*4)
      00 00             ; (reserved)
      68 62 91 a9 d2 ab c5 8c aa 32 94 b6 e8 5b 44 84
      6c 44 e5 dc b2 de 8b 9e 80 d6 9d 49 85 8a 5d b8
      4c dc 1c 9b c9 5c 01 b9 6b 6e ca 31 34 74 ae a6
      d3 14 16 e1 9d aa 9d f7 0f 05 00 88 41 ca 80 14
      96 4d 3b 30 a4 9b cf 43 e4 d3 f1 8e 86 29 5a 4a
      2b 38 d9 6c 97 05 c2 bb b0 5c 4a ac e9 7d 5e af
      f5 64 04 6c 8b d3 0b c3 9b e5 e1 7a ce 2b 10 a6
   0b                   ; Attribute type: AT_MAC
      05                ; Attribute length: 20 octets (5*4)
      00 00             ; (reserved)
      48 3a 17 99       ; MAC value
      b8 3d 7c d3
      d0 a1 e4 01
      d9 ee 47 70
```

The MAC is calculated over the EAP packet above (with MAC value set
to zero; a total of 164 bytes).

Finally, the server derives new keys. The XKEY' is calculated as
described in Section 7*:

XKEY' = 863dc120 32e08343 c1a2308d b48377f6 801f58d4

   The new MSK and EMSK are derived using the PRNG (note that K_encr and
   K_aut stay the same).

```
        MSK   = 6263f614 973895e1 335f7e30 cff028ee
                2176f519 002c9abe 732fe0ef 00cf167c
                756d9e4c ed6d5ed6 40eb3fe3 8565ca07
                6e7fb8a8 17cfe8d9 adbce441 d47c4f5e
        EMSK  = 3d8ff786 3a630b2b 06e2cf20 9684c13f
                6b82f992 f2b06f1b 54bf51ef 237f2a40
                1ef5e0d7 e098a34c 533eaebf 34578854
                b7721526 20a777f0 e0340884 a294fb73
```

A.10. EAP-Response/SIM/Re-authentication

   The client's response includes the counter as well. The following
   plaintext will be encrypted and stored in the AT_ENCR_DATA attribute:

```
        13                   ; Attribute type: AT_COUNTER
           01                ; Attribute length: 4 octets (1*4)
           00 01             ; Counter value
        06                   ; Attribute type: AT_PADDING
           03                ; Attribute length: 12 octets (3*4)
           00 00 00 00
           00 00 00 00
           00 00
```

   The EAP packet looks like this:

```
     02                      ; Code: Response
     01                      ; Identifier: 1
     00 44                   ; Length: 68 octets
     12                      ; Type: EAP-SIM
        0d                   ; EAP-SIM subtype: Re-authentication
        00 00                ; (reserved)
        81                   ; Attribute type: AT_IV
           05                ; Attribute length: 20 octets (5*4)
           00 00             ; (reserved)
           cd f7 ff a6       ; IV value
           5d e0 4c 02
           6b 56 c8 6b
           76 b1 02 ea
        82                   ; Attribute type: AT_ENCR_DATA
           05                ; Attribute length: 20 octets (5*4)
           00 00             ; (reserved)
           b6 ed d3 82
           79 e2 a1 42
           3c 1a fc 5c
           45 5c 7d 56
```

```
      0b                      ; Attribute type: AT_MAC
         05                   ; Attribute length: 20 octets (5*4)
         00 00                ; (reserved)
         fa f7 6b 71          ; MAC value
         fb e2 d2 55
         b9 6a 35 66
         c9 15 c6 17
```

   The MAC is calculated over the EAP packet above (with MAC value set
   to zero), followed by the NONCE_S value (a total of 84 bytes).

   The next packet will be EAP-Success:

```
   03                         ; Code: Success
   01                         ; Identifier: 1
   00 04                      ; Length: 4 octets
```

Appendix B. Pseudo-Random Number Generator

   The "|" character denotes concatenation, and "^" denotes
   exponentiation.

   Step 1: Choose a new, secret value for the seed-key, XKEY

   Step 2: In hexadecimal notation let
       t = 67452301 EFCDAB89 98BADCFE 10325476 C3D2E1F0
       This is the initial value for H0|H1|H2|H3|H4
       in the FIPS SHS [SHA-1]

   Step 3: For j = 0 to m - 1 do
         3.1 XSEED_j = 0 /* no optional user input */
         3.2 For i = 0 to 1 do
             a. XVAL = (XKEY + XSEED_j) mod 2^b
             b. w_i = G(t, XVAL)
             c. XKEY = (1 + XKEY + w_i) mod 2^b
         3.3 x_j = w_0|w_1

```

Authors' Addresses

   Henry Haverinen (editor)
   Nokia Enterprise Solutions
   P.O. Box 12
   FIN-40101 Jyvaskyla
   Finland

   EMail: henry.haverinen@nokia.com


   Joseph Salowey (editor)
   Cisco Systems
   2901 Third Avenue
   Seattle, WA  98121
   USA

   Phone: +1 206 256 3380
   EMail: jsalowey@cisco.com