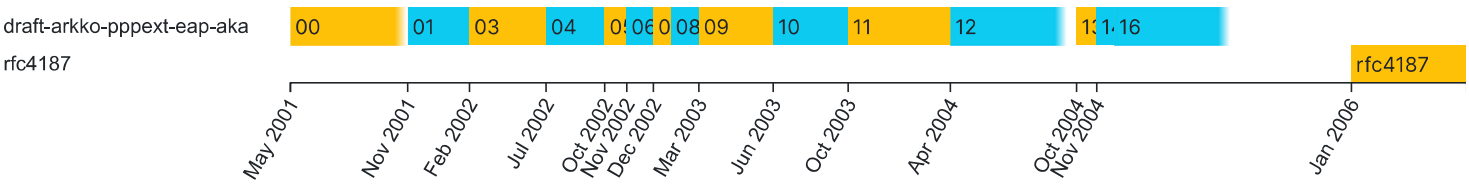
Datatracker

Sign in

Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA)

RFC 4187

Status | Email expansions | History



Document Type		RFC - Informational (January 2006) Errata Updated by RFC 9048, RFC 5448 Was draft-arkko-pppext-eap-aka (individual in int area)
	Authors	Jari Arkko ✉, Henry Haverinen ✉
	Last updated	2020-01-21
	RFC stream	Internet Engineering Task Force (IETF)
	Formats	txt html pdf htmlized w/errata bibtex
IESG	**Responsible AD**	Dr. Thomas Narten ✉
	Send notices to	(None)

Email authors | IPR | ← References | → Referenced by | Search Lists

RFC 4187

```
Network Working Group                                          J. Arkko
Request for Comments: 4187                                     Ericsson
Category: Informational                                    H. Haverinen
                                                                  Nokia
                                                            January 2006


      Extensible Authentication Protocol Method for 3rd Generation
               Authentication and Key Agreement (EAP-AKA)

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
```

Datatracker

Sign in

The EAP-AKA protocol was developed by 3GPP. The documentation of EAP-AKA is provided as information to the Internet community. While the EAP WG has verified that EAP-AKA 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 underlying UMTS AKA algorithms.

Abstract

This document specifies an Extensible Authentication Protocol (EAP) mechanism for authentication and session key distribution that uses the Authentication and Key Agreement (AKA) mechanism. AKA is used in the 3rd generation mobile networks Universal Mobile Telecommunications System (UMTS) and CDMA2000. AKA is based on symmetric keys, and typically runs in a Subscriber Identity Module, which is a UMTS Subscriber Identity Module, USIM, or a (Removable) User Identity Module, (R)UIM, similar to a smart card.

EAP-AKA includes optional identity privacy support, optional result indications, and an optional fast re-authentication procedure.

Table of Contents

1. Introduction and Motivation 4
2. Terms and Conventions Used in This Document 5
3. Protocol Overview 9
4. Operation 15
4.1. Identity Management 15
4.1.1. Format, Generation, and Usage of Peer Identities ...15
4.1.2. Communicating the Peer Identity to the Server 21
4.1.3. Choice of Identity for the EAP-Response/Identity ...23
4.1.4. Server Operation in the Beginning of EAP-AKA Exchange 23
4.1.5. Processing of EAP-Request/AKA-Identity by the Peer 24
4.1.6. Attacks against Identity Privacy 25
4.1.7. Processing of AT_IDENTITY by the Server 26
4.2. Message Sequence Examples (Informative) 27
4.2.1. Usage of AT_ANY_ID_REQ 27
4.2.2. Fall Back on Full Authentication 28
4.2.3. Requesting the Permanent Identity 1 29
4.2.4. Requesting the Permanent Identity 2 30
4.2.5. Three EAP/AKA-Identity Round Trips 30
5. Fast Re-Authentication 32
5.1. General 32
5.2. Comparison to AKA 33
5.3. Fast Re-Authentication Identity 33
5.4. Fast Re-Authentication Procedure 35
5.5. Fast Re-Authentication Procedure when Counter is Too Small 37
6. EAP-AKA Notifications 38
6.1. General 38

Samsung Ex. 1017, Page 2 of 66

Datatracker Sign in

6.3.2. Server Operation 41
6.3.3. EAP-Failure 42
6.3.4. EAP-Success 42
7. Key Generation 43
8. Message Format and Protocol Extensibility 45
8.1. Message Format 45
8.2. Protocol Extensibility 47
9. Messages 48
9.1. EAP-Request/AKA-Identity 48
9.2. EAP-Response/AKA-Identity 48
9.3. EAP-Request/AKA-Challenge 49
9.4. EAP-Response/AKA-Challenge 49
9.5. EAP-Response/AKA-Authentication-Reject 50
9.6. EAP-Response/AKA-Synchronization-Failure 50

9.7. EAP-Request/AKA-Reauthentication 50
9.8. EAP-Response/AKA-Reauthentication 51
9.9. EAP-Response/AKA-Client-Error 52
9.10. EAP-Request/AKA-Notification 52
9.11. EAP-Response/AKA-Notification 52
10. Attributes 53
10.1. Table of Attributes 53
10.2. AT_PERMANENT_ID_REQ 54
10.3. AT_ANY_ID_REQ 54
10.4. AT_FULLAUTH_ID_REQ 54
10.5. AT_IDENTITY 55
10.6. AT_RAND 55
10.7. AT_AUTN 56
10.8. AT_RES 56
10.9. AT_AUTS 57
10.10. AT_NEXT_PSEUDONYM 57
10.11. AT_NEXT_REAUTH_ID 58
10.12. AT_IV, AT_ENCR_DATA, and AT_PADDING 58
10.13. AT_CHECKCODE 60
10.14. AT_RESULT_IND 62
10.15. AT_MAC 63
10.16. AT_COUNTER 64
10.17. AT_COUNTER_TOO_SMALL 64
10.18. AT_NONCE_S 65
10.19. AT_NOTIFICATION 65
10.20. AT_CLIENT_ERROR_CODE 66
11. IANA and Protocol Numbering Considerations 66
12. Security Considerations 68
12.1. Identity Protection 69
12.2. Mutual Authentication 69
12.3. Flooding the Authentication Centre 69
12.4. Key Derivation 70
12.5. Brute-Force and Dictionary Attacks 70
12.6. Protection, Replay Protection, and Confidentiality 70
12.7. Negotiation Attacks 71
12.8. Protected Result Indications 72
12.9. Man-in-the-Middle Attacks 72
12.10. Generating Random Numbers 73

Samsung Ex. 1017, Page 3 of 66

Datatracker Sign in

15.1. Normative References 74
15.2. Informative References 76
Appendix A. Pseudo-Random Number Generator 77

1. Introduction and Motivation

This document specifies an Extensible Authentication Protocol (EAP) mechanism for authentication and session key distribution that uses the 3rd generation Authentication and Key Agreement mechanism, specified for Universal Mobile Telecommunications System (UMTS) in [TS33.102] and for CDMA2000 in [S.S0055-A]. UMTS and CDMA2000 are global 3rd generation mobile network standards that use the same AKA mechanism.

2nd generation mobile networks and 3rd generation mobile networks use different authentication and key agreement mechanisms. The Global System for Mobile communications (GSM) is a 2nd generation mobile network standard, and EAP-SIM [EAP-SIM] specifies an EAP mechanism that is based on the GSM authentication and key agreement primitives.

AKA is based on challenge-response mechanisms and symmetric cryptography. AKA typically runs in a UMTS Subscriber Identity Module (USIM) or a CDMA2000 (Removable) User Identity Module ((R)UIM). In this document, both modules are referred to as identity modules. Compared to the 2nd generation mechanisms such as GSM AKA, the 3rd generation AKA provides substantially longer key lengths and mutual authentication.

The introduction of AKA inside EAP allows several new applications. These include the following:

- The use of the AKA also as a secure PPP authentication method in devices that already contain an identity module.
- The use of the 3rd generation mobile network authentication infrastructure in the context of wireless LANs
- Relying on AKA and the existing infrastructure in a seamless way with any other technology that can use EAP.

AKA works in the following manner:

- The identity module and the home environment have agreed on a secret key beforehand. (The "home environment" refers to the home operator's authentication network infrastructure.)
- The actual authentication process starts by having the home environment produce an authentication vector, based on the secret key and a sequence number. The authentication vector contains a random part RAND, an authenticator part AUTN used for authenticating the network to the identity module, an expected result part XRES, a 128-bit session key for integrity check IK, and a 128-bit session key for encryption CK.

Samsung Ex. 1017, Page 4 of 66

Datatracker Sign in

key and the sequence number. If this process is successful (the AUTN is valid and the sequence number used to generate AUTN is within the correct range), the identity module produces an authentication result RES and sends it to the home environment.

- The home environment verifies the correct result from the identity module. If the result is correct, IK and CK can be used to protect further communications between the identity module and the home environment.

When verifying AUTN, the identity module may detect that the sequence number the network uses is not within the correct range. In this case, the identity module calculates a sequence number synchronization parameter AUTS and sends it to the network. AKA authentication may then be retried with a new authentication vector generated using the synchronized sequence number.

For a specification of the AKA mechanisms and how the cryptographic values AUTN, RES, IK, CK and AUTS are calculated, see [TS33.102] for UMTS and [S.S0055-A] for CDMA2000.

In EAP-AKA, the EAP server node obtains the authentication vectors, compares RES and XRES, and uses CK and IK in key derivation.

In the 3rd generation mobile networks, AKA is used for both radio network authentication and IP multimedia service authentication purposes. Different user identities and formats are used for these; the radio network uses the International Mobile Subscriber Identifier (IMSI), whereas the IP multimedia service uses the Network Access Identifier (NAI) [RFC4282].

2. Terms and Conventions Used in This Document

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. The AKA parameters are specified in detail in [TS33.102] for UMTS and [S.S0055-A] for CDMA2000.

AAA protocol

Authentication, Authorization and Accounting protocol

AKA

Authentication and Key Agreement

Samsung Ex. 1017, Page 5 of 66

Datatracker Sign in

authenticate subscribers in the mobile networks.

AUTN

AKA parameter. AUTN is an authentication value generated by the AuC, which, together with the RAND, authenticates the server to the peer, 128 bits.

AUTS

AKA parameter. A value generated by the peer upon experiencing a synchronization failure, 112 bits.

EAP

Extensible Authentication Protocol [RFC3748]

Fast Re-Authentication

An EAP-AKA authentication exchange that is based on keys derived upon a preceding full authentication exchange. The 3rd Generation AKA is 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 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-AKA authentication exchange that is based on the 3rd Generation AKA procedure.

GSM

Global System for Mobile communications.

NAI

Network Access Identifier [RFC4282]

Identity Module

Identity module is used in this document to refer to the part of the mobile device that contains authentication and key agreement primitives. The identity module may be an

Datatracker Sign in

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). Because 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.

RAND

An AKA parameter. Random number generated by the AuC, 128 bits.

RES

Authentication result from the peer, which, together with the RAND, authenticates the peer to the server, 128 bits.

(R)UIM

CDMA2000 (Removable) User Identity Module. (R)UIM is an application that is resident on devices such as smart cards, which may be fixed in the terminal or distributed by CDMA2000 operators (when removable).

Datatracker Sign in

process, 48 bits.

SIM

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

SRES

The authentication result parameter in GSM, corresponds to the RES parameter in 3G AKA, 32 bits.

UAK

UIM Authentication Key, used in CDMA2000 AKA. Both the identity module and the network can optionally generate the UAK during the AKA computation in CDMA2000. UAK is not used in this version of EAP-AKA.

UIM

Please see (R)UIM.

USIM

UMTS Subscriber Identity Module. USIM is an application that is resident on devices such as smart cards distributed by UMTS operators.

3. Protocol Overview

Figure 1 shows the basic, successful full authentication exchange in EAP-AKA, when optional result 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. At the minimum, EAP-AKA uses two roundtrips to authenticate and authorize the peer and generate session keys. As in other EAP schemes, an identity request/response message pair is usually exchanged first. On full authentication, the peer's identity 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 4.1. (As specified in [RFC3748], the initial identity request is not required, and MAY be bypassed in cases where the network can presume the identity, such as when using leased lines, dedicated dial-ups, etc. Please see Section 4.1.2 for specification of how to obtain the identity via EAP AKA messages.)

After obtaining the subscriber identity, the EAP server obtains an authentication vector (RAND, AUTN, RES, CK, IK) for use in authenticating the subscriber. From the vector, the EAP server

Datatracker Sign in

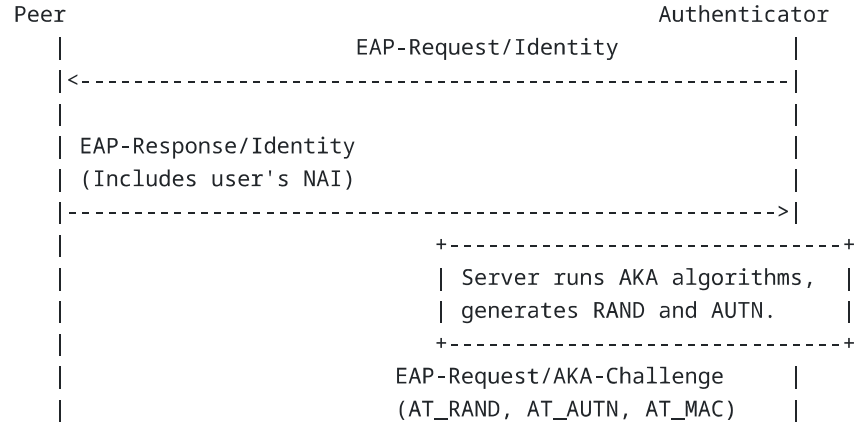
may be obtained at a time. Vectors may be stored in the EAP server for use at a later time, but they may not be reused.

In CDMA2000, the vector may include a sixth value called the User Identity Module Authentication Key (UAK). This key is not used in EAP-AKA.

Next, the EAP server starts the actual AKA protocol by sending an EAP-Request/AKA-Challenge message. EAP-AKA 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/AKA-Challenge message contains a RAND random number (AT_RAND), a network authentication token (AT_AUTN), and a message authentication code (AT_MAC). The EAP-Request/AKA-Challenge message MAY optionally contain encrypted data, which is used for identity privacy and fast re-authentication support, as described in Section 4.1. The AT_MAC attribute contains a message authentication code covering the EAP packet. The encrypted data is not shown in the figures of this section.

The peer runs the AKA algorithm (typically using an identity module) and verifies the AUTN. If this is successful, the peer is talking to a legitimate EAP server and proceeds to send the EAP-Response/AKA-Challenge. This message contains a result parameter that allows the EAP server, in turn, to authenticate the peer, and the AT_MAC attribute to integrity protect the EAP message.

The EAP server verifies that the RES and the MAC in the EAP-Response/AKA-Challenge packet are 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.



Datatracker Sign in

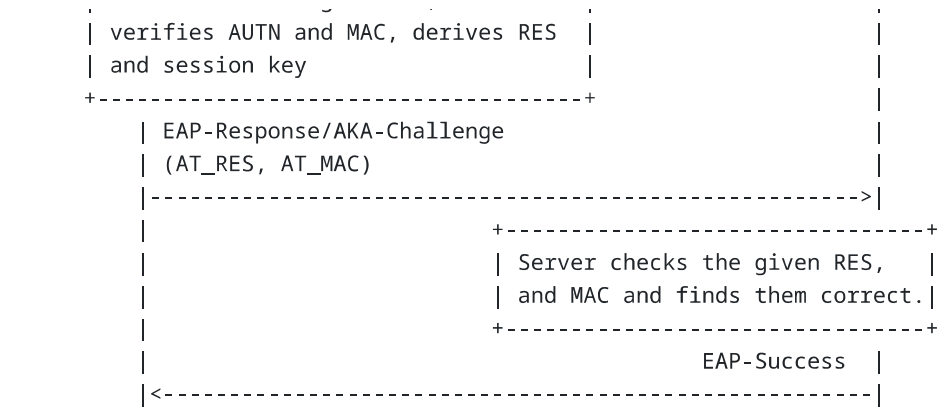


Figure 1: EAP-AKA full authentication procedure

Figure 2 shows how the EAP server rejects the Peer due to a failed authentication.

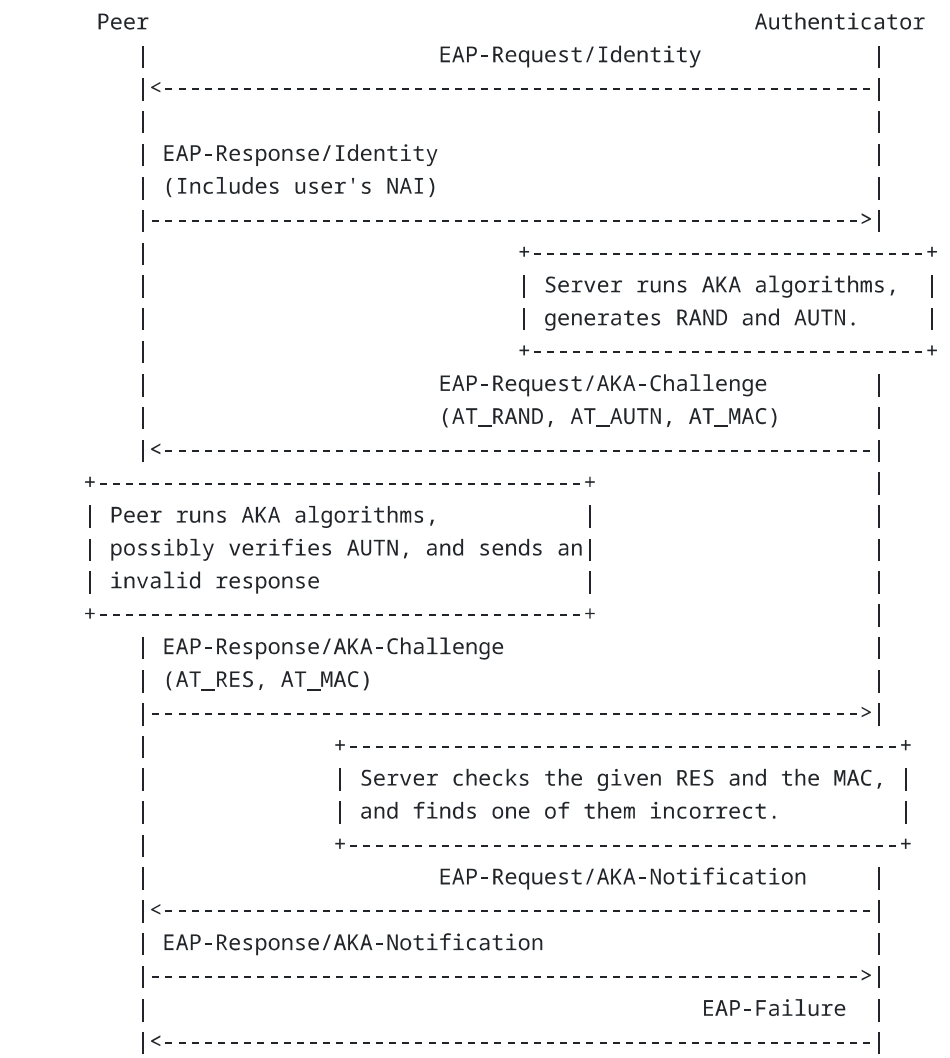


Figure 2: Peer authentication fails

Samsung Ex. 1017, Page 10 of 66

Datatracker Sign in

Figure 3 shows the peer rejecting the AUTN of the EAP server.

The peer sends an explicit error message (EAP-Response/ AKA-Authentication-Reject) to the EAP server, as usual in AKA when AUTN is incorrect. This allows the EAP server to produce the same error statistics that AKA generally produces in UMTS or CDMA2000.

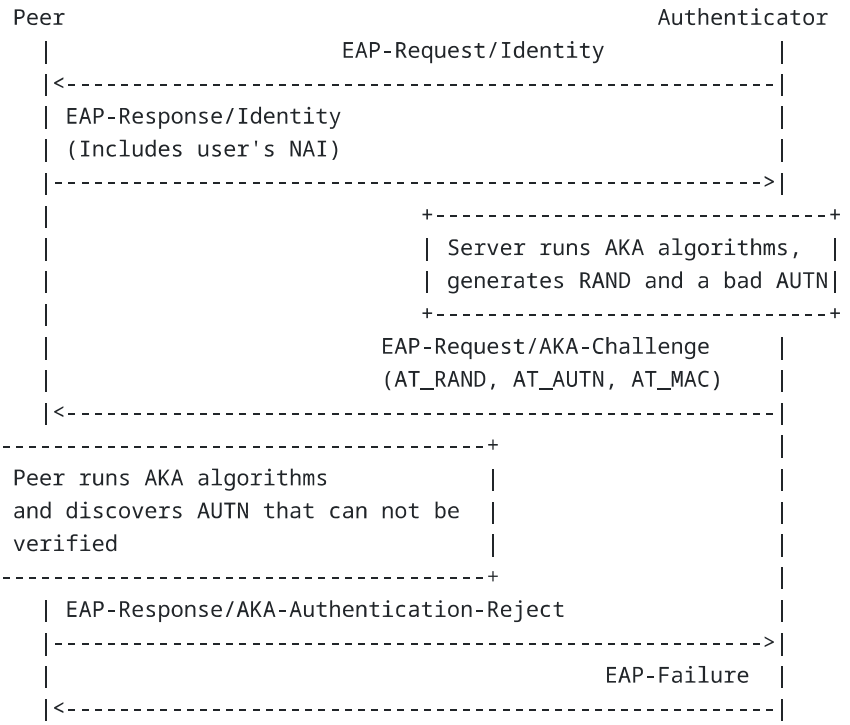
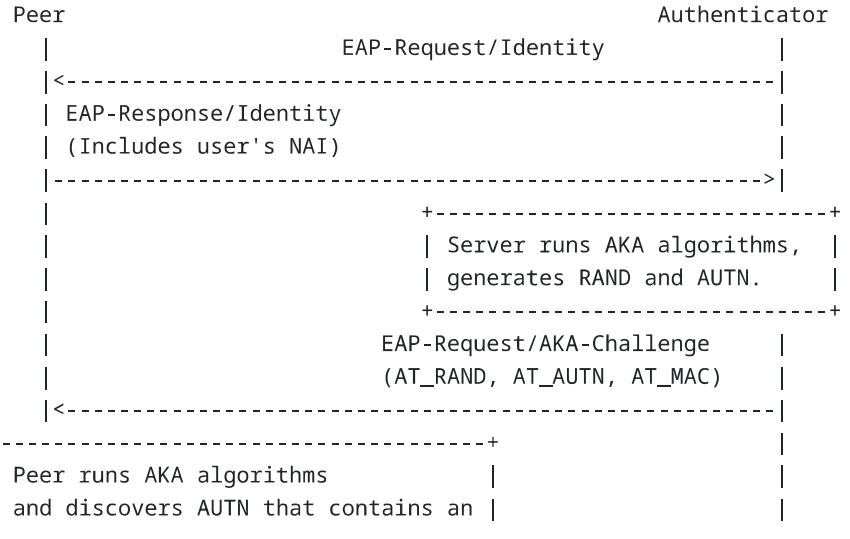


Figure 3: Network authentication fails

The AKA uses shared secrets between the Peer and the Peer's home operator, together with a sequence number, to actually perform an authentication. In certain circumstances, shown in Figure 4, it is possible for the sequence numbers to get out of sequence.



Datatracker Sign in

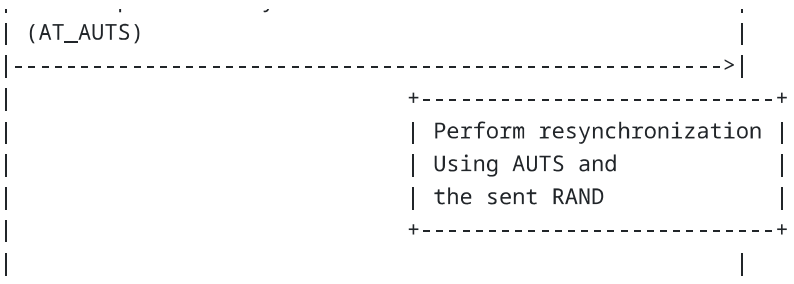


Figure 4: Sequence number synchronization

After the resynchronization process has taken place in the server and AAA side, the process continues by the server side sending a new EAP-Request/AKA-Challenge message.

In addition to the full authentication scenarios described above, EAP-AKA includes a fast re-authentication procedure, which is specified in Section 5. Fast re-authentication is based on keys derived on full authentication. If the peer has maintained state information for 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.

4. Operation

4.1. Identity Management

4.1.1. Format, Generation, and Usage of Peer Identities

4.1.1.1. 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 EAP-Response/Identity, which contains the user's identity. The formats of these packets are specified in [RFC3748].

Subscribers of mobile networks are identified with the International Mobile Subscriber Identity (IMSI) [TS23.003]. The IMSI is a string of not more than 15 digits. It is composed of a Mobile Country Code (MCC) of 3 digits, a Mobile Network Code (MNC) of 2 or 3 digits, and a Mobile Subscriber Identification Number (MSIN) of not 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.

Samsung Ex. 1017, Page 12 of 66

Datatracker Sign in

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.

4.1.1.2. Identity Privacy Support

EAP-AKA 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 12.1 for security considerations regarding identity privacy.

4.1.1.3. Username Types in EAP-AKA Identities

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

(1) Permanent usernames. For example, 0123456789098765@myoperator.com might be a valid permanent identity. In this example, 0123456789098765 is the permanent username.

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

(3) Fast re-authentication usernames. For example, 43953754@myoperator.com might be a valid fast re-authentication identity. In this case, 43953754 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 type 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.

4.1.1.4. 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 to be 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,

Datatracker Sign in

or the EAP server MUST NOT decorate any other peer identities that are used in various EAP-AKA attributes. Only the identity used in EAP-Response/Identity may be decorated.

4.1.1.5. 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-AKA 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-AKA users. This convention makes it easy to recognize that the NAI identifies an AKA subscriber. Such a reserved NAI realm may be useful as a hint of 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 to how it selects 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, will be specified in [TS23.003].

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 [TS23.003] 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 MSIN. Hence, when configuring AAA networks for operators that have 2-digit MNC's, the network SHOULD also be prepared for realm names with incorrect 3-digit MNC's.

4.1.1.6. 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 "0" | IMSI, where the character "|" denotes concatenation. In other words, the first character of the username is the digit zero (ASCII value 30 hexadecimal), followed by the IMSI. The IMSI is an ASCII string that consists of not more than 15 decimal digits (ASCII values between 30

Datatracker Sign in

and 39 hexadecimal), one character per IMSI digit, in the order as specified in [TS23.003]. For example, a permanent username derived from the IMSI 295023820005424 would be encoded as the ASCII string "0295023820005424" (byte values in hexadecimal notation: 30 32 39 35 30 32 33 38 32 30 30 30 35 34 32 34)

The EAP server MAY use the leading "0" as a hint to try EAP-AKA as the first authentication method during method negotiation, rather than using, for example, EAP-SIM. The EAP-AKA server MAY propose EAP-AKA even if the leading character was not "0".

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.

4.1.1.7. 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-AKA 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 peer to send the permanent identity.

When issuing a fast re-authentication identity, the EAP server may include a realm name in the identity that will cause the fast re-authentication request to 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. The 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 the peer's fast re-authentication state information (in other words,

Samsung Ex. 1017, Page 15 of 66

Datatracker Sign in

initiate a fast re-authentication exchange to get the server's EAP-Request/AKA-Reauthentication packet.

Transmitting pseudonyms and fast re-authentication identities from the server to the peer is discussed in Section 4.1.1.8. 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 in order to allow the server to include a server-specific realm.

Regardless of construction method, the pseudonym username MUST conform to the grammar specified for the username portion of an NAI. Also, the fast re-authentication identity 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 that 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 usernames be recognizable from each other as an aid to the server when deciding 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 recognize if an NAI username is an EAP-AKA pseudonym username or an EAP-AKA 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 "2" 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 that was sent in an EAP-Request/AKA-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.

4.1.1.8. Transmitting Pseudonyms and Fast Re-Authentication Identities to the Peer

The server transmits pseudonym usernames and fast re-authentication

Samsung Ex. 1017, Page 16 of 66

Datatracker Sign in

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 to implement, 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/AKA-Reauthentication message.

On receipt of the EAP-Request/AKA-Challenge, the peer MAY decrypt the encrypted data in AT_ENCR_DATA; and if a pseudonym username is included, 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 peer does not receive a new pseudonym username in the EAP-Request/AKA-Challenge message, the peer MAY use an old pseudonym username instead of the permanent username on 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.

4.1.1.9. 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.

4.1.1.10. Usage of the Fast Re-Authentication Identity by the Peer

On fast re-authentication, the peer uses the fast re-authentication identity received as part of the previous authentication sequence. A new fast 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

Datatracker Sign in

appended or prepended with another string.

4.1.2. Communicating the Peer Identity to the Server

4.1.2.1. 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, 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-AKA messages instead of 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/AKA-Identity message; and the peer includes the AT_IDENTITY attribute, which contains the peer's identity, in the EAP-Response/AKA-Identity 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 EAP-Request/AKA-Challenge, EAP-Response/AKA-Challenge, or the packets used on fast re-authentication may optionally include the AT_CHECKCODE attribute, which enables the protocol peers to ensure the integrity of the AKA-Identity packets. AT_CHECKCODE is specified in Section 10.13.

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 only the EAP-AKA peer and the EAP-AKA server process the AT_IDENTITY attribute and entities that 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.

4.1.2.2. Relying on EAP-Response/Identity Discouraged

The EAP-Response/Identity packet is not method specific; therefore,

Datatracker Sign in

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-AKA, 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 the peer to send an unmodified and undecorated copy of the identity in AT_IDENTITY.

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

If EAP-AKA peer is started upon receiving an EAP-Request/Identity message, then the peer MAY use an EAP-AKA 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 if the peer 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.

4.1.4. Server Operation in the Beginning of EAP-AKA Exchange

As discussed in Section 4.1.2.2, the server SHOULD NOT rely on an identity string received in EAP-Response/Identity. Therefore, the RECOMMENDED way to start an EAP-AKA exchange is to ignore any received identity strings. The server SHOULD begin the EAP-AKA exchange by issuing the EAP-Request/AKA-Identity 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/AKA-Identity message. Three methods to request an identity from the peer are discussed below.

If the server chooses to not ignore the contents of EAP-Response/Identity, then the server may already receive an EAP-AKA identity in this packet. However, if the EAP server has not received any EAP-AKA peer identity (permanent identity, pseudonym identity, or fast re-authentication identity) from the peer when sending the first EAP-AKA 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

Samsung Ex. 1017, Page 19 of 66

Datatracker Sign in

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/AKA-Identity message. This is done in the following cases:

- The server does not support fast re-authentication or identity privacy.
- 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/AKA-Identity 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/AKA-Identity message. This is done in the following cases:

- The server does not support fast re-authentication and the server supports identity privacy
- 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/AKA-Identity 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/AKA-Identity 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, when the server does not have any identity, or when the server does not recognize the format of a received identity.

4.1.5. Processing of EAP-Request/AKA-Identity by the Peer

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

If the EAP-Request/AKA-Identity includes AT_PERMANENT_ID_REQ, and if the peer does not have a pseudonym available, then the peer MUST respond with EAP-Response/AKA-Identity 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/AKA-Identity and include the permanent identity in AT_IDENTITY or respond with EAP-Response/AKA-Client-Error packet with code "unable to process packet".

If the EAP-Request/AKA-Identity includes AT_FULL_AUTH_ID_REQ, and if the peer has a pseudonym available, then the peer SHOULD respond with EAP-Response/AKA-Identity and include the pseudonym identity in

Datatracker Sign in

AT_IDENTITY. If the peer does not have a pseudonym when it receives this message, then the peer MUST respond with EAP-Response/ AKA-Identity and include the permanent identity in AT_IDENTITY. The Peer MUST NOT use a fast re-authentication identity in the AT_IDENTITY attribute.

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

An EAP-AKA exchange may include several EAP/AKA-Identity rounds. The server may issue a second EAP-Request/AKA-Identity, if it was not able to recognize the identity the peer used in the previous AT_IDENTITY attribute. At most three EAP/AKA-Identity rounds can be used, so the peer MUST NOT respond to more than three EAP-Request/AKA-Identity messages within an EAP exchange. The peer MUST verify that the sequence of EAP-Request/AKA-Identity packets 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/AKA-Identity; in other words, AT_ANY_ID_REQ MUST NOT be used in the second or third EAP-Request/AKA-Identity. AT_FULLAUTH_ID_REQ MUST NOT be used if the previous EAP-Request/AKA-Identity 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.

4.1.6. Attacks against Identity Privacy

The section above specifies two possible ways the peer can operate upon receipt of AT_PERMANENT_ID_REQ because a received AT_PERMANENT_ID_REQ does not necessarily originate from the valid network. However, an active attacker may transmit an EAP-Request/AKA-Identity 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/AKA-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/AKA-Client-Error to the EAP packet with AT_PERMANENT_ID_REQ, because the peer believes that the valid network

Samsung Ex. 1017, Page 21 of 66

Datatracker Sign in

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.

4.1.7. Processing of AT_IDENTITY by the Server

When the server receives an EAP-Response/AKA-Identity 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 an EAP-Request/AKA-Notification packet with AT_NOTIFICATION code "General failure" (16384) to terminate the EAP exchange. If the server recognizes the permanent identity and is able to continue, then the server proceeds with full authentication by sending EAP-Request/AKA-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/AKA-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/ AKA-Identity 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/ AKA-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/AKA-Reauthentication (Section 5).

If the server used AT_ANY_ID_REQ, and if the peer sent an EAP-Response/AKA-Identity with AT_IDENTITY that contains an identity that the server recognizes as a fast re-authentication identity, but the server is not able to map the identity to a permanent identity, then the server sends EAP-Request/AKA-Identity with AT_FULLAUTH_ID_REQ.

If the server used AT_ANY_ID_REQ, and if AT_IDENTITY contains a valid fast re-authentication identity, which 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 proceeds with full authentication by sending EAP-Request/AKA-Challenge.

Samsung Ex. 1017, Page 22 of 66

Datatracker Sign in

server is not able to map the pseudonym identity to a permanent identity, then the server sends EAP-Request/AKA-Identity 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/AKA-Identity with AT_FULLAUTH_ID_REQ.

4.2. Message Sequence Examples (Informative)

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

4.2.1. Usage of AT_ANY_ID_REQ

Obtaining the peer identity with EAP-AKA attributes is illustrated in Figure 5 below.

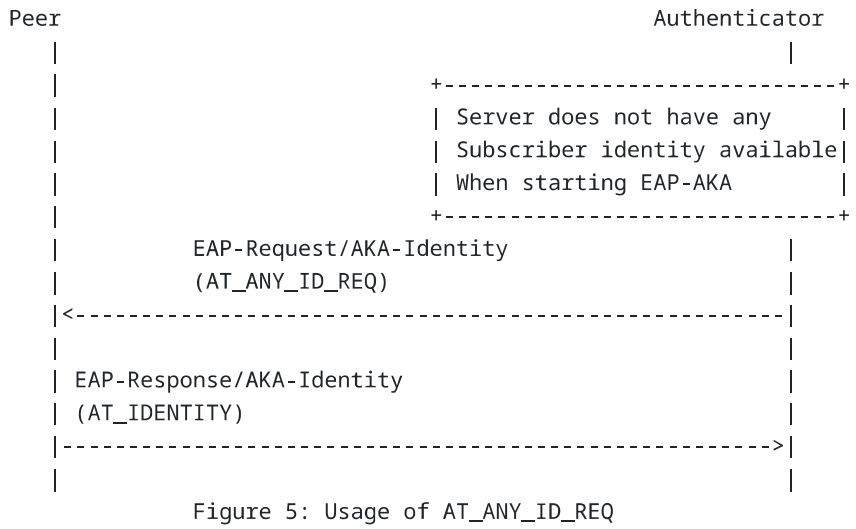
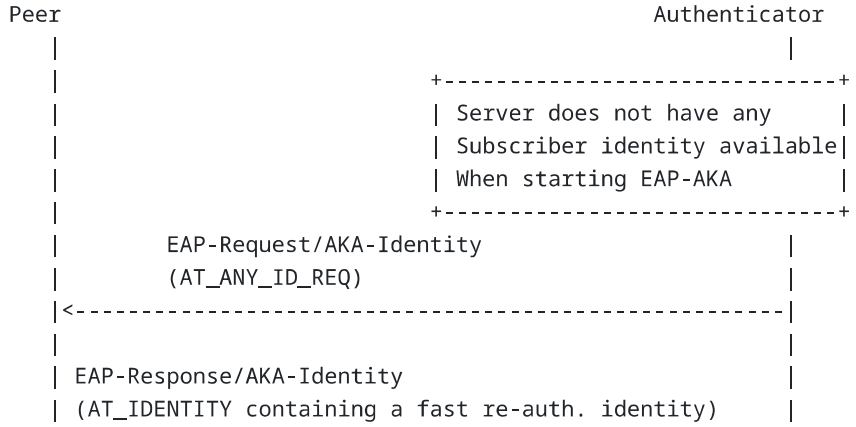


Figure 5: Usage of AT_ANY_ID_REQ

4.2.2. Fall Back on Full Authentication

Figure 6 illustrates the case when the server does not recognize the fast re-authentication identity the peer used in AT_IDENTITY.



Datatracker Sign in

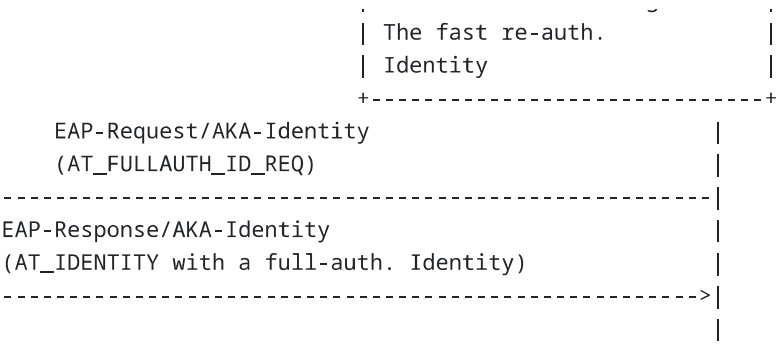


Figure 6: Fall back on full authentication

If the server recognizes the fast re-authentication identity, but still wants to fall back on full authentication, the server may issue the EAP-Request/AKA-Challenge packet. In this case, the full authentication procedure proceeds as usual.

4.2.3. Requesting the Permanent Identity 1

Figure 7 illustrates the case when the EAP server fails to decode a pseudonym identity included in the EAP-Response/Identity packet.

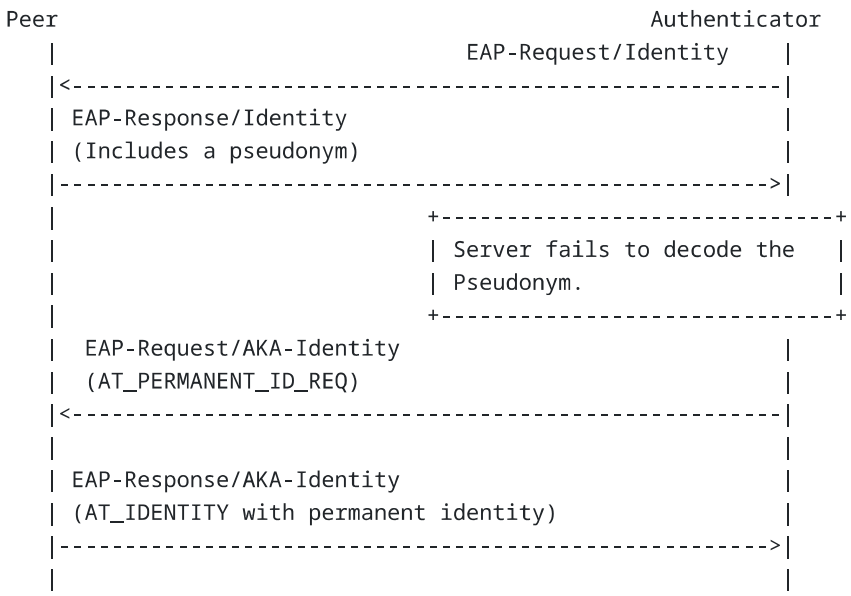


Figure 7: Requesting the permanent identity 1

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

4.2.4. Requesting the Permanent Identity 2

Samsung Ex. 1017, Page 24 of 66

Datatracker Sign in

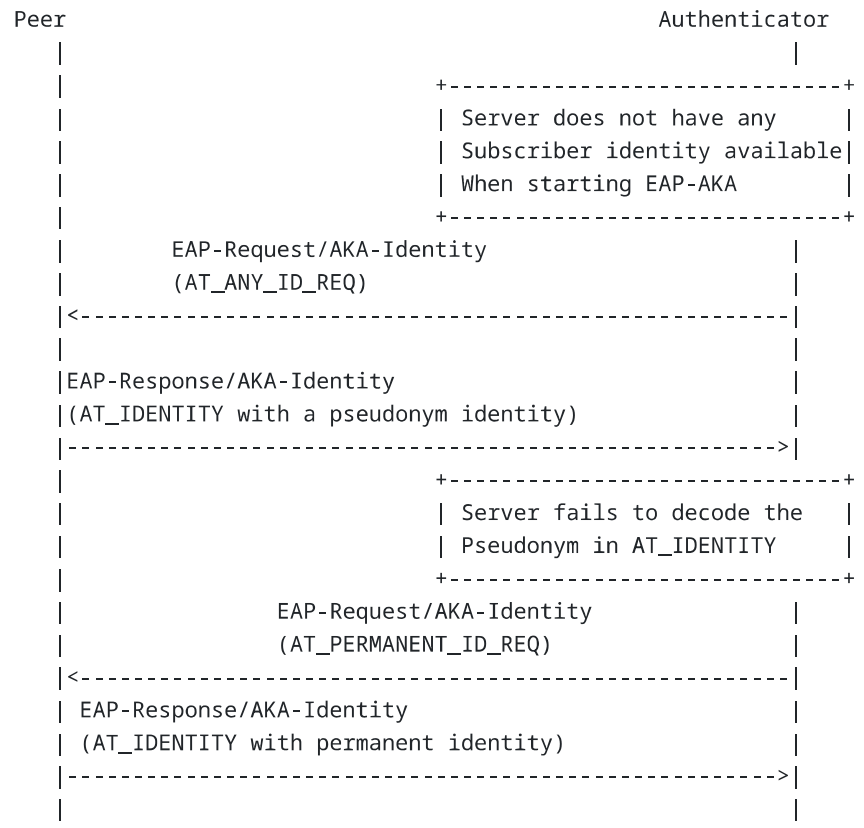
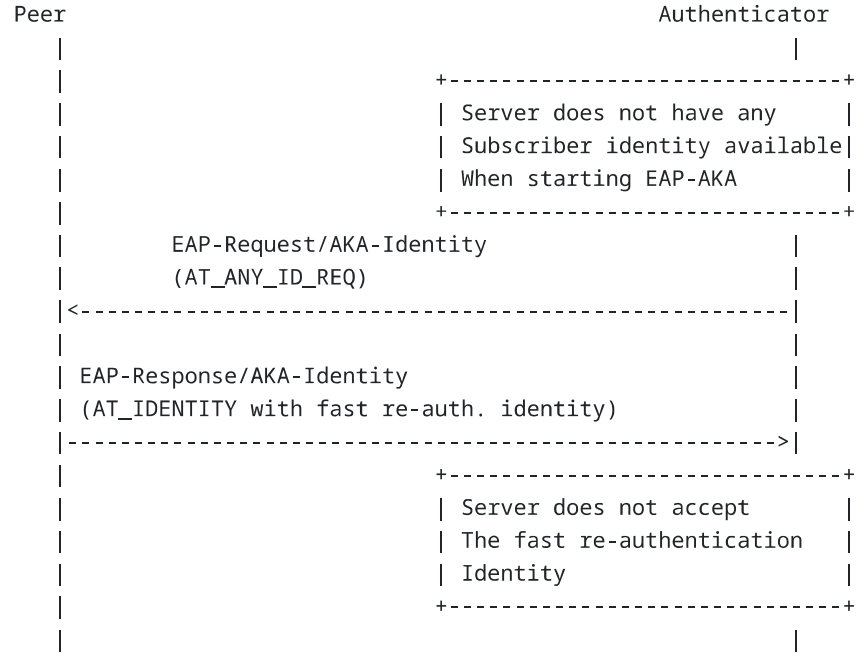


Figure 8: Requesting the permanent identity 2

4.2.5. Three EAP/AKA-Identity Round Trips

Figure 9 illustrates the case with three EAP/AKA-Identity round trips.



Datatracker Sign in

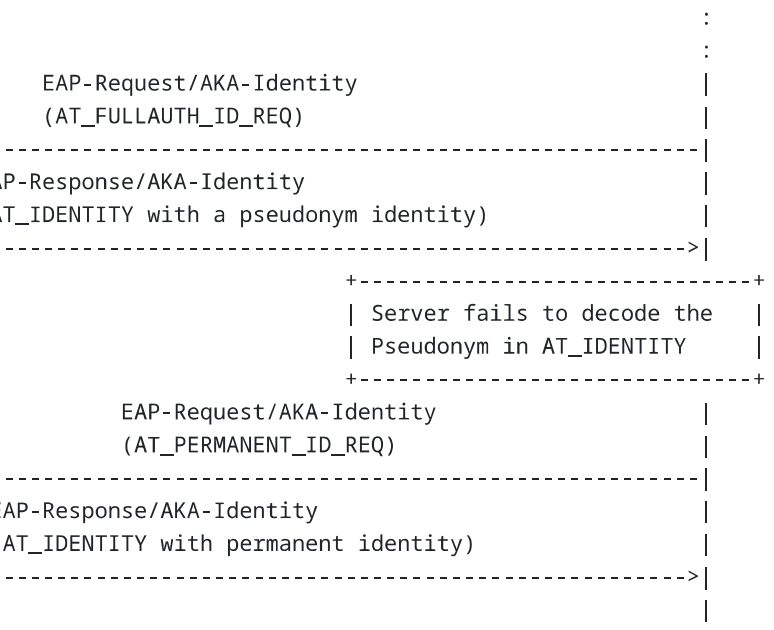


Figure 9: Three EAP-AKA Start rounds

After the last EAP-Response/AKA-Identity message, the full authentication sequence proceeds as usual.

5. Fast Re-Authentication

5.1. General

In some environments, EAP authentication may be performed frequently. Because the EAP-AKA full authentication procedure uses the AKA algorithms, and therefore requires fresh authentication vectors from the Authentication Centre, the full authentication procedure may result in many network operations when used very frequently. Therefore, EAP-AKA includes a more inexpensive fast re-authentication procedure that does not make use of the AKA algorithms and does not need new vectors from the Authentication Centre.

Fast re-authentication is optional to implement for both the EAP-AKA server and peer. On each EAP authentication, either one of the entities may fall back on full authentication if is 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 used in full authentication are used to protect EAP-AKA 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

Samsung Ex. 1017, Page 26 of 66

Datatracker Sign in

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 fast re-authentications. For example, on the second fast re-authentication, counter value is two or greater, etc. 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 and 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 AKA

When analyzing the fast re-authentication exchange, it may be helpful to compare it with the 3rd generation Authentication and Key Agreement (AKA) exchange used on full authentication. The counter corresponds to the AKA sequence number, NONCE_S corresponds to RAND, the AT_MAC in EAP-Request/AKA-Reauthentication corresponds to AUTN, the AT_MAC in EAP-Response/AKA-Reauthentication 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 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 fast re-authentication identity is lost and the network does not recognize

Datatracker Sign in

AKA-Challenge message. This attribute contains a new re-authentication identity for the next fast re-authentication. The attribute also works as a capability flag that indicates 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 will use full authentication next time. If the peer wants to use fast 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 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/AKA-Challenge or EAP-Request/AKA-Reauthentication, 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/ AKA-Identity 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. If the server was able to decode the fast re-authentication identity to the permanent identity, the server issues the EAP-Request/AKA-Challenge packet to initiate full authentication. If the server was not able to recover the peer's identity from the fast re-authentication identity, the server starts the full authentication procedure by issuing an

Datatracker

Sign in

5.4. Fast Re-Authentication Procedure

Figure 10 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 fast re-authentication identity in the EAP-Response/Identity packet. As discussed above, an alternative way to communicate the fast re-authentication identity to the server is for the peer to use the AT_IDENTITY attribute in the EAP-Response/AKA-Identity 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/AKA-Identity 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/AKA-Reauthentication 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/AKA-Reauthentication packet, including the AT_COUNTER attribute with the same counter value and the 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/AKA-Reauthentication packet. If these checks are successful, the fast 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-AKA notification round.

Samsung Ex. 1017, Page 29 of 66

Datatracker Sign in

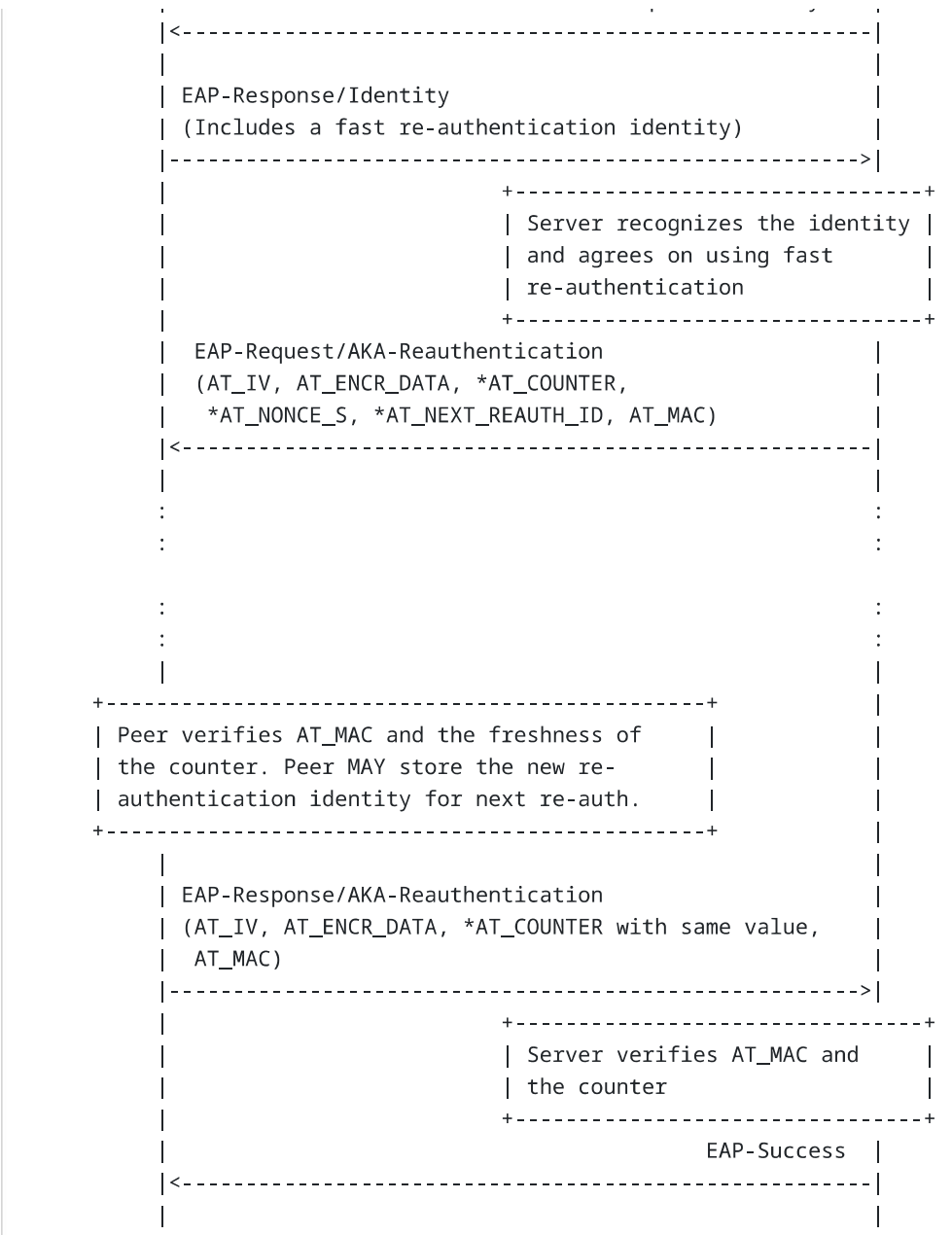


Figure 10: Reauthentication

5.5. Fast Re-Authentication Procedure when Counter is Too Small

If the peer does not accept the counter value of EAP-Request/AKA-Reauthentication, it indicates the counter synchronization problem by including the encrypted AT_COUNTER_TOO_SMALL in EAP-Response/AKA-Reauthentication. The server responds with EAP-Request/AKA-Challenge to initiate a normal full authentication procedure. This is illustrated in Figure 11. Encrypted attributes are denoted with '*'.



Samsung Ex. 1017, Page 30 of 66

Datatracker Sign in

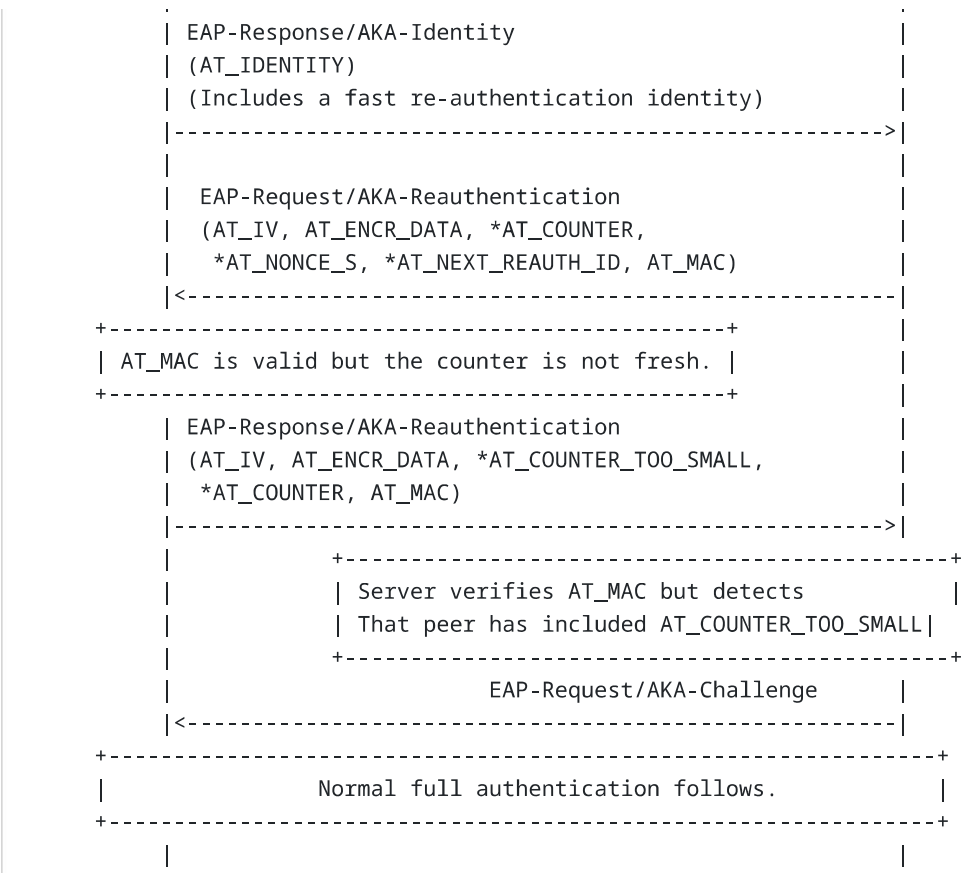


Figure 11: Fast re-authentication counter too small

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/AKA-Reauthentication. 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/AKA-Reauthentication packet. If not, the server terminates the authentication exchange by sending the EAP-Request/AKA-Notification packet with AT_NOTIFICATION code "General failure" (16384). If all checks on the packet are successful, the server transmits an EAP-Request/AKA-Challenge packet and the full authentication procedure is performed as usual. Because the server already knows the subscriber identity, it MUST NOT use the EAP-Request/AKA-Identity packet to request the identity.

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).

Datatracker Sign in

6.1. General

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

The EAP server can use EAP-AKA 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/AKA-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/AKA-Notification packet, even if the peer did not recognize the notification code.

An EAP-AKA full authentication exchange or a fast re-authentication exchange MUST NOT include more than one EAP-AKA 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 failed EAP exchange, so the peer can use the notification as a failure indication. After receiving the EAP-Response/AKA-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-AKA exchange the notification can be used. If the P bit is set to zero, the notification can only be used after a successful EAP/AKA-Challenge round in full authentication or a successful EAP/AKA-Reauthentication round in re-authentication. 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/AKA-Reauthentication.

If the P bit is set to one, the notification can only by used before the EAP/AKA-Challenge round in full authentication or before the EAP/AKA-Reauthentication 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

Datatracker Sign in

included in the notification packets.

Some of the notification codes are authorization related and hence not usually considered as part of the responsibility of an EAP method. However, they are included as part of EAP-AKA 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-AKA server implementation may decide never to send these EAP-AKA 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-AKA notification to indicate failure to the peer. In this case, the result indication is integrity and replay protected.

By sending an EAP-Response/AKA-Challenge packet or an EAP-Response/AKA-Reauthentication 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-AKA 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/AKA-Challenge or the EAP-Request/AKA-Reauthentication 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/AKA-Challenge or EAP-Response/AKA-Reauthentication. 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-AKA notification round will follow. The following EAP-AKA 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/AKA-Challenge or EAP-Request/AKA-Reauthentication 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-AKA response to this notification. Hence, the server MUST ignore the contents of the EAP-AKA response it receives to the EAP-Request/AKA-Notification with this code. Regardless of the contents of the EAP-AKA response,

Datatracker

Sign in

This section specifies the operation of the peer and the server in error cases. The subsections below require the EAP-AKA peer and server to send an error packet (EAP-Response/AKA-Client-Error, EAP-Response/AKA-Authentication-Reject or EAP-Response/AKA-Synchronization-Failure from the peer and EAP-Request/AKA-Notification from the server) in error cases. However, implementations SHOULD NOT rely upon the correct error reporting behavior of the peer, authenticator, or server. It is possible for error messages 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

Two special error messages have been specified for error cases that are related to the processing of the AKA AUTN parameter, as described in Section 3: (1) if the peer does not accept AUTN, the peer responds with EAP-Response/AKA-Authentication-Reject (Section 9.5), and the server issues EAP-Failure, and (2) if the peer detects that the sequence number in AUTN is not correct, the peer responds with EAP-Response/AKA-Synchronization-Failure (Section 9.6), and the server proceeds with a new EAP-Request/AKA-Challenge.

In other error cases, when an EAP-AKA peer detects an error in a received EAP-AKA packet, the EAP-AKA peer responds with the EAP-Response/AKA-Client-Error packet. In response to the EAP-Response/AKA-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:

- EAP exchange is not acceptable according to the peer's local policy.
- The peer is not able to parse the EAP request, i.e., the EAP request is malformed.
- The peer encountered a malformed attribute.
- Wrong attribute types or duplicate attributes have been included in the EAP request.
- A mandatory attribute is missing.
- Unrecognized non-skippable attribute.
- Unrecognized or unexpected EAP-AKA Subtype in the EAP request.
- Invalid AT_MAC. The peer SHOULD log this event.
- Invalid AT_CHECKCODE. The peer SHOULD log this event.
- Invalid pad bytes in AT_PADDING.
- The peer does not want to process AT_PERMANENT_ID_REQ.

6.3.2. Server Operation

Datatracker Sign in

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-AKA exchange, see Section 6. The error cases when the server issues an EAP-Request/AKA-Notification that implies failure include the following:

- The server is not able to parse the peer's EAP response.
- The server encounters a malformed attribute, a non-recognized non-skippable attribute, or a duplicate attribute.
- A mandatory attribute is missing or an invalid attribute was included.
- Unrecognized or unexpected EAP-AKA Subtype in the EAP Response.
- Invalid AT_MAC. The server SHOULD log this event.
- Invalid AT_CHECKCODE. The server SHOULD log this event.
- Invalid AT_COUNTER.

6.3.3. EAP-Failure

The EAP-AKA server sends EAP-Failure in three cases:

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

2. In response to an EAP-Response/AKA-Authentication-Reject packet the server has received from the peer, or

3. Following an EAP-AKA notification round, when the AT_NOTIFICATION code implies failure.

The EAP-AKA server MUST NOT send EAP-Failure in other cases than these three. However, it should be noted that even though the EAP-AKA server would not send an EAP-Failure, an authorization decision that happens outside EAP-AKA, 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), case 2), and case 3) 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/AKA-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/AKA-Challenge packet.

If the peer did not indicate that it wants to use protected success

Datatracker Sign in

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/AKA-Challenge round. In this case, the peer MUST only accept EAP-Success after receiving an EAP-AKA Notification with the AT_NOTIFICATION code "Success" (32768).

On fast re-authentication, EAP-Success can only be sent after the EAP/AKA-Reauthentication 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/AKA-Reauthentication 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/AKA-Reauthentication 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/AKA-Reauthentication round. In this case, the peer MUST only accept EAP-Success after receiving an EAP-AKA Notification with the AT_NOTIFICATION code "Success" (32768).

If the peer receives an EAP-AKA 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-AKA full authentication, a Master Key (MK) is derived from the underlying AKA values (CK and IK keys), and the identity, as follows.

MK = SHA1(Identity|IK|CK)

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, the identity from the EAP-Response/Identity packet. The identity string is included as-is, without any changes. As discussed in Section 4.1.2.2, relying on EAP-Response/Identity for conveying the EAP-AKA peer identity is discouraged, and the server SHOULD use the EAP-AKA method-specific identity attributes. The hash function SHA-1 is specified in [SHA-1].

The Master Key is fed into a Pseudo-Random number Function (PRF),

Samsung Ex. 1017, Page 36 of 66

Datatracker Sign in

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-AKA requires two TEKs for its own purposes: the authentication key K_aut, to be used with the AT_MAC attribute, and the encryption key K_encr, 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 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 Annex A.

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/AKA-Identity packet, or, if EAP-Response/AKA-Identity 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/AKA-Reauthentication packet. The counter is used in network byte order. NONCE_S denotes the 16-byte random NONCE_S value from the AT_NONCE_S attribute used in the EAP-Request/AKA-Reauthentication packet. The MK is the Master Key derived on the preceding full authentication.

Datatracker

Sign in

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.

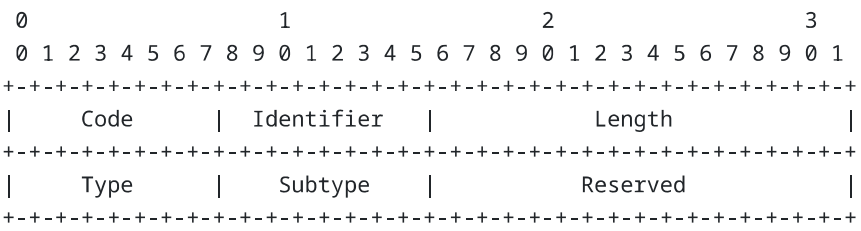
When the RADIUS attributes specified in [RFC2548] are used to transport keying material, then the first 32 bytes of the MSK correspond to MS-MPPE-RECV-KEY and the second 32 bytes to MS-MPPE-SEND-KEY. In this case, only 64 bytes of keying material (the MSK) are used.

8. Message Format and Protocol Extensibility

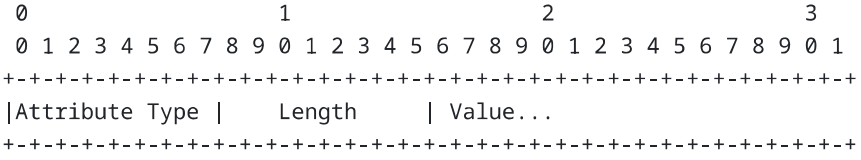
8.1. Message Format

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 Length and Identifier fields in the EAP header is also specified in [RFC3748]. In EAP-AKA, the Type field is set to 23.

In EAP-AKA, the Type-Data begins with an EAP-AKA header that consists of a 1-octet Subtype field, and a 2-octet reserved field. The Subtype values used in EAP-AKA are defined in Section 11. The formats of the EAP header and the EAP-AKA header are shown below.



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



Attribute Type

Indicates the particular type of attribute. The attribute type values are listed in Section 11.

Datatracker Sign in

Indicates the length of this attribute in multiples of 4 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 is two or more bytes in length. The type and length fields determine the format and length of the value field.

Attributes numbered within the range 0 through 127 are called non-skippable attributes. When an EAP-AKA peer encounters a non-skippable attribute type that the peer does not recognize, the peer MUST send the EAP-Response/AKA-Client-Error packet, and the authentication exchange terminates. If an EAP-AKA server encounters a non-skippable attribute that the server does not recognize, then the server sends EAP-Request/AKA-Notification packet with an

AT_NOTIFICATION code that implies general failure ("General failure after authentication" (0), or "General failure" (16384), depending on the phase of the exchange), and the authentication exchange terminates.

When an attribute numbered in the range 128 through 255 is encountered but not recognized, that particular attribute is ignored, but 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 when searching for the next attribute. These attributes are called skippable attributes.

Unless otherwise specified, the order of the attributes in an EAP-AKA message is insignificant, and an EAP-AKA implementation should not assume a certain order will 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-AKA can be extended by specifying new attribute types. If skippable attributes are used, it is possible to extend the protocol without breaking old implementations. As specified in Section 10.13, if new attributes are specified for EAP-Request/AKA-Identity or EAP-Response/AKA-Identity, then the AT_CHECKCODE MUST be used to integrity protect the new attributes.

When specifying new attributes, it should be noted that EAP-AKA 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

Datatracker Sign in

there be a reason to revise this protocol in the future, new non-skippable or skippable attributes could be specified in order to implement revised EAP-AKA versions in a backward-compatible manner. It is possible to introduce version negotiation in the EAP-Request/AKA-Identity and EAP-Response/AKA-Identity messages by specifying new skippable attributes.

9. Messages

This section specifies the messages used in EAP-AKA. 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. Message format is specified in Section 8.1.

9.1. EAP-Request/AKA-Identity

The EAP/AKA-Identity roundtrip MAY be used for obtaining the peer identity from the server. As discussed in Section 4.1, several AKA-Identity rounds may be required in order to obtain a valid peer identity.

The server MUST include one of the following identity requesting attributes: AT_PERMANENT_ID_REQ, AT_FULLAUTH_ID_REQ, 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 previously issued an EAP-Request/AKA-Identity message with the AT_PERMANENT_ID_REQ attribute, and if the server has received a response from the peer, then the server MUST NOT issue a new EAP-Request/AKA-Identity packet.

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

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

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

9.2. EAP-Response/AKA-Identity

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

The peer MUST include the AT_IDENTITY attribute. The usage of AT_IDENTITY is defined in Section 4.1.

Datatracker Sign in

9.3. EAP-Request/AKA-Challenge

The server sends the EAP-Request/AKA-Challenge on full authentication after successfully obtaining the subscriber identity.

The AT_RAND attribute MUST be included.

AT_MAC MUST be included. In EAP-Request/AKA-Challenge, there is no message-specific data covered by the MAC, see Section 10.15.

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

The AT_CHECKCODE attribute MAY be included, and in certain cases specified in Section 10.13, it MUST be included.

The EAP-Request/AKA-Challenge packet MAY include encrypted attributes for identity privacy and for communicating the next 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. Later versions of this protocol MAY specify additional attributes to be included within the encrypted data.

When processing this message, the peer MUST process AT_RAND and AT_AUTN before processing other attributes. Only if these attributes are 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/AKA-Challenge

The peer sends EAP-Response/AKA-Challenge in response to a valid EAP-Request/AKA-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/AKA-Challenge, but the peer MUST send EAP-Response/AKA-Client-Error.

The AT_MAC attribute MUST be included. In EAP-Response/AKA-Challenge, there is no message-specific data covered

Samsung Ex. 1017, Page 41 of 66

Datatracker Sign in

The AT_CHECKCODE attribute MAY be included, and in certain cases specified in Section 10.13, it MUST be included.

The AT_RESULT_IND attribute MAY be included, if it was included in EAP-Request/AKA-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/AKA-Challenge messages that include these attributes even if the server did not implement these optional attributes.

9.5. EAP-Response/AKA-Authentication-Reject

The peer sends the EAP-Response/AKA-Authentication-Reject packet if it does not accept the AUTN parameter. This version of the protocol does not specify any attributes for this message. Future versions of the protocol MAY specify attributes for this message.

The AT_MAC, AT_ENCR_DATA, or AT_IV attributes MUST NOT be used in this message.

9.6. EAP-Response/AKA-Synchronization-Failure

The peer sends the EAP-Response/AKA-Synchronization-Failure, when the sequence number in the AUTN parameter is incorrect.

The peer MUST include the AT_AUTS attribute. Future versions of the protocol MAY specify other additional attributes for this message.

The AT_MAC, AT_ENCR_DATA, or AT_IV attributes MUST NOT be used in this message.

9.7. EAP-Request/AKA-Reauthentication

The server sends the EAP-Request/AKA-Reauthentication 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/AKA-Identity.

The AT_MAC attribute MUST be included. No message-specific data is included in the MAC calculation, see Section 10.15.

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

The AT_CHECKCODE attribute MAY be included, and in certain cases specified in Section 10.13, it MUST be included.

The AT_IV and AT_ENCR_DATA attributes MUST be included. The plaintext consists of the following nested encrypted attributes,

Datatracker Sign in

9.8. EAP-Response/AKA-Reauthentication

The client sends the EAP-Response/AKA-Reauthentication packet in response to a valid EAP-Request/AKA-Reauthentication.

The AT_MAC attribute MUST be included. For EAP-Response/AKA-Reauthentication, 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_CHECKCODE attribute MAY be included, and in certain cases specified in Section 10.13, it MUST be included.

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/AKA-Reauthentication. 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/AKA-Reauthentication, but the peer MUST send EAP-Response/ AKA-Client-Error.

9.9. EAP-Response/AKA-Client-Error

The peer sends EAP-Response/AKA-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.

9.10. EAP-Request/AKA-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 AT_NOTIFICATION code is set to zero, and MUST NOT be included if 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 10.15.

Datatracker Sign in

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/AKA-Reauthentication packet on the same fast re-authentication exchange.

9.11. EAP-Response/AKA-Notification

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

The AT_MAC attribute MUST be included in cases when the P bit of the notification code in AT_NOTIFICATION of EAP-Request/AKA-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.

If EAP-Request/AKA-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/AKA-Reauthentication 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 Section 11.

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/AKA-Identity, (2) EAP-Response/AKA-Identity, (3) EAP-Request/AKA-Challenge, (4) EAP-Response/AKA-Challenge, (5) EAP-Request/AKA-Notification, (6) EAP-Response/AKA-Notification, (7) EAP-Response/AKA-Client-Error (8) EAP-Request/AKA-Reauthentication, (9) EAP-Response/AKA-Reauthentication, (10) EAP-Response/AKA-Authentication-Reject, and (11) EAP-Response/AKA-Synchronization-Failure. The column denoted with "E" indicates whether the attribute is a nested attribute that MUST be included within AT_ENCR_DATA.

"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 the future versions of the protocol.

Attribute	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	E
AT_PERMANENT_ID_REQ	0-1	0	0	0	0	0	0	0	0	0	0	N

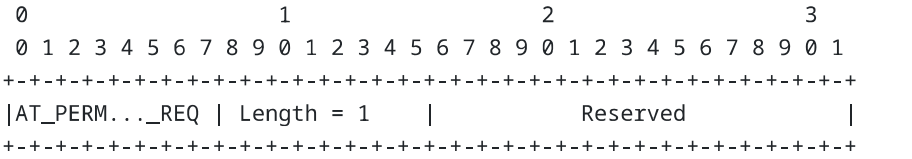
Datatracker Sign in

AT_RAND	0	0	1	0	0	0	0	0	0	0	N
AT_AUTN	0	0	1	0	0	0	0	0	0	0	N
AT_RES	0	0	0	1	0	0	0	0	0	0	N
AT_AUTS	0	0	0	0	0	0	0	0	0	1	N
AT_NEXT_PSEUDONYM	0	0	0-1	0	0	0	0	0	0	0	Y
AT_NEXT_REAUTH_ID	0	0	0-1	0	0	0	0	0-1	0	0	Y
AT_IV	0	0	0-1	0*	0-1	0-1	0	1	1	0	N
AT_ENCR_DATA	0	0	0-1	0*	0-1	0-1	0	1	1	0	N
AT_PADDING	0	0	0-1	0*	0-1	0-1	0	0-1	0-1	0	Y
AT_CHECKCODE	0	0	0-1	0-1	0	0	0	0-1	0-1	0	N
AT_RESULT_IND	0	0	0-1	0-1	0	0	0	0-1	0-1	0	N
AT_MAC	0	0	1	1	0-1	0-1	0	1	1	0	N
AT_COUNTER	0	0	0	0	0-1	0-1	0	1	1	0	Y
AT_COUNTER_TOO_SMALL	0	0	0	0	0	0	0	0	0-1	0	Y
AT_NONCE_S	0	0	0	0	0	0	0	1	0	0	Y
AT_NOTIFICATION	0	0	0	0	1	0	0	0	0	0	N
AT_CLIENT_ERROR_CODE	0	0	0	0	0	0	1	0	0	0	N

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

10.2. AT_PERMANENT_ID_REQ

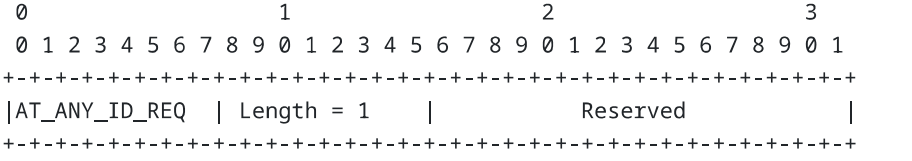
The format of the AT_PERMANENT_ID_REQ attribute is shown below.



The use of the AT_PERMANENT_ID_REQ is defined in Section 4.1. The value field only contains two reserved bytes, which are set to zero on sending and ignored on reception.

10.3. AT_ANY_ID_REQ

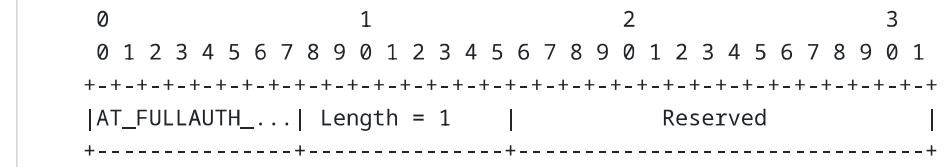
The format of the AT_ANY_ID_REQ attribute is shown below.



The use of the AT_ANY_ID_REQ is defined in Section 4.1. The value field only contains two reserved bytes, which are set to zero on sending and ignored on reception.

10.4. AT_FULLAUTH_ID_REQ

Datatracker Sign in

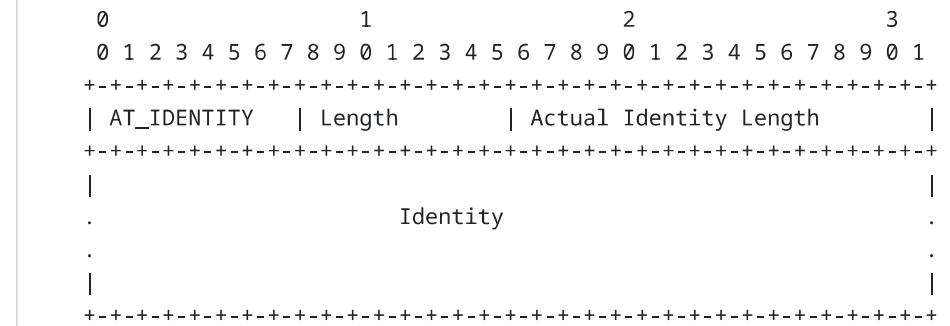


```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|AT_FULLAUTH_...| Length = 1    |           Reserved            |
+---------------+---------------+-------------------------------+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

The use of the AT_FULLAUTH_ID_REQ is defined in Section 4.1. The value field only contains two reserved bytes, which are set to zero on sending and ignored on reception.

10.5. AT_IDENTITY

The format of the AT_IDENTITY attribute is shown below.

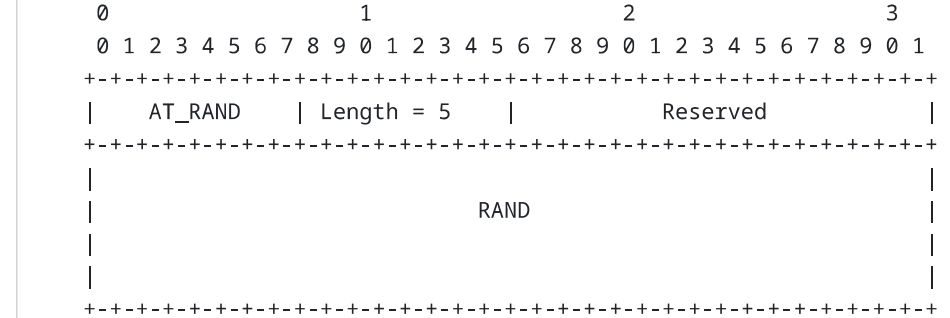


```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| AT_IDENTITY   | Length        | Actual Identity Length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+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

The use of the AT_IDENTITY is defined in Section 4.1. The value field of this attribute begins with 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 4.1.1. 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.

10.6. AT_RAND

The format of the AT_RAND attribute is shown below.



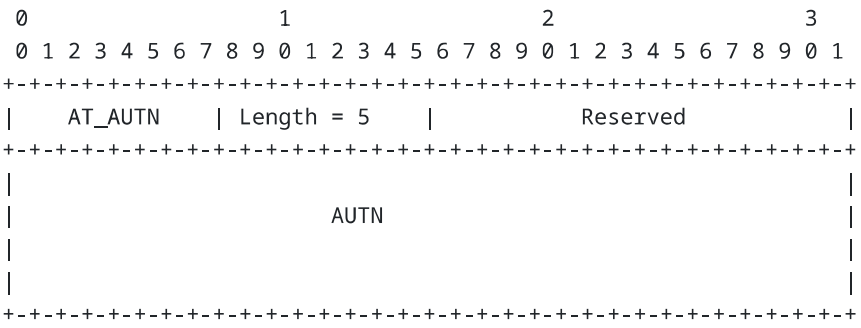
```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    AT_RAND    | Length = 5    |           Reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+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

Datatracker Sign in

reserved bytes are set to zero when sending and ignored on reception.

10.7. AT_AUTN

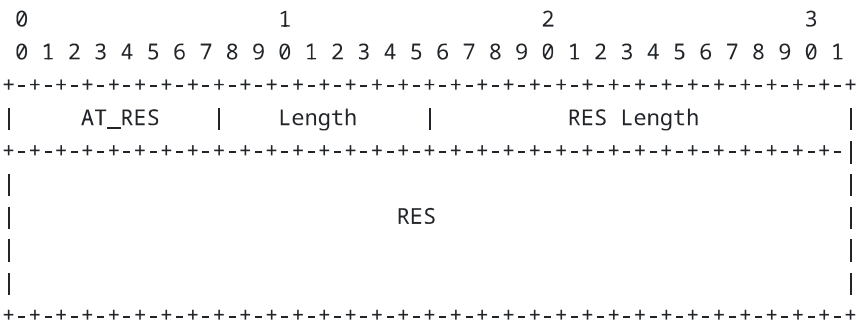
The format of the AT_AUTN attribute is shown below.



The value field of this attribute contains two reserved bytes followed by the AKA AUTN parameter, 16 bytes (128 bits). The reserved bytes are set to zero when sending and ignored on reception.

10.8. AT_RES

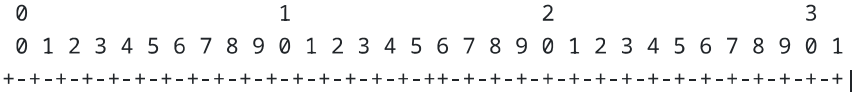
The format of the AT_RES attribute is shown below.



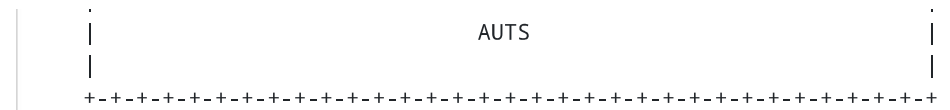
The value field of this attribute begins with the 2-byte RES Length, which identifies the exact length of the RES in bits. The RES length is followed by the AKA RES parameter. According to [TS33.105], the length of the AKA RES can vary between 32 and 128 bits. Because the length of the AT_RES attribute must be a multiple of 4 bytes, the sender pads the RES with zero bits where necessary.

10.9. AT_AUTS

The format of the AT_AUTS attribute is shown below.



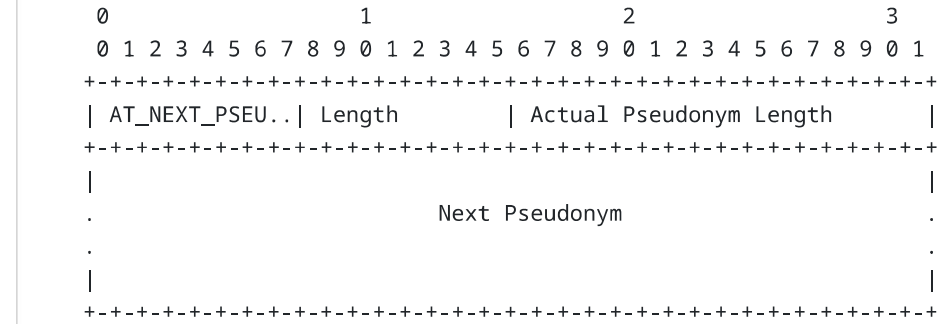
Datatracker Sign in



The value field of this attribute contains the AKA AUTS parameter, 112 bits (14 bytes).

10.10. AT_NEXT_PSEUDONYM

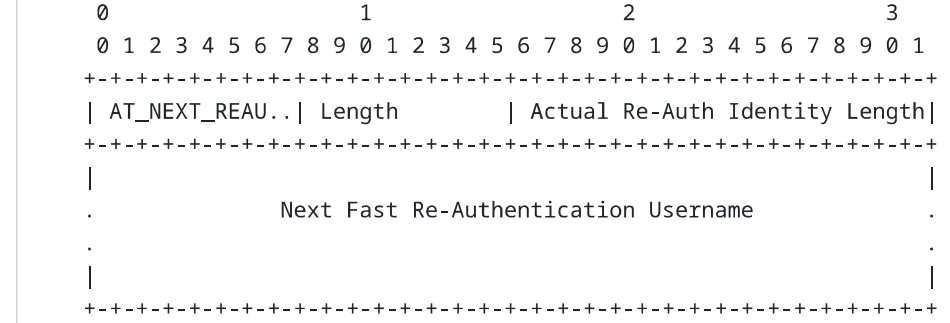
The format of the AT_NEXT_PSEUDONYM attribute is shown below.



The value field of this attribute begins with a 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.



The value field of this attribute begins with a 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

Samsung Ex. 1017, Page 48 of 66

Datatracker Sign in

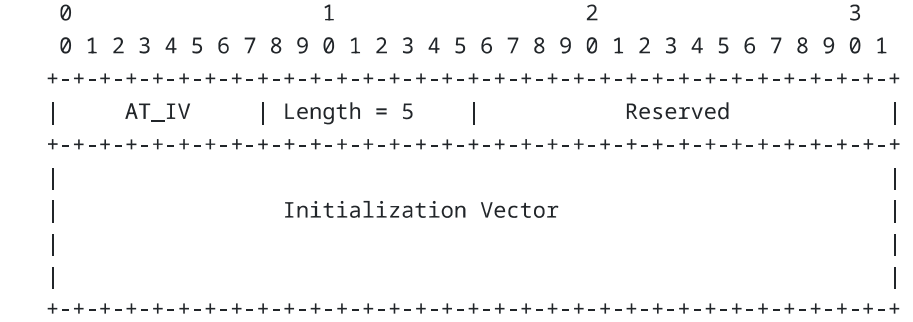
include any terminating null characters. Because the length of the attribute must be a multiple of 4 bytes, the sender pads the fast re-authentication identity with zero bytes when necessary. The identity encoding MUST follow the UTF-8 transformation format [RFC3629]. This attribute MUST always be encrypted by encapsulating it within the AT_ENCR_DATA attribute.

10.12. AT_IV, AT_ENCR_DATA, and AT_PADDING

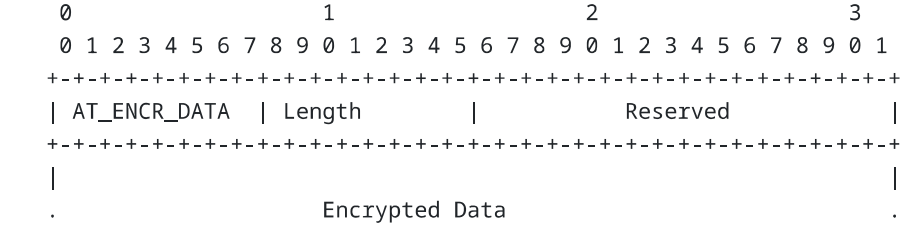
AT_IV and AT_ENCR_DATA attributes can be used to transmit encrypted information between the EAP-AKA peer and server.

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 reuse the initialization vector value from previous EAP-AKA 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.



The value field of the AT_ENCR_DATA attribute consists of two reserved bytes followed by cipher text bytes. The cipher text bytes are encrypted using the Advanced Encryption Standard (AES) [AES] with a 128-bit key in the Cipher Block Chaining (CBC) mode of operation, which uses 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.



Datatracker

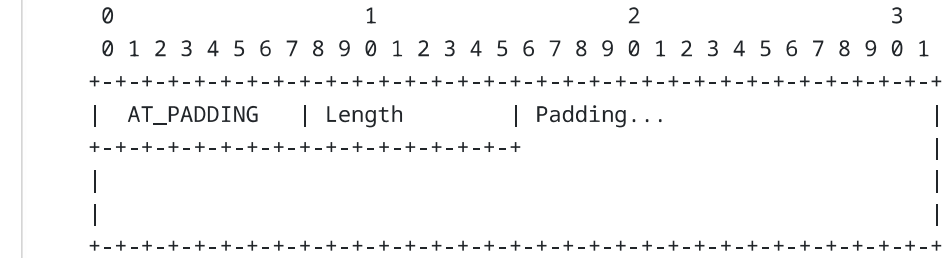
Sign in

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

The plaintext consists of nested EAP-AKA 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/AKA-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 EAP-Response/AKA-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.



10.13. AT_CHECKCODE

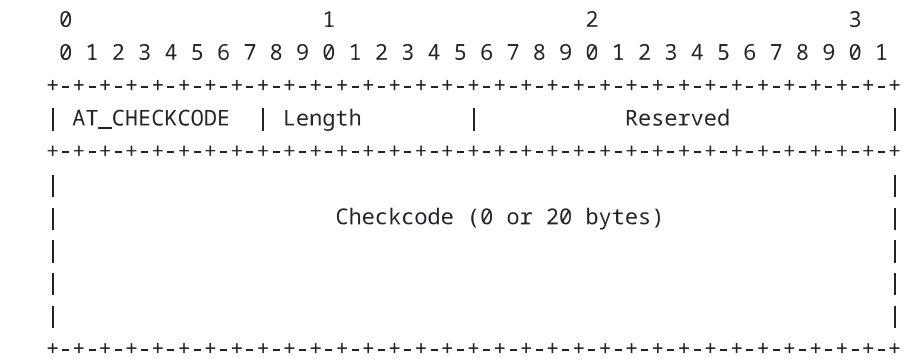
The AT_MAC attribute is not used in the very first EAP-AKA messages during the AKA-Identity round, because keying material has not been derived yet. The peer and the server may exchange one or more pairs of EAP-AKA messages of the Subtype AKA-Identity before keys are derived and before the AT_MAC attribute can be applied. The EAP/-AKA-Identity messages may also be used upon fast re-authentication.

The AT_CHECKCODE attribute MAY be used to protect the EAP/AKA-Identity messages. In full authentication, the server MAY include the AT_CHECKCODE in EAP-Request/AKA-Challenge, and the peer MAY include AT_CHECKCODE in EAP-Response/AKA-Challenge. In fast re-authentication, the server MAY include AT_CHECKCODE in EAP-Request/ AKA-Reauthentication, and the peer MAY include AT_CHECKCODE in EAP-Response/AKA-Reauthentication. The fact that the peer receives an EAP-Request with AT_CHECKCODE does not imply that

Datatracker

Sign in

attribute is used in these messages, AT_CHECKCODE will be integrity protected with AT_MAC. The format of the AT_CHECKCODE attribute is shown below.



The value field of AT_CHECKCODE begins with two reserved bytes, which may be followed by a 20-byte checkcode. If the checkcode is not included in AT_CHECKCODE, then the attribute indicates that no EAP/-AKA-Identity messages were exchanged. This may occur in both full authentication and fast re-authentication. The reserved bytes are set to zero when sending and ignored on reception.

The checkcode is a hash value, calculated with SHA1 [SHA-1], over all EAP-Request/AKA-Identity and EAP-Response/AKA-Identity packets exchanged in this authentication exchange. The packets are included in the order that they were transmitted, that is, starting with the first EAP-Request/AKA-Identity message, followed by the corresponding EAP-Response/AKA-Identity, followed by the second EAP-Request/AKA-Identity (if used), etc.

EAP packets are included in the hash calculation "as-is" (as they were transmitted or received). All reserved bytes, padding bytes, etc., that are specified for various attributes are included as such, and the receiver must not reset them to zero. No delimiter bytes, padding, or any other framing are included between the EAP packets when calculating the checkcode.

Messages are included in request/response pairs; in other words, only full "round trips" are included. Packets that are silently discarded are not included, and retransmitted packets (that have the same Identifier value) are only included once. (The base EAP protocol [RFC3748] ensures that requests and responses "match".) The EAP server must only include an EAP-Request/AKA-Identity in the calculation after it has received a corresponding response with the same Identifier value.

The peer must include the EAP-Request/AKA-Identity and the corresponding response in the calculation only if the peer receives a subsequent EAP-Request/AKA-Challenge or a follow-up EAP-Request/AKA-Identity with a different Identifier value than in the first EAP-Request/AKA-Identity.

Datatracker Sign in

The AT_CHECKCODE attribute is optional to implement. It is specified in order to allow protection of the EAP/AKA-Identity messages and any future extensions to them. The implementation of AT_CHECKCODE is RECOMMENDED.

If the receiver of AT_CHECKCODE implements this attribute, then the receiver MUST check that the checkcode is correct. If the checkcode is invalid, the receiver must operate as specified in Section 6.3.

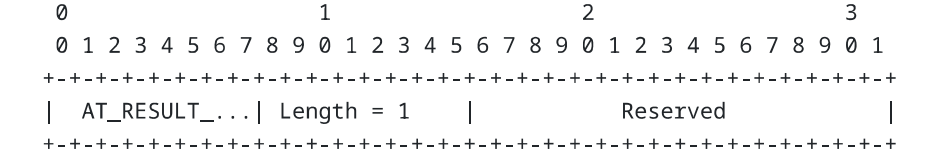
If the EAP/AKA-Identity messages are extended with new attributes, then AT_CHECKCODE MUST be implemented and used. More specifically, if the server includes any attributes other than AT_PERMANENT_ID_REQ, AT_FULLAUTH_ID_REQ, or AT_ANY_ID_REQ in the EAP-Request/AKA-Identity packet, then the server MUST include AT_CHECKCODE in EAP-Request/ AKA-Challenge or EAP-Request/AKA-Reauthentication. If the peer includes any attributes other than AT_IDENTITY in the EAP-Response/ AKA-Identity message, then the peer MUST include AT_CHECKCODE in EAP-Response/AKA-Challenge or EAP-Response/AKA-Reauthentication.

If the server implements the processing of any other attribute than AT_IDENTITY for the EAP-Response/AKA-Identity message, then the server MUST implement AT_CHECKCODE. In this case, if the server receives any attribute other than AT_IDENTITY in the EAP-Response/AKA-Identity message, then the server MUST check that AT_CHECKCODE is present in EAP-Response/AKA-Challenge or EAP-Response/ AKA-Reauthentication. The operation when a mandatory attribute is missing is specified in Section 6.3.

Similarly, if the peer implements the processing of any attribute other than AT_PERMANENT_ID_REQ, AT_FULLAUTH_ID_REQ, or AT_ANY_ID_REQ for the EAP-Request/AKA-Identity packet, then the peer MUST implement AT_CHECKCODE. In this case, if the peer receives any attribute other than AT_PERMANENT_ID_REQ, AT_FULLAUTH_ID_REQ, or AT_ANY_ID_REQ in the EAP-Request/AKA-Identity packet, then the peer MUST check that AT_CHECKCODE is present in EAP-Request/AKA-Challenge or EAP-Request/AKA-Reauthentication. The operation when a mandatory attribute is missing is specified in Section 6.3.

10.14. AT_RESULT_IND

The format of the AT_RESULT_IND attribute is shown below.



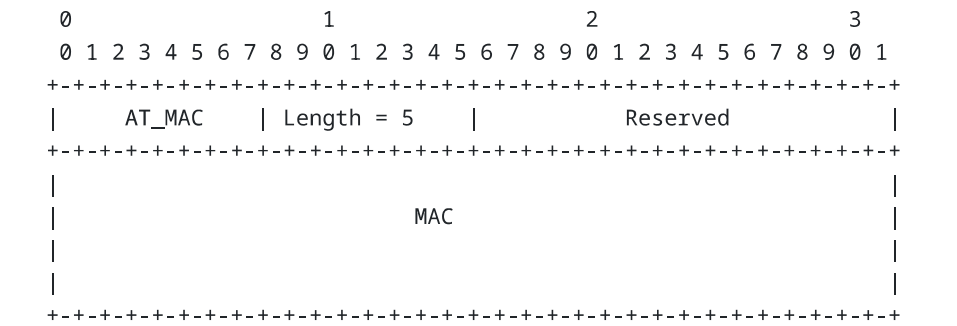
The value field of this attribute consists of two reserved bytes, which are set to zero upon sending and ignored upon reception. This attribute is always sent unencrypted, so it MUST NOT be encapsulated within the AT_ENCR_DATA attribute.

Datatracker Sign in

Section 9 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-AKA 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-AKA message in Section 9.

The format of the AT_MAC attribute is shown below.



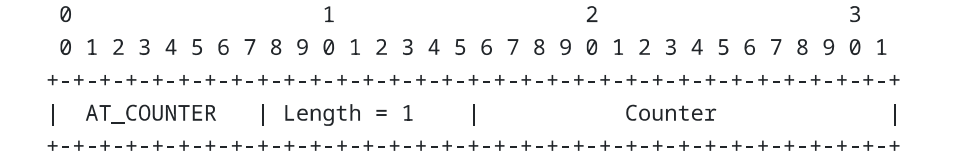
The MAC algorithm is HMAC-SHA1-128 [RFC2104] keyed hash value. (The HMAC-SHA1-128 value is obtained from the 20-byte HMAC-SHA1 value by truncating the output to 16 bytes. Hence, the length of the MAC is 16 bytes.) The derivation of the authentication key (K_aut) used in the calculation of the MAC is specified in Section 7.

When the AT_MAC attribute is included in an EAP-AKA message, the recipient MUST process the AT_MAC attribute before looking at any other attributes, except when processing EAP-Request/AKA-Challenge. The processing of EAP-Request/AKA-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.

10.16. AT_COUNTER

The format of the AT_COUNTER attribute is shown below.

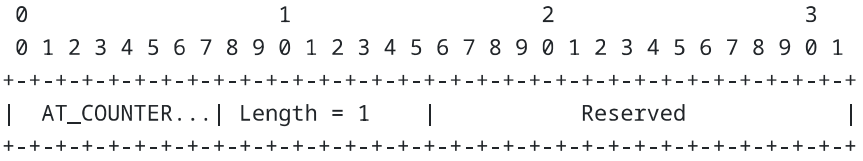


Datatracker Sign in

the AT_ENCR_DATA attribute.

10.17. AT_COUNTER_TOO_SMALL

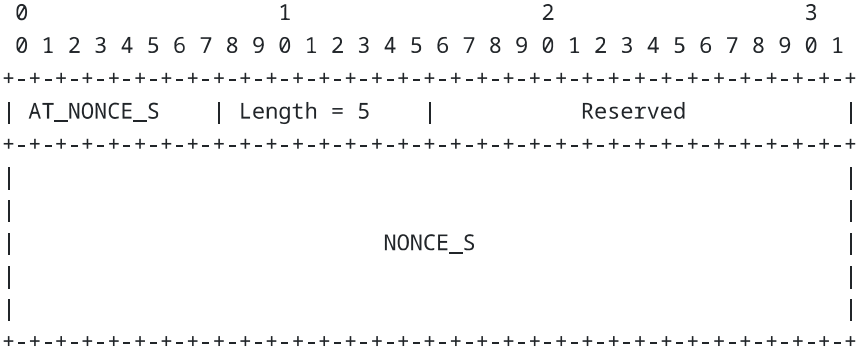
The format of the AT_COUNTER_TOO_SMALL attribute is shown below.



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.

10.18. AT_NONCE_S

The format of the AT_NONCE_S attribute is shown below.



The value field of the AT_NONCE_S attribute contains two reserved bytes followed by a random number (16 bytes) that is freshly generated by the server for this EAP-AKA fast re-authentication. The random number is used as 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 reuse the NONCE_S value from a previous EAP-AKA 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.

10.19. AT_NOTIFICATION

The format of the AT_NOTIFICATION attribute is shown below.



Datatracker Sign in

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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-AKA 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.)

10.20. 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

11. IANA and Protocol Numbering Considerations

IANA has assigned the EAP type number 23 for EAP-AKA authentication.

EAP-AKA shares most of the protocol design, such as attributes and message Subtypes, with EAP-SIM [EAP-SIM]. EAP-AKA protocol numbers should be administered in the same IANA registry with EAP-SIM. This document establishes the registries and lists the initial protocol numbers for both protocols.

Datatracker Sign in

EAP-AKA and EAP-SIM messages include a Subtype field. The Subtype is a new numbering space for which IANA administration is required. The Subtype is an 8-bit integer. The following Subtypes are specified in this document and in [EAP-SIM]:

```
AKA-Challenge.................................1
AKA-Authentication-Reject.....................2
AKA-Synchronization-Failure...................4
AKA-Identity..................................5
SIM-Start....................................10
SIM-Challenge................................11
AKA-Notification and SIM-Notification.........12
AKA-Reauthentication and SIM-Reauthentication..13
AKA-Client-Error and SIM-Client-Error.........14
```

The messages are composed of attributes, which have 8-bit attribute type numbers. Attributes numbered within the range 0 through 127 are called non-skippable attributes, and attributes within the range of 128 through 255 are called skippable attributes. The EAP-AKA and EAP-SIM attribute type number is a new numbering space for which IANA administration is required. The following attribute types are specified in this document in [EAP-SIM]:

```
AT_RAND.......................................1
AT_AUTN.......................................2
AT_RES........................................3
AT_AUTS.......................................4
AT_PADDING....................................6
AT_NONCE_MT...................................7
AT_PERMANENT_ID_REQ..........................10
AT_MAC.......................................11
AT_NOTIFICATION..............................12
AT_ANY_ID_REQ................................13
AT_IDENTITY..................................14
AT_VERSION_LIST..............................15
AT_SELECTED_VERSION..........................16
AT_FULLAUTH_ID_REQ...........................17
AT_COUNTER...................................19
AT_COUNTER_TOO_SMALL.........................20
AT_NONCE_S...................................21
AT_CLIENT_ERROR_CODE.........................22
AT_IV......................................129
AT_ENCR_DATA...............................130
AT_NEXT_PSEUDONYM..........................132
AT_NEXT_REAUTH_ID..........................133
AT_CHECKCODE...............................134
AT_RESULT_IND..............................135
```


The AT_NOTIFICATION attribute contains a 16-bit notification code value. The most significant bit of the notification code is called the S bit (success) and the second most significant bit is called the P bit (phase). If the S bit is set to zero, then the notification code indicates failure; notification codes with the S bit set to one

Datatracker Sign in

only be used after authentication. The notification code is a new numbering space for which IANA administration is required. The following values have been specified in this document and in [EAP-SIM].

```
General failure after authentication.....................0
User has been temporarily denied access.................1026
User has not subscribed to the requested service.......1031
General failure.....................................16384
Success.............................................32768
```

The AT_VERSION_LIST and AT_SELECTED_VERSION attributes, specified in [EAP-SIM], contain 16-bit EAP method version numbers. The EAP method version number is a new numbering space for which IANA administration is required. Value 1 for "EAP-SIM Version 1" has been specified in [EAP-SIM]. Version numbers are not currently used in EAP-AKA.

The AT_CLIENT_ERROR_CODE attribute contains a 16-bit client error code. The client error code is a new numbering space for which IANA administration is required. Values 0, 1, 2, and 3 have been specified in this document and in [EAP-SIM].

All requests for value assignment from the various number spaces described in this document require proper documentation, according to the "Specification Required" policy described in [RFC2434]. Requests must be specified in sufficient detail so that interoperability between independent implementations is possible. Possible forms of documentation include, but are not limited to, RFCs, the products of another standards body (e.g., 3GPP), or permanently and readily available vendor design notes.

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-AKA as well as vulnerabilities and security recommendations.

12.1. Identity Protection

EAP-AKA 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-AKA exchange. Hence, a peer that has not yet performed any EAP-AKA exchanges does not typically have a pseudonym available. If the peer does not have a pseudonym available, then the privacy mechanism cannot be used, and the permanent identity will have to be sent in the clear. The terminal SHOULD store the pseudonym in 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 (Section 4.1.2) to learn the subscriber's IMSI. However, as discussed in Section 4.1.2, the terminal can refuse to send the cleartext IMSI if it believes

Datatracker Sign in

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. The benefits and the security considerations of using an external security mechanism with EAP-AKA are beyond the scope of this document.

12.2. Mutual Authentication

EAP-AKA provides mutual authentication via the 3rd generation AKA mechanisms [TS33.102] and [S.S0055-A].

Note that this mutual authentication is 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], [EAPKeying], [ServiceIdentity].

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

12.3. Flooding the Authentication Centre

The EAP-AKA server typically obtains authentication vectors from the Authentication Centre (AuC). EAP-AKA introduces a new usage for the AuC. The protocols between the EAP-AKA server and the AuC are out of the scope of this document. However, it should be noted that a

malicious EAP-AKA peer may generate a lot of protocol requests to mount a denial-of-service attack. The EAP-AKA 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-AKA to other users of the AuC.

12.4. Key Derivation

EAP-AKA supports key derivation with 128-bit effective key strength. The key hierarchy is specified in Section 7.

The Transient EAP Keys used to protect EAP-AKA packets (K_encr, K_aut), the Master Session Keys, and the Extended Master Session Keys are cryptographically separate. 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 from AKA IK, AKA CK, EAP-AKA K_encr, EAP-AKA K_aut, the Master Session Key, or the Extended Master Session Key.

12.5. Brute-Force and Dictionary Attacks

Datatracker Sign in

passphrase, or derived from a passphrase), EAP-AKA is not vulnerable to dictionary attacks.

12.6. Protection, 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-AKA Requests and Responses. Integrity protection with AT_MAC includes the EAP header. 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.

Because keys are not available in the beginning of the EAP methods, the AT_MAC attribute cannot be used for protecting EAP/AKA-Identity messages. However, the AT_CHECKCODE attribute can optionally be used to protect the integrity of the EAP/AKA-Identity roundtrip.

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 12.1.

On full authentication, replay protection of the EAP exchange is provided by RAND and AUTN values from the underlying AKA scheme. Protection against replays of EAP-AKA messages is also based on the fact that messages that can include AT_MAC can only be sent once with a certain EAP-AKA 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-AKA notifications, if they are used after successful authentication in order to provide replay protection between re-authentication exchanges.

The contents of the user identity string are implicitly integrity protected by including them in key derivation.

Because EAP-AKA 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. On physically insecure networks, this may enable an attacker 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 before the peer successfully authenticates the server or after the peer failed to authenticate the server.

The security considerations of EAP-AKA result indications are covered

Datatracker Sign in

EAP-Failure packets sent in the clear. With EAP-AKA, confidential information MUST NOT be transmitted in EAP Notification packets.

12.7. Negotiation Attacks

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

As described in Section 8, EAP-AKA allows the protocol to be extended by defining new attribute types. When defining such attributes, it should be noted that any extra attributes included in EAP-Request/AKA-Identity or EAP-Response/AKA-Identity packets are not

included in the MACs later on, and thus some other precautions must be taken to avoid modifications to them.

EAP-AKA does not support ciphersuite negotiation or EAP-AKA protocol version negotiation.

12.8. Protected Result Indications

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

Even if an EAP-Failure packet is lost when using EAP-AKA over an unreliable medium, then the EAP-AKA 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-AKA success indication. If the optional success indications are not used, then the peer may end up believing the server completed successful authentication, when actually it failed. Because 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.

12.9. 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-AKA 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

Datatracker Sign in

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-AKA Master Session Key MAY be used to provide the cryptographic binding. However, the mechanism that provides the binding depends on the tunneling protocol and is beyond the scope of this document.

12.10. Generating Random Numbers

An EAP-AKA 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-AKA is based on the AKA mechanism, which is an authentication and key agreement mechanism based on a symmetric 128-bit pre-shared secret.

Ciphersuite negotiation: No

Mutual authentication: Yes (Section 12.2)

Integrity protection: Yes (Section 12.6)

Replay protection: Yes (Section 12.6)

Confidentiality: Yes, except method-specific success and failure indications (Section 12.1, Section 12.6)

Key derivation: Yes

Key strength: EAP-AKA supports key derivation with 128-bit effective key strength.

Description of key hierarchy: Please see Section 7.

Dictionary attack protection: N/A (Section 12.5)

Fast reconnect: Yes

Cryptographic binding: N/A

Session independence: Yes (Section 12.4)

Fragmentation: No

Channel binding: No

Indication of vulnerabilities. Vulnerabilities are discussed in

Datatracker Sign in

14. Acknowledgements and Contributions

The authors wish to thank Rolf Blom of Ericsson, Bernard Aboba of Microsoft, Arne Norefors of Ericsson, N.Asokan of Nokia, Valtteri Niemi of Nokia, Kaisa Nyberg of Nokia, Jukka-Pekka Honkanen of Nokia, Pasi Eronen of Nokia, Olivier Paridaens of Alcatel, and Ilkka Uusitalo of Ericsson for interesting discussions in this problem space.

Many thanks to Yoshihiro Ohba for reviewing the document.

This protocol has been partly developed in parallel with EAP-SIM [EAP-SIM], and hence this specification incorporates many ideas from EAP-SIM, and many contributions from the reviewer's of EAP-SIM.

The attribute format is based on the extension format of Mobile IPv4 [RFC3344].

15. References

15.1. Normative References

[TS33.102] 3rd Generation Partnership Project, "3GPP Technical Specification 3GPP TS 33.102 V5.1.0: "Technical Specification Group Services and System Aspects; 3G Security; Security Architecture (Release 5)"", December 2002.

[S.S0055-A] 3rd Generation Partnership Project 2, "3GPP2 Enhanced Cryptographic Algorithms", September 2003.

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

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

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

[TS23.003] 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.

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

Samsung Ex. 1017, Page 62 of 66

Datatracker Sign in

"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, http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf.

[TS33.105] 3rd Generation Partnership Project, "3GPP Technical Specification 3GPP TS 33.105 4.1.0: "Technical Specification Group Services and System Aspects; 3G Security; Cryptographic Algorithm Requirements (Release 4)"", June 2001.

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

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

15.2. Informative References

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

[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.

[EAPKeying] Aboba, B., Simon, D., Arkko, J., Eronen, P., and H. Levkowetz, "Extensible Authentication Protocol

Datatracker Sign in

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

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

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

[EAP-SIM] Haverinen, H., Ed. and J. Salowey, Ed., "Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM)", RFC 4186, January 2006.

Appendix A. 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

Jari Arkko
Ericsson
FIN-02420 Jorvas
Finland

EMail: jari.Arkko@ericsson.com

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

Datatracker

Sign in

Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

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

IETF IESG IAB IRTF IETF LLC IETF Trust RFC Editor IANA Privacy Statement