

Network Working Group  
Internet-Draft  
Updates: 5448 (if approved)  
Intended status: Informational  
Expires: January 3, 2019

J. Arkko  
K. Norrman  
V. Torvinen  
Ericsson  
July 2, 2018

Perfect-Forward Secrecy for the Extensible Authentication Protocol  
Method for Authentication and Key Agreement (EAP-AKA' PFS)  
draft-arkko-eap-aka-pfs-02

## Abstract

Many different attacks have been reported as part of revelations associated with pervasive surveillance. Some of the reported attacks involved compromising smart cards, such as attacking SIM card manufacturers and operators in an effort to compromise shared secrets stored on these cards. Since the publication of those reports, manufacturing and provisioning processes have gained much scrutiny and have improved. However, the danger of resourceful attackers for these systems is still a concern.

This specification is an optional extension to the EAP-AKA' authentication method which was defined in RFC 5448. The extension provides Perfect Forward Secrecy for the session key generated as a part of the authentication run in EAP-AKA'. This prevents an attacker who has gained access to the long-term pre-shared secret in a SIM card from merely passively eavesdropping the EAP-AKA' exchanges and deriving associated session keys, forcing attackers to use active attacks instead.

## Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 3, 2019.

## Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Protocol Design and Deployment Objectives . . . . .	4
3. Background . . . . .	5
3.1. AKA . . . . .	5
3.2. EAP-AKA' Protocol . . . . .	6
3.3. Attacks Against Long-Term Shared Secrets in Smart Cards . . . . .	8
4. Requirements Language . . . . .	8
5. Protocol Overview . . . . .	8
6. Extensions to EAP-AKA' . . . . .	11
6.1. AT_PUB_DH . . . . .	11
6.2. AT_KDF_DH . . . . .	11
6.3. New Key Derivation Function . . . . .	13
6.4. Diffie-Hellman Groups . . . . .	14
6.5. Message Processing . . . . .	14
6.5.1. EAP-Request/AKA'-Identity . . . . .	15
6.5.2. EAP-Response/AKA'-Identity . . . . .	15
6.5.3. EAP-Request/AKA'-Challenge . . . . .	15
6.5.4. EAP-Response/AKA'-Challenge . . . . .	16
6.5.5. EAP-Request/AKA'-Reauthentication . . . . .	16
6.5.6. EAP-Response/AKA'-Reauthentication . . . . .	16
6.5.7. EAP-Response/AKA'-Synchronization-Failure . . . . .	16
6.5.8. EAP-Response/AKA'-Authentication-Reject . . . . .	16
6.5.9. EAP-Response/AKA'-Client-Error . . . . .	17
6.5.10. EAP-Request/AKA'-Notification . . . . .	17
6.5.11. EAP-Response/AKA'-Notification . . . . .	17
7. Security Considerations . . . . .	17
8. IANA Considerations . . . . .	19
9. References . . . . .	19
9.1. Normative References . . . . .	19
9.2. Informative References . . . . .	20
Appendix A. Acknowledgments . . . . .	22

Authors' Addresses	22
--------------------	----

## 1. Introduction

Many different attacks have been reported as part of revelations associated with pervasive surveillance. Some of the reported attacks involved compromising smart cards, such as attacking SIM card manufacturers and operators in an effort to compromise shared secrets stored on these cards. Such attacks are conceivable, for instance, during the manufacturing process of cards, or during the transfer of cards and associated information to the operator. Since the publication of reports about such attacks, manufacturing and provisioning processes have gained much scrutiny and have improved.

However, the danger of resourceful attackers attempting to gain information about SIM cards is still a concern. They are a high-value target and concern a large number of people. Note that the attacks are largely independent of the used authentication technology; the issue is not vulnerabilities in algorithms or protocols, but rather the possibility of someone gaining unlawful access to key material. While the better protection of manufacturing and other processes is essential in protecting against this, there is one question that we as protocol designs can ask. Is there something that we can do to limit the consequences of attacks, should they occur?

The authors want to provide a public specification of an extension that helps defend against one aspect of pervasive surveillance. This is important, given the large number of users such practices may affect. It is also a stated goal of the IETF to ensure that we understand the surveillance concerns related to IETF protocols and take appropriate countermeasures [RFC7258]. This document does that for EAP-AKA'.

This specification is an optional extension to the EAP-AKA' authentication method [RFC5448]. The extension provides Perfect Forward Secrecy for the session key generated as a part of the authentication run in EAP-AKA'. This prevents an attacker who has gained access to the long-term pre-shared secret in a SIM card from merely passively eavesdropping the EAP-AKA' exchanges and deriving associated session keys, forcing attackers to use active attacks instead.

Attacks against AKA authentication via compromising the long-term secrets in the SIM cards have been an active discussion topic in many contexts. Perfect forward secrecy is on the list of features for the next release of 3GPP (5G Phase 2), and this document provides a basis for providing this feature in a particular fashion.

It should also be noted that 5G network architecture includes the use of the EAP framework for authentication. While any methods can be run, the default authentication method within that context will be EAP-AKA. As a result, improvements in EAP-AKA' security have a potential to improve security for large number of users.

## 2. Protocol Design and Deployment Objectives

This extension specified here re-uses large portions of the current structure of 3GPP interfaces and functions, with the rationale that this will make the construction more easily adopted. In particular, the construction maintains the interface between the Universal Subscriber Identification Module (USIM) and the mobile terminal intact. As a consequence, there is no need to roll out new credentials to existing subscribers. The work is based on an earlier paper [TrustCom2015], and uses much of the same material, but applied to EAP rather than the underlying AKA method. This specification is an initial proposal for ensuring SIM-based authentication in EAP makes attacks difficult. Comments and suggestions are much appreciated, including design improvements.

It has been a goal to implement this change as an extension of the widely supported EAP-AKA' method, rather than a completely new authentication method. The extension is implemented as a set of new, optional attributes, that are provided alongside the base attributes in EAP-AKA'. Old implementations can ignore these attributes, but their presence will nevertheless be verified as part of base EAP-AKA' integrity verification process, helping protect against bidding down attacks. This extension does not increase the number of rounds necessary to complete the protocol.

The use of this extension is at the discretion of the authenticating parties. It should be noted that PFS and defenses against passive attacks are by no means a panacea, but they can provide a partial defense that increases the cost and risk associated with pervasive surveillance.

While adding perfect forward secrecy to the existing mobile network infrastructure can be done in multiple different ways, the authors believe that the approach chosen here is relatively easily deployable. In particular:

- o As noted above, no new credentials are needed; there is no change to SIM cards.
- o PFS property can be incorporated into any current or future system that supports EAP, without changing any network functions beyond the EAP endpoints.

- o Key generation happens at the endpoints, enabling highest grade key material to be used both by the endpoints and the intermediate systems (such as access points that are given access to specific keys).
- o While EAP-AKA' is just one EAP method, for practical purposes perfect forward secrecy being available for both EAP-TLS [RFC5216] [I-D.mattsson-eap-tls13] and EAP-AKA' ensures that for many practical systems perfect forward secrecy can be enabled for either all or significant fraction of users.

### 3. Background

#### 3.1. AKA

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 contrast with its earlier GSM counterparts, 3rd generation AKA provides long key lengths and mutual authentication.

AKA works in the following manner:

- o The identity module and the home environment have agreed on a secret key beforehand.
- o 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.
- o The authentication vector is passed to the serving network, which uses it to authenticate the device.
- o The RAND and the AUTN are delivered to the identity module.
- o The identity module verifies the AUTN, again based on the secret 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 serving network.
- o The serving network 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.

### 3.2. EAP-AKA' Protocol

When AKA (and AKA') are embedded into EAP, the authentication on the network side is moved to the home environment; the serving network performs the role of a pass-through authenticator. Figure 1 describes the basic flow in the EAP-AKA' authentication process. The definition of the full protocol behaviour, along with the definition of attributes AT\_RANDOM, AT\_AUTN, AT\_MAC, and AT\_RES can be found in [RFC5448] and [RFC4187].

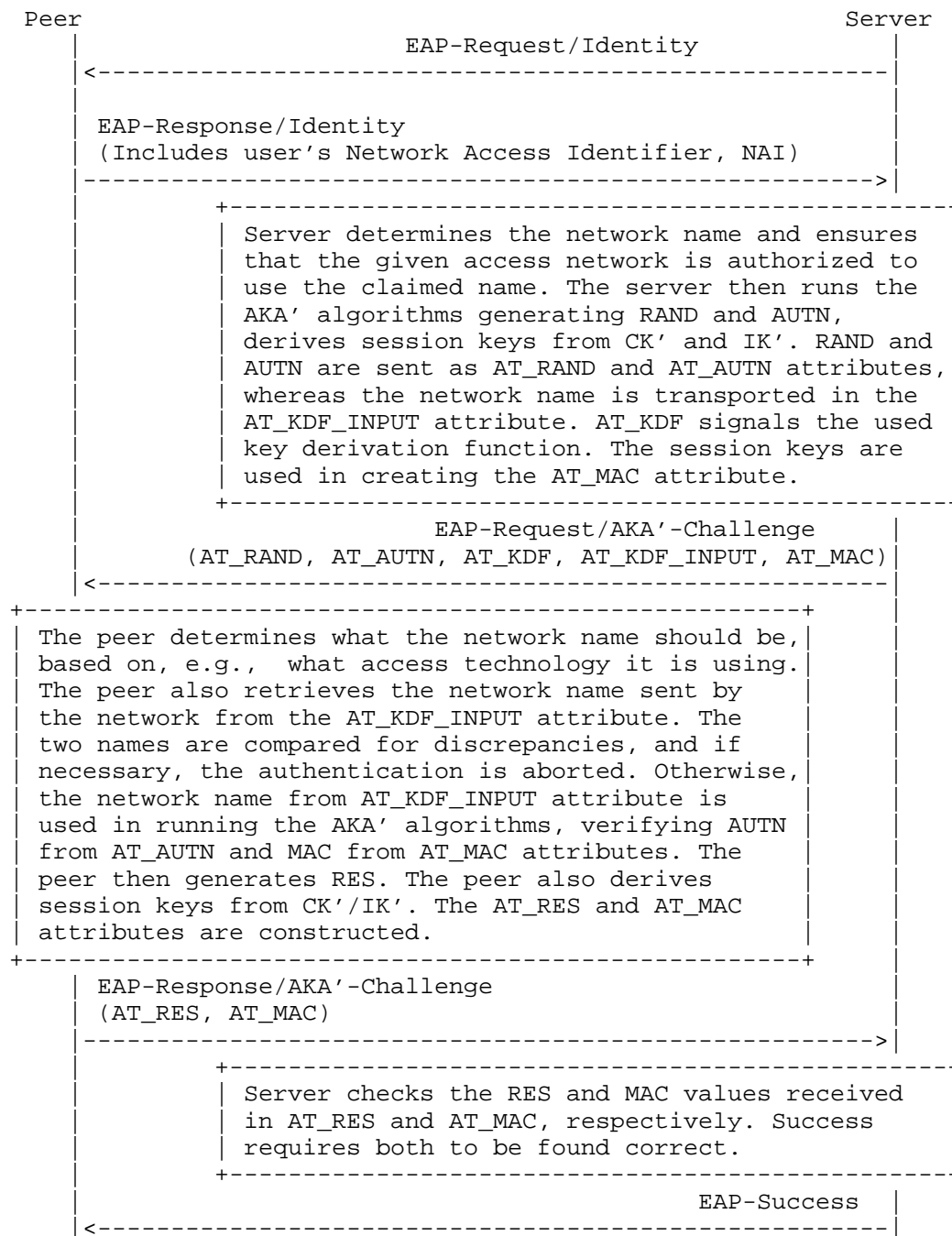


Figure 1: EAP-AKA' Authentication Process

### 3.3. Attacks Against Long-Term Shared Secrets in Smart Cards

Current 3GPP systems use (U)SIM pre-shared key based protocols to authenticate subscribers. Since the addition of replay protection and mutual authentication in the third generation 3GPP systems, there have been no published attacks that violate the security properties defined for the Authentication and Key Agreement (AKA) in, at least not within the assumed trust model. (However, there have been attacks using a different trust model [CB2014] [MT2012]; the protocol was not designed to counter those situations. There have also been attacks against systems where AKA is used in a different setting than initially intended, e.g. [BT2013].)

Recent reports of compromised long term pre-shared keys used in AKA [Heist2015] indicate a need to look into solutions that allow a weaker trust model, in particular for future 5G systems. It is also noted in [Heist2015] that, even if the current trust model is kept, some security can be retained in this situation by providing Perfect Forward Security (PFS) [DOW1992] for the session key. If AKA would have provided PFS, compromising the pre-shared key would not be sufficient to perform passive attacks; the attacker is, in addition, forced to be a Man-In-The-Middle (MITM) during the AKA run. Introducing PFS for authentication in 3GPP systems can be achieved by adding a Diffie-Hellman (DH) exchange.

## 4. Requirements Language

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].

## 5. Protocol Overview

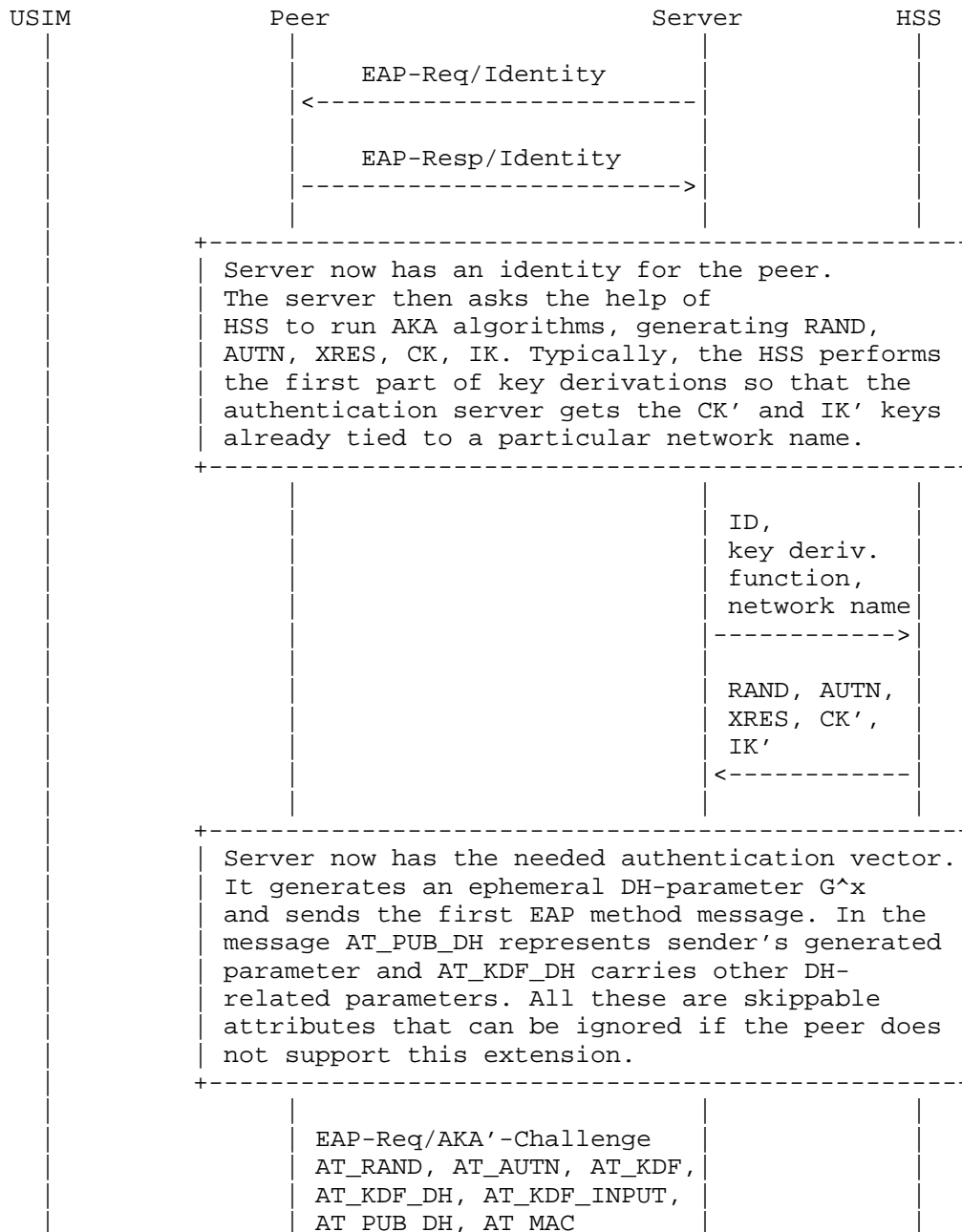
The enhancements in the protocol specified here are compatible with the signaling flow and other basic structures of both AKA and EAP-AKA'. The intent is to implement the enhancement as optional attributes that legacy implementations can ignore.

The purpose of the protocol is to achieve mutual authentication between the EAP server and peer, and to establish keying material for secure communication between the two. The enhancements brought in this document change the calculation of key material, providing new properties that are not present in key material provided by EAP-AKA' in its original form.

Figure 2 below describes the overall process. Since our goal has been to not require new infrastructure or credentials, the flow diagrams also show the conceptual interaction with the USIM card and



the 3GPP authentication server (HSS). The details of those interactions are outside the scope of this document, however, and the reader is referred to the 3GPP specifications .



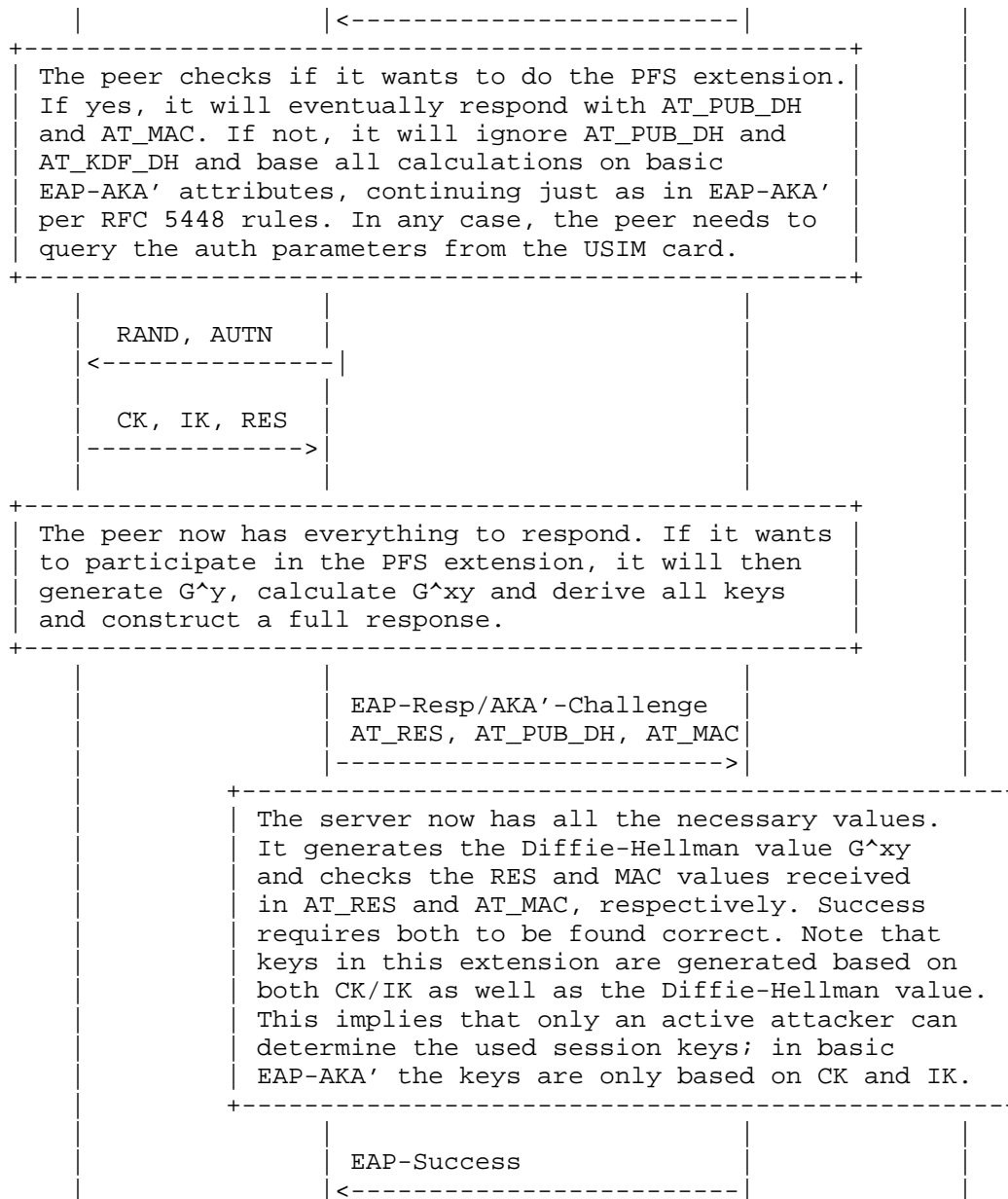


Figure 2: EAP-AKA' PFS Authentication Process

## 6. Extensions to EAP-AKA'

## 6.1. AT\_PUB\_DH

The AT\_PUB\_DH carries a Diffie-Hellman value.

The format of the AT\_PUB\_DH 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	2	3	4	5	6	7	8	9
AT_PUB_DH										Length										Value ...																			

The fields are as follows:

AT\_PUB\_DH

This is set to TBA1 BY IANA.

Length

The length of the attribute, set as other attributes in EAP-AKA [RFC4187].

Value

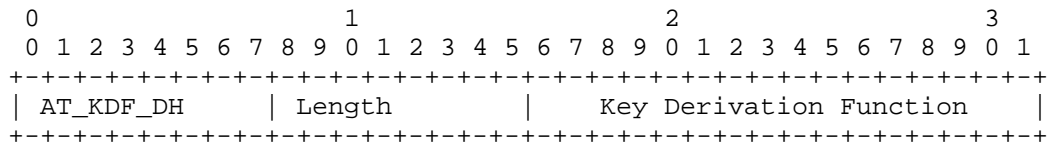
This value is the sender's Diffie-Hellman public value. For Curve25519, the length of this value is 32 bytes, represented as specified in [RFC8031] and [RFC7748].

To retain the security of the keys, the sender SHALL generate a fresh value for each run of the protocol.

## 6.2. AT\_KDF\_DH

The AT\_KDF\_DH indicates the used or desired key generation function, if the Perfect Forward Secrecy extension is taken into use. It will also at the same time indicate the used or desired Diffie-Hellman group. A new attribute is needed to carry this information, as AT\_KDF carries the legacy KDF value for those EAP peers that cannot or do not want to use this extension.

The format of the AT\_KDF\_DH attribute is shown below.



The fields are as follows:

#### AT\_KDF\_DH

This is set to TBA2 BY IANA.

#### Length

The length of the attribute, MUST be set to 1.

#### Key Derivation Function

An enumerated value representing the key derivation function that the server (or peer) wishes to use. See Section 6.3 for the functions specified in this document. Note: This field has a different name space than the similar field in the AT\_KDF attribute Key Derivation Function defined in [RFC5448].

Servers MUST send one or more AT\_KDF\_DH attributes in the EAP-Request/AKA'-Challenge message. These attributes represent the desired functions ordered by preference, the most preferred function being the first attribute.

Upon receiving a set of these attributes, if the peer supports and is willing to use the key derivation function indicated by the first attribute, and is willing and able to use the extension defined in this specification, the function is taken into use without any further negotiation. However, if the peer does not support this function or is unwilling to use it, it responds to the server with an indication that a different function is needed. Similarly with the negotiation process defined in [RFC5448] for AT\_KDF, the peer sends EAP-Response/AKA'-Challenge message that contains only one attribute, AT\_KDF\_DH with the value set to the desired alternative function from among the ones suggested by the server earlier. If there is no suitable alternative, the peer has a choice of either falling back to EAP-AKA' or behaving as if AUTN had been incorrect and failing authentication (see Figure 3 of [RFC4187]). The peer MUST fail the authentication if there are any duplicate values within the list of AT\_KDF\_DH attributes (except where the duplication is due to a request to change the key derivation function; see below for further information).

If the peer does not recognize the extension defined in this specification or is unwilling to use it, it ignores the AT\_KDF\_DH attribute.

Upon receiving an EAP-Response/AKA'-Challenge with AT\_KDF\_DH from the peer, the server checks that the suggested AT\_KDF\_DH value was one of the alternatives in its offer. The first AT\_KDF\_DH value in the message from the server is not a valid alternative. If the peer has replied with the first AT\_KDF\_DH value, the server behaves as if AT\_MAC of the response had been incorrect and fails the authentication. For an overview of the failed authentication process in the server side, see Section 3 and Figure 2 in [RFC4187]. Otherwise, the server re-sends the EAP-Response/AKA'-Challenge message, but adds the selected alternative to the beginning of the list of AT\_KDF\_DH attributes, and retains the entire list following it. Note that this means that the selected alternative appears twice in the set of AT\_KDF values. Responding to the peer's request to change the key derivation function is the only legal situation where such duplication may occur.

When the peer receives the new EAP-Request/AKA'-Challenge message, it MUST check that the requested change, and only the requested change occurred in the list of AT\_KDF\_DH attributes. If yes, it continues. If not, it behaves as if AT\_MAC had been incorrect and fails the authentication. If the peer receives multiple EAP-Request/AKA'-Challenge messages with differing AT\_KDF\_DH attributes without having requested negotiation, the peer MUST behave as if AT\_MAC had been incorrect and fail the authentication.

### 6.3. New Key Derivation Function

A new Key Derivation Function type is defined for "EAP-AKA' with DH and Curve25519", represented by value 1. It represents a particular choice of key derivation function and at the same time selects a Diffie-Hellman group to be used.

The Key Derivation Function type value is only used in the AT\_KDF\_DH attribute, and should not be confused with the different range of key derivation functions that can be represented in the AT\_KDF attribute as defined in [RFC5448].

Key derivation in this extension produces exactly the same keys for internal use within one authentication run as RFC 5448 EAP-AKA' did. For instance, K\_aut that is used in AT\_MAC is still exactly as it was in EAP-AKA'. The only change to key derivation is in re-authentication keys and keys exported out of the EAP method, MSK and EMSK. As a result, EAP-AKA' attributes such as AT\_MAC continue to be usable even when this extension is in use.

When the Key Derivation Function field in the AT\_KDF\_DH attribute is set to 1 and the Key Derivation Function field in the AT\_KDF attribute is also set to 1, the Master Key (MK) is derived and as follows below.

```
MK = PRF'(IK'|CK',"EAP-AKA'"|Identity)
MK_DH = PRF'(IK'|CK'|G^xy,"EAP-AKA' PFS"|Identity)
K_encr = MK[0..127]
K_aut = MK[128..383]
K_re = MK_DH[0..255]
MSK = MK_DH[256..767]
EMSK = MK_DH[768..1279]
```

The rest of computation proceeds as defined in Section 3.3 of [RFC5448].

For readability, an explanation of the notation used above is copied here: [n..m] denotes the substring from bit n to m. PRF' is a new pseudo-random function specified in [RFC5448]. K\_encr is the encryption key, 128 bits, K\_aut is the authentication key, 256 bits, K\_re is the re-authentication key, 256 bits, MSK is the Master Session Key, 512 bits, and EMSK is the Extended Master Session Key, 512 bits. MSK and EMSK are outputs from a successful EAP method run [RFC3748].

CK and IK are produced by the AKA algorithm. IK' and CK' are derived as specified in [RFC5448] from IK and CK.

The value "EAP-AKA'" is an eight-characters-long ASCII string. It is used as is, without any trailing NUL characters. Similarly, "EAP-AKA' PFS" is a twelve-characters-long ASCII string, also used as is.

Identity is the peer identity as specified in Section 7 of [RFC4187].

#### 6.4. Diffie-Hellman Groups

The selection of suitable groups for the Diffie-Hellman computation is necessary. The choice of a group is made at the same time as deciding to use of particular key derivation function in AT\_KDF\_DH. For "EAP-AKA' with DH and Curve25519" the Diffie-Hellman group is the Curve25519 group specified in [RFC8031].

#### 6.5. Message Processing

This section specifies the changes related to message processing when this extension is 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, where those details are different for this extension than the base EAP-AKA' or EAP-AKA protocol. Unless otherwise specified here, the rules from [RFC5448] or [RFC4187] apply.

#### 6.5.1. EAP-Request/AKA'-Identity

No changes, except that the AT\_KDF\_DH or AT\_PUB\_DH attributes MUST NOT be added to this message. The appearance of these messages in a received message MUST be ignored.

#### 6.5.2. EAP-Response/AKA'-Identity

No changes, except that the AT\_KDF\_DH or AT\_PUB\_DH attributes MUST NOT be added to this message. The appearance of these messages in a received message MUST be ignored.

#### 6.5.3. EAP-Request/AKA'-Challenge

The server sends the EAP-Request/AKA'-Challenge on full authentication as specified by [RFC4187] and [RFC5448]. The attributes AT\_RANDOM, AT\_AUTN, and AT\_MAC MUST be included and checked on reception as specified in [RFC4187]. They are also necessary for backwards compatibility.

In EAP-Request/AKA'-Challenge, there is no message-specific data covered by the MAC for the AT\_MAC attribute. The AT\_KDF\_DH and AT\_PUB\_DH attributes MUST be included. The AT\_PUB\_DH attribute carries the server's public Diffie-Hellman key. If either AT\_KDF\_DH or AT\_PUB\_DH is missing on reception, the peer MUST treat them as if neither one was sent, and assume that the extension defined in this specification is not in use.

The AT\_RESULT\_IND, AT\_CHECKCODE, AT\_IV, AT\_ENCR\_DATA, AT\_PADDING, AT\_NEXT\_PSEUDONYM, AT\_NEXT\_REAUTH\_ID and other attributes may be included as specified in Section 9.3 of [RFC4187].

When processing this message, the peer MUST process AT\_RANDOM, AT\_AUTN, AT\_KDF\_DH, AT\_PUB\_DH before processing other attributes. Only if these attributes are verified to be valid, the peer derives keys and verifies AT\_MAC. If the peer is unable or unwilling to perform the extension specified in this document, it proceeds as defined in [RFC5448]. Finally, the operation in case an error occurs is specified in Section 6.3.1. of [RFC4187].

#### 6.5.4. EAP-Response/AKA'-Challenge

The peer sends EAP-Response/AKA'-Challenge in response to a valid EAP-Request/AKA'-Challenge message, as specified by [RFC4187] and [RFC5448]. If the peer supports and is willing to perform the extension specified in this protocol, and the server had made a valid request involving the attributes specified in Section 6.5.3, the peer responds per the rules specified below. Otherwise, the peer responds as specified in [RFC4187] and [RFC5448] and ignores the attributes related to this extension.

The AT\_MAC attribute MUST be included and checked as specified in [RFC5448]. In EAP-Response/AKA'-Challenge, there is no message-specific data covered by the MAC. The AT\_PUB\_DH attribute MUST be included, and carries the peer's public Diffie-Hellman key.

The AT\_RES attribute MUST be included and checked as specified in [RFC4187].

The AT\_CHECKCODE, AT\_RESULT\_IND, AT\_IV, AT\_ENCR\_DATA and other attributes may be included as specified in Section 9.4 of [RFC4187].

#### 6.5.5. EAP-Request/AKA'-Reauthentication

No changes, but note that the re-authentication process uses the keys generated in the original EAP-AKA' authentication, which, if the extension specified in this documents is in use, employs key material from the Diffie-Hellman procedure.

#### 6.5.6. EAP-Response/AKA'-Reauthentication

No changes, but as discussed in Section 6.5.5, re-authentication is based on the key material generated by EAP-AKA' and the extension defined in this document.

#### 6.5.7. EAP-Response/AKA'-Synchronization-Failure

No changes, except that the AT\_KDF\_DH or AT\_PUB\_DH attributes MUST NOT be added to this message. The appearance of these messages in a received message MUST be ignored.

#### 6.5.8. EAP-Response/AKA'-Authentication-Reject

No changes, except that the AT\_KDF\_DH or AT\_PUB\_DH attributes MUST NOT be added to this message. The appearance of these messages in a received message MUST be ignored.



#### 6.5.9. EAP-Response/AKA'-Client-Error

No changes, except that the AT\_KDF\_DH or AT\_PUB\_DH attributes MUST NOT be added to this message. The appearance of these messages in a received message MUST be ignored.

#### 6.5.10. EAP-Request/AKA'-Notification

No changes.

#### 6.5.11. EAP-Response/AKA'-Notification

No changes.

### 7. Security Considerations

This section deals only with the changes to security considerations as they differ from EAP-AKA', or as new information has been gathered since the publication of [RFC5448].

The possibility of attacks against key storage offered in SIM or other smart cards has been a known threat. But as the discussion in Section 3.3 shows, the likelihood of practically feasible attacks has increased. Many of these attacks can be best dealt with improved processes, e.g., limiting the access to the key material within the factory or personnel, etc. But not all attacks can be entirely ruled out for well-resourced adversaries, irrespective of what the technical algorithms and protection measures are.

This extension can provide assistance in situations where there is a danger of attacks against the key material on SIM cards by adversaries that can not or who are unwilling to mount active attacks against large number of sessions. This extension is most useful when used in a context where EAP keys are used without further mixing that can provide Perfect Forward Secrecy. For instance, when used with IKEv2, the session keys produced by IKEv2 have this property, so better characteristics of EAP keys is not that useful. However, typical link layer usage of EAP does not involve running Diffie-Hellman, so using EAP to authenticate access to a network is one situation where the extension defined in this document can be helpful.

The following security properties of EAP-AKA' are impacted through this extension:

Protected ciphersuite negotiation

EAP-AKA' has a negotiation mechanism for selecting the key derivation functions, and this mechanism has been extended by the extension specified in this document. The resulting mechanism continues to be secure against bidding down attacks.

There are two specific needs in the negotiation mechanism:

Negotiating key derivation function within the extension

The negotiation mechanism allows changing the offered key derivation function, but the change is visible in the final EAP-Request/AKA'-Challenge message that the server sends to the peer. This message is authenticated via the AT\_MAC attribute, and carries both the chosen alternative and the initially offered list. The peer refuses to accept a change it did not initiate. As a result, both parties are aware that a change is being made and what the original offer was.

Negotiating the use of this extension

This extension is offered by the server through presenting the AT\_KDF\_DH and AT\_PUB\_DH attributes in the EAP-Request/AKA'-Challenge message. These attributes are protected by AT\_MAC, so attempts to change or omit them by an adversary will be detected. (Except of course, if the adversary holds the long-term shared secret and is willing to engage in an active attack, but that is a case that cannot be solved by a technical change in this protocol.) However, as discussed in the introduction, even an attacker with access to the long-term keys is required to be MITM on each AKA run, which makes mass surveillance slightly more laborious.

Key derivation

This extension provides key material that is based on the Diffie-Hellman keys, yet bound to the authentication through the (U)SIM card. This means that subsequent payload communications between the parties are protected with keys that are not solely based on information in the clear (such as the RAND) and information derivable from the long-term shared secrets on the (U)SIM card. As a result, if anyone successfully recovers shared secret information, they are unable to decrypt communications protected by the keys generated through this extension. Note that the recovery of shared secret information could occur either before or after the time that the protected communications are used. When this extension is used, communications at time t0 can be protected if at some later time t1 an adversary learns of long-term shared

secret and has access to a recording of the encrypted communications.

Obviously, this extension is still vulnerable to attackers that are willing to perform an active attack and who at the time of the attack have access to the long-term shared secret.

This extension does not change the properties of related to re-authentication. No new Diffie-Hellman run is performed during the re-authentication allowed by EAP-AKA'. However, if this extension was in use when the original EAP-AKA' authentication was performed, the keys used for re-authentication ( $K_{re}$ ) are based on the Diffie-Hellman keys, and hence continue to be equally safe against expose of the long-term secrets as the original authentication.

## 8. IANA Considerations

This extension of EAP-AKA' shares its attribute space and subtypes with EAP-SIM [RFC4186], EAP-AKA [RFC4186], and EAP-AKA' [RFC5448].

Two new Attribute Type value (TBA1, TBA2) in the skippable range need to be assigned for AT\_PUB\_DH (Section 6.1) and AT\_KDF\_DH (Section 6.2) in the EAP-AKA and EAP-SIM Parameters registry under Attribute Types.

Also, a new registry should be created to represent Diffie-Hellman Key Derivation Function types. The "EAP-AKA' with DH and Curve25519" type (1, see Section 6.3) needs to be assigned, along with one reserved value. The initial contents of this namespace are therefore as below; new values can be created through the Specification Required policy [RFC8126].

Value	Description	Reference
-----	-----	-----
0	Reserved	[TBD BY IANA: THIS RFC]
1	EAP-AKA' with DH and Curve25519	[TBD BY IANA: THIS RFC]
2-65535	Unassigned	

## 9. References

### 9.1. Normative References

- [RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <<https://www.rfc-editor.org/info/rfc2104>>.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <<https://www.rfc-editor.org/info/rfc3748>>.
- [RFC4187] Arkko, J. and H. Haverinen, "Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA)", RFC 4187, DOI 10.17487/RFC4187, January 2006, <<https://www.rfc-editor.org/info/rfc4187>>.
- [RFC5448] Arkko, J., Lehtovirta, V., and P. Eronen, "Improved Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA')", RFC 5448, DOI 10.17487/RFC5448, May 2009, <<https://www.rfc-editor.org/info/rfc5448>>.
- [RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <<https://www.rfc-editor.org/info/rfc7296>>.
- [RFC7748] Langley, A., Hamburg, M., and S. Turner, "Elliptic Curves for Security", RFC 7748, DOI 10.17487/RFC7748, January 2016, <<https://www.rfc-editor.org/info/rfc7748>>.
- [RFC8031] Nir, Y. and S. Josefsson, "Curve25519 and Curve448 for the Internet Key Exchange Protocol Version 2 (IKEv2) Key Agreement", RFC 8031, DOI 10.17487/RFC8031, December 2016, <<https://www.rfc-editor.org/info/rfc8031>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.

## 9.2. Informative References

- [RFC4186] 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, DOI 10.17487/RFC4186, January 2006, <<https://www.rfc-editor.org/info/rfc4186>>.

- [RFC5216] Simon, D., Aboba, B., and R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, DOI 10.17487/RFC5216, March 2008, <<https://www.rfc-editor.org/info/rfc5216>>.
- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <<https://www.rfc-editor.org/info/rfc7258>>.
- [I-D.mattsson-eap-tls13] Mattsson, J. and M. Sethi, "Using EAP-TLS with TLS 1.3", draft-mattsson-eap-tls13-02 (work in progress), March 2018.
- [TrustCom2015] Arkko, J., Norrman, K., Naslund, M., and B. Sahlin, "A USIM compatible 5G AKA protocol with perfect forward secrecy", August 2015 in Proceedings of the TrustCom 2015, IEEE.
- [CB2014] Choudhary, A. and R. Bhandari, "3GPP AKA Protocol: Simplified Authentication Process", December 2014, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 12.
- [MT2012] Mjolsnes, S. and J-K. Tsay, "A vulnerability in the UMTS and LTE authentication and key agreement protocols", October 2012, in Proceedings of the 6th international conference on Mathematical Methods, Models and Architectures for Computer Network Security: computer network security.
- [BT2013] Beekman, J. and C. Thompson, "Breaking Cell Phone Authentication: Vulnerabilities in AKA, IMS and Android", August 2013, in 7th USENIX Workshop on Offensive Technologies, WOOT '13.
- [Heist2015] Scahill, J. and J. Begley, "The great SIM heist", February 2015, in <https://firstlook.org/theintercept/2015/02/19/great-sim-heist/> .
- [DOW1992] Diffie, W., vanOorschot, P., and M. Wiener, "Authentication and Authenticated Key Exchanges", June 1992, in Designs, Codes and Cryptography 2 (2): pp. 107-125.

## Appendix A. Acknowledgments

The authors would like to note that the technical solution in this document came out of the TrustCom paper [TrustCom2015], whose authors were J. Arkko, K. Norrman, M. Naslund, and B. Sahlin. This document uses also a lot of material from [RFC4187] by J. Arkko and H. Haverinen as well as [RFC5448] by J. Arkko, V. Lehtovirta, and P. Eronen.

The authors would also like to thank Tero Kivinen, John Mattson, Mohit Sethi, Vesa Lehtovirta, Joseph Salowey, Kathleen Moriarty, Zhang Fu, Bengt Sahlin, Ben Campbell, Prajwol Kumar Nakarmi, Goran Rune, Tim Evans, Helena Vahidi Mazinani, Anand R. Prasad, and many other people at the GSMA and 3GPP groups for interesting discussions in this problem space.

## Authors' Addresses

Jari Arkko  
Ericsson  
Jorvas 02420  
Finland

Email: jari.arkko@piuha.net

Karl Norrman  
Ericsson  
Stockholm 16483  
Sweden

Email: karl.norrman@ericsson.com

Vesa Torvinen  
Ericsson  
Jorvas 02420  
Finland

Email: vesa.torvinen@ericsson.com

Network Working Group  
Internet-Draft  
Updates: 5216 (if approved)  
Intended status: Standards Track  
Expires: November 30, 2018

J. Mattsson  
M. Sethi  
Ericsson  
May 29, 2018

Using EAP-TLS with TLS 1.3  
draft-ietf-emu-eap-tls13-00

Abstract

This document specifies the use of EAP-TLS with TLS 1.3 while remaining backwards compatible with existing implementations of EAP-TLS. TLS 1.3 provides significantly improved security, privacy, and reduced latency when compared to earlier versions of TLS. This document updates RFC 5216.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on November 30, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
1.1. Requirements and Terminology . . . . .	3
2. Protocol Overview . . . . .	3
2.1. Overview of the EAP-TLS Conversation . . . . .	3
2.1.1. Base Case . . . . .	3
2.1.2. Resumption . . . . .	6
2.1.3. Termination . . . . .	7
2.1.4. Privacy . . . . .	10
2.1.5. Fragmentation . . . . .	12
2.2. Identity Verification . . . . .	12
2.3. Key Hierarchy . . . . .	12
2.4. Parameter Negotiation and Compliance Requirements . . . . .	13
3. Detailed Description of the EAP-TLS Protocol . . . . .	13
4. IANA considerations . . . . .	13
5. Security Considerations . . . . .	14
5.1. Security Claims . . . . .	14
5.2. Peer and Server Identities . . . . .	14
5.3. Certificate Validation . . . . .	14
5.4. Certificate Revocation . . . . .	14
5.5. Packet Modification Attacks . . . . .	15
6. References . . . . .	15
6.1. Normative References . . . . .	15
6.2. Informative references . . . . .	16
Appendix A. Updated references . . . . .	17
Appendix B. Acknowledgments . . . . .	18
Authors' Addresses . . . . .	18

## 1. Introduction

The Extensible Authentication Protocol (EAP), defined in [RFC3748], provides a standard mechanism for support of multiple authentication methods. EAP-Transport Layer Security (EAP-TLS) [RFC5216] specifies an EAP authentication method with certificate-based mutual authentication and key derivation utilizing the TLS handshake protocol for cryptographic algorithms and protocol version negotiation, mutual authentication, and establishment of shared secret keying material. EAP-TLS is widely supported for authentication in IEEE 802.11 [IEEE-802.11] networks (Wi-Fi) using IEEE 802.1X [IEEE-802.1X] and it's the default mechanism for certificate based authentication in MulteFire [MulteFire] and 3GPP 5G [TS.33.501] networks. EAP-TLS [RFC5216] references TLS 1.0 [RFC2246] and TLS 1.1 [RFC4346], but works perfectly also with TLS 1.2 [RFC5246].



Weaknesses found in previous versions of TLS, as well as new requirements for security, privacy, and reduced latency has led to the development of TLS 1.3 [I-D.ietf-tls-tls13], which in large parts is a complete remodeling of the TLS handshake protocol including a different message flow, different handshake messages, different key schedule, different cipher suites, different resumption, and different privacy protection. This means that significant parts of the normative text in the previous EAP-TLS specification [RFC5216] are not applicable to EAP-TLS with TLS 1.3 (or higher). Therefore, aspects such as resumption, privacy handling, and key derivation need to be appropriately addressed for EAP-TLS with TLS 1.3 (or higher).

This document defines how to use EAP-TLS with TLS 1.3 (or higher) and does not change how EAP-TLS is used with older versions of TLS. While this document updates EAP-TLS [RFC5216], it remains backwards compatible with it and existing implementations of EAP-TLS. This document only describes differences compared to [RFC5216].

In addition to the improved security and privacy offered by TLS 1.3, there are other significant benefits of using EAP-TLS with TLS 1.3. When EAP-TLS is used with support for privacy, TLS 1.3 requires two fewer round-trips. TLS 1.3 also introduces more possibilities to reduce fragmentation when compared to earlier versions of TLS.

## 1.1. Requirements and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]. Readers are expected to be familiar with the terms and concepts used in EAP-TLS [RFC5216] and TLS 1.3 [I-D.ietf-tls-tls13].

## 2. Protocol Overview

### 2.1. Overview of the EAP-TLS Conversation

#### 2.1.1. Base Case

TLS 1.3 changes both the message flow and the handshake messages compared to earlier versions of TLS. Therefore, much of Section 2.1 of RFC5216 [RFC5216] does not apply for TLS 1.3 (or higher).

After receiving an EAP-Request packet with EAP-Type=EAP-TLS as described in [RFC5216] the conversation will continue with the TLS handshake protocol encapsulated in the data fields of EAP-Response and EAP-Request packets. When EAP-TLS is used with TLS version 1.3 or higher, the formatting and processing of the TLS handshake SHALL be done as specified in that version of TLS. This document only

lists additional and different requirements, restrictions, and processing compared to [I-D.ietf-tls-tls13] and [RFC5216].

The EAP server MUST authenticate with a certificate and SHOULD require the EAP peer to authenticate with a certificate. Certificates can be of any type supported by TLS including raw public keys. Pre-Shared Key (PSK) authentication SHALL NOT be used except for resumption. SessionID is deprecated in TLS 1.3 and the EAP server SHALL ignore the `legacy_session_id` field if TLS 1.3 is negotiated. Resumption is handled as described in Section 2.1.2. After the TLS handshake has completed, the EAP server sends EAP-Success.

As stated in [RFC5216], the TLS cipher suite shall not be used to protect application data. This applies also for early application data. When EAP-TLS is used with TLS 1.3, early application data SHALL NOT be used.

In the case where EAP-TLS with mutual authentication is successful, the conversation will appear as shown in Figure 1.

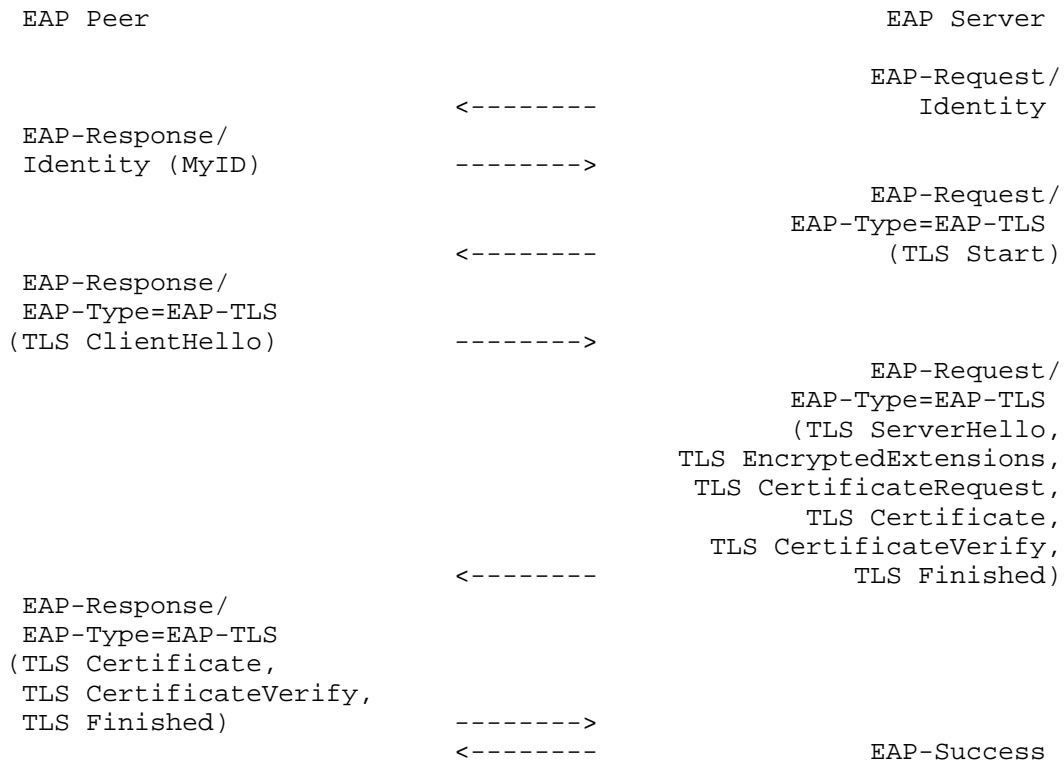


Figure 1: EAP-TLS mutual authentication

When using EAP-TLS with TLS 1.3, the EAP server MUST indicate support of resumption in the initial authentication. To indicate support of resumption, the EAP server sends a NewSessionTicket message (containing a PSK and other parameters) after it has received the Finished message.

In the case where EAP-TLS with mutual authentication and ticket establishment is successful, the conversation will appear as shown in Figure 2.

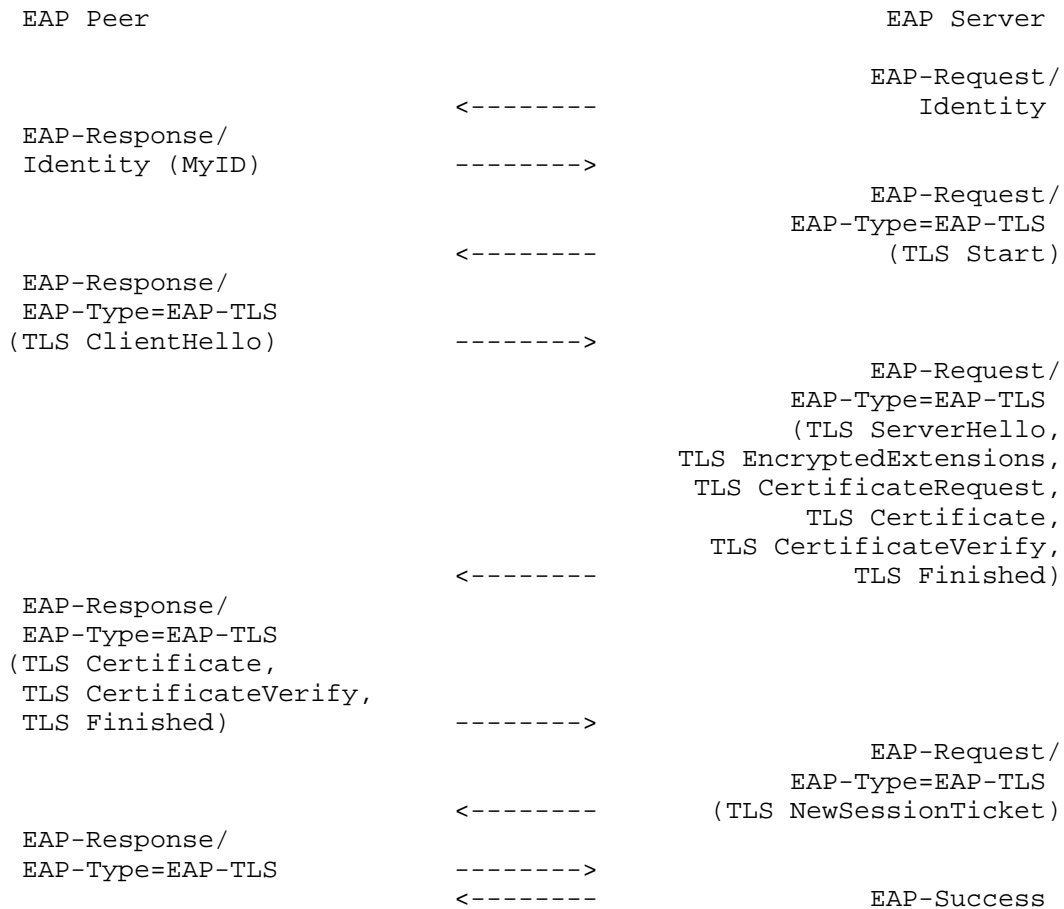


Figure 2: EAP-TLS ticket establishment

### 2.1.2. Resumption

TLS 1.3 replaces the session resumption mechanisms in earlier versions of TLS with a new PSK exchange. When EAP-TLS is used with TLS version 1.3 or higher, EAP-TLS SHALL use a resumption mechanism compatible with that version of TLS.

For TLS 1.3, resumption is described in Section 2.2 of [I-D.ietf-tls-tls13]. If the client has received a NewSessionTicket message from the server, the client can use the PSK identity received in the ticket to negotiate the use of the associated PSK. If the server accepts it, then the security context of the new connection is tied to the original connection and the key derived from the initial handshake is used to bootstrap the cryptographic state instead of a

full handshake. It is left up to the EAP peer whether to use resumption, but a EAP peer SHOULD use resumption as long as it has a valid ticket cached. It is RECOMMENDED that the EAP server accept resumption as long as the ticket is valid. However, the server MAY choose to require a full authentication.

A subsequent authentication using resumption, where both sides authenticate successfully is shown in Figure 3.

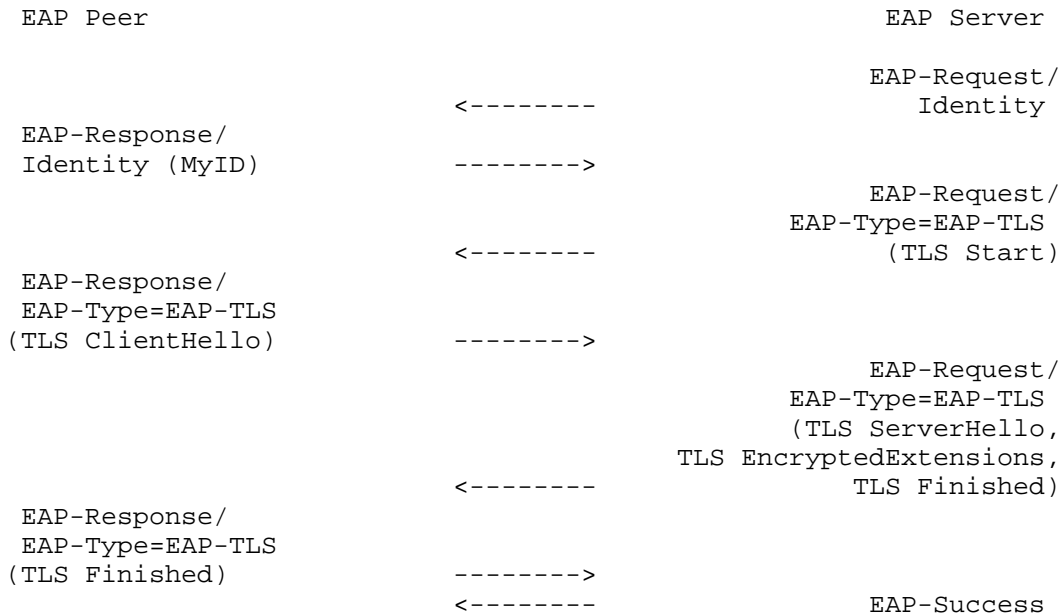


Figure 3: EAP-TLS resumption

As specified in Section 2.2 of [I-D.ietf-tls-tls13], the EAP peer SHOULD supply a "key\_share" extension when offering resumption, which allows the EAP server to decline resumption and continue the handshake as a full handshake. The message flow in this case is given by Figure 1 or Figure 2. If the EAP peer did not supply a "key\_share" extension when offering resumption, the EAP server needs to reject the ClientHello and the EAP peer needs to restart a full handshake. The message flow in this case is given by Figure 4 followed by Figure 1 or Figure 2.

### 2.1.3. Termination

TLS 1.3 changes both the message flow and the handshake messages compared to earlier versions of TLS. Therefore, some normative text in Section 2.1.3 of RC5216 [RFC5216] does not apply for TLS 1.3 or

higher. The two paragraphs below replaces the corresponding paragraphs in Section 2.1.3 of RC5216 [RFC5216] when EAP-TLS is used with TLS 1.3 or higher. The other paragraphs in Section 2.1.3 of RC5216 [RFC5216] still apply with the exception that SessionID is deprecated.

If the EAP Server authenticates successfully the EAP Peer MUST send an EAP-Response message with EAP-Type=EAP-TLS containing TLS records confirming the processing in the version of TLS used.

If the EAP Peer authenticates successfully the EAP Server MUST send an EAP-Request packet with EAP-Type=EAP-TLS containing TLS records confirming to the processing in the version of TLS used. The message flow ends with the EAP Server sending a EAP-Success message.

In the case where the server rejects the ClientHello, the conversation will appear as shown in Figure 4.

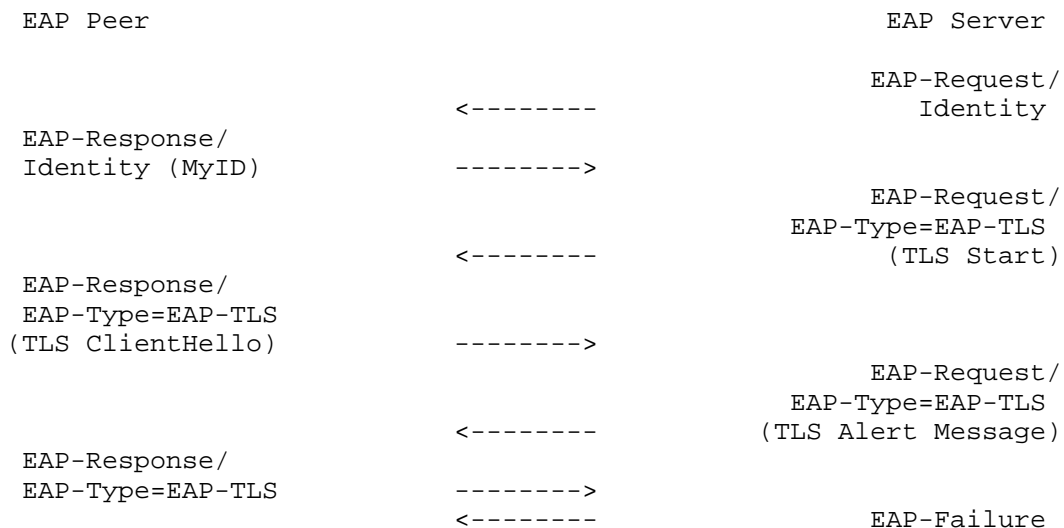


Figure 4: EAP-TLS server rejection of ClientHello

In the case where server authentication is unsuccessful, the conversation will appear as shown in Figure 5.

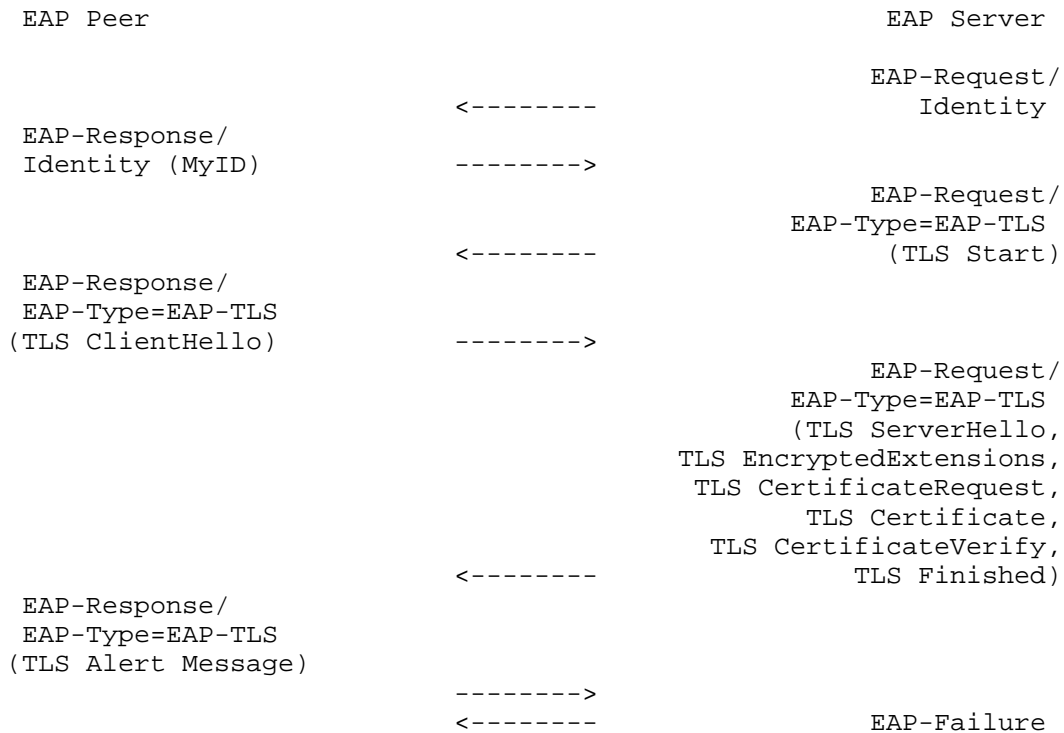


Figure 5: EAP-TLS unsuccessful server authentication

In the case where the server authenticates to the peer successfully, but the peer fails to authenticate to the server, the conversation will appear as shown in Figure 6.

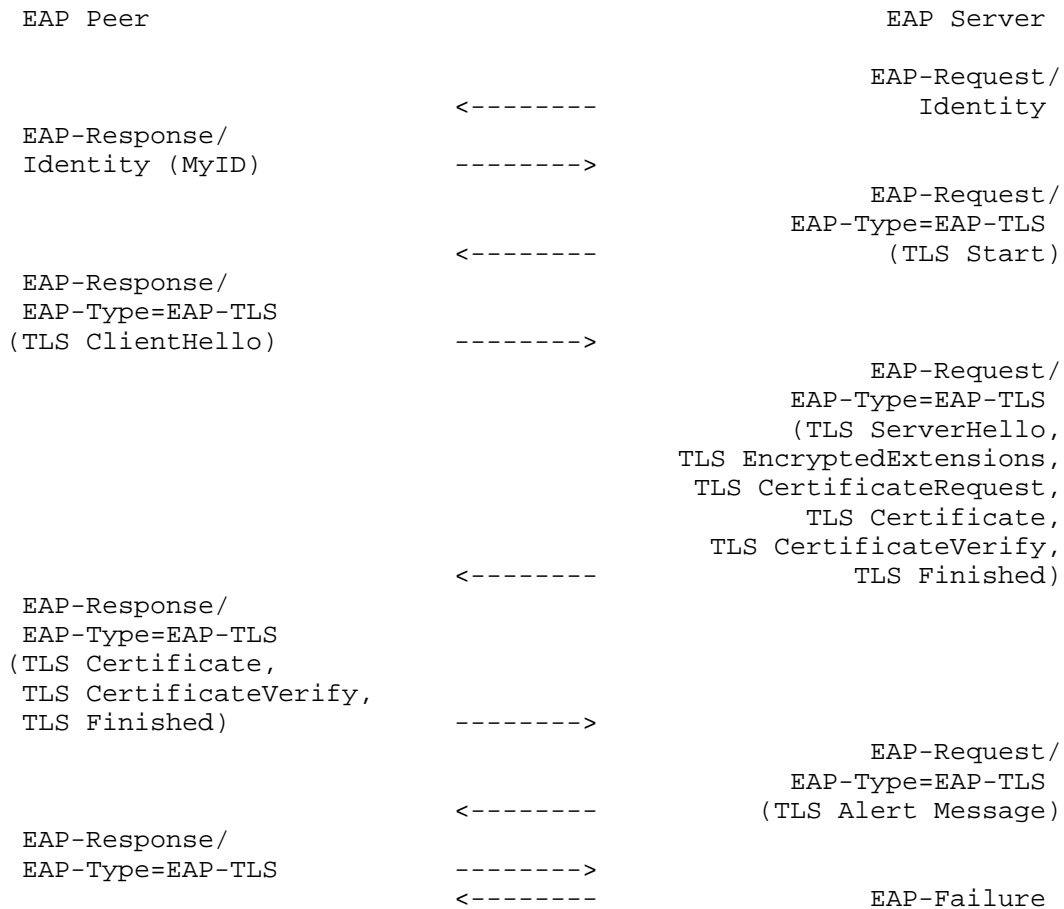


Figure 6: EAP-TLS unsuccessful client authentication

## 2.1.4. Privacy

TLS 1.3 significantly increases privacy when compared to earlier version of TLS by forbidding cipher suites without confidentiality and encrypting large parts of the TLS handshake including the certificate messages.

EAP-TLS peer and server implementations supporting TLS 1.3 or higher MUST support anonymous NAIs (Network Access Identifiers) (Section 2.4 in [RFC7542]) and the client MUST confidentiality protect its identity (e.g. using Anonymous NAIs) when the EAP-TLS server is known to support TLS 1.3 or higher.



As the certificate messages in TLS 1.3 are encrypted, there is no need to send an empty `certificate_list` or perform a second handshake (as needed by EAP-TLS when with earlier versions of TLS). When EAP-TLS is used with TLS version 1.3 or higher the EAP-TLS peer and EAP-TLS server SHALL follow the processing specified by the used version of TLS. For TLS 1.3 this means that the EAP-TLS peer only sends an empty `certificate_list` if it does not have an appropriate certificate to send and the EAP-TLS server MAY treat an empty `certificate_list` as a terminal condition.

When EAP-TLS is used with TLS 1.3 and privacy, no extra round-trips are added and the message flow looks just like a normal message flow with the only difference that an anonymous NAI is used. In the case where EAP-TLS with mutual authentication and privacy is successful, the conversation will appear as shown in Figure 7.

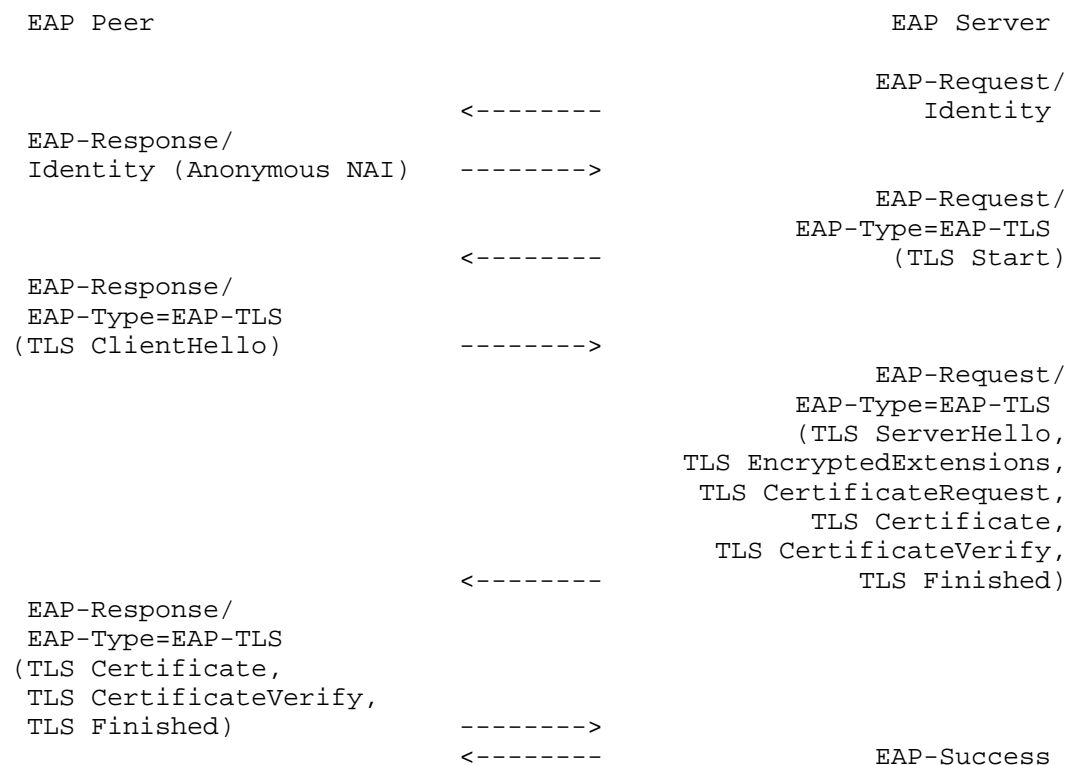


Figure 7: EAP-TLS privacy

#### 2.1.5. Fragmentation

Including ContentType and ProtocolVersion a single TLS record may be up to 16387 octets in length. Some EAP implementations and access networks may limit the number of EAP packet exchanges that can be handled. To avoid fragmentation, it is RECOMMENDED to keep the sizes of client, server, and trust anchor certificates small and the length of the certificate chains short. In addition, it is RECOMMENDED to use mechanisms that reduce the sizes of Certificate messages.

While Elliptic Curve Cryptography (ECC) was optional for earlier version of TLS, TLS 1.3 mandates support of ECC (see Section 9 of [I-D.ietf-tls-tls13]). To avoid fragmentation, the use of ECC in certificates, signature algorithms, and groups are RECOMMENDED when using EAP-TLS with TLS 1.3 or higher. At a 128-bit security level, this reduces public key sizes from 384 bytes (RSA and DHE) to 32 bytes (ECDHE) and signatures from 384 bytes (RSA) to 64 bytes (ECDSA and EdDSA). An EAP-TLS deployment MAY further reduce the certificate sizes by limiting the number of Subject Alternative Names.

Endpoints SHOULD reduce the sizes of Certificate messages by omitting certificates that the other endpoint is known to possess. When using TLS 1.3, all certificates that specifies a trust anchor may be omitted (see Section 4.4.2 of [I-D.ietf-tls-tls13]). When using TLS 1.2 or earlier, only the self-signed certificate that specifies the root certificate authority may be omitted (see Section 7.4.2 of [RFC5246]). EAP-TLS peers and servers SHOULD support and use the Cached Information Extension as specified in [RFC7924]. EAP-TLS peers and servers MAY use other extensions for reducing the sizes of Certificate messages, e.g. certificate compression [I-D.ietf-tls-certificate-compression].

#### 2.2. Identity Verification

No updates to [RFC5216].

#### 2.3. Key Hierarchy

TLS 1.3 replaces the TLS pseudorandom function (PRF) used in earlier versions of TLS with HKDF and completely changes the Key Schedule. The key hierarchies shown in Section 2.3 of [RFC5216] are therefore not correct when EAP-TLS is used with TLS version 1.3 or higher. For TLS 1.3 the key schedule is described in Section 7.1 of [I-D.ietf-tls-tls13].

When EAP-TLS is used with TLS version 1.3 or higher the Key\_Material, IV, and Session-Id SHALL be derived from the exporter\_master\_secret

using the TLS exporter interface [RFC5705] (for TLS 1.3 this is defined in Section 7.5 of [I-D.ietf-tls-tls13]).

```
Key_Material = TLS-Exporter("EXPORTER_EAP_TLS_Key_Material", "", 128)
IV           = TLS-Exporter("EXPORTER_EAP_TLS_IV", "", 64)
Session-Id   = TLS-Exporter("EXPORTER_EAP_TLS_Session-Id", "", 64)
```

By using the TLS exporter, EAP-TLS can use any TLS 1.3 implementation without having to extract the Master Secret, ClientHello.random, and ServerHello.random in a non-standard way.

All other parameters such as MSK and EMSK are derived as specified in EAP-TLS [RFC5216], Section 2.3. The use of these keys is specific to the lower layer, as described [RFC5247].

#### 2.4. Parameter Negotiation and Compliance Requirements

TLS 1.3 cipher suites are defined differently than in earlier versions of TLS (see Section B.4 of [I-D.ietf-tls-tls13]), and the cipher suites discussed in Section 2.4 of [RFC5216] can therefore not be used when EAP-TLS is used with TLS version 1.3 or higher. The requirements on protocol version and compression given in Section 2.4 of [RFC5216] still apply.

When EAP-TLS is used with TLS version 1.3 or higher, the EAP-TLS peers and servers MUST comply with the requirements for the TLS version used. For TLS 1.3 the compliance requirements are defined in Section 9 of [I-D.ietf-tls-tls13].

#### 3. Detailed Description of the EAP-TLS Protocol

No updates to [RFC5216].

#### 4. IANA considerations

This section provides guidance to the Internet Assigned Numbers Authority (IANA) regarding registration of values related to the EAP-TLS 1.3 protocol in accordance with [RFC8126].

This memo requires IANA to add the following labels to the TLS Exporter Label Registry defined by [RFC5705]. These labels are used in derivation of Key\_Material, IV and Session-Id as defined in Section 2.3:

- o "EXPORTER\_EAP\_TLS\_Key\_Material"
- o "EXPORTER\_EAP\_TLS\_IV"

- o "EXPORTER\_EAP\_TLS\_Session-Id"

## 5. Security Considerations

### 5.1. Security Claims

Using EAP-TLS with TLS 1.3 does not change the security claims for EAP-TLS as given in Section 4.1 of [RFC5216]. However, it strengthens several of the claims as described in the following updates to the notes given in Section 4.1 of [RFC5216].

[2] Confidentiality: The TLS 1.3 handshake offers much better confidentiality than earlier versions of TLS by mandating cipher suites with confidentiality and encrypting certificates and some of the extensions, see [I-D.ietf-tls-tls13]. When using EAP-TLS with TLS 1.3, the use of privacy does not cause any additional round-trips.

[3] Key strength: TLS 1.3 forbids all algorithms with known weaknesses including 3DES, CBC mode, RC4, SHA-1, and MD5. TLS 1.3 only supports cryptographic algorithms offering at least 112-bit security, see [I-D.ietf-tls-tls13].

[4] Cryptographic Negotiation: TLS 1.3 increases the number of cryptographic parameters that are negotiated in the handshake. When EAP-TLS is used with TLS 1.3, EAP-TLS inherits the cryptographic negotiation of AEAD algorithm, HKDF hash algorithm, key exchange groups, and signature algorithm, see Section 4.1.1 of [I-D.ietf-tls-tls13].

### 5.2. Peer and Server Identities

No updates to [RFC5216].

### 5.3. Certificate Validation

No updates to [RFC5216].

### 5.4. Certificate Revocation

The OCSP status handling in TLS 1.3 is different from earlier versions of TLS, see Section 4.4.2.1 of [I-D.ietf-tls-tls13]. In TLS 1.3 the OCSP information is carried in the CertificateEntry containing the associated certificate instead of a separate CertificateStatus message as in [RFC4366]. This enables sending OCSP information for all certificates in the certificate chain.

EAP-TLS peers and servers supporting TLS 1.3 SHOULD support Certificate Status Requests (OCSP stapling) as specified in [RFC6066] and Section 4.4.2.1 of [I-D.ietf-tls-tls13]. The use of Certificate Status Requests to determine the current status of the EAP server's certificate is RECOMMENDED.

## 5.5. Packet Modification Attacks

No updates to [RFC5216].

## 6. References

### 6.1. Normative References

- [I-D.ietf-tls-tls13]  
Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-28 (work in progress), March 2018.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <<https://www.rfc-editor.org/info/rfc3748>>.
- [RFC5216] Simon, D., Aboba, B., and R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, DOI 10.17487/RFC5216, March 2008, <<https://www.rfc-editor.org/info/rfc5216>>.
- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <<https://www.rfc-editor.org/info/rfc5280>>.
- [RFC5705] Rescorla, E., "Keying Material Exporters for Transport Layer Security (TLS)", RFC 5705, DOI 10.17487/RFC5705, March 2010, <<https://www.rfc-editor.org/info/rfc5705>>.
- [RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <<https://www.rfc-editor.org/info/rfc6066>>.

- [RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, DOI 10.17487/RFC6960, June 2013, <<https://www.rfc-editor.org/info/rfc6960>>.
- [RFC7542] DeKok, A., "The Network Access Identifier", RFC 7542, DOI 10.17487/RFC7542, May 2015, <<https://www.rfc-editor.org/info/rfc7542>>.
- [RFC7924] Santesson, S. and H. Tschofenig, "Transport Layer Security (TLS) Cached Information Extension", RFC 7924, DOI 10.17487/RFC7924, July 2016, <<https://www.rfc-editor.org/info/rfc7924>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.

## 6.2. Informative references

- [I-D.ietf-tls-certificate-compression]  
Ghedini, A. and V. Vasiliev, "Transport Layer Security (TLS) Certificate Compression", draft-ietf-tls-certificate-compression-03 (work in progress), April 2018.
- [IEEE-802.11]  
Institute of Electrical and Electronics Engineers, "IEEE Standard for Information technology--Telecommunications and information exchange between systems Local and metropolitan area networks--Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012) , December 2016.
- [IEEE-802.1X]  
Institute of Electrical and Electronics Engineers, "IEEE Standard for Local and metropolitan area networks -- Port-Based Network Access Control", IEEE Standard 802.1X-2010 , February 2010.
- [MulleFire]  
MulleFire, "MulleFire Release 1.0.1 specification", 2017.
- [RFC2246] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, DOI 10.17487/RFC2246, January 1999, <<https://www.rfc-editor.org/info/rfc2246>>.

- [RFC2560] Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 2560, DOI 10.17487/RFC2560, June 1999, <<https://www.rfc-editor.org/info/rfc2560>>.
- [RFC3280] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, DOI 10.17487/RFC3280, April 2002, <<https://www.rfc-editor.org/info/rfc3280>>.
- [RFC4282] Aboba, B., Beadles, M., Arkko, J., and P. Eronen, "The Network Access Identifier", RFC 4282, DOI 10.17487/RFC4282, December 2005, <<https://www.rfc-editor.org/info/rfc4282>>.
- [RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, DOI 10.17487/RFC4346, April 2006, <<https://www.rfc-editor.org/info/rfc4346>>.
- [RFC4366] Blake-Wilson, S., Nystrom, M., Hopwood, D., Mikkelsen, J., and T. Wright, "Transport Layer Security (TLS) Extensions", RFC 4366, DOI 10.17487/RFC4366, April 2006, <<https://www.rfc-editor.org/info/rfc4366>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.
- [RFC5247] Aboba, B., Simon, D., and P. Eronen, "Extensible Authentication Protocol (EAP) Key Management Framework", RFC 5247, DOI 10.17487/RFC5247, August 2008, <<https://www.rfc-editor.org/info/rfc5247>>.
- [TS.33.501] 3GPP, "Security architecture and procedures for 5G System", 3GPP TS 33.501 0.7.1, February 2018.

#### Appendix A. Updated references

All the following references in [RFC5216] are updated as specified below when EAP-TLS is used with TLS 1.3 or higher.

All references to [RFC2560] are updated with [RFC6960].

All references to [RFC3280] are updated with [RFC5280].

All references to [RFC4282] are updated with [RFC7542].

#### Appendix B. Acknowledgments

The authors want to thank Alan DeKok, Ari Keraenen, Bernard Aboba, Jari Arkko, and Vesa Torvinen for comments and suggestions on the draft.

#### Authors' Addresses

John Mattsson  
Ericsson  
164 40 Stockholm  
Sweden

Email: john.mattsson@ericsson.com

Mohit Sethi  
Ericsson  
Jorvas 02420  
Finland

Email: mohit@piuha.net



Network Working Group  
Internet-Draft  
Obsoletes: 5448 (if approved)  
Updates: 4187 (if approved)  
Intended status: Informational  
Expires: January 3, 2019

J. Arkko  
V. Lehtovirta  
V. Torvinen  
Ericsson  
P. Eronen  
Nokia  
July 2, 2018

Improved Extensible Authentication Protocol Method for 3rd Generation  
Authentication and Key Agreement (EAP-AKA')  
draft-ietf-emu-rfc5448bis-01

## Abstract

This specification defines a new EAP method, EAP-AKA', a small revision of the EAP-AKA method. The change is a new key derivation function that binds the keys derived within the method to the name of the access network. The new key derivation mechanism has been defined in the 3rd Generation Partnership Project (3GPP). This specification allows its use in EAP in an interoperable manner. In addition, EAP-AKA' employs SHA-256 instead of SHA-1.

This specification also updates RFC 4187 EAP-AKA to prevent bidding down attacks from EAP-AKA'.

This version of the EAP-AKA' specification updates a reference to constructing one field in the protocol, so that EAP-AKA' becomes compatible with 5G deployments as well.

## Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 3, 2019.

## Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Requirements Language . . . . .	5
3. EAP-AKA' . . . . .	5
3.1. AT_KDF_INPUT . . . . .	7
3.2. AT_KDF . . . . .	10
3.3. Key Generation . . . . .	12
3.4. Hash Functions . . . . .	14
3.4.1. PRF' . . . . .	14
3.4.2. AT_MAC . . . . .	14
3.4.3. AT_CHECKCODE . . . . .	14
4. Bidding Down Prevention for EAP-AKA . . . . .	15
5. Identifier Usage in 5G . . . . .	16
5.1. Key Derivation . . . . .	17
5.2. EAP Identity Response and EAP-AKA' AT_IDENTITY Attribute . . . . .	18
6. Exported Parameters . . . . .	19
7. Security Considerations . . . . .	19
7.1. Security Properties of Binding Network Names . . . . .	22
8. IANA Considerations . . . . .	23
8.1. Type Value . . . . .	23
8.2. Attribute Type Values . . . . .	23
8.3. Key Derivation Function Namespace . . . . .	23
9. Contributors . . . . .	24
10. Acknowledgments . . . . .	24
11. References . . . . .	24
11.1. Normative References . . . . .	24
11.2. Informative References . . . . .	26
Appendix A. Changes from RFC 5448 . . . . .	27
Appendix B. Changes from RFC 4187 to RFC 5448 . . . . .	27
Appendix C. Changes from Previous Version of This Draft . . . . .	28
Appendix D. Importance of Explicit Negotiation . . . . .	28
Appendix E. Test Vectors . . . . .	29

Authors' Addresses . . . . .	33
------------------------------	----

## 1. Introduction

This specification defines a new Extensible Authentication Protocol (EAP)[RFC3748] method, EAP-AKA', a small revision of the EAP-AKA method originally defined in [RFC4187]. What is new in EAP-AKA' is that it has a new key derivation function, specified in [TS-3GPP.33.402]. This function binds the keys derived within the method to the name of the access network. This limits the effects of compromised access network nodes and keys. This specification defines the EAP encapsulation for AKA when the new key derivation mechanism is in use.

3GPP has defined a number of applications for the revised AKA mechanism, some based on native encapsulation of AKA over 3GPP radio access networks and others based on the use of EAP.

For making the new key derivation mechanisms usable in EAP-AKA, additional protocol mechanisms are necessary. Given that RFC 4187 calls for the use of CK (the encryption key) and IK (the integrity key) from AKA, existing implementations continue to use these. Any change of the key derivation must be unambiguous to both sides in the protocol. That is, it must not be possible to accidentally connect old equipment to new equipment and get the key derivation wrong or attempt to use wrong keys without getting a proper error message. The change must also be secure against bidding down attacks that attempt to force the participants to use the least secure mechanism.

This specification therefore introduces a variant of the EAP-AKA method, called EAP-AKA'. This method can employ the derived keys CK' and IK' from the 3GPP specification and updates the used hash function to SHA-256 [FIPS.180-4]. But it is otherwise equivalent to RFC 4187. Given that a different EAP method type value is used for EAP-AKA and EAP-AKA', a mutually supported method may be negotiated using the standard mechanisms in EAP [RFC3748].

Note: Appendix D explains why it is important to be explicit about the change of semantics for the keys, and why other approaches would lead to severe interoperability problems.

This version of the EAP-AKA' specification obsoletes RFC 5448. The changes consist of three things:

- o Update the reference on how the Network Name field is constructed in the protocol. The update helps ensure that EAP-AKA' becomes compatible with 5G deployments as well. RFC 5448 referred to the

Release 8 version of [TS-3GPP.24.302] and this update points to the first 5G version, Release 15.

- o Specify how EAP and EAP-AKA' use identifiers in 5G, as additional identifiers are introduced, and for interoperability, it is important that implementations use the right ones.
- o Specify session identifiers and other exported parameters, as those were not specified in [RFC5448] despite requirements set forward in [RFC5247] to do so. Also, while [RFC5247] specified session identifiers for EAP-AKA, it only did so for the full authentication case, not for the case of fast re-authentication.

Arguably, the updates are small. For the first update, the 3GPP specification number for the updated calculation has not changed, only the version. But this reference is crucial in correct calculation of the keys resulting from running the EAP-AKA' method, so an update of the RFC with the newest version pointer may be warranted. As always, feedback is welcome on that point as well as on any other topic within this document.

Note: It is an open issue whether this update should refer to only the 5G version of the definition, or be explicit that any further update of that specification is something that EAP-AKA' implementations should take into account. Note that one should keep in mind that specification being automatically updated is different from implementations taking notice of new things.

The second update is needed to ensure that implementations use the correct identifiers in the context of 5G, as it introduces additional privacy-protected identifiers, and it is no longer clear which identifiers are used in EAP-AKA'.

The third update is necessary in order to fix a problem in previous RFCs.

It is an explicit non-goal of this draft to include any other technical modifications, addition of new features or other changes. The EAP-AKA' base protocol is stable and needs to stay that way. If there are any extensions or variants, those need to be proposed as standalone extensions or even as different authentication methods.

The rest of this specification is structured as follows. Section 3 defines the EAP-AKA' method. Section 4 adds support to EAP-AKA to prevent bidding down attacks from EAP-AKA'. Section 7 explains the security differences between EAP-AKA and EAP-AKA'. Section 8 describes the IANA considerations and Appendix A and Appendix B explains what updates to RFC 5448 AKA' and RFC 4187 EAP-AKA have been

made in this specification. Appendix D explains some of the design rationale for creating EAP-AKA'. Finally, Appendix E provides test vectors.

Editor's Note: The publication of this RFC depends on its normative references [TS-3GPP.24.302] and [TS-3GPP.33.501] reaching a stable status for Release 15, as indicated by 3GPP. This is expected to happen shortly. The RFC Editor should check with the 3GPP liaisons that this has happened. RFC Editor: Please delete this note upon publication of this specification as an RFC.

## 2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 3. EAP-AKA'

EAP-AKA' is a new EAP method that follows the EAP-AKA specification [RFC4187] in all respects except the following:

- o It uses the Type code 50, not 23 (which is used by EAP-AKA).
- o It carries the AT\_KDF\_INPUT attribute, as defined in Section 3.1, to ensure that both the peer and server know the name of the access network.
- o It supports key derivation function negotiation via the AT\_KDF attribute (Section 3.2) to allow for future extensions.
- o It calculates keys as defined in Section 3.3, not as defined in EAP-AKA.
- o It employs SHA-256, not SHA-1 [FIPS.180-4] (Section 3.4).

Figure 1 shows an example of the authentication process. Each message AKA'-Challenge and so on represents the corresponding message from EAP-AKA, but with EAP-AKA' Type code. The definition of these messages, along with the definition of attributes AT\_RANDOM, AT\_AUTN, AT\_MAC, and AT\_RES can be found in [RFC4187].

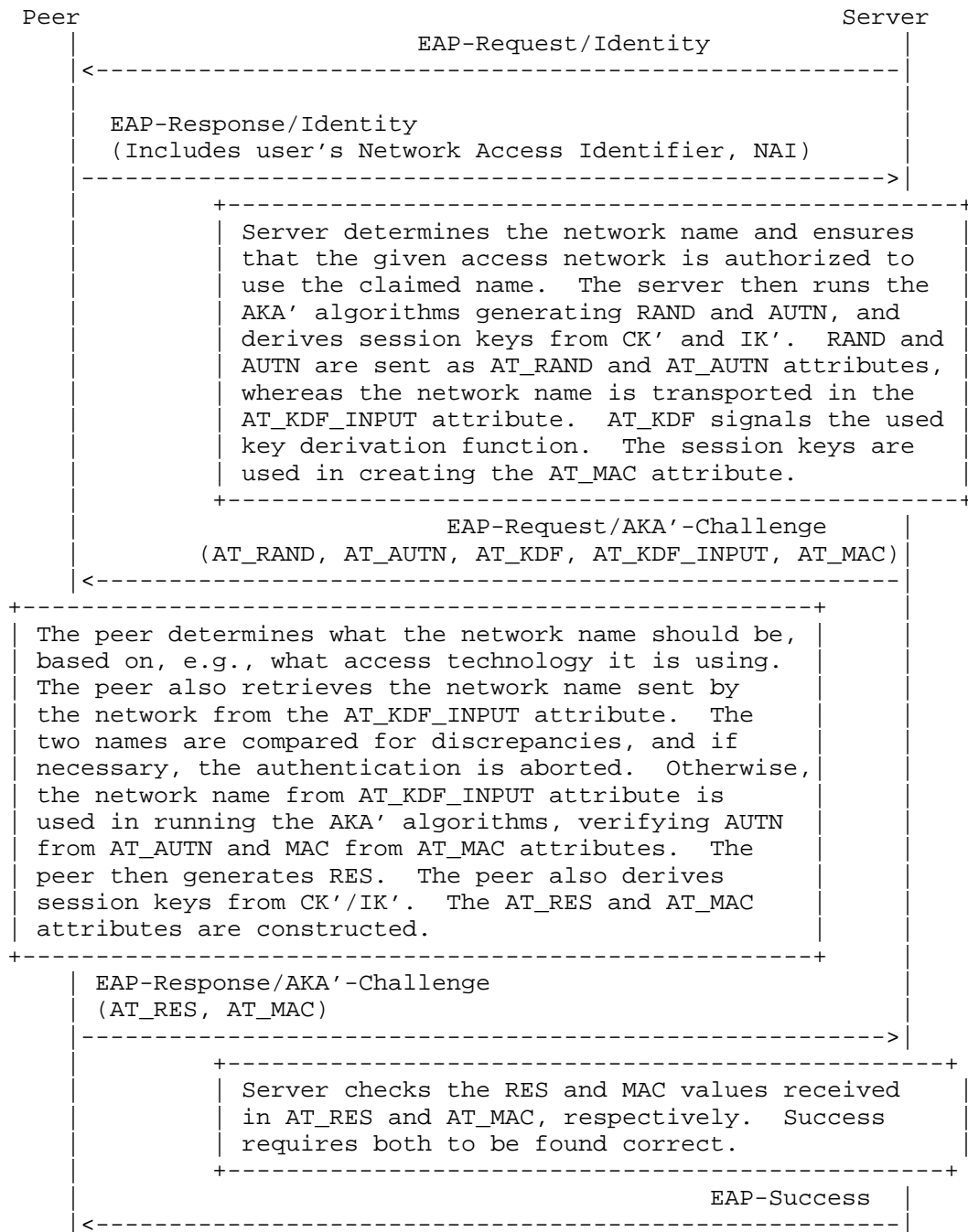


Figure 1: EAP-AKA' Authentication Process

EAP-AKA' can operate on the same credentials as EAP-AKA and employ the same identities. However, EAP-AKA' employs different leading characters than EAP-AKA for the conventions given in Section 4.1.1 of [RFC4187] for International Mobile Subscriber Identifier (IMSI) based usernames. EAP-AKA' MUST use the leading character "6" (ASCII 36 hexadecimal) instead of "0" for IMSI-based permanent usernames. All other usage and processing of the leading characters, usernames, and identities is as defined by EAP-AKA [RFC4187]. For instance, the pseudonym and fast re-authentication usernames need to be constructed so that the server can recognize them. As an example, a pseudonym could begin with a leading "7" character (ASCII 37 hexadecimal) and a fast re-authentication username could begin with "8" (ASCII 38 hexadecimal). Note that a server that implements only EAP-AKA may not recognize these leading characters. According to Section 4.1.4 of [RFC4187], such a server will re-request the identity via the EAP-Request/AKA-Identity message, making obvious to the peer that EAP-AKA and associated identity are expected.

### 3.1. AT\_KDF\_INPUT

The format of the AT\_KDF\_INPUT 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	2	3	4	5	6	7	8	9
AT_KDF_INPUT										Length										Actual Network Name Length																			
										Network Name																													

The fields are as follows:

AT\_KDF\_INPUT

This is set to 23.

Length

The length of the attribute, calculated as defined in [RFC4187], Section 8.1.

Actual Network Name Length

This is a 2 byte actual length field, needed due to the requirement that the previous field is expressed in multiples of 4 bytes per the usual EAP-AKA rules. The Actual Network Name Length field provides the length of the network name in bytes.

#### Network Name

This field contains the network name of the access network for which the authentication is being performed. The name does not include any terminating null characters. Because the length of the entire attribute must be a multiple of 4 bytes, the sender pads the name with 1, 2, or 3 bytes of all zero bits when necessary.

Only the server sends the AT\_KDF\_INPUT attribute. The value is sent as specified in [TS-3GPP.24.302] for non-3GPP access networks, and as specified in [TS-3GPP.33.501] for 5G access networks. Per [TS-3GPP.33.402], the server always verifies the authorization of a given access network to use a particular name before sending it to the peer over EAP-AKA'. The value of the AT\_KDF\_INPUT attribute from the server MUST be non-empty. If it is empty, the peer behaves as if AUTN had been incorrect and authentication fails. See Section 3 and Figure 3 of [RFC4187] for an overview of how authentication failures are handled.

Note: Currently, [TS-3GPP.24.302] or [TS-3GPP.33.501] specify separate values. The former specifies what is called "Access Network ID" and the latter specifies what is called "Serving Network Name". However, from an EAP-AKA' perspective both occupy the same field, and need to be distinguishable from each other. Currently specified values are distinguishable, but it would be useful that this be specified explicitly in the 3GPP specifications.

In addition, the peer MAY check the received value against its own understanding of the network name. Upon detecting a discrepancy, the peer either warns the user and continues, or fails the authentication process. More specifically, the peer SHOULD have a configurable policy that it can follow under these circumstances. If the policy indicates that it can continue, the peer SHOULD log a warning message or display it to the user. If the peer chooses to proceed, it MUST use the network name as received in the AT\_KDF\_INPUT attribute. If the policy indicates that the authentication should fail, the peer behaves as if AUTN had been incorrect and authentication fails.

The Network Name field contains a UTF-8 string. This string MUST be constructed as specified in [TS-3GPP.24.302] for "Access Network Identity". The string is structured as fields separated by colons



(:). The algorithms and mechanisms to construct the identity string depend on the used access technology.

On the network side, the network name construction is a configuration issue in an access network and an authorization check in the authentication server. On the peer, the network name is constructed based on the local observations. For instance, the peer knows which access technology it is using on the link, it can see information in a link-layer beacon, and so on. The construction rules specify how this information maps to an access network name. Typically, the network name consists of the name of the access technology, or the name of the access technology followed by some operator identifier that was advertised in a link-layer beacon. In all cases, [TS-3GPP.24.302] is the normative specification for the construction in both the network and peer side. If the peer policy allows running EAP-AKA' over an access technology for which that specification does not provide network name construction rules, the peer SHOULD rely only on the information from the AT\_KDF\_INPUT attribute and not perform a comparison.

If a comparison of the locally determined network name and the one received over EAP-AKA' is performed on the peer, it MUST be done as follows. First, each name is broken down to the fields separated by colons. If one of the names has more colons and fields than the other one, the additional fields are ignored. The remaining sequences of fields are compared, and they match only if they are equal character by character. This algorithm allows a prefix match where the peer would be able to match "", "FOO", and "FOO:BAR" against the value "FOO:BAR" received from the server. This capability is important in order to allow possible updates to the specifications that dictate how the network names are constructed. For instance, if a peer knows that it is running on access technology "FOO", it can use the string "FOO" even if the server uses an additional, more accurate description, e.g., "FOO:BAR", that contains more information.

The allocation procedures in [TS-3GPP.24.302] ensure that conflicts potentially arising from using the same name in different types of networks are avoided. The specification also has detailed rules about how a client can determine these based on information available to the client, such as the type of protocol used to attach to the network, beacons sent out by the network, and so on. Information that the client cannot directly observe (such as the type or version of the home network) is not used by this algorithm.

The AT\_KDF\_INPUT attribute MUST be sent and processed as explained above when AT\_KDF attribute has the value 1. Future definitions of

new AT\_KDF values MUST define how this attribute is sent and processed.

### 3.2. AT\_KDF

AT\_KDF is an attribute that the server uses to reference a specific key derivation function. It offers a negotiation capability that can be useful for future evolution of the key derivation functions.

The format of the AT\_KDF 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	2	3	4	5	6	7	8	9
AT_KDF										Length										Key Derivation Function																			

The fields are as follows:

#### AT\_KDF

This is set to 24.

#### Length

The length of the attribute, MUST be set to 1.

#### Key Derivation Function

An enumerated value representing the key derivation function that the server (or peer) wishes to use. Value 1 represents the default key derivation function for EAP-AKA', i.e., employing CK' and IK' as defined in Section 3.3.

Servers MUST send one or more AT\_KDF attributes in the EAP-Request/ AKA'-Challenge message. These attributes represent the desired functions ordered by preference, the most preferred function being the first attribute.

Upon receiving a set of these attributes, if the peer supports and is willing to use the key derivation function indicated by the first attribute, the function is taken into use without any further negotiation. However, if the peer does not support this function or is unwilling to use it, it does not process the received EAP-Request/ AKA'-Challenge in any way except by responding with the EAP-Response/ AKA'-Challenge message that contains only one attribute, AT\_KDF with the value set to the selected alternative. If there is no suitable

alternative, the peer behaves as if AUTN had been incorrect and authentication fails (see Figure 3 of [RFC4187]). The peer fails the authentication also if there are any duplicate values within the list of AT\_KDF attributes (except where the duplication is due to a request to change the key derivation function; see below for further information).

Upon receiving an EAP-Response/AKA'-Challenge with AT\_KDF from the peer, the server checks that the suggested AT\_KDF value was one of the alternatives in its offer. The first AT\_KDF value in the message from the server is not a valid alternative. If the peer has replied with the first AT\_KDF value, the server behaves as if AT\_MAC of the response had been incorrect and fails the authentication. For an overview of the failed authentication process in the server side, see Section 3 and Figure 2 of [RFC4187]. Otherwise, the server re-sends the EAP-Response/AKA'-Challenge message, but adds the selected alternative to the beginning of the list of AT\_KDF attributes and retains the entire list following it. Note that this means that the selected alternative appears twice in the set of AT\_KDF values. Responding to the peer's request to change the key derivation function is the only legal situation where such duplication may occur.

When the peer receives the new EAP-Request/AKA'-Challenge message, it MUST check that the requested change, and only the requested change, occurred in the list of AT\_KDF attributes. If so, it continues with processing the received EAP-Request/AKA'-Challenge as specified in [RFC4187] and Section 3.1 of this document. If not, it behaves as if AT\_MAC had been incorrect and fails the authentication. If the peer receives multiple EAP-Request/AKA'-Challenge messages with differing AT\_KDF attributes without having requested negotiation, the peer MUST behave as if AT\_MAC had been incorrect and fail the authentication.

Note that the peer may also request sequence number resynchronization [RFC4187]. This happens after AT\_KDF negotiation has already completed. An AKA'-Synchronization-Failure message is sent as a response to the newly received EAP-Request/AKA'-Challenge (the last message of the AT\_KDF negotiation). The AKA'-Synchronization-Failure message MUST contain the AUTS parameter as specified in [RFC4187] and a copy the AT\_KDF attributes as they appeared in the last message of the AT\_KDF negotiation. If the AT\_KDF attributes are found to differ from their earlier values, the peer and server MUST behave as if AT\_MAC had been incorrect and fail the authentication.

### 3.3. Key Generation

Both the peer and server MUST derive the keys as follows.

AT\_KDF set to 1

In this case, MK is derived and used as follows:

```
MK = PRF'(IK'|CK',"EAP-AKA'|Identity)
K_encr = MK[0..127]
K_aut  = MK[128..383]
K_re   = MK[384..639]
MSK    = MK[640..1151]
EMSK   = MK[1152..1663]
```

Here [n..m] denotes the substring from bit n to m. PRF' is a new pseudo-random function specified in Section 3.4. The first 1664 bits from its output are used for K\_encr (encryption key, 128 bits), K\_aut (authentication key, 256 bits), K\_re (re-authentication key, 256 bits), MSK (Master Session Key, 512 bits), and EMSK (Extended Master Session Key, 512 bits). These keys are used by the subsequent EAP-AKA' process. K\_encr is used by the AT\_ENCR\_DATA attribute, and K\_aut by the AT\_MAC attribute. K\_re is used later in this section. MSK and EMSK are outputs from a successful EAP method run [RFC3748].

IK' and CK' are derived as specified in [TS-3GPP.33.402]. The functions that derive IK' and CK' take the following parameters: CK and IK produced by the AKA algorithm, and value of the Network Name field comes from the AT\_KDF\_INPUT attribute (without length or padding) .

The value "EAP-AKA'" is an eight-characters-long ASCII string. It is used as is, without any trailing NUL characters.

Identity is the peer identity as specified in Section 7 of [RFC4187].

When the server creates an AKA challenge and corresponding AUTN, CK, CK', IK, and IK' values, it MUST set the Authentication Management Field (AMF) separation bit to 1 in the AKA algorithm [TS-3GPP.33.102]. Similarly, the peer MUST check that the AMF separation bit is set to 1. If the bit is not set to 1, the peer behaves as if the AUTN had been incorrect and fails the authentication.

On fast re-authentication, the following keys are calculated:

```
MK = PRF'(K_re, "EAP-AKA' re-auth" | Identity | counter | NONCE_S)
MSK = MK[0..511]
EMSK = MK[512..1023]
```

MSK and EMSK are the resulting 512-bit keys, taking the first 1024 bits from the result of PRF'. Note that K\_encr and K\_aut are not re-derived on fast re-authentication. K\_re is the re-authentication key from the preceding full authentication and stays unchanged over any fast re-authentication(s) that may happen based on it. The value "EAP-AKA' re-auth" is a sixteen-characters-long ASCII string, again represented without any trailing NUL characters. Identity is the fast re-authentication identity, counter is the value from the AT\_COUNTER attribute, NONCE\_S is the nonce value from the AT\_NONCE\_S attribute, all as specified in Section 7 of [RFC4187]. To prevent the use of compromised keys in other places, it is forbidden to change the network name when going from the full to the fast re-authentication process. The peer SHOULD NOT attempt fast re-authentication when it knows that the network name in the current access network is different from the one in the initial, full authentication. Upon seeing a re-authentication request with a changed network name, the server SHOULD behave as if the re-authentication identifier had been unrecognized, and fall back to full authentication. The server observes the change in the name by comparing where the fast re-authentication and full authentication EAP transactions were received at the Authentication, Authorization, and Accounting (AAA) protocol level.

AT\_KDF has any other value

Future variations of key derivation functions may be defined, and they will be represented by new values of AT\_KDF. If the peer does not recognize the value, it cannot calculate the keys and behaves as explained in Section 3.2.

AT\_KDF is missing

The peer behaves as if the AUTN had been incorrect and MUST fail the authentication.

If the peer supports a given key derivation function but is unwilling to perform it for policy reasons, it refuses to calculate the keys and behaves as explained in Section 3.2.

### 3.4. Hash Functions

EAP-AKA' uses SHA-256, not SHA-1 (see [FIPS.180-4]) as in EAP-AKA. This requires a change to the pseudo-random function (PRF) as well as the AT\_MAC and AT\_CHECKCODE attributes.

#### 3.4.1. PRF'

The PRF' construction is the same one IKEv2 uses (see Section 2.13 of [RFC4306]). The function takes two arguments. K is a 256-bit value and S is an octet string of arbitrary length. PRF' is defined as follows:

$$\text{PRF}'(K, S) = T1 \mid T2 \mid T3 \mid T4 \mid \dots$$

where:

T1 = HMAC-SHA-256 (K, S | 0x01)  
T2 = HMAC-SHA-256 (K, T1 | S | 0x02)  
T3 = HMAC-SHA-256 (K, T2 | S | 0x03)  
T4 = HMAC-SHA-256 (K, T3 | S | 0x04)  
...

PRF' produces as many bits of output as is needed. HMAC-SHA-256 is the application of HMAC [RFC2104] to SHA-256.

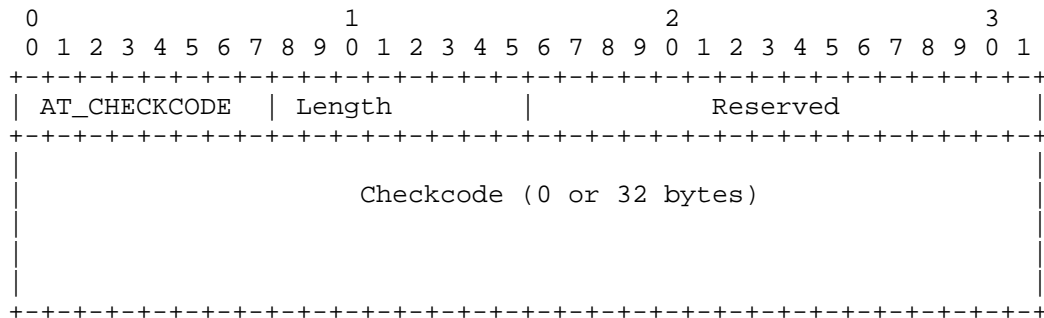
#### 3.4.2. AT\_MAC

When used within EAP-AKA', the AT\_MAC attribute is changed as follows. The MAC algorithm is HMAC-SHA-256-128, a keyed hash value. The HMAC-SHA-256-128 value is obtained from the 32-byte HMAC-SHA-256 value by truncating the output to the first 16 bytes. Hence, the length of the MAC is 16 bytes.

Otherwise, the use of AT\_MAC in EAP-AKA' follows Section 10.15 of [RFC4187].

#### 3.4.3. AT\_CHECKCODE

When used within EAP-AKA', the AT\_CHECKCODE attribute is changed as follows. First, a 32-byte value is needed to accommodate a 256-bit hash output:



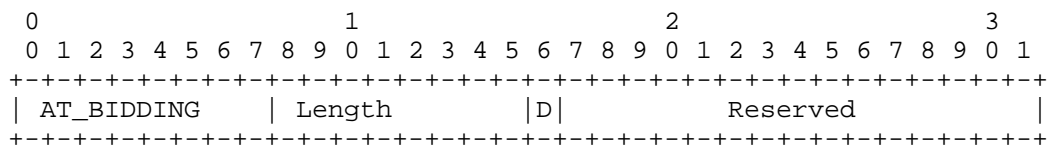
Second, the checkcode is a hash value, calculated with SHA-256 [FIPS.180-4], over the data specified in Section 10.13 of [RFC4187].

#### 4. Bidding Down Prevention for EAP-AKA

As discussed in [RFC3748], negotiation of methods within EAP is insecure. That is, a man-in-the-middle attacker may force the endpoints to use a method that is not the strongest that they both support. This is a problem, as we expect EAP-AKA and EAP-AKA' to be negotiated via EAP.

In order to prevent such attacks, this RFC specifies a new mechanism for EAP-AKA that allows the endpoints to securely discover the capabilities of each other. This mechanism comes in the form of the AT\_BIDDING attribute. This allows both endpoints to communicate their desire and support for EAP-AKA' when exchanging EAP-AKA messages. This attribute is not included in EAP-AKA' messages as defined in this RFC. It is only included in EAP-AKA messages. This is based on the assumption that EAP-AKA' is always preferable (see Section 7). If during the EAP-AKA authentication process it is discovered that both endpoints would have been able to use EAP-AKA', the authentication process SHOULD be aborted, as a bidding down attack may have happened.

The format of the AT\_BIDDING attribute is shown below.



The fields are as follows:

AT\_BIDDING

This is set to 136.

#### Length

The length of the attribute, MUST be set to 1.

#### D

This bit is set to 1 if the sender supports EAP-AKA', is willing to use it, and prefers it over EAP-AKA. Otherwise, it should be set to zero.

#### Reserved

This field MUST be set to zero when sent and ignored on receipt.

The server sends this attribute in the EAP-Request/AKA-Challenge message. If the peer supports EAP-AKA', it compares the received value to its own capabilities. If it turns out that both the server and peer would have been able to use EAP-AKA' and preferred it over EAP-AKA, the peer behaves as if AUTN had been incorrect and fails the authentication (see Figure 3 of [RFC4187]). A peer not supporting EAP-AKA' will simply ignore this attribute. In all cases, the attribute is protected by the integrity mechanisms of EAP-AKA, so it cannot be removed by a man-in-the-middle attacker.

Note that we assume (Section 7) that EAP-AKA' is always stronger than EAP-AKA. As a result, there is no need to prevent bidding "down" attacks in the other direction, i.e., attackers forcing the endpoints to use EAP-AKA'.

## 5. Identifier Usage in 5G

In EAP-AKA', the peer identity may be communicated to the server in one of three ways:

- o As a part of link layer establishment procedures, externally to EAP.
- o With the EAP-Response/Identity message in the beginning of the EAP exchange, but before the selection of EAP-AKA'.
- o 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 identity carried above may be a permanent identity or a pseudonym identity or fast re-authentication identity as defined in this RFC.

In networks where EAP is the only part handling such pseudonym or fast re-authentication identities, this usage is clear. However, 5G supports the concept of pseudonym or privacy identifiers, and it is important for interoperability that the right type of identifiers are used in the right place.

5G defines the Subscription Permanent Identifier (SUPI) and Subscription Concealed Identifier (SUCI) [TS-3GPP.23.501] [TS-3GPP.33.501]. SUPI is globally unique and allocated to each subscriber. However, it is only used internally in the 5G network, and is privacy sensitive. The SUCI is a privacy preserving identifier containing the concealed SUPI, using public key cryptography to encrypt the SUPI.

Given the choice between these two types of identifiers, two areas need further specification in EAP-AKA' to ensure that different implementations understand each other and stay interoperable:

- o Where identifiers are used within EAP-AKA' -- such as key derivation -- specify what values exactly should be used, to avoid ambiguity.
- o Where identifiers are carried within EAP-AKA' packets -- such as in the AT\_IDENTITY attribute -- specify which identifiers should be filled in.

In 5G, the normal mode of operation is that identifiers are only transmitted outside EAP. However, in a system involving terminals from many generations and several connectivity options via 5G and other mechanisms, implementations and the EAP-AKA' specification need to prepare for many different situations, including sometimes having to communicate identities within EAP.

The following sections clarify which identifiers are used and how.

### 5.1. Key Derivation

In EAP-AKA', the peer identity is used in the Section 3.3 key derivation formula.

If the AT\_KDF\_INPUT parameter contains the prefix "5G:", the AT\_KDF parameter has the value 1, and this authentication is not a fast re-

authentication, then the peer identity used in the key derivation MUST be the 5G SUPI for the peer. This rule applies to all full EAP-AKA' authentication processes, even if the peer sent some other identifier at a lower layer or as a response to an EAP Identity Request or if no identity was sent.

In all other cases, the following applies:

The identity used in the key derivation formula MUST be exactly the one sent in EAP-AKA' AT\_IDENTITY attribute, if one was sent, regardless of the kind of identity that it may have been. If no AT\_IDENTITY was sent, the identity MUST be the exactly the one sent in the generic EAP Identity exchange, if one was made. Again, the identity MUST be used exactly as sent.

If no identity was communicated inside EAP, then the identity is the one communicated outside EAP in link layer messaging.

In this case, the used identity MUST be the identity most recently communicated by the peer to the network, again regardless of what type of identity it may have been.

## 5.2. EAP Identity Response and EAP-AKA' AT\_IDENTITY Attribute

The EAP authentication option is only available in 5G when the new 5G core network is also in use. However, in other networks an EAP-AKA' peer may be connecting to other types of networks and existing equipment.

When the EAP peer is connecting to a 5G access network and uses the 5G core network signalling mechanisms, it can assume that the EAP server is in a 5G network. The EAP level identity exchanges are not generally used in this case, but if there is, the EAP peer SHOULD employ only the privacy preserving SUCI identifier within EAP (either in EAP Identity Response or EAP-AKA' AT\_IDENTITY attribute).

Similarly, if the peer is explicitly communicating through mechanisms developed for 5G to connect to 5G networks over WLAN, it MUST assume that the EAP server is in a 5G network, and again employ the SUCI within EAP.

Otherwise, the peer SHOULD employ IMSI, SUPI, or a NAI as it is configured to use.

## 6. Exported Parameters

The EAP-AKA' Session-Id is the concatenation of the EAP Type Code (50, one octet) with the contents of the RAND field from the AT\_RAND attribute, followed by the contents of the AUTN field in the AT\_AUTN attribute:

$$\text{Session-Id} = 50 \parallel \text{RAND} \parallel \text{AUTN}$$

When using fast re-authentication, the EAP-AKA' Session-Id is the concatenation of the EAP Type Code (50) with the contents of the NONCE\_S field from the AT\_NONCE\_S attribute, followed by the contents of the MAC field from the AT\_MAC attribute from EAP-Request/AKA-Reauthentication:

$$\text{Session-Id} = 50 \parallel \text{NONCE\_S} \parallel \text{MAC}$$

The Peer-Id is the contents of the Identity field from the AT\_IDENTITY attribute, using only the Actual Identity Length octets from the beginning. Note that the contents are used as they are transmitted, regardless of whether the transmitted identity was a permanent, pseudonym, or fast EAP re-authentication identity. The Server-Id is the null string (zero length).

## 7. Security Considerations

A summary of the security properties of EAP-AKA' follows. These properties are very similar to those in EAP-AKA. We assume that SHA-256 is at least as secure as SHA-1. This is called the SHA-256 assumption in the remainder of this section. Under this assumption, EAP-AKA' is at least as secure as EAP-AKA.

If the AT\_KDF attribute has value 1, then the security properties of EAP-AKA' are as follows:

Protected ciphersuite negotiation

EAP-AKA' has no ciphersuite negotiation mechanisms. It does have a negotiation mechanism for selecting the key derivation functions. This mechanism is secure against bidding down attacks. The negotiation mechanism allows changing the offered key derivation function, but the change is visible in the final EAP-Request/AKA'-Challenge message that the server sends to the peer. This message is authenticated via the AT\_MAC attribute, and carries both the chosen alternative and the initially offered list. The peer refuses to accept a change it did not initiate. As a result, both parties are aware that a change is being made and what the original offer was.

### Mutual authentication

Under the SHA-256 assumption, the properties of EAP-AKA' are at least as good as those of EAP-AKA in this respect. Refer to [RFC4187], Section 12 for further details.

### Integrity protection

Under the SHA-256 assumption, the properties of EAP-AKA' are at least as good (most likely better) as those of EAP-AKA in this respect. Refer to [RFC4187], Section 12 for further details. The only difference is that a stronger hash algorithm, SHA-256, is used instead of SHA-1.

### Replay protection

Under the SHA-256 assumption, the properties of EAP-AKA' are at least as good as those of EAP-AKA in this respect. Refer to [RFC4187], Section 12 for further details.

### Confidentiality

The properties of EAP-AKA' are exactly the same as those of EAP-AKA in this respect. Refer to [RFC4187], Section 12 for further details.

### Key derivation

EAP-AKA' supports key derivation with an effective key strength against brute force attacks equal to the minimum of the length of the derived keys and the length of the AKA base key, i.e., 128 bits or more. The key hierarchy is specified in Section 3.3.

The Transient EAP Keys used to protect EAP-AKA packets ( $K_{encr}$ ,  $K_{aut}$ ,  $K_{re}$ ), the MSK, and the EMSK are cryptographically separate. If we make the assumption that SHA-256 behaves as a pseudo-random function, an attacker is incapable of deriving any non-trivial information about any of these keys based on the other keys. An attacker also cannot calculate the pre-shared secret from  $IK$ ,  $CK$ ,  $IK'$ ,  $CK'$ ,  $K_{encr}$ ,  $K_{aut}$ ,  $K_{re}$ , MSK, or EMSK by any practically feasible means.

EAP-AKA' adds an additional layer of key derivation functions within itself to protect against the use of compromised keys. This is discussed further in Section 7.1.

EAP-AKA' uses a pseudo-random function modeled after the one used in IKEv2 [RFC4306] together with SHA-256.

#### Key strength

See above.

#### Dictionary attack resistance

Under the SHA-256 assumption, the properties of EAP-AKA' are at least as good as those of EAP-AKA in this respect. Refer to [RFC4187], Section 12 for further details.

#### Fast reconnect

Under the SHA-256 assumption, the properties of EAP-AKA' are at least as good as those of EAP-AKA in this respect. Refer to [RFC4187], Section 12 for further details. Note that implementations MUST prevent performing a fast reconnect across method types.

#### Cryptographic binding

Note that this term refers to a very specific form of binding, something that is performed between two layers of authentication. It is not the same as the binding to a particular network name. The properties of EAP-AKA' are exactly the same as those of EAP-AKA in this respect, i.e., as it is not a tunnel method, this property is not applicable to it. Refer to [RFC4187], Section 12 for further details.

#### Session independence

The properties of EAP-AKA' are exactly the same as those of EAP-AKA in this respect. Refer to [RFC4187], Section 12 for further details.

#### Fragmentation

The properties of EAP-AKA' are exactly the same as those of EAP-AKA in this respect. Refer to [RFC4187], Section 12 for further details.

#### Channel binding

EAP-AKA', like EAP-AKA, does not provide channel bindings as they're defined in [RFC3748] and [RFC5247]. New skippable attributes can be used to add channel binding support in the future, if required.

However, including the Network Name field in the AKA' algorithms (which are also used for other purposes than EAP-AKA') provides a form of cryptographic separation between different network names, which resembles channel bindings. However, the network name does not typically identify the EAP (pass-through) authenticator. See the following section for more discussion.

### 7.1. Security Properties of Binding Network Names

The ability of EAP-AKA' to bind the network name into the used keys provides some additional protection against key leakage to inappropriate parties. The keys used in the protocol are specific to a particular network name. If key leakage occurs due to an accident, access node compromise, or another attack, the leaked keys are only useful when providing access with that name. For instance, a malicious access point cannot claim to be network Y if it has stolen keys from network X. Obviously, if an access point is compromised, the malicious node can still represent the compromised node. As a result, neither EAP-AKA' nor any other extension can prevent such attacks; however, the binding to a particular name limits the attacker's choices, allows better tracking of attacks, makes it possible to identify compromised networks, and applies good cryptographic hygiene.

The server receives the EAP transaction from a given access network, and verifies that the claim from the access network corresponds to the name that this access network should be using. It becomes impossible for an access network to claim over AAA that it is another access network. In addition, if the peer checks that the information it has received locally over the network-access link layer matches with the information the server has given it via EAP-AKA', it becomes impossible for the access network to tell one story to the AAA network and another one to the peer. These checks prevent some "lying NAS" (Network Access Server) attacks. For instance, a roaming partner, R, might claim that it is the home network H in an effort to lure peers to connect to itself. Such an attack would be beneficial for the roaming partner if it can attract more users, and damaging for the users if their access costs in R are higher than those in other alternative networks, such as H.

Any attacker who gets hold of the keys CK and IK, produced by the AKA algorithm, can compute the keys CK' and IK' and, hence, the Master Key (MK) according to the rules in Section 3.3. The attacker could then act as a lying NAS. In 3GPP systems in general, the keys CK and IK have been distributed to, for instance, nodes in a visited access network where they may be vulnerable. In order to reduce this risk, the AKA algorithm MUST be computed with the AMF separation bit set to 1, and the peer MUST check that this is indeed the case whenever it

runs EAP-AKA'. Furthermore, [TS-3GPP.33.402] requires that no CK or IK keys computed in this way ever leave the home subscriber system.

The additional security benefits obtained from the binding depend obviously on the way names are assigned to different access networks. This is specified in [TS-3GPP.24.302]. See also [TS-3GPP.23.003]. Ideally, the names allow separating each different access technology, each different access network, and each different NAS within a domain. If this is not possible, the full benefits may not be achieved. For instance, if the names identify just an access technology, use of compromised keys in a different technology can be prevented, but it is not possible to prevent their use by other domains or devices using the same technology.

## 8. IANA Considerations

### 8.1. Type Value

EAP-AKA' has the EAP Type value 50 in the Extensible Authentication Protocol (EAP) Registry under Method Types. Per Section 6.2 of [RFC3748], this allocation can be made with Designated Expert and Specification Required.

### 8.2. Attribute Type Values

EAP-AKA' shares its attribute space and subtypes with EAP-SIM [RFC4186] and EAP-AKA [RFC4187]. No new registries are needed.

However, a new Attribute Type value (23) in the non-skipable range has been assigned for AT\_KDF\_INPUT (Section 3.1) in the EAP-AKA and EAP-SIM Parameters registry under Attribute Types.

Also, a new Attribute Type value (24) in the non-skipable range has been assigned for AT\_KDF (Section 3.2).

Finally, a new Attribute Type value (136) in the skipable range has been assigned for AT\_BIDDING (Section 4).

### 8.3. Key Derivation Function Namespace

IANA has also created a new namespace for EAP-AKA' AT\_KDF Key Derivation Function Values. This namespace exists under the EAP-AKA and EAP-SIM Parameters registry. The initial contents of this namespace are given below; new values can be created through the Specification Required policy [RFC8126].

Value	Description	Reference
-----	-----	-----
0	Reserved	[RFC 5448]
1	EAP-AKA' with CK'/IK'	[RFC 5448]
2-65535	Unassigned	

## 9. Contributors

The test vectors in Appendix C were provided by Yogendra Pal and Jouni Malinen, based on two independent implementations of this specification.

Jouni Malinen provided suggested text for Section 6.

## 10. Acknowledgments

The authors would like to thank Guenther Horn, Joe Salowey, Mats Naslund, Adrian Escott, Brian Rosenberg, Laksminath Dondeti, Ahmad Muhanna, Stefan Rommer, Miguel Garcia, Jan Kall, Ankur Agarwal, Jouni Malinen, Brian Weis, Russ Housley, Alfred Hoenes, Vesa Torvinen, Anand Palanigounder, and Mohit Sethi for their in-depth reviews and interesting discussions in this problem space.

## 11. References

### 11.1. Normative References

- [TS-3GPP.23.501]  
3GPP, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security architecture and procedures for 5G System; (Release 15)", 3GPP Technical Specification 23.501, December 2017.
- [TS-3GPP.24.302]  
3GPP, "3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; Access to the 3GPP Evolved Packet Core (EPC) via non-3GPP access networks; Stage 3; (Release 15)", 3GPP Draft Technical Specification 24.302, June 2018.
- [TS-3GPP.33.102]  
3GPP, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security architecture (Release 15)", 3GPP Draft Technical Specification 33.102, June 2018.



- [TS-3GPP.33.402] 3GPP, "3GPP System Architecture Evolution (SAE); Security aspects of non-3GPP accesses (Release 15)", 3GPP Draft Technical Specification 33.402, June 2018.
- [TS-3GPP.33.501] 3GPP, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security architecture and procedures for 5G System (Release 15)", 3GPP Draft Technical Specification 33.501, June 2018.
- [FIPS.180-4] National Institute of Standards and Technology, "Secure Hash Standard", FIPS PUB 180-4, August 2015, <<https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf>>.
- [RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <<https://www.rfc-editor.org/info/rfc2104>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <<https://www.rfc-editor.org/info/rfc3748>>.
- [RFC4187] Arkko, J. and H. Haverinen, "Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA)", RFC 4187, DOI 10.17487/RFC4187, January 2006, <<https://www.rfc-editor.org/info/rfc4187>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

## 11.2. Informative References

- [TS-3GPP.23.003]  
3GPP, "3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; Numbering, addressing and identification (Release 15)", 3GPP Draft Technical Specification 23.003, June 2018.
- [TS-3GPP.35.208]  
3GPP, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the MILENAGE Algorithm Set: An example algorithm set for the 3GPP authentication and key generation functions f1, f1\*, f2, f3, f4, f5 and f5\*; Document 4: Design Conformance Test Data (Release 14)", 3GPP Technical Specification 35.208, March 2017.
- [FIPS.180-1]  
National Institute of Standards and Technology, "Secure Hash Standard", FIPS PUB 180-1, April 1995, <<http://www.itl.nist.gov/fipspubs/fip180-1.htm>>.
- [FIPS.180-2]  
National Institute of Standards and Technology, "Secure Hash Standard", FIPS PUB 180-2, August 2002, <<http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf>>.
- [RFC4186] 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, DOI 10.17487/RFC4186, January 2006, <<https://www.rfc-editor.org/info/rfc4186>>.
- [RFC4284] Adrangi, F., Lortz, V., Bari, F., and P. Eronen, "Identity Selection Hints for the Extensible Authentication Protocol (EAP)", RFC 4284, DOI 10.17487/RFC4284, January 2006, <<https://www.rfc-editor.org/info/rfc4284>>.
- [RFC4306] Kaufman, C., Ed., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, DOI 10.17487/RFC4306, December 2005, <<https://www.rfc-editor.org/info/rfc4306>>.
- [RFC5113] Arkko, J., Aboba, B., Korhonen, J., Ed., and F. Bari, "Network Discovery and Selection Problem", RFC 5113, DOI 10.17487/RFC5113, January 2008, <<https://www.rfc-editor.org/info/rfc5113>>.

- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <<https://www.rfc-editor.org/info/rfc5226>>.
- [RFC5247] Aboba, B., Simon, D., and P. Eronen, "Extensible Authentication Protocol (EAP) Key Management Framework", RFC 5247, DOI 10.17487/RFC5247, August 2008, <<https://www.rfc-editor.org/info/rfc5247>>.
- [RFC5448] Arkko, J., Lehtovirta, V., and P. Eronen, "Improved Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA')", RFC 5448, DOI 10.17487/RFC5448, May 2009, <<https://www.rfc-editor.org/info/rfc5448>>.

#### Appendix A. Changes from RFC 5448

The changes consist first of all, referring to a newer version of [TS-3GPP.24.302]. The new version includes an updated definition of the Network Name field, to include 5G.

Secondly, identifier usage for 5G has been specified in Section 5.

Thirdly, exported parameters for EAP-AKA' have been defined in Section 6, as required by [RFC5247], including the definition of those parameters for both full authentication and fast re-authentication.

Finally, the references to [RFC2119], [RFC5226], [FIPS.180-1] and [FIPS.180-2] have been updated to their most recent versions and language in this document changed accordingly. Similarly, references to all 3GPP technical specifications have been updated to their 5G (Release 15) versions or otherwise most recent version when there has not been a 5G-related update.

#### Appendix B. Changes from RFC 4187 to RFC 5448

The changes to RFC 4187 relate only to the bidding down prevention support defined in Section 4. In particular, this document does not change how the Master Key (MK) is calculated in RFC 4187 (it uses CK and IK, not CK' and IK'); neither is any processing of the AMF bit added to RFC 4187.

## Appendix C. Changes from Previous Version of This Draft

RFC Editor: Please delete this section at the time of publication.

The -00 version of the working group draft is merely a republication of an earlier individual draft.

The -01 version of the working group clarifies updates relationship to RFC 4187, clarifies language relating to obsoleting RFC 5448, clarifies when the 3GPP references are expected to be stable, updates several past references to their more recently published versions, specifies what identifiers should be used in key derivation formula for 5G, specifies how to construct the network name in manner that is compatible with both 5G and previous versions, and has some minor editorial changes.

## Appendix D. Importance of Explicit Negotiation

Choosing between the traditional and revised AKA key derivation functions is easy when their use is unambiguously tied to a particular radio access network, e.g., Long Term Evolution (LTE) as defined by 3GPP or evolved High Rate Packet Data (eHRPD) as defined by 3GPP2. There is no possibility for interoperability problems if this radio access network is always used in conjunction with new protocols that cannot be mixed with the old ones; clients will always know whether they are connecting to the old or new system.

However, using the new key derivation functions over EAP introduces several degrees of separation, making the choice of the correct key derivation functions much harder. Many different types of networks employ EAP. Most of these networks have no means to carry any information about what is expected from the authentication process. EAP itself is severely limited in carrying any additional information, as noted in [RFC4284] and [RFC5113]. Even if these networks or EAP were extended to carry additional information, it would not affect millions of deployed access networks and clients attaching to them.

Simply changing the key derivation functions that EAP-AKA [RFC4187] uses would cause interoperability problems with all of the existing implementations. Perhaps it would be possible to employ strict separation into domain names that should be used by the new clients and networks. Only these new devices would then employ the new key derivation mechanism. While this can be made to work for specific cases, it would be an extremely brittle mechanism, ripe to result in problems whenever client configuration, routing of authentication requests, or server configuration does not match expectations. It also does not help to assume that the EAP client and server are

running a particular release of 3GPP network specifications. Network vendors often provide features from future releases early or do not provide all features of the current release. And obviously, there are many EAP and even some EAP-AKA implementations that are not bundled with the 3GPP network offerings. In general, these approaches are expected to lead to hard-to-diagnose problems and increased support calls.

#### Appendix E. Test Vectors

Test vectors are provided below for four different cases. The test vectors may be useful for testing implementations. In the first two cases, we employ the Milenage algorithm and the algorithm configuration parameters (the subscriber key K and operator algorithm variant configuration value OP) from test set 19 in [TS-3GPP.35.208].

The last two cases use artificial values as the output of AKA, and is useful only for testing the computation of values within EAP-AKA', not AKA itself.

## Case 1

The parameters for the AKA run are as follows:

Identity: "0555444333222111"

Network name: "WLAN"

RAND: 81e9 2b6c 0ee0 e12e bceb a8d9 2a99 dfa5

AUTN: bb52 e91c 747a c3ab 2a5c 23d1 5ee3 51d5

IK: 9744 871a d32b f9bb d1dd 5ce5 4e3e 2e5a

CK: 5349 fbe0 9864 9f94 8f5d 2e97 3a81 c00f

RES: 28d7 b0f2 a2ec 3de5

Then the derived keys are generated as follows:

CK': 0093 962d 0dd8 4aa5 684b 045c 9edf fa04

IK': ccfc 230c a74f cc96 c0a5 d611 64f5 a76c

K\_encr: 766f a0a6 c317 174b 812d 52fb cd11 a179

K\_aut: 0842 ea72 2ff6 835b fa20 3249 9fc3 ec23  
c2f0 e388 b4f0 7543 ffc6 77f1 696d 71ea

K\_re: cf83 aa8b c7e0 aced 892a cc98 e76a 9b20  
95b5 58c7 795c 7094 715c b339 3aa7 d17a

MSK: 67c4 2d9a a56c 1b79 e295 e345 9fc3 d187  
d42b e0bf 818d 3070 e362 c5e9 67a4 d544  
e8ec fe19 358a b303 9aff 03b7 c930 588c  
055b abee 58a0 2650 b067 ec4e 9347 c75a

EMSK: f861 703c d775 590e 16c7 679e a387 4ada  
8663 11de 2907 64d7 60cf 76df 647e a01c  
313f 6992 4bdd 7650 ca9b ac14 1ea0 75c4  
ef9e 8029 c0e2 90cd bad5 638b 63bc 23fb

## Case 2

The parameters for the AKA run are as follows:

Identity: "0555444333222111"

Network name: "HRPD"

RAND: 81e9 2b6c 0ee0 e12e bceb a8d9 2a99 dfa5

AUTN: bb52 e91c 747a c3ab 2a5c 23d1 5ee3 51d5

IK: 9744 871a d32b f9bb d1dd 5ce5 4e3e 2e5a

CK: 5349 fbe0 9864 9f94 8f5d 2e97 3a81 c00f

RES: 28d7 b0f2 a2ec 3de5

Then the derived keys are generated as follows:

CK': 3820 f027 7fa5 f777 32b1 fb1d 90c1 a0da

IK': db94 a0ab 557e f6c9 ab48 619c a05b 9a9f

K\_encr: 05ad 73ac 915f ce89 ac77 e152 0d82 187b

K\_aut: 5b4a caef 62c6 ebb8 882b 2f3d 534c 4b35  
2773 37a0 0184 f20f f25d 224c 04be 2afd

K\_re: 3f90 bf5c 6e5e f325 ff04 eb5e f653 9fa8  
cca8 3981 94fb d00b e425 b3f4 0dba 10ac

MSK: 87b3 2157 0117 cd6c 95ab 6c43 6fb5 073f  
f15c f855 05d2 bc5b b735 5fc2 1ea8 a757  
57e8 f86a 2b13 8002 e057 5291 3bb4 3b82  
f868 a961 17e9 1a2d 95f5 2667 7d57 2900

EMSK: c891 d5f2 0f14 8a10 0755 3e2d ea55 5c9c  
b672 e967 5f4a 66b4 bafa 0273 79f9 3aee  
539a 5979 d0a0 042b 9d2a e28b ed3b 17a3  
1dc8 ab75 072b 80bd 0c1d a612 466e 402c

## Case 3

The parameters for the AKA run are as follows:

Identity: "0555444333222111"  
Network name: "WLAN"  
RAND: e0e0 e0e0 e0e0 e0e0 e0e0 e0e0 e0e0 e0e0  
AUTN: a0a0 a0a0 a0a0 a0a0 a0a0 a0a0 a0a0 a0a0  
IK: b0b0 b0b0 b0b0 b0b0 b0b0 b0b0 b0b0 b0b0  
CK: c0c0 c0c0 c0c0 c0c0 c0c0 c0c0 c0c0 c0c0  
RES: d0d0 d0d0 d0d0 d0d0 d0d0 d0d0 d0d0 d0d0

Then the derived keys are generated as follows:

CK': cd4c 8e5c 68f5 7dd1 d7d7 dfd0 c538 e577  
IK': 3ece 6b70 5dbb f7df c459 a112 80c6 5524  
K\_encr: 897d 302f a284 7416 488c 28e2 0dcb 7be4  
K\_aut: c407 00e7 7224 83ae 3dc7 139e b0b8 8bb5  
58cb 3081 eccd 057f 9207 d128 6ee7 dd53  
K\_re: 0a59 1a22 dd8b 5b1c f29e 3d50 8c91 dbbd  
b4ae e230 5189 2c42 b6a2 de66 ea50 4473  
MSK: 9f7d ca9e 37bb 2202 9ed9 86e7 cd09 d4a7  
0d1a c76d 9553 5c5c ac40 a750 4699 bb89  
61a2 9ef6 f3e9 0f18 3de5 861a d1be dc81  
ce99 1639 1b40 1aa0 06c9 8785 a575 6df7  
EMSK: 724d e00b db9e 5681 87be 3fe7 4611 4557  
d501 8779 537e e37f 4d3c 6c73 8cb9 7b9d  
c651 bc19 bfad c344 ffe2 b52c a78b d831  
6b51 dacc 5f2b 1440 cb95 1552 1cc7 ba23



## Case 4

The parameters for the AKA run are as follows:

Identity: "0555444333222111"  
Network name: "HRPD"  
RAND: e0e0 e0e0 e0e0 e0e0 e0e0 e0e0 e0e0 e0e0  
AUTN: a0a0 a0a0 a0a0 a0a0 a0a0 a0a0 a0a0 a0a0  
IK: b0b0 b0b0 b0b0 b0b0 b0b0 b0b0 b0b0 b0b0  
CK: c0c0 c0c0 c0c0 c0c0 c0c0 c0c0 c0c0 c0c0  
RES: d0d0 d0d0 d0d0 d0d0 d0d0 d0d0 d0d0 d0d0

Then the derived keys are generated as follows:

CK': 8310 a71c e6f7 5488 9613 da8f 64d5 fb46  
IK': 5adf 1436 0ae8 3819 2db2 3f6f cb7f 8c76  
K\_encr: 745e 7439 ba23 8f50 fcac 4d15 d47c d1d9  
K\_aut: 3e1d 2aa4 e677 025c fd86 2a4b e183 61a1  
3a64 5765 5714 63df 833a 9759 e809 9879  
K\_re: 99da 835e 2ae8 2462 576f e651 6fad 1f80  
2f0f a119 1655 dd0a 273d a96d 04e0 fcd3  
MSK: c6d3 a6e0 cee a 951e b20d 74f3 2c30 61d0  
680a 04b0 b086 ee87 00ac e3e0 b95f a026  
83c2 87be ee44 4322 94ff 98af 26d2 cc78  
3bac e75c 4b0a f7fd feb5 511b a8e4 cbd0  
EMSK: 7fb5 6813 838a dafa 99d1 40c2 f198 f6da  
cebf b6af ee44 4961 1054 02b5 08c7 f363  
352c b291 9644 b504 63e6 a693 5415 0147  
ae09 cbc5 4b8a 651d 8787 a689 3ed8 536d

Authors' Addresses

Jari Arkko  
Ericsson  
Jorvas 02420  
Finland

Email: jari.arkko@piuha.net

Vesa Lehtovirta  
Ericsson  
Jorvas 02420  
Finland

Email: vesa.lehtovirta@ericsson.com

Vesa Torvinen  
Ericsson  
Jorvas 02420  
Finland

Email: vesa.torvinen@ericsson.com

Pasi Eronen  
Nokia Research Center  
P.O. Box 407  
FIN-00045 Nokia Group  
Finland

Email: pasi.eronen@nokia.com