Extensible Authentication Protocol
Working Group
Internet-Draft
Expires: October 1, 2004

J. Arkko
Ericsson
P. Eronen
Nokia Research Center
April 2, 2004

Authenticated Service Identities for the Extensible Authentication Protocol (EAP)
draft-arkko-eap-service-identity-auth-00

Status of this Memo

Copyright Notice

Abstract

A common arrangement in network access is the separation of the actual network access device (such as a wireless LAN access point) from the authentication servers. In the Extensible Authentication Protocol (EAP) framework, different authentication methods can provide varying security properties. If the EAP methods support authentication of service identities, it becomes possible for the clients to verify not only that the access device is trusted, but also that the parameters advertised by the access device are correct.

This document specifies a backward compatible extension to popular EAP methods for supporting such service identity authentication. A common parameter name space is created in order to ensure that the same parameters can be communicated independent of the choice of the authentication method.

Table of Contents

1. Introduction

In the Extensible Authentication Protocol (EAP) framework, different authentication methods can provide varying security properties. If the EAP methods support authentication of service identities, it becomes possible for the clients to verify not only that the access device is trusted, but also that the parameters advertised by the access device are correct. [4] uses the term channel bindings for this property, and defines it as follows:

> The communication within an EAP method of integrity-protected channel properties such as endpoint identifiers which can be compared to values communicated via out of band mechanisms (such as via a AAA or lower layer protocol).

The document continues by describing the security implications of not being able to verify service identities:

> It is possible for a compromised or poorly implemented EAP authenticator to communicate incorrect information to the EAP peer and/or server. This may enable an authenticator to impersonate another authenticator or communicate incorrect information via out-of-band mechanisms (such as via a AAA or lower layer protocol).
>
> Where EAP is used in pass-through mode, the EAP peer typically does not verify the identity of the pass-through authenticator, it only verifies that the pass-through authenticator is trusted by the EAP server. This creates a potential security vulnerability.
>
> Section 4.3.7 of [11] describes how an EAP pass-through authenticator acting as a AAA client can be detected if it attempts to impersonate another authenticator (such by sending incorrect NAS-Identifier [9], NAS-IP-Address [9] or NAS-IPv6-Address [10] attributes via the AAA protocol). However,

it is possible for a pass-through authenticator acting as a AAA client to provide correct information to the AAA server while communicating misleading information to the EAP peer via a lower layer protocol.

For example, it is possible for a compromised authenticator to utilize another authenticator's Called-Station-Id or NAS-Identifier in communicating with the EAP peer via a lower layer protocol, or for a pass-through authenticator acting as a AAA client to provide an incorrect peer Calling-Station-Id [9, 12] to the AAA server via the AAA protocol.

In order to address this vulnerability, EAP methods may support a

protected exchange of channel properties such as endpoint identifiers, including (but not limited to): Called-Station-Id [9, 12], Calling-Station-Id [9, 12], NAS-Identifier [9], NAS-IP-Address [9], and NAS-IPv6-Address [10].

Using such a protected exchange, it is possible to match the channel properties provided by the authenticator via out-of-band mechanisms against those exchanged within the EAP method. Where discrepancies are found, these SHOULD be logged; additional actions MAY also be taken, such as denying access.

Unfortunately, such verification is currently not possible in popular network scenarios. For instance, in IEEE 802.11 networks a rogue operator can actually advertise the same identity (SSID) as the local operator; the parameters advertised by the access point information are not authenticated end-to-end to the home network. There is no support is in the commonly used EAP methods for authentication of service identities, and there are no alternative verification means in the lower layer. Hence, rogue access points can present a different set of parameters to the client and to the home network.

This document specifies a backwards compatible extension to popular EAP methods for supporting authentication service identities. A common parameter name space is created in order to ensure that the same parameters can be communicated independent of the choice of the authentication method.

This document is organized as follows. Section 2 gives an overview of

the protocol operation. Section 3 describes the parameters that can be verified. We have provided only an initial list of parameters for the most popular lower layers, but additional parameters can be defined through IANA. Section 4 describes the extensions necessary for certain popular EAP methods. Support for other EAP methods can be added in other specifications.

2. Protocol Overview

In order to provide authentication of service identities, an EAP method needs to be able to pass data between the EAP peer and server, and be able to protect this exchange using keys known only them and not the access device. The Transient EAP Keys (TEKs) can be used for this purpose, as these keys are only known to the EAP endpoints and not communicated to the access device.

The data exchange needs to be bidirectional. After exchanging the information, the EAP peer can compare the information provided from the EAP server to the information it has received directly from the access device. If the information does not match, the access device

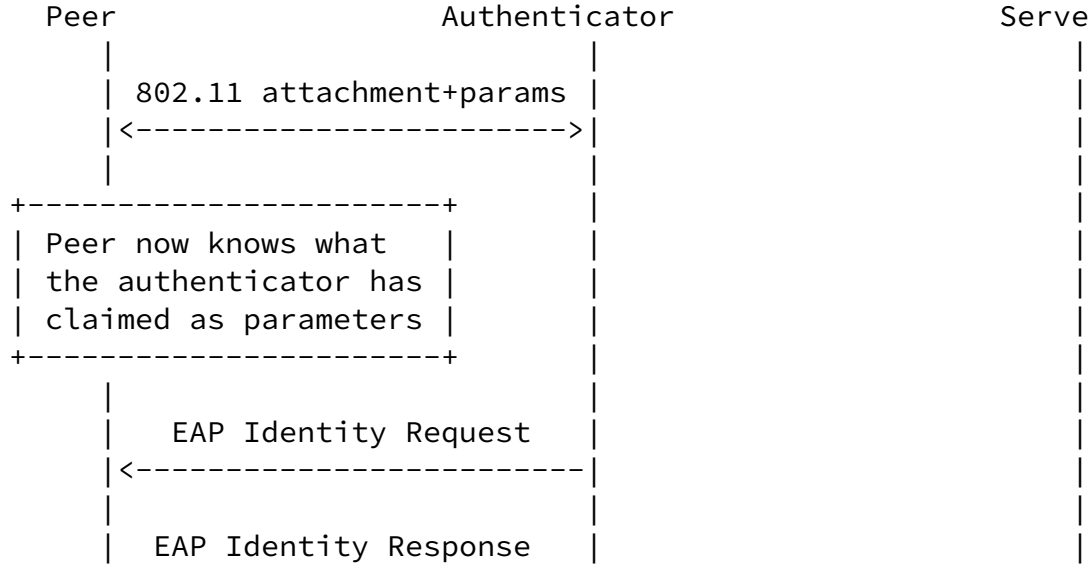
has provided different information to the peer and to the AAA protocol. This is disallowed, and the authentication SHOULD be terminated in this case. Similarly, the EAP server can compare the information the peer has provided to the information given from the access device via AAA.

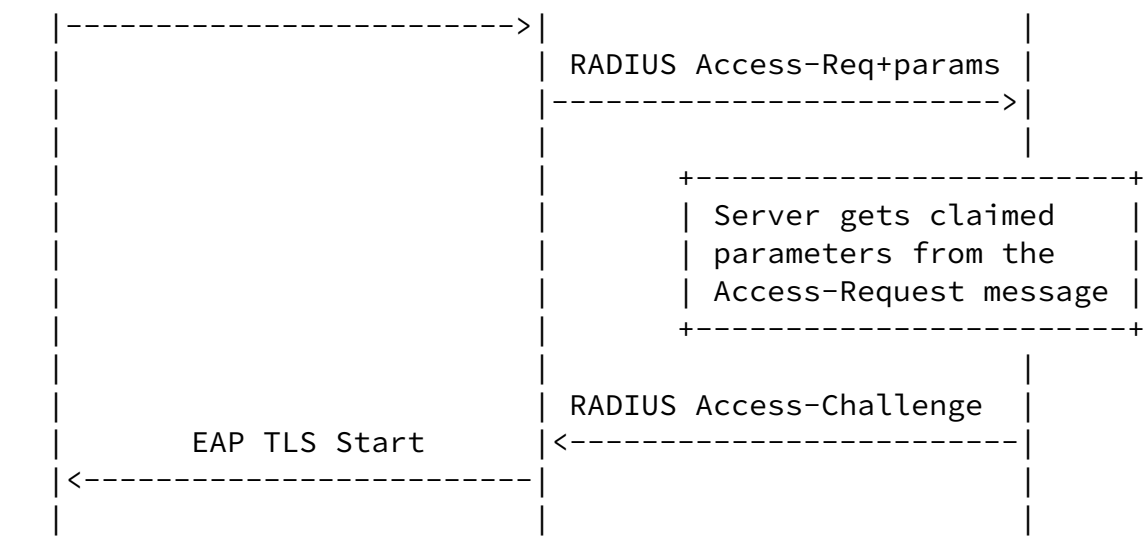
Simply comparing the information from the two sources ensures only that the service (NAS) has provided the same information for the involved parties. However, this does not guarantee that the information is correct in any sense. For many parameters it is necessary for the EAP server to check that the NAS is providing it is authorized to do. There are two authorization checks the EAP server must do. The first is that the authenticating user is allowed to access this service. The second is that the NAS node making the AAA request is allowed to provide this service. Both of these checks use the authentication information (who is the user being authenticated and who is the AAA node providing the service) and policy configured either on the EAP server or provided to it by other means such as certificates containing authorization information.

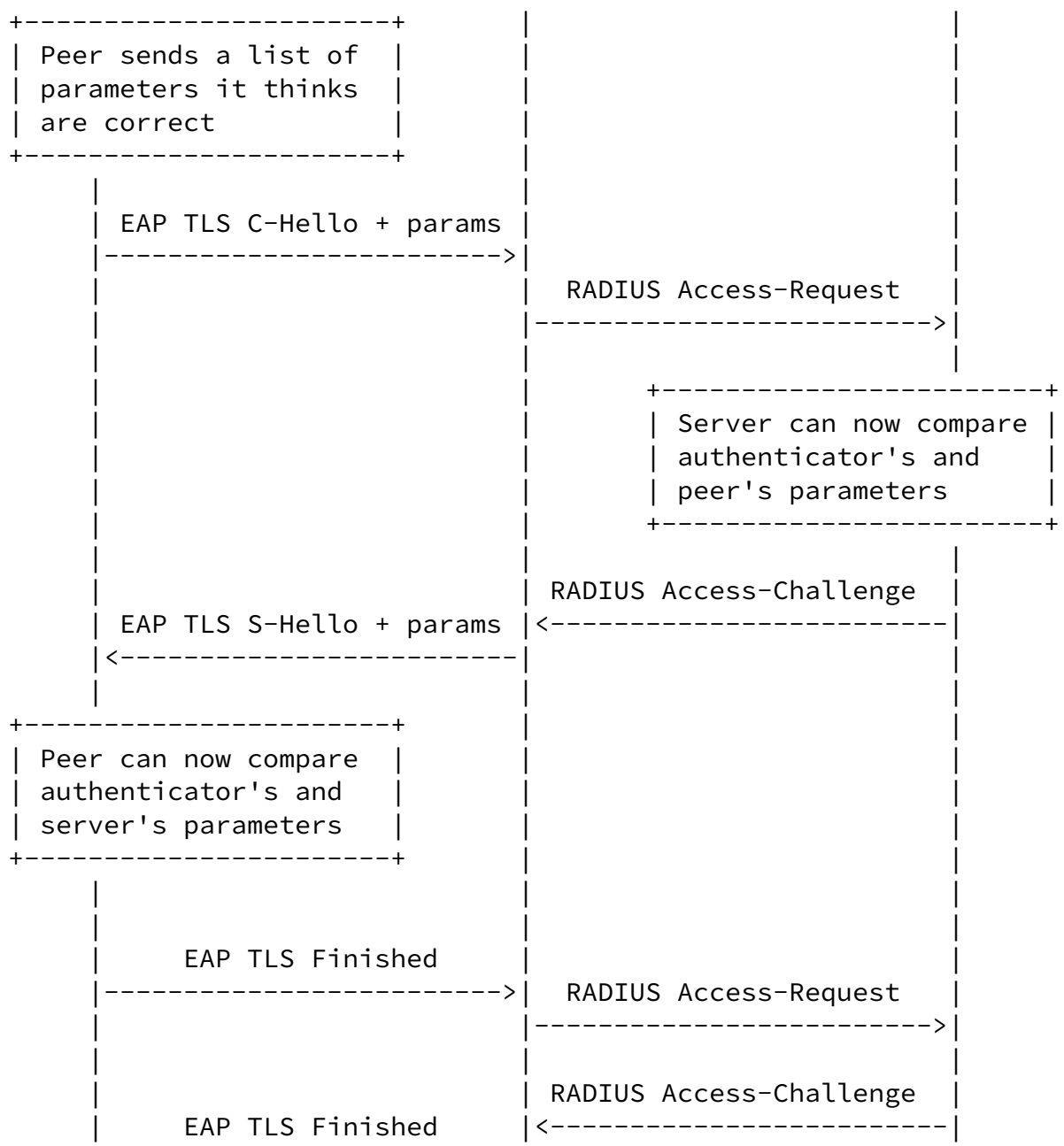
In order to provide a generic solution where any EAP method can be

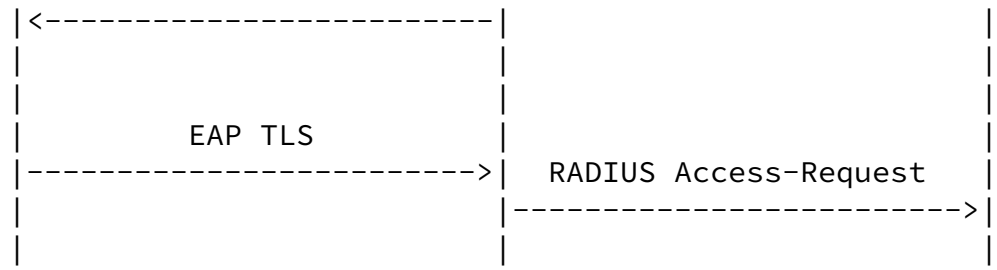
used on a given lower layer, the same format is used for the exchanged information. This format consists of Tag-Length-Value triplets with IANA managed tag space.

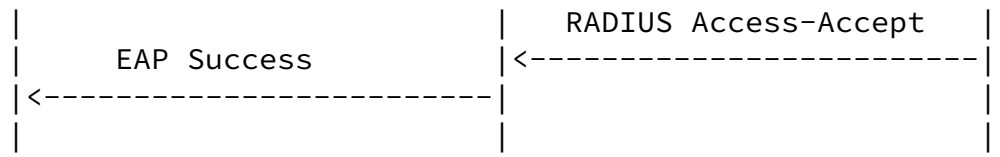
The parameter information is sent along the other messages in an EAP method. Both the server and the peer send their information to the other. Both parties make their own, independent decision about the correctness of the information. When mismatching information is received from EAP and authenticator, authentication MUST be terminated. The exact message sequences depend on the used EAP method, but Figure 1 shows a typical sequence.






Peer sends a list of parameters it thinks are correct
EAP TLS C-Hello + params
RADIUS Access-Request
Server can now compare authenticator's and peer's parameters
RADIUS Access-Challenge
EAP TLS S-Hello + params
Peer can now compare authenticator's and server's parameters
EAP TLS Finished
RADIUS Access-Request
RADIUS Access-Challenge
EAP TLS Finished


EAP TLS
RADIUS Access-Request

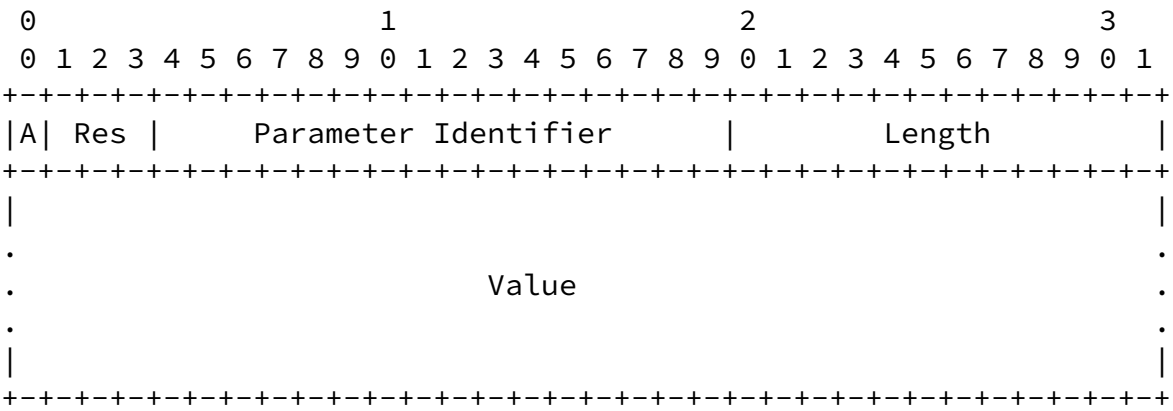


Zero or more parameters are exchanged in each direction. Each parameter is of the format explained in the next section.

3. Parameters

3.1 Format

Nodes supporting this extension pass parameters in the following format:



The meaning of the fields is described as follows:

A

The authenticated information flag. Value of zero means that the information is claimed by the service, but the EAP server is unable to tell whether the service is authorized to claim this or not. Value of one means that the EAP server knows that the service is authorized to claim this.

Res

A 3-bit field reserved for future use. The value MUST be initialized to zero by the sender, and MUST be ignored by the

receiver.

Parameter Identifier

A 16-bit field that specifies what parameter is being communicated.

Length

A 12-bit field that indicates the length of the Value field, in bytes.

Value

The actual parameter value. The interpretation of this value depends on the Parameter Identifier field.

The encapsulation of this sequence of parameters is EAP method dependent.

3.2 General Parameters

These parameters are for any type of nodes and lower layers. The Service Type parameter MUST be supported by all nodes conforming to this specification, and MUST be the first parameter in all messages containing a sequence of parameters defined here.

3.2.1 Service Type Parameter

The Parameter Identifier for this parameter is 0, and the Value is a 32-bit integer, represented in network byte order. The following values have been currently defined:

0 PPP

1 IEEE 802.11

2 PANA

3 IKEv2

The 'A' flag MUST always be set with the Service Type parameter. The receiver SHOULD fail the authentication if the Value field is either not recognized by it or is not the same one for which it thinks access is being provided.

3.2.2 Service Provider Parameter

The Parameter Identifier for this parameter is 1, and the Value is an UTF-8 encoded string describing the human readable name of the service provider. As EAP is used primarily for network access, this is typically the name of the access network provider.

3.2.3 Country Code Parameter

The Parameter Identifier for this parameter is 2, and the Value is an ASCII string of at most 3 characters, conforming to the ISO 3166 [8] country code.

3.3 Parameters for IEEE 802.11 wireless LANs

All the following parameters MUST be supported when IEEE 802.11 is accepted as a Service Type.

3.3.1 SSID Parameter

The Parameter Identifier for this parameter is 3, and the Value is an octet string containing the Service Set Identifier (SSID).

3.3.2 BSSID Parameter

The Parameter Identifier for this parameter is 4, and the Value is a 6-octet string containing the BSSID.

3.3.3 STA_MAC Parameter

The Parameter Identifier for this parameter is 5, and the Value is a 6-octet string containing the STA MAC address.

3.3.4 Protection Mechanism Parameter

The Parameter Identifier for this parameter is 6, and the Value is a 32-bit integer, represented in network byte order. The following values have been currently defined:

0 WPA/802.11i

1 WEP/802.1X

2 802.1X only

3.4 Parameters for PPP

All the following parameters MUST be supported when PPP is accepted as the Service Type.

3.4.1 Encapsulation Parameter

The Parameter Identifier for this parameter is 7, and the Value is a 32-bit integer, represented in network byte order. The following values have been currently defined:

0 Framed

1 PPPoE

2 PPTP

3 L2TP

3.4.2 Called-Station-Id Parameter

The Parameter Identifier for this parameter is 8, and the Value is an ASCII string containing the called station identity.

3.4.3 Protection Mechanism Parameter

The Parameter Identifier for this parameter is 9, and the Value is a 32-bit integer, represented in network byte order. The following values have been currently defined:

0 None

1 MPPE

3.5 Parameters for PANA

All the following parameters MUST be supported when PANA is accepted as the Service Type.

3.5.1 PaC Device-Id Parameter

The Parameter Identifier for this parameter is 10, and the Value is an octet string containing the PaC Device-Id.

3.5.2 PAA Device-Id Parameter

The Parameter Identifier for this parameter is 11, and the Value is an octet string containing the PAA Device-Id.

3.5.3 Protection Mechanism Parameter

The Parameter Identifier for this parameter is 12, and the Value is a 32-bit integer, represented in network byte order. The following values have been currently defined:

0 None

1 Layer 2

3 IPsec

3.6 Parameters for IKEv2

All the following parameters MUST be supported when IKEv2 is accepted as the Service Type.

3.6.1 Initiator Address Parameter

The Parameter Identifier for this parameter is 13, and the Value is the IP address of the node who initiated this IKEv2 EAP exchange. The Value is either 4 or 16 bytes depending on whether IPv4 or IPv6 is used.

3.6.2 Responder Address Parameter

The Parameter Identifier for this parameter is 14, and the Value is the IP address of the node who acted as the responder for this IKEv2 EAP exchange. The Value is either 4 or 16 bytes depending on whether IPv4 or IPv6 is used.

3.6.3 IDi Parameter

The Parameter Identifier for this parameter is 15, and the Value is an octet string containing the IKEv2 initiator identity payload (IDi).

3.6.4 IDr Parameter

The Parameter Identifier for this parameter is 16, and the Value is an octet string containing the IKEv2 initiator identity payload (IDr).

4. EAP Method Extensions

This section describes an initial set of extensions to some current EAP methods so that they can be transport the parameter information.

The extensions are optional and backwards compatible, so that, where allowed by policy, EAP peers without these extensions can still contact EAP servers with these extensions and vice versa. The default policy SHOULD be that such usage is allowed.

4.1 EAP-TLS

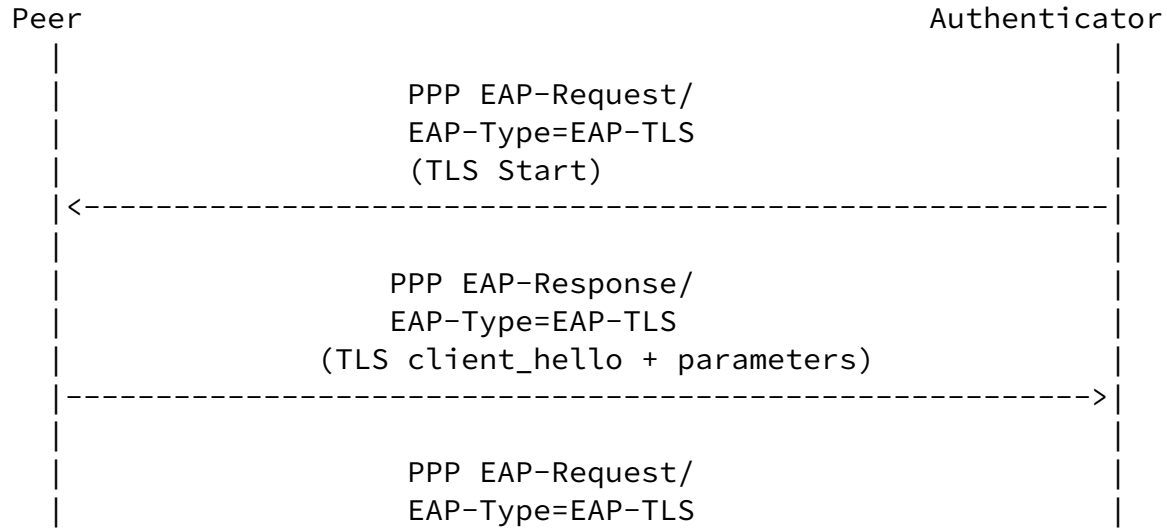
A TLS extension [3] is added to the EAP TLS [2] client_hello/ server_hello messages. The extension type of the extension is EAP Service Identity and it has the number < To Be Assigned By IANA >. The extension contains a sequence of parameters, followed by each other.

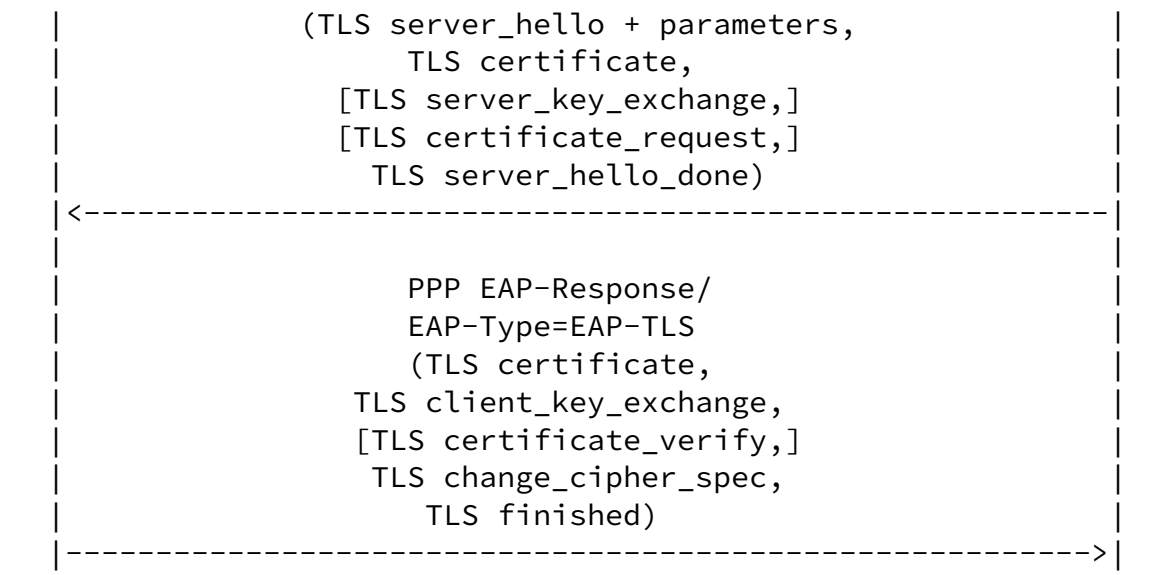
As discussed in RFC 3546, when these extensions appear in a client hello message, they are ignored by old server implementations. The lack of this extension in the authenticator's server hello response SHOULD be taken as an indication that the authenticator does not

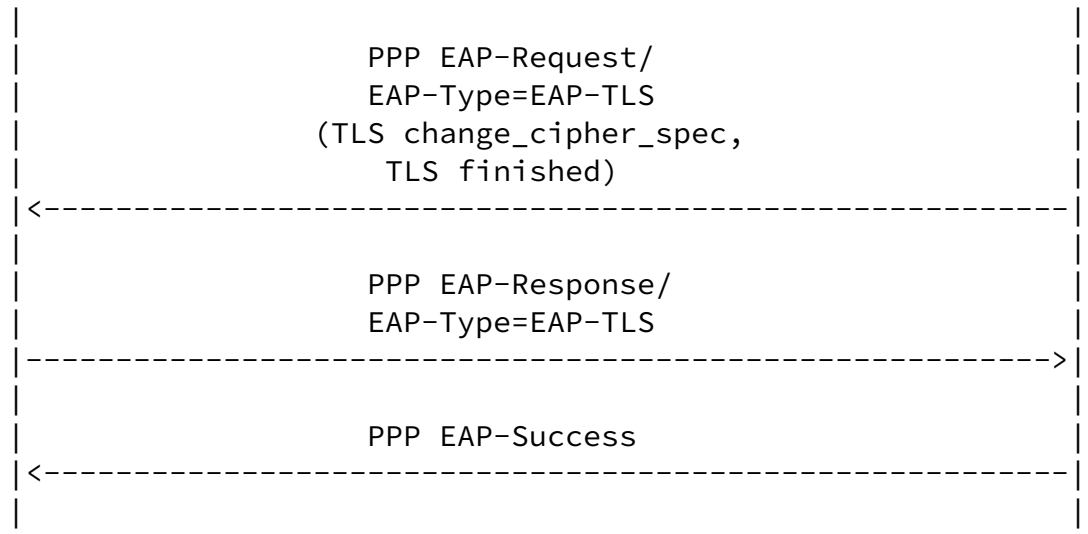
support the mechanisms defined in this document. The authenticator MUST NOT use this extension unless the client provided the same extension in its own hello message, as per RFC 3546 the client is required to terminate the TLS session otherwise.

The client_hello/server_hello messages are included in MACs in the TLS Finished messages, which ensures that modifications will be detected.

The following sequence illustrates the operation of the EAP TLS protocol with this extension:



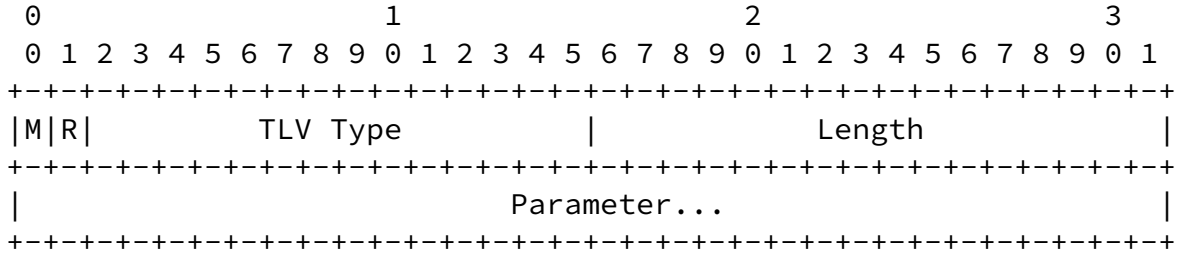




This works the same way when resuming session. Note that the parameters can change from the initial authentication.

4.2 PEAPv2

In PEAPv2 [7], the Connection-Binding TLV is used to carry parameter objects. One Connection-Binding TLV for this purpose is exchanged in each direction, containing all the parameters that need to be exchanged. The Connection-Binding TLV carries a set of PEAPv2 TLVs. The transport of parameters for the purposes of this document takes place through the PEAPv2 Service Identity Parameter TLV defined in the following:



The fields of this TLV are as follows:

M

0 - Optional TLV.

R

Reserved, set to zero (0).

TLV Type

< To Be Assigned By IANA >

Length

Length of the TLV.

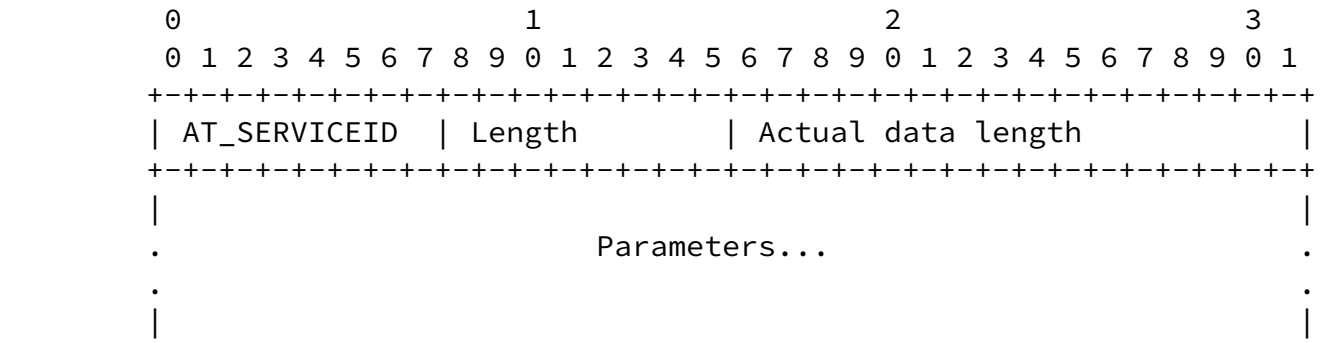
Parameter...

The parameter in the format described in Section 3.1.

4.3 EAP-AKA

For EAP-AKA, a new attribute AT_SERVICEID is added to the EAP-Request/AKA/Challenge and EAP-Response/AKA/Challenge messages.

The format of the AT_SERVICEID attribute is shown below:



The fields of this attribute are as follows:

AT_SERVICEID

< To Be Assigned By IANA >

Length

Length of the attribute.

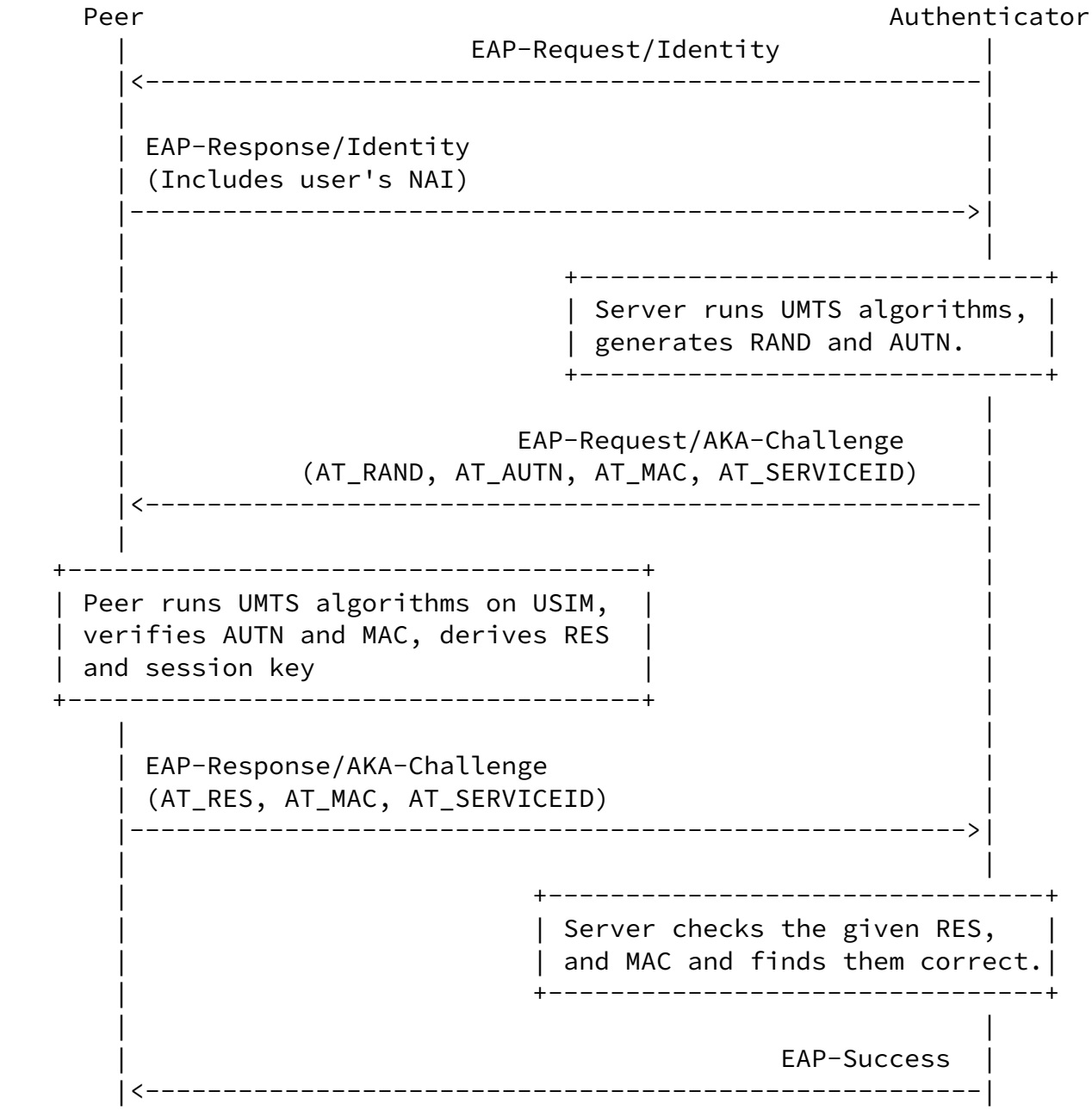
Actual data length

This field specifies the length of the following field in bytes, because the length of the parameter must be a multiple of 4 bytes, the sender pads the data with zero bytes when necessary.

Parameters...

The parameters in the format described in Section 3.1.

The following sequence illustrates the operation of the EAP-AKA protocol with this extension:



Note that the AT_SERVICEID attribute is used also in the EAP-Request/AKA/AKA-Reauthentication and EAP-Response/AKA/AKA-Reauthentication messages, and that the set of parameters exchanged in this case may differ from those agreed upon earlier in the initial authentication.

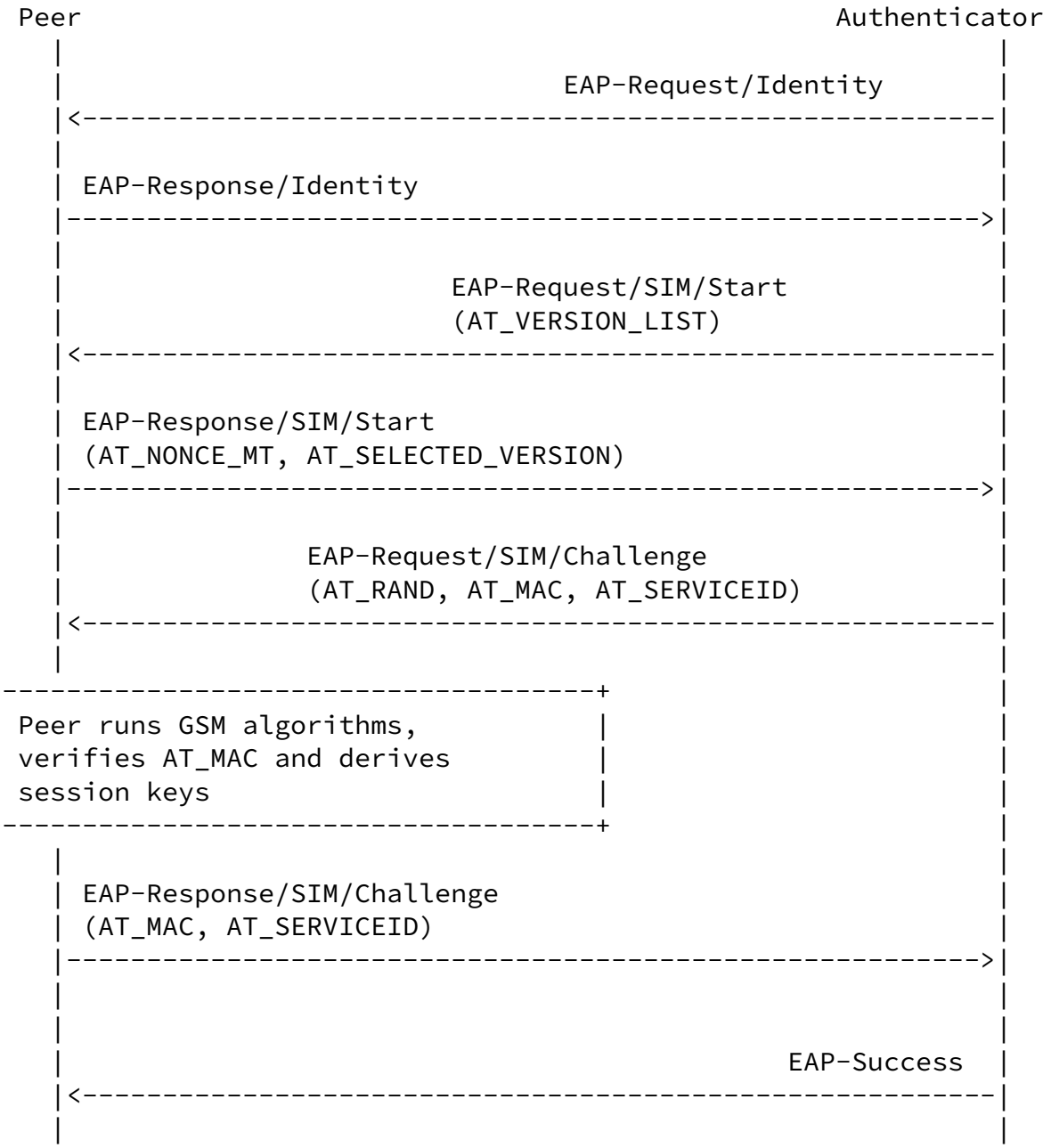
The use of the AT_SERVICEID attribute is backwards compatible, because existing implementations ignore unknown parameters.

4.4 EAP-SIM

For EAP-SIM, a new attribute AT_SERVICEID is added to the

EAP-Request/SIM/Challenge and EAP-Response/SIM/Challenge messages. The format of the AT_SERVICEID attribute is as shown for EAP-AKA.

The following sequence illustrates the operation of the EAP-SIM protocol with this extension:



As with EAP-AKA, the AT_SERVICEID attribute must be passed also in

the EAP-Request/SIM/SIM-Reauthentication and EAP-Response/SIM/ SIM-Reauthentication messages.

5. Security Considerations

The implications of being unable to verify service identities have been described in Section 7.15 of [4]. These include vulnerabilities related to compromised access points or fraudulent service providers. The mechanism provided in this document removes these

vulnerabilities. The mechanism is generic and not tied to any specific EAP method or use of EAP over a specific link layer, and as such can be expected to be more easily deployed as alternative suggestions such as those described in PEAPv2 [7] or EAP FAST [13].

In order to operate, however, the mechanism requires that the used AAA protocol is able to transport the same information (such as the SSID that an access point claims the user requested) to the home AAA server, so that the server can compare this claim to the authenticated information received from the client. Where such information is not available, vulnerabilities still remain.

In the deployment phase, it is possible that clients and servers do not get support for the mechanism described in this document at the same time. It is a policy decision to accept an EAP exchange from a party that does not support this mechanism. This decision is protected from a bidding down attack by a man-in-the-middle, because EAP methods have integrity protection for the exchanged messages. Therefore, the removal or modification of the parameter block would be detected.

6. IANA Considerations

6.1 Allocations Requested in This Document

This document requests an IANA allocation of TLS Extension type [3] for EAP Service Identity (see Section 4.1).

This document requests an IANA allocation of a PEAPv2 [7] TLV type number for the Service Identity Parameter TLV (see Section 4.2).

This document requests an IANA allocation for the attribute type

number AT_SERVICEID in the [6] and [5] protocols (see Section 4.3 and Section 4.4). The same value should be allocated for both protocols.

6.2 Future Allocation Policy

New Parameter Identifier values can be defined through Specification Required [1]. The following values have been currently allocated:

0 Service Type

1 Service Provider

2 Country Code

3 802.11/SSID

4 802.11/BSSID

5 802.11/STA_MAC

6 802.11/Protection Mechanism

7 PPP/Encapsulation

8 PPP/Called-Station-Id

9 PPP/Protection Mechanism

10 PANA/PaC Device-Id

11 PANA/PAA Device-Id

12 PANA/Protection Mechanism

13 IKEv2/Initiator Address

14 IKEv2/Responder Address

15 IKEv2/IDi

16 IKEv2/IDr

New Service Type values can be defined through IETF Consensus [1]. The following values have been currently allocated:

0 PPP

1 IEEE 802.11

2 PANA

3 IKEv2

Values in other enumerated parameters can be defined through First Come, First Served [1].

Normative References

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

[2] Aboba, B. and D. Simon, "PPP EAP TLS Authentication Protocol",

RFC 2716, October 1999.

[3] Blake-Wilson, S., Nystrom, M., Hopwood, D., Mikkelsen, J. and T. Wright, "Transport Layer Security (TLS) Extensions", RFC 3546, June 2003.

[4] Blunk, L., "Extensible Authentication Protocol (EAP)", draft-ietf-eap-rfc2284bis-07 (work in progress), December 2003.

[5] Haverinen, H. and J. Salowey, "EAP SIM Authentication", draft-haverinen-pppext-eap-sim-12 (work in progress), October 2003.

[6] Arkko, J. and H. Haverinen, "EAP AKA Authentication", draft-arkko-pppext-eap-aka-11 (work in progress), October 2003.

[7] Josefsson, S., Palekar, A., Simon, D. and G. Zorn, "Protected EAP Protocol (PEAP)", draft-josefsson-pppext-eap-tls-eap-07

(work in progress), October 2003.

[8] International Organization for Standardization, "Codes for the representation of names of countries, 3rd edition", ISO Standard 3166, August 1988.

Informative References

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

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

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

[12] Congdon, P., Aboba, B., Smith, A., Zorn, G. and J. Roese, "IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines", RFC 3580, September 2003.

[13] Salowey, J., "EAP Flexible Authentication via Secure Tunneling (EAP-FAST)", draft-cam-winget-eap-fast-00 (work in progress), February 2004.

Authors' Addresses

Jari Arkko
Ericsson

FI-02420 Jorvas
Finland

EMail: jari.arkko@ericsson.com

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

EMail: pasi.eronen@nokia.com

Appendix A. Acknowledgments

The authors would like to thank Bernard Aboba, Mohan Parthasarathy, and David Mariblanca for interesting discussions in this problem space.

Intellectual Property Statement

pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the IETF's procedures with respect to rights in IETF Documents can be found in BCP 78 and BCP 79.

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

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

The IETF has been notified of intellectual property rights claimed in regard to some or all of the specification contained in this document. For more information consult the online list of claimed rights.

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.