Authenticated Service Information for the Extensible Authentication Protocol (EAP)
draft-arkko-eap-service-identity-auth-02

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on November 4, 2005.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

EAP is usually used in an arrangement where the actual service (such as a wireless LAN access point) is separated from the authentication server. However, EAP itself does not have a concept of a service identity or its parameters, and thus the client usually does not authenticate any information about the service itself, even when a mutually authenticating EAP method is used. This document specifies...
a backward compatible extension to popular EAP methods for authenticating service related information, such as the identity and type of the offered service. A common parameter name space is created in order to ensure that the same kinds of identifiers can be authenticated independent of the choice of the EAP authentication method.

Table of Contents

1. Introduction ........................................... 3
2. Authenticated Service Information ...................... 3
3. Protocol Overview ...................................... 6
4. Parameters ............................................ 8
   4.1 Format ........................................... 8
   4.2 General Parameters ................................ 9
       4.2.1 Service Type Parameter ....................... 9
       4.2.2 Service Provider Parameter ................. 10
       4.2.3 Country Code Parameter ..................... 10
   4.3 Parameters for IEEE 802.11 wireless LANs ........... 10
       4.3.1 SSID Parameter ................................ 10
       4.3.2 BSSID Parameter ............................. 10
   4.4 Parameters for IKEv2 ............................ 10
       4.4.1 Responder Address Parameter ............... 10
       4.4.2 IDr Parameter .............................. 11
5. EAP Method Extensions .................................. 11
   5.1 EAP-TLS ......................................... 11
   5.2 PEAPv2 ........................................... 13
   5.3 EAP-AKA ......................................... 13
   5.4 EAP-SIM ......................................... 15
6. Security Considerations .............................. 16
7. IANA Considerations ................................. 17
   7.1 Allocations Requested in This Document .......... 17
   7.2 Future Allocation Policy .......................... 17
8. References ............................................ 18
   8.1 Normative References ............................. 18
   8.2 Informative References ........................... 19
A. Acknowledgments ..................................... 20

Intellectual Property and Copyright Statements .......... 21
1. Introduction

EAP is usually used in an arrangement where the actual service (such as a wireless LAN access point) is separated from the authentication server. However, EAP itself does not have a concept of a service identity or its parameters, and thus the client usually does not authenticate any information about the service itself, even when a mutually authenticating EAP method is used.

However, if a method supports channel bindings as specified in RFC 3748 [4] it becomes possible to ensure that the client, the node providing the service, and the authentication server all have the same information about this information. This does not, by itself, ensure that the information is correct, just that everyone has the same information; a service node might be providing a service that this particular node should not be providing. A method that supports authenticated service information ensures in addition that the authentication server knows this information to be correct.

This document specifies a backwards compatible extension to popular EAP methods for supporting both channel bindings and authenticated service information. A common parameter name space is created in order to ensure that the same kinds of information can be authenticated independent of the choice of the EAP method. While this draft specifies the actual parameters and their formats, it is not intended that either the EAP layer or EAP methods interpret or understand the information in any manner. They should treat is as opaque data. The information is expected to be conveyed to upper layers where authorization decisions can be made.

This extension is intended for the verification of service information. It is not intended as a means for communicating information about parameters that EAP clients would not otherwise be aware of based on their communication with the node providing the service.

This rest of the document is organized as follows. Section 3 gives an overview of the protocol operation. Section 4 describes the kind of information that can be verified. We have provided only an initial list of parameters for IEEE 802.11 and IKEv2, but additional parameters can be defined through IANA. Section 5 describes the extensions necessary for certain popular EAP methods. Support for other EAP methods can be added in other specifications.

2. Authenticated Service Information

EAP is run for the purposes of providing granting access to a service, such as network access. The nodes providing such services
(called authenticators in EAP) typically have an identifier or identifiers, and offer a specific type of a service with an associated set of parameters. Collectively, this identifier, type and parameter information is called service information.

In the Extensible Authentication Protocol (EAP) framework, different authentication methods can provide varying security properties. One such property is called "channel bindings", which is described in RFC 3748 [4] 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 information:

"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 (BSSID or 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 in the commonly used EAP methods for authentication of service information, and there are no alternative verification means in the IEEE 802 lower layer. For instance, rogue access points can present a different identity to the client and to the home network. Or a rogue IKEv2 gateway can claim to be a 802.11 access point to its clients, but still appear as an IKEv2 gateway towards the authentication server.

There are cases where the lower layer does provide its own means of authenticating the service information. For instance, in IKE2, EAP is used together with certificate-based authentication of the responder. However, this document may be useful with proposed IKEv2 extensions like [14] that remove the need to deploy a PKI.

This situation is further complicated by the fact that services do not necessarily have just a single identifier, but several different identifiers of different types. For instance, an IEEE 802.11 access point could be identified by a BSSID, an IPv4 address (e.g., NAS-IP-Address), or a domain name (e.g., NAS-Identifier). Other identifiers, such as SSID, do not necessarily identify a single access point, but may be more interesting to the client (if you consider the "service" to be wireless LAN network access in some hotspot, rather than a single physical box).

It is important to make a distinction between channel bindings and authenticating information related to the pass-through authenticator. Channel bindings only ensure that the same information is available to the EAP peer and the AAA server. This alone does not prevent an authenticator from impersonating another authenticator if the AAA server blindly accepts any information received from the authenticator. To provide authentication, the AAA
server has to verify that the information actually corresponds to the entity the AAA-Key is sent to.

3. Protocol Overview

The basic idea in this extension is that the AAA server sends the EAP peer a statement that it has sent (or is going to send) the AAA-Key (usually the MSK) to an access device associated with particular set of identifiers and other information.

In order to protect this statement, an EAP method needs to be able to pass data from the EAP server to the EAP peer, 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.

After receiving this 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.

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. 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.

```
<table>
<thead>
<tr>
<th>Peer</th>
<th>Authenticator</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>802.11 attachment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------</td>
<td>--------</td>
</tr>
<tr>
<td>Information received</td>
<td></td>
<td></td>
</tr>
<tr>
<td>at this point is not authenticated</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EAP Identity Request</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Arkko & Eronen Expires November 4, 2005 [Page 6]
EAP Identity Response
-------------------------> RADIUS Access-Request

EAP TLS Start
<-------------------------

EAP TLS C-Hello
-------------------------> RADIUS Access-Request

RADIUS Access-Challenge
<-------------------------

RADIUS Access-Challenge
<-------------------------

RADIUS Access-Challenge
<-------------------------

RADIUS Access-Challenge
<-------------------------

EAP TLS S-Hello + id.
EAP TLS Finished
------------------------->

Peer can now verify that the information is what was expected

EAP TLS Finished
------------------------->

EAP TLS Finished
------------------------->

RADIUS Access-Challenge
<-------------------------

RADIUS Access-Challenge
<-------------------------

RADIUS Access-Challenge
<-------------------------

RADIUS Access-Challenge
<-------------------------

RADIUS Access-Challenge
<-------------------------

Arkko & Eronen          Expires November 4, 2005
4. Parameters

4.1 Format

Nodes supporting this extension pass parameters in the following format:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|A| Res |     Parameter Identifier      |        Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                                                               |
|                          Value                                |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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 information is correct or not (i.e., this corresponds to channel bindings without authentication). Value of one means that the EAP server knows that the service is authorized to claim this ("authenticated service information").
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 EAP or the EAP method layer SHOULD NOT attempt to interpret the information beyond this format. In other words, the Parameter Identifier and Value fields are interpreted as opaque data in order to ensure EAP media independence. EAP implementations SHOULD pass the information to higher layers that are in charge of authorization decisions, such as AAA server authorization logic.

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

4.2 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.

4.2.1 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  IEEE 802.11
1  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.

4.2.2 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.

4.2.3 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.

4.3 Parameters for IEEE 802.11 wireless LANs

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

4.3.1 SSID Parameter

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

4.3.2 BSSID Parameter

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

4.4 Parameters for IKEv2

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

4.4.1 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.
4.4.2 IDr Parameter

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

5. 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.

5.1 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 Information and it has the number < To Be Assigned By IANA >. The extension contains a sequence of parameters, followed by each other.

The extension sent in the client_hello message SHOULD contain zero parameters, and is only used for capability detection. 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:
Peer                                               Authenticator

PPP EAP-Request/
EAP-Type=EAP-TLS
(TLS Start)

-------------------------------------------------------->

PPP EAP-Response/
EAP-Type=EAP-TLS
(TLS client_hello + extension)

PPP EAP-Request/
EAP-Type=EAP-TLS
(TLS server_hello + extension,
  TLS certificate,
  [TLS server_key_exchange,]
  [TLS certificate_request,]
  TLS server_hello_done)

PPP EAP-Response/
EAP-Type=EAP-TLS
(TLS certificate,
  TLS client_key_exchange,
  [TLS certificate_verify,]
  TLS change_cipher_spec,
  TLS finished)

PPP EAP-Request/
EAP-Type=EAP-TLS
(TLS change_cipher_spec,
  TLS finished)

PPP EAP-Response/
EAP-Type=EAP-TLS

PPP EAP-Success

This works the same way when resuming session. Note that the parameters can change from the initial authentication.
5.2 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 Information 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 4.1.

5.3 EAP-AKA

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

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 4.1.

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 message, 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 backward compatible, because existing implementations ignore unknown parameters.

5.4 EAP-SIM

For EAP-SIM, a new attribute AT_SERVICEID is added to the EAP-Request/SIM/Challenge message. 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:

Peer

- EAP-Request/Identity
- EAP-Response/Identity
- EAP-Request/SIM/Start
  (AT_VERSION_LIST)
- EAP-Response/SIM/Start
  (AT_NONCE_MT, AT_SELECTED_VERSION)
- EAP-Request/SIM/Challenge
  (AT_RAND, AT_MAC, AT_SERVICEID)
- EAP-Response/SIM/Challenge
  (AT_MAC)
- EAP-Success

Authenticator

As with EAP-AKA, the AT_SERVICEID attribute must be passed also in the EAP-Request/SIM/SIM-Reauthentication message.

6. Security Considerations

The implications of being unable to verify service information have been described in Section 7.15 of RFC 3748 [4]. These include vulnerabilities related to compromised access points or fraudulent service providers. When properly used, 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].

Authenticating the service information may complicate operation in some deployment scenarios, since it requires that the AAA server is able to authenticate the expected kinds of information. For instance, RADIUS is often deployed in situations where the only authenticated information related to the RADIUS client is the IP address; other information may be present in the Access-Request message (such as BSSID/SSID in the Called-Station-Id attribute), but this is simply claimed information not authenticated information. Where such information is not available, some 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.

7. IANA Considerations

7.1 Allocations Requested in This Document

This document requests an IANA allocation of TLS Extension type [3] for EAP Service Identity (see Section 5.1).

This document requests an IANA allocation of a PEAPv2 [7] TLV type number for the Service Identity Parameter TLV (see Section 5.2).

This document requests an IANA allocation for the attribute type number AT_SERVICEID in the [6] and [5] protocols (see Section 5.3 and Section 5.4). The same value should be allocated for both protocols.

7.2 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
6  IKEv2/Responder Address
7  IKEv2/IDr

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

0  IEEE 802.11
1  IKEv2

Values in other enumerated parameters can be defined through First Come, First Served [1]. However, this extension is intended only for the verification of service information. Its use for communicating other information not already known by the EAP client (such as for service discovery) is discouraged.

8. References

8.1  Normative References


Arkko & Eronen  Expires November 4, 2005  [Page 18]
8.2 Informative References


Authors’ Addresses

Jari Arkko
Ericsson
FI-02420 Jorvas
Finland

Email: jari.arkko@ericsson.com
Appendix A. Acknowledgments

The authors would like to thank Bernard Aboba, Mohan Parthasarathy, Hannes Tschofenig, Yoshihiro Ohba, Joe Salowey, Glen Zorn, and David Mariblanca for interesting discussions in this problem space.
Intellectual Property Statement

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

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

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

Disclaimer of Validity

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

Copyright Statement

Copyright (C) The Internet Society (2005). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

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