Channel Binding Support for EAP Methods
draft-clancy-emu-chbind-00

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 draft 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 August 21, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2008).

Abstract

This document defines how to implement channel bindings for Extensible Authentication Protocol (EAP) methods.
1. Introduction

A well-documented problem with the current Extensible Authentication Protocol (EAP) architecture [RFC3748] when used in pass-through authenticator mode is the so-called "lying NAS" problem. Here, a Network Access Server (NAS), or pass-through authenticator, may authenticate to the backend AAA infrastructure using one set of credentials, while representing contrary information to EAP clients.

A concrete example of this may be an IEEE 802.11 access point with a security association to a particular AAA server. While there may be some identity tied to that security association, there's no reason the access point need advertise the same identity to clients. In fact, it may include whatever information in its beacons (to include different SSID or security properties) it desires. This could lead to situations where, for example, a client joins one network that's masquerading as another.

In this document, we utilize "AAA Payloads" [I-D.clancy-emu-aaapay] to transport key channel binding characteristics from the client to the server, allowing the server to verify the authenticator has advertised valid information to the peer. The server can also respond back with additional information that could be useful for the client to decide whether or not to continue its session with the authenticator.

2. Terminology

In this document, several words are used to signify the requirements of the specification. These words are often capitalized. 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].

3. Overview

In a [RFC4017] and [RFC4962]-compliant EAP authentication, the EAP client and EAP server mutually authenticate each other, and derive keying material. When operating in pass-through mode, the EAP server then transports the resulting Master Session Key (MSK) to the authenticator. However, just because a security association exists between the client and server, and between the server and authenticator, doesn’t mean the authenticator can’t provide false information to the client.

Channel bindings [RFC5056] seek to enforce a stronger security
relationship between the three entities, by allowing the client and server to compare their relative perception of the authenticator’s identity in a secure channel. This document defines how to use "AAA Payloads" [I-D.clancy-emu-aaapay] within EAP methods to implement channel bindings. Unlike [RFC5056], channel binding in EAP context does not ensure binding different layers of a session but rather the information advertised to client and authentication server by an authenticator acting as pass-through device during an EAP session.

It is preferable to use this idea, rather than hashing identity information directly into keys, for a number of reasons:

- It allows for fuzzy comparisons of entity names, rather than requiring absolute. This can facilitate deployment and debugging.
- Any EAP method that provides mutual authentication and key derivation can easily add channel binding as proposed in this draft. On the other hand, including identities into key derivations requires the modification of EAP methods. For instance, many EAP methods specify how MSK and other keying material is derived and adding channel binding by adding identifiers would need to be specified for each method individually.
- Client and authentication server may communicate on different layers with the authenticator and thus receive different identifiers that may belong to the same device. A server may be able to recognize which different identifiers belong to the same device (e.g. by performing a table look up). This scenario cannot be solved by hashing identities into keys.

4. Trust Model

Channel bindings are always provided between two communication endpoints, i.e. here between client and server, who communicate through an authenticator in pass-trough mode (as illustrated in Figure 1). Channel binding prevents attacks by rogue authenticators in pass-trough mode (see Example Attacks).
During an EAP execution, an authenticator presents information info1 and info2 to client and server, respectively. Such information can include the authenticator’s identifier/address and/or advertised network information (such as offered services and billing information). To prevent attacks by rogue authenticators, the server must be able to recognize any inconsistencies in info1 and info2. Therefore, the client sends info1 to the server and upon performing an inconsistency check, the server notifies the client about the result. Only in case of a positive result, channel binding is provided and the EAP session continues. Note that under some circumstances, the client could be able to check for inconsistencies; however the check is generally easier to implement on the server.

The trust model for channel binding consists of three parties, namely a client, an authentication server and an authenticator in pass-through mode as illustrated in Figure 1. In this trust model, client and authentication server are honest while the authenticator is maliciously sending false information to client and/or server. The three parties have the following trust relationships:

- The server trusts that the channel binding information received from the client is the information that the client received from the authenticator (i.e. info1)
- The client trusts the channel binding result received from the server
- The server trusts the identifier received from the authenticator

Note that the model includes all cases in which authenticator and authentication server communicate over one or more intermediate nodes as long as 1) these nodes only forward the messages (e.g. routers) and 2) authenticator and server communicate using an end-to-end protocol (e.g. AAA). However, roaming scenarios with proxies are out of the scope of this document.
In order to establish the first two trust relationships during an EAP execution, an EAP method MUST provide the following:

- mutual authentication between client and server
- derivation of keying material including a key for integrity protection of channel binding messages
- sending info1 from client to server over an integrity-protected channel
- sending the result and optionally info2 from server to client over an integrity-protected channel

Not all requirements for enabling channel binding capabilities can be achieved by EAP [HC07]. Such requirements are outside the scope of channel binding as described in this document. The following discussion points out the prerequisites any system implementing EAP methods with channel binding MUST provide. For example, the third trust relationship cannot be achieved by EAP methods. The server MUST share an authentic channel with the authenticator, e.g. using an AAA protocol. This is necessary for the server to be able to authenticate the authenticator. Otherwise, the server cannot verify the received identifier/address as part of info2 and thus an authenticator sending false but identical information to client and server would remain undetected.

Furthermore, the server’s capability of checking inconsistencies of provided information might not be achievable by EAP and states another necessary system prerequisite. Once the server verified the validity of info2, it MUST verify whether the provided information is consistent with the information in info1. While comparing provided service information as part of info1 and info2 is simple, comparing the provided identifiers/addresses is not always that straightforward. The authenticator communicates on different protocol layers with client and server and thus may provide different identifiers/addresses to both parties. 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 authenticator, but may be of interest to a client. Hence, the server MUST be capable to verify whether both identifiers/addresses belong to the same entity.

5. Example Attacks

Section TBD.
6. Recommendations for Lower-Layer Bindings

This section discusses bindings to EAP lower layers, and what would be appropriate AVPs to transport to the server. We only define bindings for protocols with secure association protocols. Protocols without secure association do not derive session keys to protect data, and as such, channel bindings can do little to improve their link security. (Note: is this true?? Do we need to include PPP, 1X, etc???) Only if such keys exist, implementing channel binding as described in this draft is useful because all information exchanged between client and authentication server used for channel binding must be integrity-protected.

For each EAP lower layer, a variety of AAA properties may be of interest to the server. These values may already be known by the server, or may be transported to the server via an existing RADIUS or Diameter connection.

[Need to describe validation process ... TBD]

6.1. IEEE 802.11

The client SHOULD transmit to the server the following fields, encapsulated within the appropriate Diameter AVPs:

- SSID
- BSSID
- RSN IE (if present)

If the received information is unsatisfactory given some validation policy, the server SHOULD respond by halting the EAP authentication and returning an EAP-Failure.

If validation is successful, the server MAY respond back to the client with information encapsulated in AVPs that may be of use to the client, and information the client may not have any way of otherwise knowing. For example, the Cost-Information AVP from the Diameter Credit-Control Application [RFC4006] may be useful in telling the client how much they will be billed for service.

6.2. IEEE 802.16

TBD

6.3. Internet Key Exchange v2 (IKEv2)

TBD
7. Security Considerations

TBD

8. IANA Considerations

This document contains no IANA considerations.

9. References

9.1. Normative References


9.2. Informative References


Authors’ Addresses

T. Charles Clancy
DoD Laboratory for Telecommunications Sciences
8080 Greenmead Drive
College Park, MD 20740
USA

Email: clancy@ltsnet.net

Katrin Hoeper
National Institute of Standards and Technology
100 Bureau Drive, mail stop 8930
Gaithersburg, MD 20878
USA

Email: khoeper@nist.gov