EAP Working Group
INTERNET-DRAFT
Category: Informational
<draft-aboba-pppext-key-problem-05.txt>
21 December 2002

Bernard Aboba
Dan Simon
Microsoft

EAP Keying Framework

Status of this Memo

Copyright Notice

Abstract

This document describes the framework for key derivation by EAP methods and provides guidelines for generation and usage of keys derived by EAP methods requesting publication as an RFC. Algorithms for key derivation or mechanisms for key transport are not specified in this document. Rather, this document provides a framework within which derivation algorithms and transport mechanisms can be discussed and evaluated.

Table of Contents

1. Introduction

The Extensible Authentication Protocol (EAP), defined in [RFC2284bis], was originally developed to provide extensible authentication for use with PPP [RFC1661]. Since then, new applications of EAP have emerged, including IEEE 802.1X network port authentication [IEEE8021X].

The primary purpose of EAP is to authenticate an EAP Client to an EAP Server, as well as to provide keys for use with a link layer ciphersuite negotiated between an EAP Client and a Network Access Server (NAS). EAP can be deployed in configurations where the EAP Server and NAS are co-located, supporting a two-party exchange; alternatively, it can be deployed in configurations where the EAP Server is located on a separate entity, known as an Authentication Server. In this case, EAP supports a three-party exchange, where the Authentication Server acts as a Key Distribution Center (KDC), and the Authentication Server and NAS communicate using a AAA protocol supporting EAP as well as key wrap. Examples of AAA protocols supporting EAP include RADIUS [RFC2869bis], and Diameter [DiamEAP]; examples of AAA key wrap specifications include [RFC2548] and [DiamCMS].

EAP methods defined in [RFC2284bis] include EAP MD5, as well as One-Time Password (OTP) and Generic Token Card methods. Each of these methods supports one-way authentication only (EAP Client to EAP Server) but not key derivation. Since those methods do not support key derivation and do not provide for mutual authentication, they are only appropriate for use in situations where the link layer can be assumed to be physically secure. Where this is not the case, a session established over the link subsequent to authentication would be subject to hijacking, since without key derivation, it is not possible to tie the initial authentication to subsequent data traffic on a per-packet basis. These limitations can be overcome via negotiation of EAP methods such as EAP TLS [RFC2716] that support mutual authentication, as well as key derivation.

In order for EAP methods to provide appropriate keying material for link

layer ciphersuites, the keying requirements of the ciphersuites need to be understood and provided for. These requirements are discussed in Appendix A. With the increasing deployment of link layer ciphersuites, particularly with wireless networks, there is a need for a clear specification of what is expected of EAP methods deriving keys, as well as of ciphersuites utilizing keying material provided by EAP methods.

An overview of the EAP architecture is provided in Section 2, including in Section 2.1, a discussion of the implications for EAP methods generating keys. Section 3 describes both the two-party (Section 3.1) and the three-party (Section 3.2) exchanges. An introduction to the EAP key hierarchy is provided in Section 3.3.

Keying requirements are discussed in Section 4. This includes requirements for the EAP methods themselves (Section 4.1), the AAA protocols (Section 4.2), and the link layer ciphersuites (Section 4.3). Section 5 analyzes the security properties of both the two-way and three-way exchanges.

Appendix A provides a summary of the keying requirements of link layer ciphersuites supported on PPP and IEEE 802.11. Appendix B provides an example EAP Master Key (EMK) hierarchy. Appendix C provides an example Master Session Key (MSK) hierarchy.

1.1. 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 BCP 14 [RFC2119].

1.2. Terminology

This document frequently uses the following terms:

Client-Server Token (CS-Token)
: The package within which the MSK is transported between the EAP Server and the EAP Client. The package MUST be integrity protected, authenticated and encrypted, so as to protect the MSK from compromise. In addition to the MSK, the CS-Token MAY include one or more attributes providing information on MSK usage. Attributes may include the NAS layer 2 and layer 3 addresses, MSK lifetime, etc. The format of the CS-Token is defined by the EAP

method. Support for the CS-Token is optional and most current EAP methods do not support it, since they derive the MSK as part of the EMK key hierarchy, and thus do not need to transport it separately. However, in the case where an EAP Client needs to handle multiple MSKs, such as when it is connected to multiple NASes simultaneously, or where an Authentication Server is sending MSKs to multiple NASes in order to support fast handoff, use of methods supporting the CS-Token may be desirable.

AAA-NAS Token (AN-Token)
: The package within which the Master Session Keys (MSKs) and one or more AAA attributes is transported between the Authentication Server and NAS. The AAA attributes provide the NAS with information on MSK usage. For example, AAA attributes might include the Client layer 2 address, the NAS layer 2 and layer 3 addresses, MSK lifetime, etc. The format and wrapping of the AN-Token, which is intended to be accessible only to the Authentication Server and NAS, is defined by the AAA key distribution specification.

Authentication Server
: An Authentication Server is an entity that provides an Authentication Service to a Network Access Server (NAS). This service verifies from the credentials provided by the Client, the claim of identity made by the Client. Where an Authentication Server is provided, it acts as the EAP Server, terminating EAP conversation with the EAP Client. In the EAP Keying architecture the Authentication Server acts as a KDC, distributing the Master Session Keys (MSKs) to the EAP Client and NAS.

Cryptographic binding
: The demonstration of the EAP Client to the EAP Server that a single entity has acted as the EAP Client for all methods executed within a sequence or tunnel. Binding MAY also imply that the EAP Server demonstrates to the Client that a single entity has acted as the EAP Server for all methods executed within a sequence or tunnel. If executed correctly, binding serves to mitigate man-in-the-middle vulnerabilities.

Cryptographic separation
: Two keys (x and y) are "cryptographically separate" if an adversary that knows all messages exchanged in the protocol cannot compute x

from y or y from x without "breaking" some cryptographic assumption. In particular, this definition allows that the adversary has the knowledge of all nonces sent in cleartext as well as all predictable counter values used in the protocol. Breaking a cryptographic assumption would typically require inverting a one-way function or predicting the outcome of a cryptographic pseudo-random number generator without knowledge of the secret state. In other words, if the keys are cryptographically separate, there is no shortcut to compute x from y or y from x, but the work an adversary must do to perform this computation is equivalent to performing exhaustive search for the secret state value.

EAP Master key (EMK)
: The key derived between the EAP Client and Server during the EAP authentication process. The EMK is unique to the EAP Client and Server, and is not shared with any other parties.

Master Session Key (MSK)
: Keying material provided to the EAP Client and NAS by the AAA Server, acting as a Key Distribution Center (KDC). The MSK is used in derivation of Transient Session Keys for the ciphersuite negotiated between the EAP Client and NAS. So that the MSK is usable with any ciphersuite, it is longer than necessary, and is truncated to fit.

Note that an Authentication Server may simultaneously provide the EAP Client with MSKs suitable for use with multiple APs, so as to enable fast handoff. Similarly the AAA Server may send MSKs to multiple APs simultaneously. Note that where the AP supports transport of multiple MSK sets to the EAP Client and NASes, the MSKs MUST be kept cryptographically separate from each other.

Network Access Server (NAS)
: The device that provides access to the network. Where no Authentication Server is present, the NAS acts as the EAP Server, terminating the EAP conversation with the Client. Where an Authentication Server is present, the NAS may act as a pass-through for one or more authentication methods and for non-local users.

Pairwise Master Key (PMK)

As defined in [RFC2716], the MSK is 192 octets (1536 bits) in length. Octets 0-31 of the MSK are known as the "Peer to Authenticator Encryption Key" or Enc-RECV-Key (reception is defined from the point of view of the EAP Authenticator or NAS). Within IEEE 802.11, the Enc-RECV-Key is also known as the Pairwise Master Key (PMK). IEEE 802.11 ciphersuites such as TKIP, WRAP and CCMP derive their Transient Session Keys (TSKs) solely from the PMK, whereas the WEP ciphersuite, when used with IEEE 802.1X-2002, derives its TSKs from both the Enc-RECV-Key and the Enc-SEND-Key. Octets 32-63 of the MSK are known as the "Authenticator to Peer Encryption Key" or End-SEND-Key. Octets 64-95 are known as the "Peer to Authenticator Authentication Key" or Auth-RECV-Key. Octets 96-127 are known as the "Authenticator to Peer Authentication Key" or Auth-SEND-Key. Octets 128-159 are known as the "Peer to Authenticator IV" or RECV-IV, and Octets 160-191 are known as the "Authenticator to Peer IV", or SEND-IV.

Within [IEEE80211i], the Enc-RECV-Key is also known as the Pairwise Master Key (PMK). IEEE 802.11 ciphersuites such as TKIP, WRAP and CCMP derive their Transient Session Keys (TSKs) solely from the PMK, whereas the WEP ciphersuite, when used with IEEE 802.1X-2002, derives its TSKs from both the Enc-RECV-Key and the Enc-SEND-Key. IEEE 802.11 ciphersuites do not utilize the Auth-RECV-Key, Auth-SEND-Key, RECV-IV or SEND-IV, largely because attributes supporting transport of these portions of the MSK were not defined in [RFC2548].

Transient EAP Keys (TEKs)
: Session keys which are used to establish a protected channel between the EAP Client and Server during the EAP authentication exchange. The TEKs are derived from the EMK, and are appropriate for use with the ciphersuite negotiated between EAP Client and Server as part the EAP authentication exchange. Note that the

ciphersuite used to set up the protected channel between the EAP Client and Server during EAP authentication is unrelated to the ciphersuite used to subsequently protect data sent between the EAP Client and NAS. In particular, the TEKs used to protect the EAP exchange MUST be cryptographically separate from TSKs used to protect data.

Transient Session Keys (TSKs)

Session keys used to protect data which are appropriate for the ciphersuite negotiated between the EAP Client and NAS. The TSKs are derived from the MSK by a process defined by the link layer. In the case of IEEE 802.11, TSK derivation is supported via a 4-way handshake that supports mutual authentication between the EAP Client and NAS. The 4-handshake also confirms mutual possession of the PMK as well as supporting protected ciphersuite negotiation.

2. EAP architecture overview

EAP authentication involves a Client, NAS and (optionally) an Authentication Server. One of the goals of EAP is to enable development of new authentication methods without requiring deployment of new code on the NAS. While the NAS may implement some methods locally and use those methods to authenticate local users, it may at the same time act as a pass-through for other users and methods. Supporting pass-through of authentication to the Authentication Server enables the NAS to support any authentication method implemented on the Authentication Server and EAP Client, not just locally implemented methods. This implies that the NAS need not implement code for each EAP method required by authenticating Clients.

EAP presumes that prior to authentication, the EAP Client has located the NAS, using an out-of-band mechanism. For example, for use with PPP, the Client might be configured with a phone book providing phone numbers for accessing the selected service. For use with IEEE 802.11 wireless LANs, the Client (a Station (STA) in IEEE 802.11 terminology) may locate NAS devices (an Access Point (AP) in IEEE 802.l1 terminology) using the IEEE 802.11 Beacon and Probe Request/Response frames.

EAP also assumes that link layer ciphersuite negotiation is handled within the link layer. For example, the EAP Client might be preconfigured with policy indicating the ciphersuite to be used in communicating with a given NAS, or alternatively, the link layer protocol may support ciphersuite negotiation. Within PPP, the ciphersuite is negotiated within the Encryption Control Protocol (ECP), after EAP authentication is completed. Within IEEE 802.11i, the AP capabilities (including ciphersuite) are advertised in the Beacon and Probe Responses, and are verified during a 4-way exchange after EAP authentication has completed. The desired ciphersuite is indicated

within the Association/Reassociation Request/Response exchange.

Figure 1 illustrates the relationship between the EAP Client, NAS and Authentication Server (EAP Server) for the case where the NAS and Authentication Server are located on separate hosts, and the NAS acts as a pass-through.

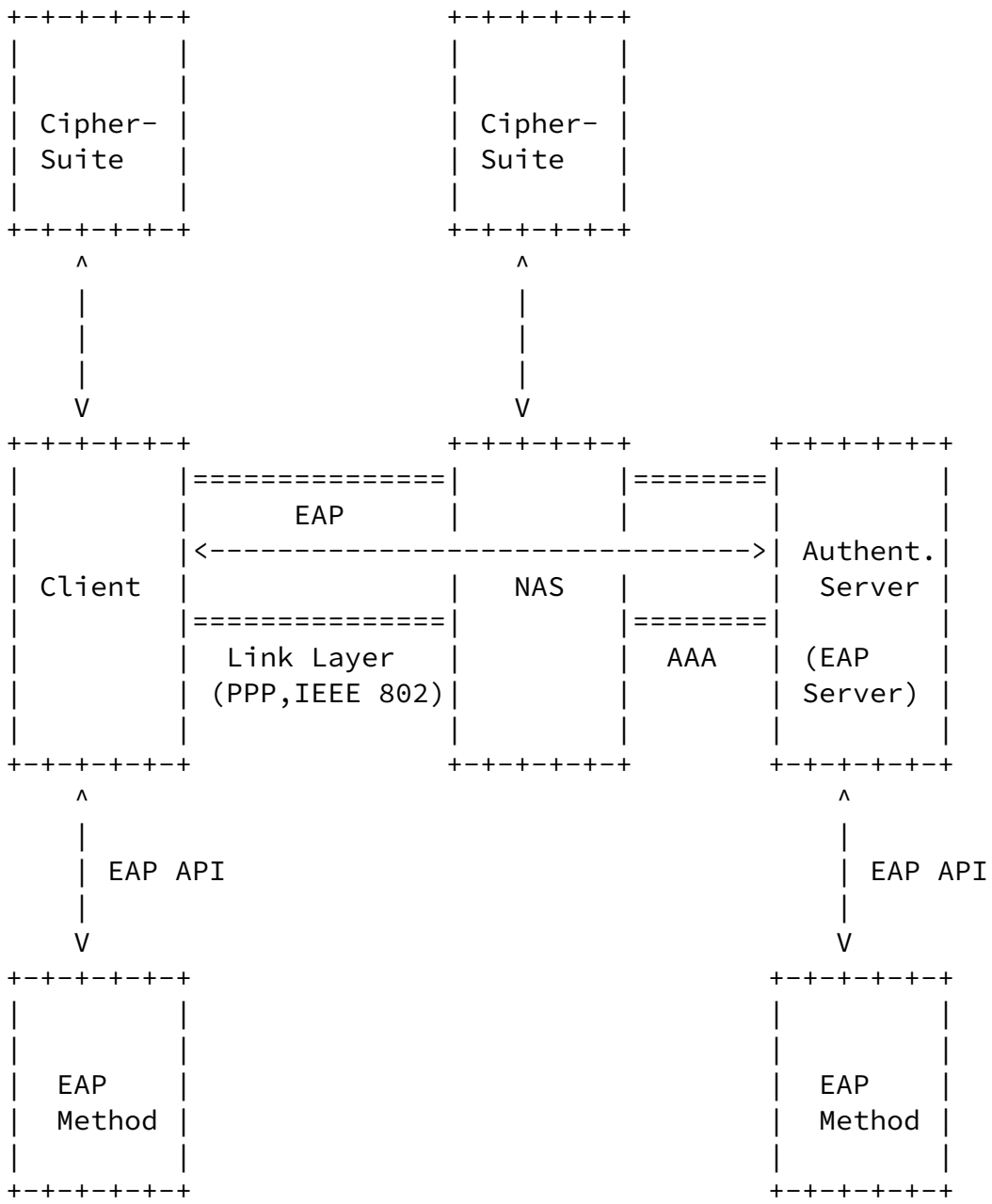


Figure 1 - Pass-through relationship between EAP Client, NAS and Authentication Server.

In the illustration, EAP is spoken between the Client and NAS, encapsulated within a link layer protocol, such as PPP, defined in [RFC1661] and IEEE 802, defined in [IEEE802]. The NAS then encapsulates

EAP within a AAA protocol such as RADIUS [RFC2869bis] or Diameter [DiamEAP], and transports this back and forth to the AAA Server, which acts as an EAP Server. Since the NAS acts as a pass-through, EAP methods reside only on the EAP Client and Server, interfacing to the operating system via an EAP API.

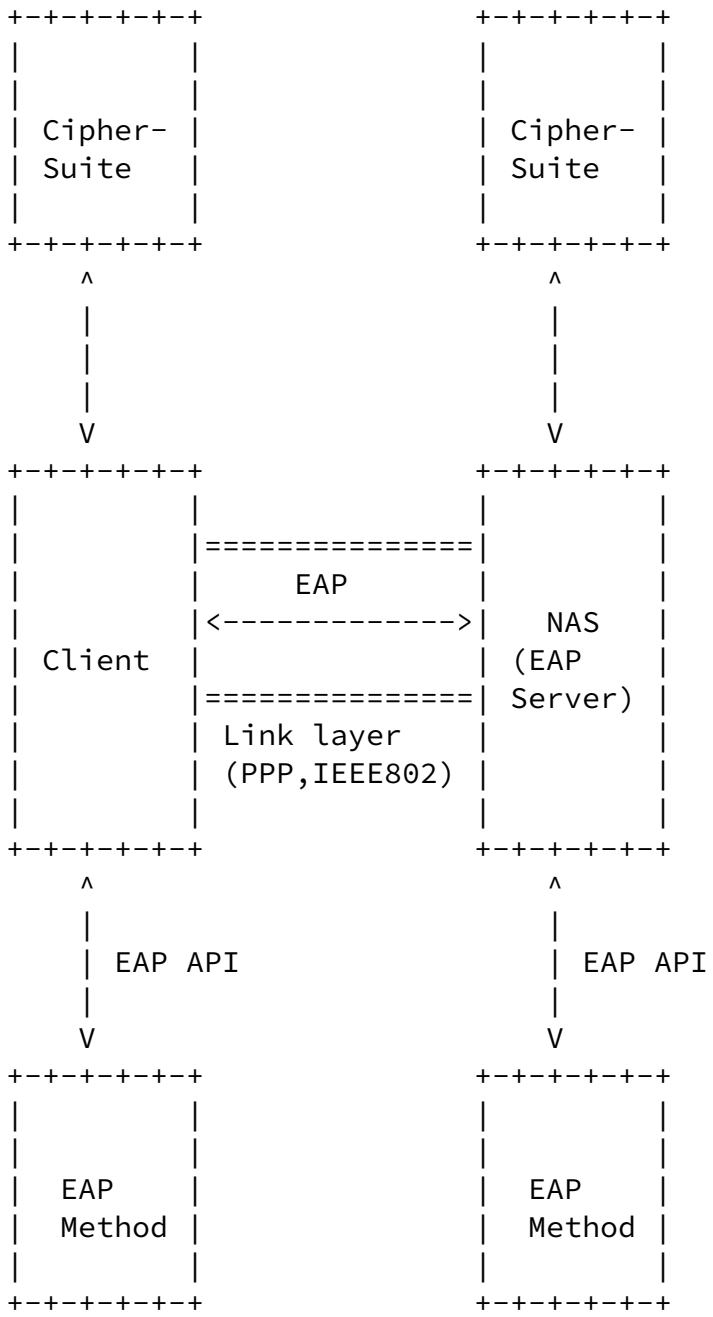


Figure 2 - Relationship between EAP Client and NAS (acting as an EAP Server) where no Authentication Server is present

Once EAP authentication is complete, the EAP Client and NAS pass data between each other, encapsulated within the link layer protocol. In order to protect the data, the Client and NAS may negotiate and subsequently implement a ciphersuite.

While pass-through operation is common with EAP, it is optional, so that EAP may also be implemented in situations where no Authentication Server is present. This is illustrated in Figure 2.

In Figure 2, EAP is spoken between the Client and NAS, encapsulated within a link layer protocol, such as PPP or IEEE 802. Since the NAS terminates the EAP conversation rather than acting as a pass-through, EAP methods are implemented on the NAS, as well as on the EAP Client, interfacing to the operating system via an EAP API.

Once EAP authentication is complete, the EAP Client and NAS pass data, encapsulated within the link layer protocol. In order to protect the data, the Client and NAS may negotiate and subsequently implement a ciphersuite.

2.1. Ciphersuite independence

Within the EAP authentication model, it is assumed that the ciphersuite is negotiated between the EAP Client and NAS using link layer mechanisms. While EAP methods which derive keys can be used to provide automated keying, the EAP method SHOULD NOT generate ciphersuite-specific keys or even contain ciphersuite-specific code. Since it is the Client and NAS that negotiate and implement the ciphersuite, knowledge of the ciphersusite is restricted to those entities.

Within the EAP 3-party model, the Authentication Server is not a party to the ciphersuite negotiation that occurs between the EAP Client and NAS, and neither is the Authentication Server involved in the passing of data between the EAP Client and NAS. Since the Authentication Server is not involved in the handling of data traffic, and may not even be aware of the ciphersuite negotiated between the EAP Client and NAS, it cannot be assumed to implement ciphersuite-specific code. In fact, the Authentication Server cannot even be assumed to have knowledge of the ciphersuites available on the NAS and EAP Client.

Since the Authentication Server may not know the ciphersuite negotiated between EAP Client and NAS, it will not necessarily be able to make this information available to a resident EAP method via the EAP APIs. As a result, ciphersuite-specific key generation implemented within an EAP method might not function correctly on every implementation.

Similarly, because the NAS is not required to implement any EAP methods, the NAS cannot be assumed to implement code specific to any EAP method.

Since operating systems provide EAP APIs in order to remain "EAP-Method Agnostic", EAP APIs cannot be assumed to implement EAP method-specific code either.

EAP methods deriving keys MUST support mutual authentication and provide for the derivation of an EAP Master Key (EMK), known only to the EAP Client and Server. EAP methods deriving keys also MUST provide for the distribution of the CS-Token between the AAA Server and EAP Client, and the AN-Token between the AAA server and NAS. The MSK contained within the CS-Token and AN-Tokens is suitable for use with any negotiated ciphersuite, and therefore an EAP method MUST NOT directly use the MSK as a Transient Session Key (TSK). Rather, the TSK(s) are derived from the MSK in a separate step, once the negotiated ciphersuite is known.

Drawbacks to utilizing the MSK as a transient session key include:

Ciphersuite negotiation
: Enabling derivation of the TSK(s) in a separate step provides for additional security. For example, the TSK derivation supported within IEEE 802.11i enables the EAP Client and NAS to mutually authenticate and conduct a protected ciphersuite negotiation. If the MSK is used directly as a TSK, then the EAP Client and NAS may not mutually authenticate each other, and a protected ciphersuite negotiation, if it occurs at all, would typically need to be supported within EAP itself. Since the ciphersuite negotiation mechanisms are typically particular to a given link layer, carrying this out within EAP may not be appropriate.

Document Revision
: If an EAP method specifies how to derive transient

session keys on a per-ciphersuite basis, the specification will need to be revised each time a new ciphersuite is developed. This would also imply that an Authentication Server supporting an EAP method might not be usable with all NASes supporting EAP, due to lack of support for a new ciphersuite implemented on a NAS.

EAP method complexity
: Requiring the EAP method to include ciphersuite-specific code for transient session key derivation increases the complexity of the EAP method, as well as Client and Authentication Server implementations.

Knowledge asymmetry
: In practice, an EAP method may not have knowledge of the ciphersuite that has been negotiated between the EAP

Client and NAS. In PPP, ciphersuite negotiation occurs via the Encryption Control Protocol (ECP), described in [RFC1968]. Since ECP negotiation occurs after authentication, unless an EAP method is utilized that supports ciphersuite negotiation, the Client, NAS and Authentication Server may not be able to anticipate the negotiated ciphersuite and therefore this information cannot be provided to the EAP method. Since ciphersuite negotiation is assumed to occur out-of-band of EAP, there is no need for ciphersuite negotiation within EAP.

3. EAP Exchanges

EAP supports two modes of exchange:

[a] Two-party exchange. The two-party exchange occurs where the EAP Client and NAS act as the endpoints of the EAP conversation, and no Authentication Server is present. Here the NAS implements the EAP method locally, rather than acting as a pass-through. In this mode, the EAP method used between the EAP Client and NAS (EAP Server) derives the EMK, as well as providing for the distribution of the Client-Server token containing the MSK.

[b] Three-party exchange. This mode is used where the NAS acts as a pass-through and an Authentication Server (acting as an EAP Server)

is present. In this mode, the EAP Server and Client derive the EMK, and the Authentication Server distributes to the CS-Token to the EAP Client and the AN-Token to the NAS. Both the CS-Token and AN-Token contain the embedded MSK.

3.1. Two-party exchange

Figure 3 illustrates the two-party exchange, where the Client is authenticated locally by the NAS using a locally implemented authentication method. In this case, the EAP Master Key (EMK) is derived on the Client and the NAS, which acts as the EAP Server during the EAP authentication exchange, and the Client-Server token is transported by the NAS to the EAP Client. The Client and NAS then use the MSK contained with the CS-Token to derive the transient session keys used with the selected ciphersuite. It is assumed that ciphersuite negotiation is handled out of band, rather than within EAP. For example, Within an IEEE 802.11 Reliable Secure Network (RSN), the TSK derivation occurs using the RSN 4-way handshake.

If the authentication occurs with a method not implemented on the NAS, or involves a non-local user whose credentials the Server is unable to validate, then the NAS functions as a "pass-through". For pass-through authentication methods, instead of implementing the authentication

method locally, the NAS delegates the authentication to an Authentication Server. The Authentication Server installs the desired EAP method, typically by interfacing with the operating system via an EAP API, such as that described in [EAPAPI].

In order to allow the Client and Authentication Server to install new EAP methods without requiring an operating system upgrade, operating systems isolate EAP method-specific code within the installed EAP methods, and thus largely operate as pass-through entities with respect to EAP.

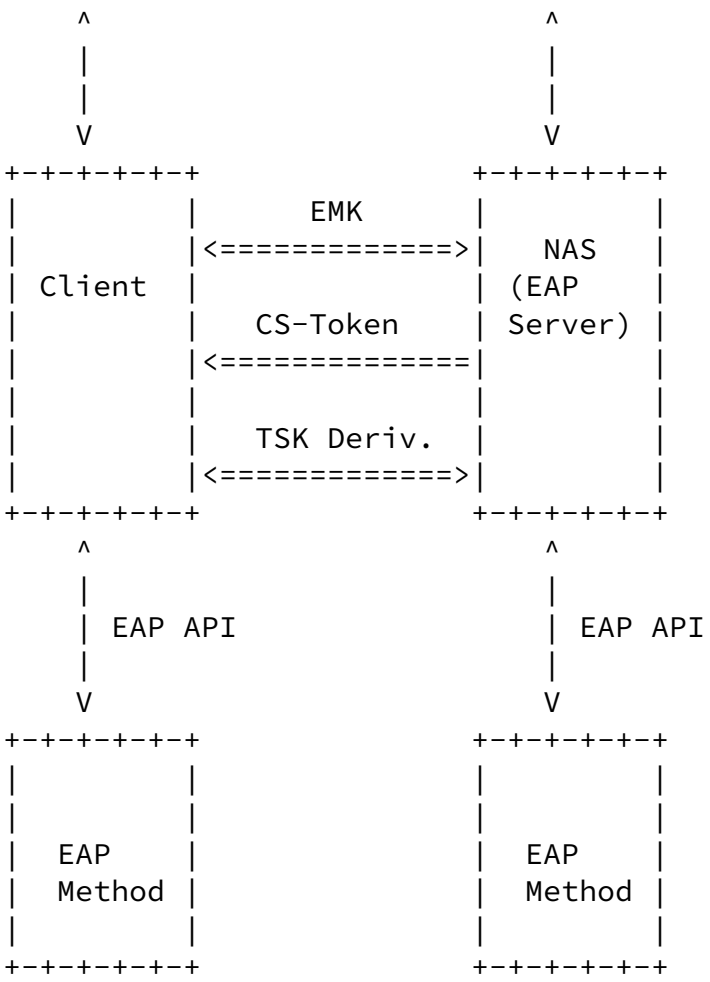


Figure 3 - Two-party exchange

3.2. Three-party exchange

Figure 4 illustrates a three-party exchange where the NAS acts as a pass-through. As described in the figure, the EAP conversation "passes through" the NAS on its way between the Client and the Authentication Server (which acts as the EAP Server).

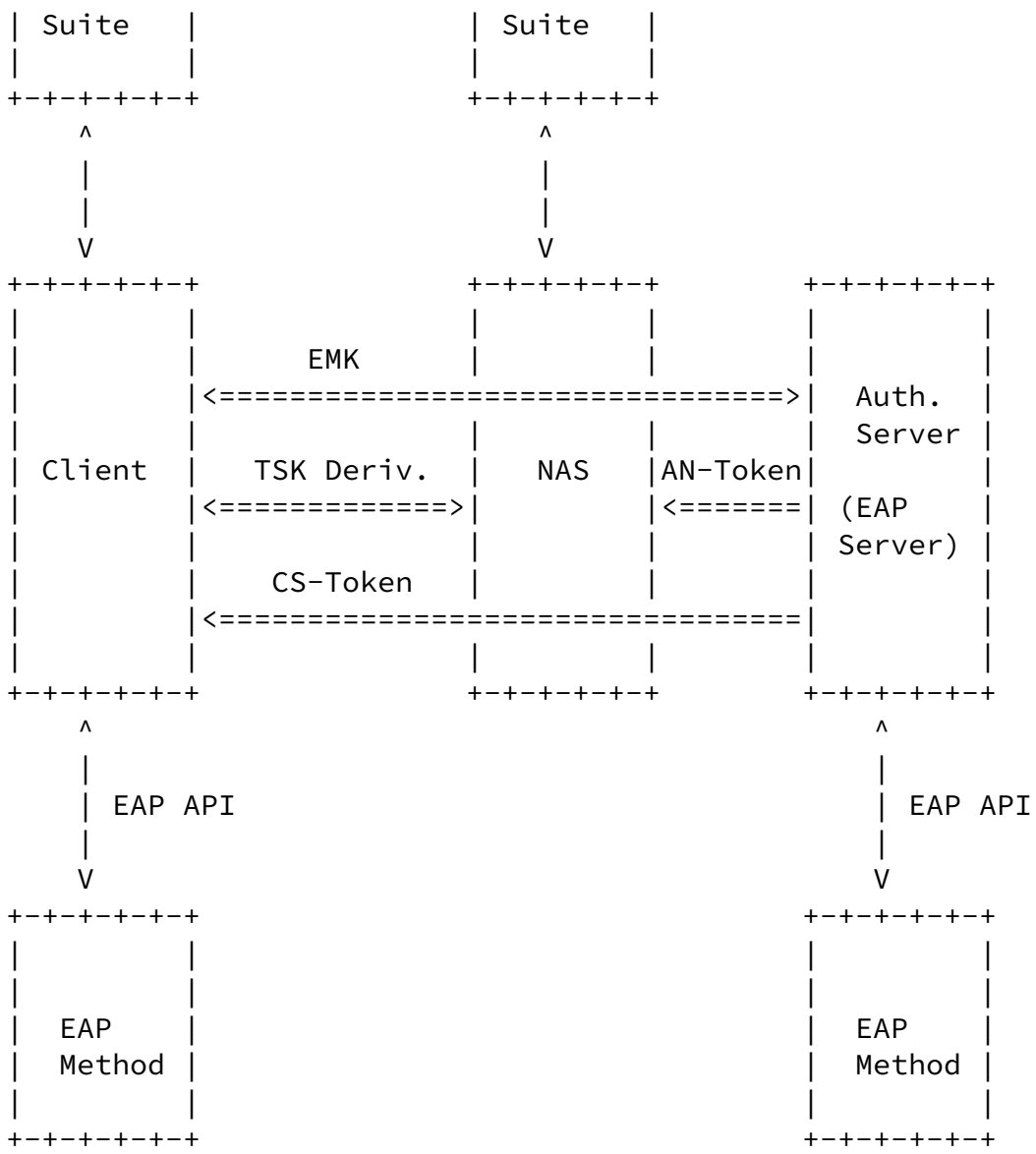


Figure 4 - Three-way exchange

The three-way EAP exchange takes part in several phases:

[a] EAP authentication. During this phase, the EAP Client and Server mutually authenticate and derive the EMK, which is known only to

the EAP Client and Server. Since possession of the EMK would enable a third party to impersonate the EAP Client or Server, the EMK MUST NOT be shared with any other party. Where the NAS acts as a pass-through, it does not participate in the EAP conversation, except to forward packets between the EAP Client and the Authentication Server. As a result, the NAS does not possess the EMK and MUST NOT be able to derive it, based on observing the EAP conversation, or obtaining the MSK.

[b] Token distribution. During this phase, the AAA Server acts as a Key Distribution Center (KDC), distributing the CS-Token to the EAP Client and the AN-Token to the NAS. These tokens, which are defined in the EAP Method and AAA key distribution specifications, respectively, contain the MSK.

[c] TSK derivation. During this phase, the EAP Client and NAS confirm mutual possession of the MSK, and derive the Transient Session Keys used in the negotiated ciphersuite. TSK derivation occurs out of band of EAP; an example is the 4-way handshake provided in IEEE 802.11 RSN.

Figure 5 below illustrates the relationship between the parties in the three-way exchange.

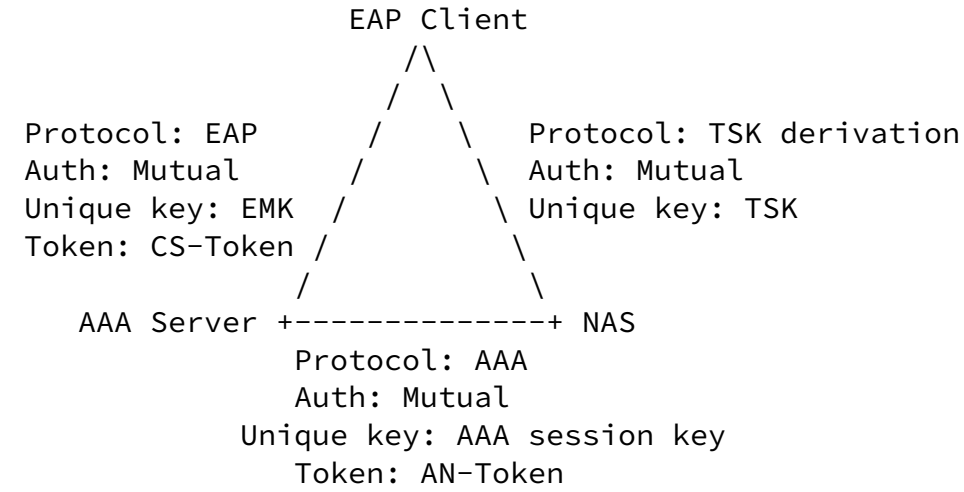


Figure 5: Three-party EAP key distribution

Where key distribution is supported, the EAP Client and Authentication Server (EAP Server) MUST mutually authenticate via the negotiated EAP method, and derive keys only known between the EAP Client and Server, known as the EMK. During EAP authentication, the CS-Token MAY be

transported from the EAP Server to the EAP Client, providing the Client with the MSK. Alternatively, the MSK MAY be derived from the EMK, via a one-way function. Whether the MSK is derived or transported, possession of the MSK MUST NOT provide information useful in determining the EMK.

Utilizing the AAA protocol, the Authentication Server and NAS MUST mutually authenticate, and derive a protected channel which MUST provide per-packet integrity protection, authentication and confidentiality. The AN-Token is distributed by the Authentication Server to the NAS over this channel. Where possible, the channel between the Authentication Server and NAS SHOULD be protected using a session key, as in [DiamEAP] and RADIUS over IPsec [RFC3162], rather than using a static key, as in RADIUS [RFC2865].

During the (optional) TSK derivation step, the EAP Client and NAS MUST mutually authenticate by providing mutual posession of the portion of the MSK used in the derivation. The TSK derivation step SHOULD also provide for a protected ciphersuite negotiation between the EAP Client and NAS.

The security of the three-party exchange is highly dependent on the security properties of the algorithms chosen. For example, if mutual authentication is not completed between the EAP Client and Authentication server, then the Client will be vulnerable to rogue Authentication Servers and NASes. If the EMK is not derived between the Client and Authentication Server, then there will be no binding between the authentication and subsequent data traffic, leaving the session vulnerable to hijack.

If the Authentication Server and NAS do not mutually authenticate, then the the EAP Client will once again be vulnerable to rogue Authentication Servers, NASes or both. If there is no per-packet authentication, integrity and replay protection between the Authentication Server and NAS, then the EAP conversation could be modified in transit, or packets can spoofed.

If the TSK derivation does not support mutual authentication, then the EAP Client will not have assurance that it is connected to the right NAS, only that the NAS and AAA server share a trust relationship (assuming that the AAA protocol supports mutual authentication). This distinction can become important when multiple NASes receive MSKs from the Authentication Server, as may be the case where fast handoff is supported. If the TSK derivation does not provide for protected ciphersuite negotiation, then downgrade attacks are possible.

3.3. EAP Key hierarchy

The EAP key hierarchy depends on two branches:

[a] EAP Master Key (EMK) branch. The EMK is derived during the EAP conversation between the EAP Client and Server, and TEKs derived from it are used to establish a protected channel between the EAP Client and Server. Therefore, the EMK branch of the EAP key hierarchy describes the derivation of keys used to protect the EAP exchange itself.

Since the EMK is uniquely held by the EAP Client and Server, and only mutually authenticating EAP methods may distribute keys, proof of possession of the EMK is proof of a completed mutual authentication. In order to ensure that the NAS does not possess the EMK, which could be used to impersonate the EAP Client or EAP Server, the EMK MUST NOT be provided to third parties such as the NAS, or be derivable from other keying material such as the MSK. In order to protect against compromise of the EMK, the EMK MUST NOT be directly used to protect data; rather the TEKs derived from the EMK are used for this purpose. Examples of the EMK branch of the key hierarchy are given in Appendix A.

[b] Master Session Key (MSK) branch. The MSK is (optionally) distributed by the Authentication Server to the EAP Client within the CS-Token (or alternatively, derived from the EMK). It is transported from the Authentication Server to the NAS within the AN-Token. Since the MSK is not ciphersuite-specific, it is larger than necessary, and is truncated to fit as part of the Transient Session Key (TSK) derivation process. As with the EMK, the MSK MUST NOT be directly used to protect data; rather TSKs derived from the MSK are used for this purpose. Examples of the MSK hierarchy are given in Appendix B.

4. Security considerations

This section describes the security requirements for EAP methods, AAA protocols, TSK derivation mechanisms and Ciphersuites involved in three-party EAP exchanges. These requirements MUST be met by specifications requesting publication as an RFC.

4.1. Three-party exchange

The security of the three-party exchange is highly dependent on the security properties of the each of the protocols. For example, if mutual authentication is not completed between the EAP Client and Authentication server, then the Client will be vulnerable to rogue Authentication Servers and NASes. If the EMK is not derived between the

Client and Authentication Server, then there will be no binding between the authentication and subsequent data traffic, leaving the session vulnerable to hijack.

If the Authentication Server and NAS do not mutually authenticate, then the the EAP Client will once again be vulnerable to rogue Authentication Servers, NASes or both. If there is no per-packet authentication, integrity and replay protection between the Authentication Server and NAS, then the EAP conversation could be modified in transit, or packets can spoofed.

If the TSK derivation does not support mutual authentication, then the EAP Client will not have assurance that it is connected to the right NAS, only that the NAS and AAA server share a trust relationship (assuming that the AAA protocol supports mutual authentication). This distinction can become important when multiple NASes receive MSKs from the Authentication Server, as may be the case where fast handoff is supported. If the TSK derivation does not provide for protected ciphersuite negotiation, then downgrade attacks are possible. As a result, where physical security cannot be assumed, or roaming is supported, the TSK derivation step SHOULD NOT be ommitted.

4.2. EAP method requirements

EMK hierarchy
: Methods deriving keys MUST support mutual authentication and derivation of the EMK, as well as specifying how TEKs are derived from the EMK. The EMK MUST NOT be used to directly protect data.

CS-Token specification
: Methods supporting key derivation MUST specify the format of the CS-Token containing the MSK. If no explicit CS-Token format is used, then the formulas for derivation of the MSK MUST be provided.

MSK hierarchy
: For a ciphersuite to be suitable for use with dynamic keying via EAP a specification MUST be provided describing how TSKs are derived from the MSK.

Cryptographic Separation
: Methods supporting key derivation MUST demonstrate cryptographic separation between the TEKs and TSKs. Also, it must be demonstrated that possession of the MSK does not provide information useful in determining the EMK.

Ciphersuite Independence
: The MSK derivation SHOULD be ciphersuite-independent and the EAP method SHOULD NOT assume knowledge of the ciphersuite.

Key size
: An EAP method supporting key derivation MUST generate a 192 octet MSK.

Key Entropy
: The strength of session keys is dependent upon the security of the EAP method. If the chosen EAP method has security vulnerabilities, or does not produce an EMK and MSK of sufficient entropy then the security of the three-party exchange is reduced. An EAP method supporting key derivation SHOULD generate an EMK and MSK with at least 128 bits of entropy.

Session Uniqueness
: In order to assure non-repetition of TSKs even in cases where one party may not have a high quality random number generator, the MSK derivation SHOULD include a two-way nonce exchange, using nonces of at least 128-bits. Note although the IEEE 802.11 RSN TSK derivation includes a nonce exchange, the TSK derivation step is link layer dependent, so that a link layer nonce exchange cannot be guaranteed to occur. As a result, a nonce exchange is still needed within the EAP method itself. A nonce exchange SHOULD also be included in the derivation of the TEKs from the EMK.

Known-good algorithms
: The development and validation of key derivation algorithms is difficult, and as a result EAP methods SHOULD reuse existing

algorithms, rather than inventing new ones. EAP methods requesting publication as an RFC MUST provide citations to literature justifying the security of the chosen algorithms. EAP methods SHOULD utilize well established and analyzed mechanisms for EMK and MSK derivation.

4.3. AAA protocol requirements

AAA protocols suitable for use with EAP MUST provide the following facilities:

AN-Token specification
: In order to enable Authentication Servers to provide keying material to the NAS in a well defined format, AAA protocols suitable for use with EAP MUST define the format and wrapping of the AN-Token.

AN-Token protection
: To ensure against compromise, the AN-Token MUST be integrity protected, authenticated and encrypted in transit, using well-established cryptographic algorithms. In order to protect the AN-Token from modification by AAA intermediaries, where untrusted

intermediaries are present, it SHOULD be protected using well-established algorithms, such as is described in Diameter CMS Security [DiamCMS], a work in progress. Proper key hygiene is critical for protection of the AN-Token, which SHOULD protected with session keys as in Diameter CMS Security [DiamCMS] (a work in progress) or RADIUS over IPsec [RFC3162] rather than static keys, as in [RFC2548].

4.4. Ciphersuite requirements

Ciphersuites suitable for keying by EAP methods MUST provide the following facilities:

TSK derivation
: In order to key a ciphersuite with EAP, it is necessary to specify how the TSKs required by the ciphersuite are derived from the MSK. Derivation of the TSKs keys from MSK requires knowledge of the negotiated ciphersuite.

TEK derivation
: In order to establish a protected channel between the EAP Client and Server as part of the EAP exchange, a ciphersuite needs to be negotiated and keyed, using TEKs derived from the EMK. The ciphersuite used to protect the EAP exchange is distinct from the ciphersuite negotiated between the EAP client and NAS, used to protect data. Where a protected channel is established within the EAP method, the method specification MUST specify the mechanism by which the EAP ciphersuite is negotiated, as well as the algorithms for derivation of TEKs from the EMK during the EAP authentication exchange.

EAP method independence
: Algorithms for deriving TSKs from the MSK MUST NOT depend on the EAP method. However, algorithms for deriving TEKs from the EMK MAY be specific to the EAP method.

Cryptographic separation
: The TSKs derived from the MSK MUST be cryptographically separate from each other. Similarly, TEKs MUST be cryptographically separate from each other. In addition, the TSKs MUST be cryptographically separate from the TEKs.

5. Normative References

[RFC1661] Simpson, W., Editor, "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, July 1994.

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

[RFC2284bis] Blunk, L., Vollbrecht, J., Aboba, B., "Extensible Authentication Protocol (EAP)", Internet draft (work in progress), draft-ietf-pppext-rfc2284bis-08.txt, December 2002.

[IEEE80211] 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. 802.11-1997, 1997.

[IEEE8021X] IEEE Standards for Local and Metropolitan Area Networks: Port based Network Access Control, IEEE Std 802.1X-2001, June 2002.

6. Informative References

[RFC1968] Meyer, G., "The PPP Encryption Protocol (ECP)", RFC 1968, June 1996.

[RFC2104] Krawczyk, et al, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[RFC2246] Dierks, T. and Allen, C. "The TLS Protocol Version 1.0", RFC 2246, November 1998.

[RFC2409] Harkins, D., Carrel, D., "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[RFC2419] Sklower, K., Meyer, G., "The PPP DES Encryption Protocol, Version 2 (DESE-bis)", RFC 2419, September 1998.

[RFC2420] Hummert, K., "The PPP Triple-DES Encryption Protocol (3DESE)", RFC 2420, September 1998.

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

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

[RFC2716] Aboba, B., Simon, D.,"PPP EAP TLS Authentication Protocol", RFC 2716, October 1999.

[RFC2865] Rigney, C., Willens, S., Rubens, A., Simpson, W., "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC3078] Pall, G. and Zorn, G. "Microsoft Point-to-Point Encryption (MPPE) RFC 3078, March 2001.

[RFC3079] Zorn, G. "Deriving Keys for use with Microsoft Point-to-Point Encryption (MPPE)," RFC 3079, March 2001.

[RFC3394] R. Housley, "Advance Encryption Standard (AES) Key Wrap Algorithm", RFC 3394, September 2002.

[FIPSDES] National Bureau of Standards, "Data Encryption Standard", FIPS PUB 46 (January 1977).

[DESMODES] National Bureau of Standards, "DES Modes of Operation", FIPS PUB 81 (December 1980).

[FIPS197] FIPS PUB 197, Advanced Encryption Standard (AES), 2001 November 26H.

[SHA] National Institute of Standards and Technology (NIST), "Announcing the Secure Hash Standard," FIPS 180-1, U.S. Department of Commerce, 04/1995

[IEEE80211i] IEEE Draft 802.11i/D3, "Draft Supplement to STANDARD FOR Telecommunications and Information Exchange between Systems - LAN/MAN Specific Requirements - Part 11: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: Specification for Enhanced Security", November 2002.

[EAPAPI] Microsoft Developer Network, "Windows 2000 EAP API", August 2000, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/eap/eapport_0fj9.asp

[RFC2869bis] Aboba, B., Calhoun, P., "RADIUS Support For Extensible Authentication Protocol (EAP)", Internet draft (work in progress), draft-aboba-radius-rfc2869bis-05.txt, December 2002.

[DiamCMS] Calhoun, P., Farrell, S., Bulley, W., "Diameter CMS Security Application", Internet draft (work in progress), draft-ietf-aaa-diameter-cms-sec-04.txt, March 2002.

[DiamEAP] Hiller, T., Zorn, G., "Diameter Extensible Authentication Protocol (EAP) Application", Internet draft (work in progress), draft-ietf-aaa-eap-00.txt, June 2002.

Appendix A - Ciphersuite keying requirements

To date, PPP and IEEE 802.11 ciphersuites are suitable for keying by EAP. This Appendix describes the transient session keying requirements of common PPP and 802.11 ciphersuites.

PPP ciphersuites include DESEbis [RFC2419], 3DES [RFC2420], and MPPE [RFC3078]. The DES algorithm is described in [FIPSDES], and DES modes (such as CBC, used in RFC 2419 and DES-EDE3-CBC, used in RFC 2420) are described in [DESMODES]. For PPP DESEbis, a single 56-bit encryption key is required, used in both directions. For PPP 3DES, a 168-bit encryption key is needed, used in both directions. As described in [RFC2419] for DESEbis and [RFC2420] for 3DES, the IV, which is different in each direction, is "deduced from an explicit 64-bit nonce, which is exchanged in the clear during the [ECP] negotiation phase."

For MPPE, 40-bit, 56-bit or 128-bit encryption keys can be required in each direction, as described in [RFC3078]. Since MPPE is based on the RC4 algorithm, no initialization vector is required.

While these PPP ciphersuites provide encryption, they do not provide a per-packet keyed message integrity check (MIC). Thus, an authentication key is not required in either direction.

Within 802.11, transient session keys are required both for unicast traffic as well as for multicast traffic, and therefore separate TSK hierarchies are required for unicast keys and multicast keys. IEEE 802.11 ciphersuites include WEP-40, described in [IEEE80211], which requires a 40-bit encryption key, the same in either direction; and WEP-128, which requires a 104-bit encryption key, the same in either direction. These ciphersuites also do not include a keyed MIC, so that an authentication key is not required in either direction. However, in order to protect the transport of the multicast keys from the Access Point to the Station, additional authentication and encryption keys are required.

Recently, new ciphersuites have been proposed for use with 802.11 that provide per-packet authentication as well as encryption [IEEE80211i]. This includes TKIP, which requires a single 128-bit encryption key and a 128-bit authentication key (used in both directions); AES CCMP, which requires a single 128-bit key (used in both directions) in order to authenticate and encrypt data; and WRAP, which requires a single 128-bit

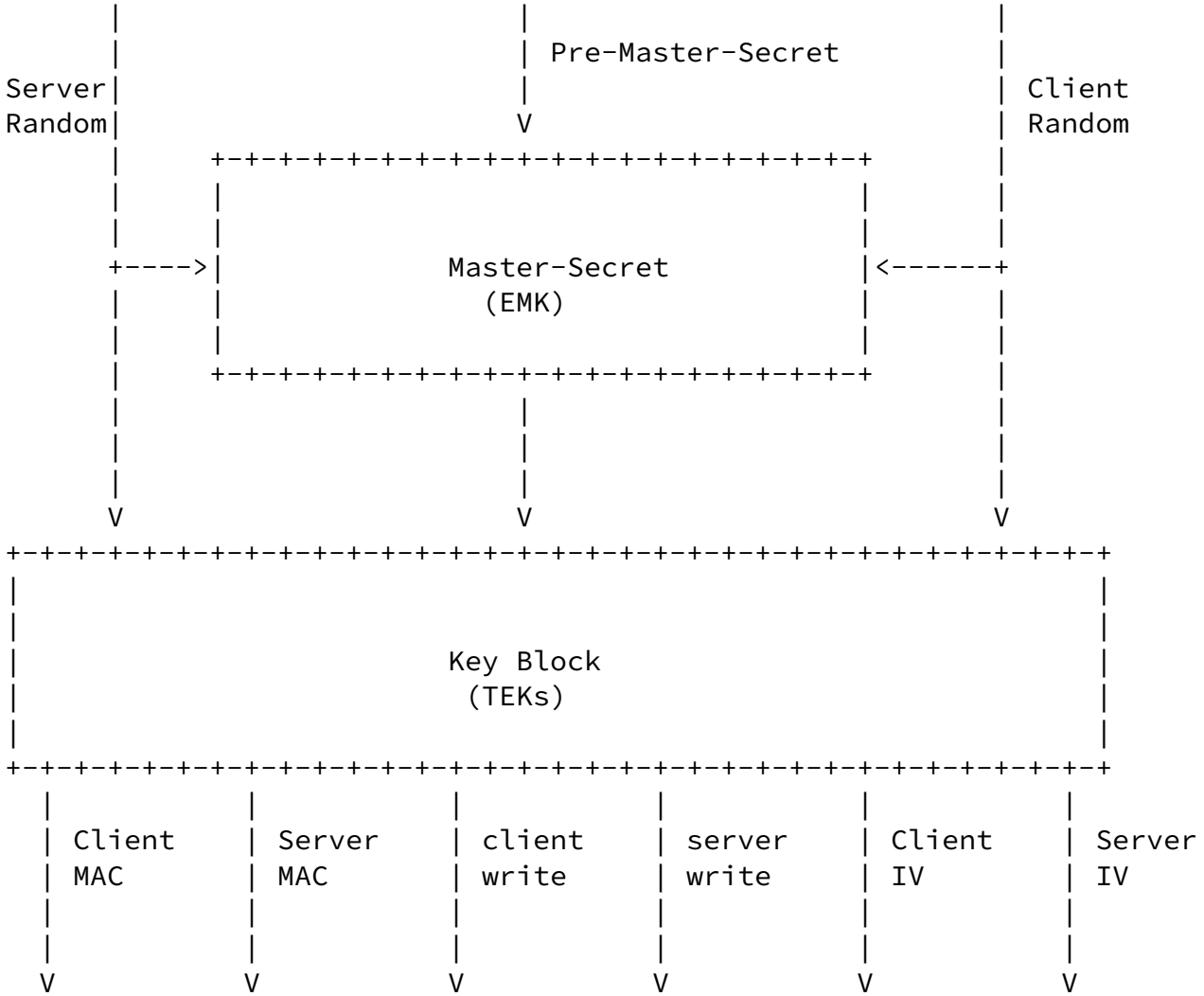
key (used in both directions).

Appendix B - Example EMK Hierarchy

In EAP TLS [RFC2716], ciphersuite negotiation and derivation of the TEKs is provided using the Transport Layer Security (TLS) key hierarchy specified in [RFC2246]. The TLS-negotiated ciphersuite is used to set up a protected channel, keyed by derived TEKs. The TEK derivations proceeds as follows:

```
Master_secret = TLS-PRF(Pre-Master-Secret, "master secret" ||
                server.random || client.random)
TEK = TLS-PRF-X(Master-Secret, "key expansion", server.random || client.random)

Where:
TLS-PRF-X =     TLS pseudo-random function defined in [RFC2246],
                computed to X octets.
Master-Secret = TLS term for the EMK.
```

Figure B-1 illustrates the EMK key hierarchy, which is derived from the TLS key hierarchy described in [RFC2246].


Pre-Master-Secret
Server Random
Client Random
Master-Secret (EMK)
Key Block (TEKs)
Client MAC
Server MAC
client write
server write
Client IV
Server IV

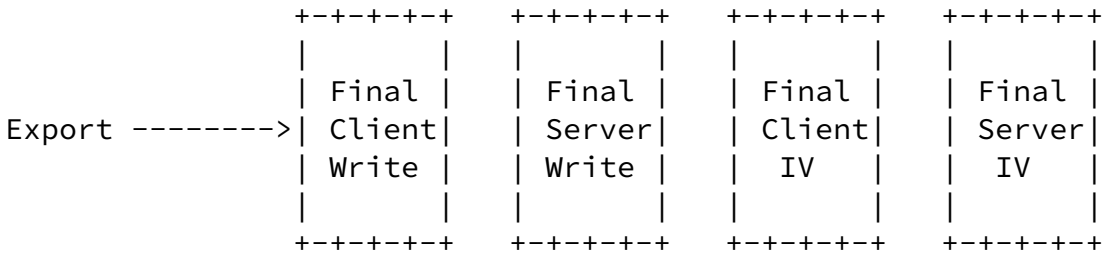


Figure B-1 - TLS [RFC2246] Key Hierarchy

Appendix C - Example MSK Hierarchy

In EAP TLS [RFC2716], the MSK is not transported within a CS-Token package. Rather, the MSK is derived from the EMK via a one-way function. This ensures that the EMK cannot be derived from the MSK unless the one-way function (TLS PRF) is broken.

Since the MSK is derived from the EMK, if the EMK is compromised then the MSK is also compromised. However, this would be the case even if the MSK were cryptographically separate from the EMK, since TEKs derived from the EMK are used to protect the CS-Token containing the MSK. Thus if the EMK is compromised, so are the TEKs, the CS-token and ultimately the MSK.

As described in [RFC2716], the formula for the derivation of the MSK from the EMK is as follows:

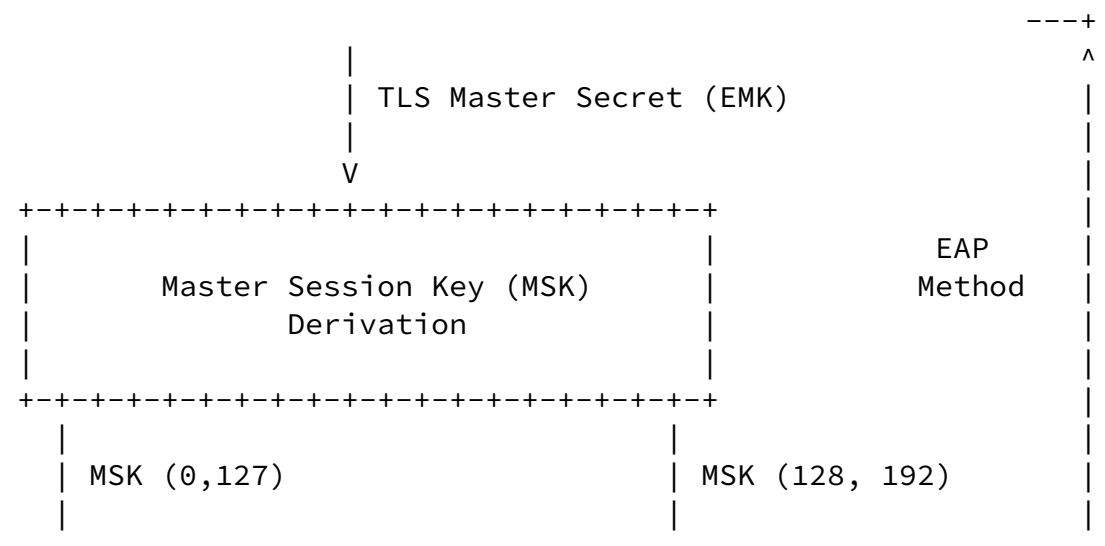
```
MSK(0,127)  = TLS-PRF-128(EMK, "client EAP encryption", client.random || server
MSK(128,191)= TLS-PRF-64("", "client EAP encryption", client.random || server.r

MSK(0,31)   = Peer to Authenticator Encryption Key (Enc-RECV-Key)
```

```
                (MS-MPPE-Recv-Key in [RFC2548])
MSK(32,63)  = Authenticator to Peer Encryption Key (Enc-SEND-Key)
                (MS-MPPE-Send-Key in [RFC2548])
MSK(64,95)  = Peer to Authenticator Authentication Key (Auth-RECV-Key)
MSK(96,127) = Authenticator to Peer Authentication Key (Auth-Send-Key)
MSK(128,159)= Peer to Authenticator Initialization Vector (RECV-IV)
MSK(160,191)= Authenticator to Peer Initialization vector (SEND-IV)
```

Where:

```
MSK(W,Z)      = Octets W through Z inclusive of the MSK.
EMK           = TLS master secret
TLS-PRF-X     = TLS PRF function defined in [RFC2246] computed to X octets
client.random = Nonce generated by the TLS client.
server.random = Nonce generated by the TLS server.
```

Figure C-1 describes the process by which the MSK, and ultimately the TSKs, are derived from the EMK. Note that in [RFC2716], the EMK is referred to as the "TLS Master Secret".



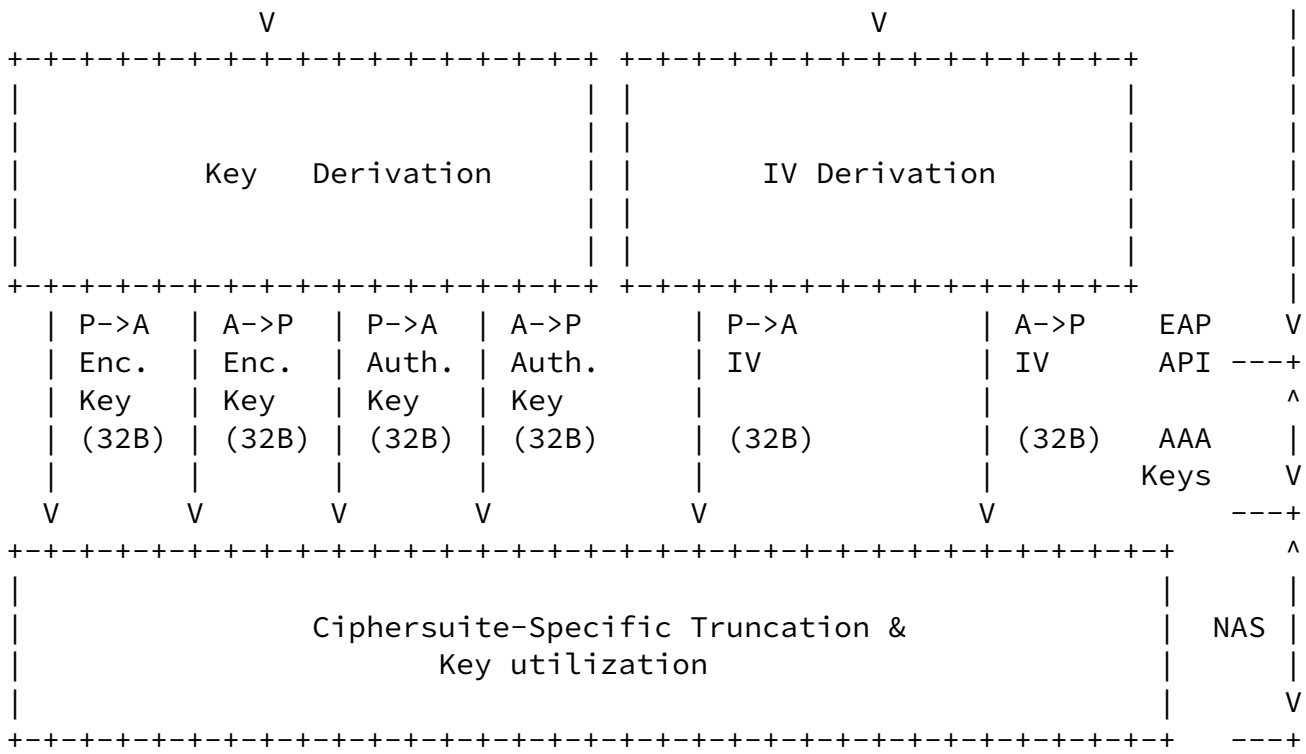


Figure C-1 - EAP TLS [RFC2716] MSK hierarchy

Within IEEE 802.11 RSN, the Pairwise Transient Key (PTK), a transient session key used to protect unicast traffic, is derived from the PMK (octets 0-31 of the MSK), otherwise known as the Peer to Authenticator Encryption Key. Within [RFC2548], the PMK is transported via the MS-MPPE-Recv-Key attribute. In IEEE 802.11 RSN, the PTK is derived from the PMK via the following formula:

```
PTK = EAPOL-PRF-X(PMK, "Pairwise key expansion" || Min(AA,SA) ||
      Max(AA, SA) || Min(ANonce,SNonce) || Max(ANonce,SNonce))
```


Where:

```
PMK             = MSK(0,31)
SA              = Station MAC address
AA              = AP MAC address
ANonce          = AP Nonce
SNonce          = Station Nonce
```

EAPOL-PRF-X = Pseudo-Random Function based on HMAC-SHA1, generating a PTK of size X.

TKIP uses X = 512, while CCMP, WRAP, and WEP use X = 384.

The EAPOL-Key Confirmation Key (KCK) is used to provide data origin authenticity in the TSK derivation. It utilizes the first 128 bits (bits 0-127) of the PTK. The EAPOL-Key Encr. Key (KEK) provides confidentiality in the TSK derivation. It utilizes bits 128-255 of the PTK. Bits 256-383 of the PTK are used by Temporal Key 1, and Bits 384-511 are used by Temporal Key 2. Usage of TK1 and TK2 is ciphersuite specific. Additional details are available in [IEEE80211i].

Acknowledgments

Thanks to Arun Ayyagari, Ashwin Palekar, and Tim Moore of Microsoft, Dorothy Stanley of Agere, Dave Halasz of Cisco Systems, and Russ Housley of RSA Security for useful feedback.

Author Addresses

Bernard Aboba
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

EMail: bernarda@microsoft.com
Phone: +1 425 706 6605
Fax: +1 425 936 7329

Dan Simon
Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

EMail: dansimon@microsoft.com
Phone: +1 425 706 6711
Fax: +1 425 936 7329

Intellectual Property Statement

Full Copyright Statement

Open issues

Open issues relating to this specification are tracked on the following web site:

http://www.drizzle.com/~aboba/EAP/eapissues.html

Expiration Date

This memo is filed as <draft-aboba-pppext-key-problem-05.txt>, and expires July 22, 2003.