EMU Working Group
Internet-Draft
Intended status: Informational
Expires: June 19, 2011

K. Hoeper
Motorola, Inc.
S. Hanna
Juniper Networks
H. Zhou
J. Salowey, Ed.
Cisco Systems, Inc.

December 16, 2010

Requirements for a Tunnel Based EAP Method
draft-ietf-emu-eaptunnel-req-09.txt

Abstract

This memo defines the requirements for a tunnel-based Extensible Authentication Protocol (EAP) Method. This tunnel method will use Transport Layer Security (TLS) to establish a secure tunnel. The tunnel will provide support for password authentication, EAP authentication and the transport of additional data for other purposes.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

An Extensible Authentication Protocol (EAP) tunnel method is an EAP method that establishes a secure tunnel and executes other EAP methods under the protection of that secure tunnel. An EAP tunnel method can be used in any lower layer protocol that supports EAP authentication. There are several existing EAP tunnel methods that use Transport Layer Security (TLS) to establish the secure tunnel. EAP methods supporting this include Protected EAP (PEAP) [PEAP], Tunneled Transport Layer Security EAP (TTLS) [RFC5281] and EAP Flexible Authentication via Secure Tunneling (EAP-FAST) [RFC4851]. In general this has worked well so there is consensus to continue to use TLS as the basis for a tunnel method. There have been various reasons for employing a protected tunnel for EAP processes. They include protecting weak authentication exchanges, such as username and password. In addition a protected tunnel can provide means to provide peer identity protection and EAP method chaining. Finally, systems have found it useful to transport additional types of data within the protected tunnel.

This document describes the requirements for a EAP tunnel method as well as for a password protocol supporting legacy password

verification within the tunnel method.

2. Conventions Used In This Document

Use of each capitalized word within a sentence or phrase carries the following meaning during the EAP Method Update (EMU) WG's method selection process:

MUST - indicates an absolute requirement

MUST NOT - indicates something absolutely prohibited

SHOULD - indicates a strong recommendation of a desired result

SHOULD NOT - indicates a strong recommendation against a result

MAY - indicates a willingness to allow an optional outcome

Lower case uses of "MUST", "MUST NOT", "SHOULD", "SHOULD NOT" and "MAY" carry their normal meaning and are not subject to these definitions.

3. Use Cases

To motivate and explain the requirements in this document, a representative set of use cases for the EAP tunnel method are supplied here. It is mandatory for a candidate tunnel method to support all of the use cases that are marked below as "MUST".

3.1. Password Authentication

Many legacy systems only support user authentication with passwords. Some of these systems require transport of the actual username and password to the authentication server. This is true for systems where the authentication server does not have access to the cleartext password or a consistent transform of the cleartext password. Example of such systems are some one time password (OTP) systems and

other systems where the username and password are submitted to an external party for validation. The tunnel method MUST support transporting cleartext username and password to the EAP server. It MUST NOT reveal information about the username and password to parties in the communication path between the peer and the EAP Server. The advantage any attacker gains against the tunnel method when employing a username and password for authentication MUST be through interaction and not computation. The tunnel MUST support protection from man-in-the-middle attacks. The combination of the tunnel authentication and password authentication MUST enable mutual authentication.

Since EAP authentication occurs before network access is granted the tunnel method SHOULD enable an inner exchange to provide support for minimal password management tasks including password change, "new PIN mode", and "next token mode" required by some systems.

3.2. Protection of Weak EAP Methods

Some existing EAP methods have vulnerabilities that could be eliminated or reduced by running them inside a protected tunnel. For example, a EAP-MD5 does not provide mutual authentication or protection from dictionary attacks. Without extra protection, EAP tunnel methods are vulnerable to a special type of tunnel man-in-the-middle attack [TUNNEL-MITM]. This attack is referred to as "tunnel MitM attack" in the remainder of this document. The additional protection needed to thwart tunnel MitM attacks depends on the inner method executed within the tunnel. When weak methods are used, these attacks can be mitigated via security policies that require the method to be used only within a tunnel. On the other hand, a technical solution (so-called cryptographic bindings) can be used whenever the inner method derives key material and is not susceptible to attacks outside a tunnel. Only the latter mitigation technique

can be made an actual requirement for EAP tunnel methods (see Section 4.6.3), while security policies are outside the scope of this requirement draft. Please refer to the NIST Recommendation for EAP Methods Used in Wireless Network Access Authentication [NIST SP 800-120] and [LCN 2010] for a discussion on security policies and complete solutions for thwarting tunnel MitM attacks.

The tunnel method MUST support protection of weak EAP methods.

Cryptographic protection from tunnel MitM attacks MUST be provided for all key generating methods. In combination with an appropriate security policy this will thwart MitM attacks against inner methods.

3.3. Chained EAP Methods

Several circumstances are best addressed by using chained EAP methods. For example, it may be desirable to authenticate the user and also authenticate the device being used. However, chained EAP methods from different conversations can be re-directed into the same conversation by an attacker giving the authenticator the impression that both conversations terminate at the same end-point. Cryptographic binding can be used to bind the results of chained key generating methods together or to an encompassing tunnel.

The tunnel method MUST support chained EAP methods while including protection against attacks on method chaining.

3.4. Identity Protection

When performing an EAP authentication, the peer may want to protect its identity and only disclose it to a trusted EAP server. This helps to maintain peer privacy.

The tunnel method MUST support identity protection, therefore the identity of the peer used for authentication purposes MUST NOT be obtainable by any entity other than the EAP server terminating the tunnel method. Peer identity protection provided by the tunnel method applies to tunnel method and inner method specific identities. Note that the peer may need to expose the realm portion of the EAP outer identity in the Network Access Identifier (NAI) [RFC4282] in a roaming scenario in order to reach the appropriate authentication server.

3.5. Anonymous Service Access

When network service is provided, it is sometimes desirable for a user to gain network access in order to access limited services for emergency communication or troubleshooting information. To avoid eavesdropping, it's best to negotiate link layer security as with any

other authentication.

Therefore, the tunnel method SHOULD allow anonymous peers or server-only authentication, while still deriving keys that can be used for link layer security. The tunnel method MAY also allow for the bypass of server authentication processing on the client.

Forgoing user or server authentication increases the chance of man-in-the-middle and other types of attacks that can compromise the derived keys used for link layer security. Therefore, passwords and other sensitive information MUST NOT be disclosed to an unauthenticated server, or to a server that is not authorized to authenticate the user.

3.6. Network Endpoint Assessment

The Network Endpoint Assessment (NEA) protocols and reference model described in [RFC5209] provide a standard way to check the health ("posture") of a device at or after the time it connects to a network. If the device does not comply with the network's requirements, it can be denied access to the network or granted limited access to remediate itself. EAP is a convenient place for conducting an NEA exchange.

The tunnel method SHOULD support carrying NEA protocols such as PB-TNC [RFC5793]. Depending on the specifics of the tunnel method, these protocols may be required to be carried in an EAP method.

3.7. Client Authentication During Tunnel Establishment

In some cases the peer will have credentials that allow it to authenticate during tunnel establishment. These credentials may only partially authenticate the identity of the peer and additional authentication may be required inside the tunnel. For example, a communication device may be authenticated during tunnel establishment, in addition user authentication may be required to satisfy authentication policy. The tunnel method MUST be capable of providing client side authentication during tunnel establishment.

3.8. Extensibility

The tunnel method MUST provide extensibility so that additional data related to authentication, authorization and network access can be carried inside the tunnel in the future. This removes the need to develop new tunneling methods for specific purposes.

An application for extensibility is credential provisioning. When a peer has authenticated with EAP, this is a convenient time to

distribute credentials to that peer that may be used for later authentication exchanges. For example, the authentication server can provide a private key or shared key to the peer that can be used by the peer to perform rapid re-authentication or roaming. In addition there have been proposals to perform enrollment within EAP, such as [I-D.mahy-eap-enrollment]. Another use for extensibility is support for alternate authentication frameworks within the tunnel.

3.9. Certificate-less Authentication and Generic EAP Method Extension

In some cases the peer will not have a way to verify a server certificate and in some cases a server might not have a certificate to verify. Therefore, it is desirable to support certificate-less authentication. An application for this is credential provisioning where the peer and server authenticate each other with a shared password and credentials for subsequent authentication (e.g. a key pair and certificate or a shared key) can be passed inside the tunnel. Another application is to extend existing EAP methods with new features such as EAP channel bindings.

Great care must be taken when using tunnels with no server authentication for the protection of an inner method. For example, the client may lack the appropriate trust roots to fully authenticate the server, but may still establish the tunnel to execute an inner EAP method to perform mutual authentication and key derivation. In these cases, the inner EAP method MUST provide resistance to dictionary attack and a cryptographic binding between the inner method and the tunnel method MUST be established. Furthermore, the cipher suite used to establish the tunnel MUST derive the master key using contribution from both client and server, as in ephemeral Diffie-Hellman cipher suites.

The tunnel method MAY allow for certificate-less authentication.

4. Requirements

4.1. General Requirements

4.1.1. RFC Compliance

The tunnel method MUST include a Security Claims section with all security claims specified in Section 7.2 in RFC 3748 [RFC3748]. In addition, it MUST meet the requirement in Sections 2.1 and 7.4 of RFC 3748 that tunnel methods MUST support protection against man-in-the-middle attacks. Furthermore, the tunnel method MUST support identity

protection as specified in Section 7.3 in RFC 3748.

The tunnel method MUST be unconditionally compliant with RFC 4017 [RFC4017] (using the definition of "unconditionally compliant" contained in section 1.1 of RFC 4017). This means that the method MUST satisfy all the MUST, MUST NOT, SHOULD, and SHOULD NOT requirements in RFC 4017.

The tunnel method MUST meet all the MUST and SHOULD requirements relevant to EAP methods contained in the EAP Key Management Framework [RFC5247] or any successor. This includes the generation of the MSK, EMSK, Peer-Id, Server-Id and Session-Id. These requirements will enable the tunnel method to properly fit into the EAP key management framework, maintaining all of the security properties and guarantees of that framework.

The tunnel method MUST NOT be tied to any single cryptographic algorithm. Instead, it MUST support run-time negotiation to select among an extensible set of cryptographic algorithms, such as algorithms used with certificates presented during tunnel establishment. This "cryptographic algorithm agility" provides several advantages. Most important, when a weakness in an algorithm is discovered or increased processing power overtakes an algorithm, users can easily transition to a new algorithm. Also, users can choose the algorithm that best meets their needs.

The tunnel method MUST meet the SHOULD and MUST requirements pertinent to EAP method contained in Section 3 of RFC 4962 [RFC4962]. This includes: cryptographic algorithm independence; strong, fresh session keys; replay detection; keying material confidentiality and integrity; and confirmation of cipher suite selection.

4.2. Tunnel Requirements

The following section discusses requirements for TLS Tunnel Establishment.

4.2.1. TLS Requirements

The tunnel based method MUST support TLS version 1.2 [RFC5246] and may support earlier versions greater than SSL 2.0 to enable the

possibility of backwards compatibility.

4.2.1.1. Cipher Suites

4.2.1.1.1. Cipher Suite Negotiation

Cipher suite negotiations always suffer from downgrading attacks when they are not secured by any kind of integrity protection. A common practice is a post integrity check in which, as soon as available,

the established keys (here the tunnel key) are used to derive integrity keys. These integrity keys are then used by peer and authentication server to verify whether the cipher suite negotiation has been maliciously altered by another party.

Integrity checks prevent downgrading attacks only if the derived integrity keys and the employed integrity algorithms cannot be broken in real-time. See Section 6.1 or [KHLC07] for more information on this. Hence, the tunnel method MUST provide integrity protected cipher suite negotiation with secure integrity algorithms and integrity keys.

TLS provides protected cipher suite negotiation as long as all the cipher suites supported provide authentication, key establishment and data integrity protection as discussed in Section 6.1.

4.2.1.1.2. Tunnel Data Protection Algorithms

In order to prevent attacks on the cryptographic algorithms employed by inner authentication methods, a tunnel protocol's protection needs to provide a basic level of algorithm strength. The tunnel method MUST provide at least one mandatory to implement cipher suite that provides the equivalent security of 128-bit AES for encryption and message authentication. See Part 1 of the NIST Recommendation for Key Management [NIST SP 800-57] for a discussion of the relative strengths of common algorithms.

4.2.1.1.3. Tunnel Authentication and Key Establishment

A tunnel method MUST provide unidirectional authentication from authentication server to EAP peer and mutual authentication between authentication server and EAP peer. The tunnel method MUST provide

at least one mandatory to implement cipher suite that provides certificate-based authentication of the server and provides optional certificate-based authentication of the client. Other types of authentication MAY be supported.

At least one mandatory to implement cipher suite MUST be approved by NIST Draft Recommendation for Key Management, Part 3 [NIST SP 800-57p3], i.e., the ciphersuite MUST be listed in Table 4-1, 4-2 or 4-3 in that document.

The mandatory to implement cipher suites MUST only include cipher suites that use strong cryptographic algorithms. They MUST NOT include cipher suites providing mutually anonymous authentication or static Diffie-Hellman cipher suites.

Other ciphersuites MAY be selected following the security

requirements for tunnel protocols in NIST DRAFT Recommendation for EAP Methods Used in Wireless Network Access Authentication [NIST SP 800-120].

4.2.1.2. Tunnel Replay Protection

In order to prevent replay attacks on a tunnel protocol, the message authentication MUST be generated using a time-variant input such as timestamps, sequence numbers, nonces, or a combination of these so that any re-use of the authentication data can be detected as invalid. TLS provides sufficient replay protection to meet this requirements as long as weak cipher suites discussed in Section 6.1 are avoided.

4.2.1.3. TLS Extensions

In order to meet the requirements in this document TLS extensions MAY be used. For example, TLS extensions may be useful in providing certificate revocation information via the TLS Online Certificate Status Protocol (OCSP) extension [I-D.ietf-tls-rfc4366-bis] (thus meeting the requirement in Section 4.5.1.3).

4.2.1.4. Peer Identity Privacy

A tunnel protocol MUST support peer privacy. This requires that the

username and other attributes associated with the peer are not transmitted in the clear or to an unauthenticated, unauthorized party. Peer identity protection provided by the tunnel method applies to establishment of the tunnel and protection of inner method specific identities. If applicable, the peer certificate is sent confidentially (i.e. encrypted).

4.2.1.5. Session Resumption

The tunnel method MUST support TLS session resumption as defined in [RFC5246]. The tunnel method MAY support other methods of session resumption such as those defined in [RFC5077].

4.2.2. Fragmentation

Tunnel establishment sometimes requires the exchange of information that exceeds what can be carried in a single EAP message. In addition information carried within the tunnel may also exceed this limit. Therefore a tunnel method MUST support fragmentation and reassembly.

4.2.3. Protection of Data External to Tunnel

A man-in-the-middle attacker can modify clear text values such as protocol version and type code information communicated outside the TLS tunnel. The tunnel method MUST provide implicit or explicit protection of the protocol version and type code. If modification of other information external to the tunnel can cause exploitable vulnerabilities, the tunnel method MUST provide protection against modification of this additional data.

4.3. Tunnel Payload Requirements

This section describes the payload requirements inside the tunnel. These requirements frequently express features that a candidate protocol must be capable of offering so that a deployer can decide whether to make use of that feature. This section does not state requirements about what features of each protocol must be used during a deployment.

4.3.1. Extensible Attribute Types

The payload MUST be extensible. Some standard payload attribute types will be defined to meet known requirements listed below, such as password authentication, inner EAP method, vendor specific attributes, and result indication. Additional payload attributes MAY be defined in the future to support additional features and data types.

4.3.2. Request/Challenge Response Operation

The payload MUST support request and response type of half-duplex operation typical of EAP. Multiple attributes may be sent in a single payload. The payload MAY support carrying on multiple authentications in a single payload packet.

4.3.3. Indicating Criticality of Attributes

It is expected that new attributes will be defined to be carried within the tunnel method. In some cases it is necessary for the sender to know if the receiver did not understand the attribute. To support this, there MUST be a way for the sender to mark attributes such that the receiver will indicate if an attribute is not understood.

4.3.4. Vendor Specific Support

The payload MUST support communication of an extensible set of vendor-specific attributes. These attributes will be segmented into

uniquely identified vendor specific name spaces. They can be used for experiments or vendor specific features.

4.3.5. Result Indication

The payload MUST support result indication and its acknowledgement, so both the EAP peer and server will end up with a synchronized state. The result indication is needed after each chained inner authentication method and at the end of the authentication, so separate result indication for intermediate and final result MUST be supported.

4.3.6. Internationalization of Display Strings

The payload MAY provide a standard attribute format that supports international strings. This attribute format MUST support encoding strings in UTF-8 [RFC3629] format. Any strings sent by the server intended for display to the user MUST be sent in UTF-8 format and SHOULD be able to be marked with language information and adapted to the user's language preference as indicated by RFC 5646 [RFC5646]. Note that in some cases, such as when transmitting error codes, it is acceptable to exchange numeric codes that can be translated by the client to support the particular local language. These numeric codes are not subject internationalization during transmission.

4.4. EAP Channel Binding Requirements

The tunnel method MUST be capable of meeting EAP channel binding requirements described in [I-D.ietf-emu-chbind]. As discussed in [RFC5056], EAP Channel bindings differ from channel bindings discussed in other contexts. Cryptographic binding between the TLS tunnel and the inner method discussed in Section 4.6.3 relates directly to the non-EAP channel binding concepts discussed in RFC 5056.

4.5. Requirements Associated with Carrying Username and Passwords

This section describes the requirements associated with tunneled password authentication. The password authentication mentioned here refers to user or machine authentication using a legacy password database or verifier, such as LDAP [RFC4511], OTP, etc. These typically require the password in its original text form in order to authenticate the peer, hence they require the peer to send the clear text user name and password to the EAP server.

4.5.1. Security

Many internal EAP methods have the peer send its password in the clear to the EAP server. Other methods (e.g. challenge-response

methods) are vulnerable to attacks if an eavesdropper can intercept the traffic. For any such methods, the security measures in the following sections MUST be met.

4.5.1.1. Confidentiality and Integrity

The clear text password exchange MUST be integrity and confidentiality protected. As long as the password exchange occurs inside an authenticated and encrypted tunnel, this requirement is met.

4.5.1.2. Authentication of Server

The EAP server MUST be authenticated before the peer sends the clear text password to the server.

4.5.1.3. Server Certificate Revocation Checking

When certificate authentication is used during tunnel establishment the EAP peer may need to present its password to the server before it has network access to check the revocation status of the server's credentials. Therefore, the tunnel method MUST support mechanisms to check the revocation status of a credential. The tunnel method SHOULD make use of Online Certificate Status Protocol (OCSP) [RFC2560] or Server-based Certificate Validation Protocol (SCVP) [RFC5055] to obtain the revocation status of the EAP server certificate.

4.5.2. Internationalization

The password authentication exchange MUST support user names and passwords in international languages. It MUST support encoding of user name and password strings in UTF-8 [RFC3629] format. The method MUST specify how username and password normalizations and/or comparisons is performed in reference to SASLPrep [RFC4013], Net-UTF-8 [RFC5198] or their replacement.

Any strings sent by the server intended for display to the user MUST be sent in UTF-8 format and SHOULD be able to be marked with language information and adapted to the user's language preference as indicated by RFC 5646 [RFC5646]. Note that in some cases, such as when transmitting error codes, it is acceptable to exchange numeric codes that can be translated by the client to support the particular local language. These numeric codes are not subject to

internationalization during transmission.

4.5.3. Meta-data

The password authentication exchange SHOULD support additional associated meta-data which can be used to indicate whether the authentication is for a user or a machine. This allows the EAP server and peer to request and negotiate authentication specifically for a user or machine. This is useful in the case of multiple inner authentications where the user and machine both need to be authenticated.

4.5.4. Password Change

The password authentication exchange MUST support password change. The exchange SHOULD be extensible to support other "housekeeping" functions, such as the management of PINs or other data, required by some systems.

4.6. Requirements Associated with Carrying EAP Methods

The tunnel method MUST be able to carry inner EAP methods without modifying them. EAP methods MUST NOT be redefined inside the tunnel.

4.6.1. Method Negotiation

The tunnel method MUST support the protected negotiation of the inner EAP method. It MUST NOT allow the inner EAP method negotiation to be manipulated by intermediaries.

4.6.2. Chained Methods

The tunnel method SHOULD support the chaining of multiple EAP methods. The tunnel method MUST allow for the communication of intermediate result and verification of compound binding between executed inner methods when chained methods are employed.

4.6.3. Cryptographic Binding with the TLS Tunnel

The tunnel method MUST provide a mechanism to bind the tunnel protocol and the inner EAP method. This property is referred to as cryptographic binding. Such bindings are an important tool for mitigating the tunnel MitM attacks on tunnel methods [TUNNEL-MITM]. Cryptographic bindings enable the complete prevention of tunnel MitM attacks without the need of additional security policies as long as the inner method derives keys and is not vulnerable to attacks outside a protected tunnel [LCN 2010]. Even though weak or non-key

deriving inner methods may be permitted, and thus security policies

preventing tunnel MitM attacks are still necessary, the tunnel method MUST provide cryptographic bindings, because only this allows migrating to more secure, policy-independent implementations.

Cryptographic bindings are typically achieved by securely mixing the established keying material (say tunnel key TK) from the tunnel protocol with the established keying material (say method key MK) from the inner authentication method(s) in order to derive fresh keying material. If chained EAP methods are executed in the tunnel, all derived inner keys are combined with the tunnel key to create a new compound tunnel key (CTK). In particular, CTK is used to derive the EAP MSK, EMSK and other transient keys (TEK), such as transient encryption keys and integrity protection keys. The key hierarchy for tunnel methods executions that derive compound keys for the purpose of cryptographic binding is depicted in Figure 1.

In the case of the sequential executions of n inner methods, a chained compound key CTK_i MUST be computed upon the completion of each inner method i such that it contains the compound key of all previous inner methods, i.e. CTK_i=f(CTK_i-1, MK_i) with 0 < i <= n and CTK_0=TK, where f() is a key derivation function, such as one that complies with NIST Recommendation for Key Derivation Using Pseudorandom Functions [NIST SP 800-108]. CTK_n SHOULD serve as the key to derive further keys. Figure 1 depicts the key hierarchy in the case of a single inner method. Transient keys derived from the compound key CTK are used in a cryptographic protocol to verify the integrity of the tunnel and the inner authentication method.

```
                   -----------
                   | TK | MK |
                   -----------
                      |   |
                      v   v
                    --------
                    | CTK  |
                    --------
                        |
                        v
                 ----------------
                 |      |       |
```

```
                                   v      v       v
                                -------  ------  -------
                                | TEK |  | MSK | | EMSK |
                                -------  ------- --------
```

Figure 1: Compound Keys

Furthermore, all compound keys CTK_i and all keys derived from it

SHOULD follow the recommendations for key derivations and key hierarchies as specified in [NIST SP 800-108]. In particular, all derived keys MUST have a lifetime assigned that does not exceed the lifetime of any key higher in the key hierarchy. The derivation MUST prevent a compromise in one part of the system from leading to compromises in other parts of the system that relay on keys at the same or higher level in the hierarchy.

4.6.4. Peer Initiated

The tunnel method SHOULD allow for the peer to initiate an inner EAP authentication in order to meet its policy requirements for authenticating the server.

4.6.5. Method Meta-data

The tunnel method SHOULD allow for the communication of additional data associated with an EAP method. This can be used to indicate whether the authentication is for a user or a machine. This allows the EAP server and peer to request and negotiate authentication specifically for a user or machine. This is useful in the case of multiple inner EAP authentications where the user and machine both need to be authenticated.

5. IANA Considerations

This document has no IANA considerations.

6. Security Considerations

A tunnel method is often deployed to provide mutual authentication

between EAP Peer and EAP Server and to generate key material for use in protecting lower layer protocols. In addition the tunnel is used to protect the communication of additional data, including peer identity between the EAP Peer and EAP Server from disclosure to or modification by an attacker. These sections cover considerations that affect the ability for a method to achieve these goals.

6.1. Cipher Suite Selection

TLS supports a wide variety of cipher suites providing a variety of security properties. The selection of cipher suites is critical to the security of the tunnel method. Selection of a cipher suite with weak or no authentication, such as an anonymous Diffie- Hellman based cipher suite will greatly increase the risk of system compromise. Since a tunnel method uses the TLS tunnel to transport data, the

selection of a ciphersuite with weak data encryption and integrity algorithms will also increase the vulnerability of the method to attacks.

A tunnel protocol is prone to downgrading attacks if the tunnel protocol supports any key establishment algorithm that can be broken on-line. In a successful downgrading attack, an adversary breaks the selected "weak" key establishment algorithm and optionally the "weak" authentication algorithm without being detected. Here, "weak" refers to a key establishment algorithm that can be broken in real-time, and an authentication scheme that can be broken off-line, respectively. See [KHLC07] for more details. The requirements in this document disapprove the use of key establishment algorithms that can be broken on-line.

Mutually anonymous tunnel protocols are prone to man-in-the-middle attacks described in [KHLC07]. During such an attack, an adversary establishes a tunnel with each the peer and the authentication server, while peer and server believe that they established a tunnel with each other. Once both tunnels have been established, the adversary can eavesdrop on all communications within the tunnels, i.e. the execution of the inner authentication method(s). Consequently, the adversary can eavesdrop on the identifiers that are exchanged as part of the EAP method and thus, the privacy of peer and/or authentication server is compromised along with any other data transmitted within the tunnels. This document requires server

authentication to avoid the risks associated with anonymous cipher suites.

6.2. Tunneled Authentication

In many cases a tunnel method provides mutual authentication by authenticating the server during tunnel establishment and authenticating the peer within the tunnel using an EAP method. As described in [TUNNEL-MITM], this mode of operation can allow tunnel man-in-the-middle attackers to authenticate to the server as the peer by tunneling the inner EAP protocol messages to and from a peer executing the method outside a tunnel or with an untrustworthy server. Cryptographic binding between the established keying material from the inner authentication method(s) and the tunnel protocol verifies that the endpoints of the tunnel and the inner authentication method(s) are the same. This can thwart the attack if the inner method derived keys of sufficient strength that they cannot be broken in real-time.

In cases where the inner authentication method does not generate any or only weak key material, security policies MUST be enforced such that the peer cannot execute the inner method with the same

credentials outside a protective tunnel or with an untrustworthy server.

6.3. Data External to Tunnel

The tunnel method will use data that is outside the TLS tunnel such as the EAP type code or version numbers. If an attacker can compromise the protocol by modifying these values the tunnel method MUST protect this data from modification. In some cases external data may not need additional protection because it is implicitly verified during the protocol operation.

6.4. Separation of TLS Tunnel and Inner Authentication Termination

Terminating the inner method at a different location than the outer tunnel needs careful consideration. The inner method data may be vulnerable to modification and eavesdropping between the server that terminates the tunnel and the server that terminates the inner method. For example if a clear text password is used then it may be

sent to the inner method server in a RADIUS password attribute which uses weak encryption that may not be suitable protection for many environments.

In some cases terminating the tunnel at a different location may make it difficult for a peer to authenticate the server and trust it for further communication. For example, if the TLS tunnel is terminated by a different organization the peer needs to be able to authenticate and authorize the tunnel server to handle secret credentials that it shares with the home server that terminates the inner method. This may not meet the security policy of many environments.

7. References

7.1. Normative References

[I-D.ietf-emu-chbind]
Hartman, S., Clancy, C., and K. Hoeper, "Channel Binding Support for EAP Methods", draft-ietf-emu-chbind-06 (work in progress), October 2010.

[RFC2560] Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 2560, June 1999.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC4017] Stanley, D., Walker, J., and B. Aboba, "Extensible Authentication Protocol (EAP) Method Requirements for Wireless LANs", RFC 4017, March 2005.

[RFC4962] Housley, R. and B. Aboba, "Guidance for Authentication, Authorization, and Accounting (AAA) Key Management", BCP 132, RFC 4962, July 2007.

[RFC5055] Freeman, T., Housley, R., Malpani, A., Cooper, D., and W. Polk, "Server-Based Certificate Validation Protocol (SCVP)", RFC 5055, December 2007.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5247] Aboba, B., Simon, D., and P. Eronen, "Extensible Authentication Protocol (EAP) Key Management Framework", RFC 5247, August 2008.

7.2. Informative References

[I-D.ietf-tls-rfc4366-bis] 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", draft-ietf-tls-rfc4366-bis-12 (work in progress), September 2010.

[I-D.mahy-eap-enrollment] Mahy, R., "An Extensible Authentication Protocol (EAP) Enrollment Method", draft-mahy-eap-enrollment-01 (work in progress), March 2006.

[KHLC07] Hoeper, K. and L. Chen, "Where EAP Security Claims Fail", ICST QShine , August 2007.

[LCN 2010] Hoeper, K. and L. Chen, "An Inconvenient Truth about Tunneled Authentications", Proceedings of 35th Annual IEEE Conference on Local Computer Networks (LCN 2010) , September 2009.

[NIST SP 800-108] Chen, L., "Recommendation for Key Derivation Using Pseudorandom Functions", Draft NIST Special Publication 800-108, April 2008.

[NIST SP 800-120] Hoeper, K. and L. Chen, "Recommendation for EAP Methods Used in Wireless Network Access Authentication", NIST Special Publication 800-120, September 2009.

[NIST SP 800-57]
Barker, E., Barker, W., Burr, W., Polk, W., and M. Smid, "Recommendation for Key Management - Part 1: General (Revised)", NIST Special Publication 800-57, part 1, March 2007.

[NIST SP 800-57p3]
Barker, E., Burr, W., Jones, A., Polk, W., , S., and M. Smid, "Recommendation for Key Management, Part 3 Application-Specific Key Management Guidance", Draft NIST Special Publication 800-57,part 3, October 2008.

[PEAP] Microsoft Corporation, "[MS-PEAP]: Protected Extensible Authentication Protocol (PEAP) Specification", August 2009.

[RFC4013] Zeilenga, K., "SASLprep: Stringprep Profile for User Names and Passwords", RFC 4013, February 2005.

[RFC4282] Aboba, B., Beadles, M., Arkko, J., and P. Eronen, "The Network Access Identifier", RFC 4282, December 2005.

[RFC4511] Sermersheim, J., "Lightweight Directory Access Protocol (LDAP): The Protocol", RFC 4511, June 2006.

[RFC4851] Cam-Winget, N., McGrew, D., Salowey, J., and H. Zhou, "The Flexible Authentication via Secure Tunneling Extensible Authentication Protocol Method (EAP-FAST)", RFC 4851, May 2007.

[RFC5056] Williams, N., "On the Use of Channel Bindings to Secure Channels", RFC 5056, November 2007.

[RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, January 2008.

[RFC5198] Klensin, J. and M. Padlipsky, "Unicode Format for Network Interchange", RFC 5198, March 2008.

[RFC5209] Sangster, P., Khosravi, H., Mani, M., Narayan, K., and J. Tardo, "Network Endpoint Assessment (NEA): Overview and Requirements", RFC 5209, June 2008.

[RFC5281] Funk, P. and S. Blake-Wilson, "Extensible Authentication Protocol Tunneled Transport Layer Security Authenticated Protocol Version 0 (EAP-TTLSv0)", RFC 5281, August 2008.

[RFC5646] Phillips, A. and M. Davis, "Tags for Identifying Languages", BCP 47, RFC 5646, September 2009.

[RFC5793] Sahita, R., Hanna, S., Hurst, R., and K. Narayan, "PB-TNC: A Posture Broker (PB) Protocol Compatible with Trusted Network Connect (TNC)", RFC 5793, March 2010.

[TUNNEL-MITM] Asokan, N., Niemi, V., and K. Nyberg, "Man-in-the-Middle in Tunnelled Authentication Protocols", Cryptology ePrint Archive: Report 2002/163, November 2002.

Appendix A. Changes from -01

- Added combined mutual authentication in section 3.1
- Changed reference from SP 800-52 to SP 800-57,part 3
- In section 6.2 changed terminology to tunnel MitM and security policy enforcement
- Reworded text in section 3.2 to clarify MITM protection
- Added more specific text about derivation of the CTK
- Removed resource constrained section
- Added support for Non EAP authentication as a use for extensibility
- Added text to emergency services section to emphasize that sensitive information should not be sent if the server is unauthenticated.
- Reworded TLS requirements
- Reworded external data protection requirements
- Added text to section 4.6 that states method must not be re-defined inside the tunnel.
- Editorial fixes

Appendix B. Changes from -02

- Editorial Fixes
- Clarified client authentication during tunnel establishment
- Changed text so that the tunnel method MUST meet all MUST and SHOULD requirements relevant to EAP methods in RFCs 4962 and 5247

Appendix C. changes from -03

- Resolution of open issues: http://trac.tools.ietf.org/wg/emu/trac/report/9

- o Revised section 2 to match other similar RFC(Issue 6)
- o Cleaned up section 3.2 (issue 8)
- o Clarified identity protection scope in section 3.4 and 4.2.1.4(issue 9)
- o Changed Emergency Services to anonymous authentication(section 3.5)(issue 10)
- o Clarified section 4.1.1 (issue 15)
- o Cleaned up TLS requirements in section 4.2.1(issue 11)
- o Replaced text in 4.2.1.1.3 with suitable reference
- o Improved wording in 4.2.3 and 6.3 (issue 13)
- o Update internationalization requirements in 4.3.6 and 4.5.2 (Issues 25,18)
- o Updated text in 4.5.1 (issue 16)
- o Changed meta-data to SHOULD in 4.5.3 and 4.6.5(Issue 20)
- o Changed chained methods to SHOULD in 4.6.2(issue 19)
- o Added security consideration for inner method termination(issue 24)
- o Updated references
- o Editorial changes(issues 7,22,17)

Authors' Addresses

Katrin Hoeper
Motorola, Inc.
1301 E Algonquin Rd
Schaumburg, IL 60196
USA

Email: khoeper@motorola.com

Stephen Hanna
Juniper Networks
3 Beverly Road
Bedford, MA 01730
USA

Email: shanna@juniper.net

Hao Zhou
Cisco Systems, Inc.
4125 Highlander Parkway
Richfield, OH 44286
USA

Email: hzhou@cisco.com

Joseph Salowey (editor)
Cisco Systems, Inc.
2901 3rd. Ave
Seattle, WA 98121
USA

Email: jsalowey@cisco.com