EAP WG R. Mahy
Internet-Draft SIP Edge LLC
Expires: January 2, 2006 Jul 2005

An Extensible Authentication Protocol (EAP) Enrollment Method
draft-mahy-eap-enrollment-00.txt

Status of this Memo

Copyright Notice

Abstract

This document introduces a new EAP Method intended to automatically enroll clients with minimal user interaction in a wireless LAN environment. The enrollment process involves the client providing weak possibly temporary credentials and the server providing stronger persistent credentials, both stages over a TLS-protected channel.

Table of Contents

1. Conventions

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 RFC-2119 [2].

2. Introduction and Motivation

Wireless LANs are increasingly popular for portable devices such as phones, PDAs, projectors, equipment carts, and factory equipment, many of which have minimal user interface capabilities. Authentication on wireless LAN networks today requires either a human-brokered web-based enrollment process, entering a shared secret key and optionally a username, or via a preprovisioned X.509 certificate [11]. The user experience of this type of manual enrollment is burdensome even on a device such as a laptop, but becomes even more problematic on devices with minimal user interface capabilities. For example, many wireless LAN phones require entry of pre-shared keys using mutiple taps per character on the phone keypad. Furthermore, many of the traditional techniques for automatic configuration such as DHCP [19] and SLP [20] cannot be used in a wireless LAN environment until layer 2 or layer 3 connectivity is fully established.

What is required is an enrollment mechanism that can bootstrap using whatever credentials are available from these devices (for example: a numeric PIN or a manufacturer's certificate) and provide them with strong credentials such as certificates rooted in the local domain or machine generated username/password pairs. Since EAP [1] is already part of the recommended method of authenticating stations in a wireless LAN network such as 802.11 [17], an EAP extension is the logical way to exchange these credentials. The mechanism takes advantage of TLS [3] to setup a secure channel over which to perform the enrollment exchange.

Since it is desirable to offer enrollment as widely as possible, it

should be possible to enroll a client with credentials for a separate wireless LAN than the one used for enrollment. Since many existing wireless LANs are deployed using username/password authentication, this document describes an enrollment mechanism that can provide both certificates and strong username/password combinations. Specifically, this mechanism was designed to provide persistent credentials suitable for authentication using WiFi Protected Access (WPA) and 802.11i [18] (WPA2) authentication: WPA Personal (which requires a machine generated key), WPA Enterprise (which needs a "username" and machine generated password), and EAP-TLS [5] with mutual certificate-based authentication (which requires a certificate and possibly the corresponding private key).

The certificates provided during this enrollment phase could correspond to a private key already in use on the client corresponding to an already existing certificate, a new private key generated by the client (if the server does not trust the distribution of the previous key), or a new private key generated by the server (if the server does not trust that the client can generate keys with sufficient entropy).

3. Protocol Overview

A client that wants to enroll, first tries to discover wireless LANs which support automatic enrollment. The client can attempt to discover this through a variety of mechanisms, including probing. For example, the author has proposed a mechanism [24] for 802.11 networks that a wireless access point can use to advertise support for this automatic enrollment protocol in beacons.

Once a client associates with a wireless LAN, the client should be able to indicate to the server that it wishes to perform automatic enrollment. In this proposal, the EAP Enrollment process can be triggered in the initial EAP Identity response by sending the following URN (Uniform Resource Name) as the identity string:

```
urn:ietf:params:eap:enroll
```

The server then starts a TLS session to protect the enrollment exchange. The TLS handshake does not require a client certificate and expects the server to present a certificate rooted in an authority trusted by the client. The client is also expected to

present the domain name from the certificate to the user for confirmation if at all practical.

Once the TLS channel is setup, the actual enrollment consists of two-stages. During the first roundtrip, the Provide phase, the server indicates what combinations of weak credentials are acceptable for enrollment and what types of strong credentials it can provide. The client responds by providing a combination of weak credentials from the server's list of acceptable weak credentials and indicates which of the offered strong credentials it can use. If the client cannot provide the weak credentials or cannot use credentials that the server would provide, the client can send an EAP Response/Nak to end the enrollment process.

During the second roundtrip, the Store phase, the server asks the client to store the credentials the client requested. The client response merely indicates it successfully stored them. It may be desirable for the server to also provide a usage policy document that indicate how these credentials are used. For example the document

could indicate which SSIDs and which authentication mechanisms are valid when used with the provided credentials. This policy information is left for future work.

```
  <------- EAP Request/Identity
  -------> EAP Response/Identity: urn:ietf:params:eap:enroll

  <------- EAP Request/EAP-TTLS   (Start)
  -------> EAP Response/EAP-TTLS: ClientHello...
  <------- EAP Request/EAP-TTLS:  ServerHello, Certificate...
  -------> EAP Response/EAP-TTLS: ClientKeyExchange...
  <------- EAP Request/EAP-TTLS:  ChangeCipherSuite...
  # <----- EAP Request/EAP-Enroll:  Provide (list ok weak
  #                                   credentials, strong
  #                                   credentials offered)
  # -----> EAP Response/EAP-Enroll: Provide (weak credentials)
  # <----- EAP Request/EAP-Enroll:  Store (strong credentials)
  # -----> EAP Response/EAP-Enroll: Store (Done)
  # <----- EAP Success
  EAP authentication optionally continues with new credentials...
```

These exchanges occur inside a TLS-protected channel.

The first EAP-Enroll Request is inside the same EAP-TTLS Request as the ChangeCipherSuite message which completes the TLS handshake

The client can now save the strong credentials provided during enrollment into persistent storage and use them to authenticate with the target wireless LAN. At this point the server can start a new EAP exchange to authenticate the client using the new credentials. Once IP connectivity is established, any additional configuration can proceed using already defined mechanisms. Alternatively the server can terminate the outermost EAP method with an EAP Request/Failure.

4. Establishing a Secure Channel

EAP clients that wish to use enrollment during an EAP exchange SHOULD respond to the first EAP Identity request from the server by providing the following URN as the client identity.

urn:ietf:params:eap:enroll

A server that receives this string in an Identity request SHOULD immediately start a TLS connecting using EAP-TTLS according to the rules defined there (this could be replaced with any other EAP method that creates an tunnel which provided confidentiality , integrity, and server authentication, for example PEAP, but we should select a single mechanism).

The TLS handshake MAY request a client certificate but MUST NOT require the client to have a valid certificate to continue the TLS handshake. The TLS client SHOULD provide a certificate if it has one, even if the certificate is self-signed. The TLS server MUST have a certificate which SHOULD be rooted in a certificate authority that is likely to be trusted by the client. If possible, the client SHOULD render to the user the CommonName (CN) in the Subject of the server's certificate and prompt the user of the device to authorize enrollment with the target domain. The CN SHOULD be a domainName choice type which is a fully-qualified domain name. The CN should also be the same CN presented when authenticating to any wireless LAN with the strong credentials provided during enrollment. If the server certificate contains a wlanSSID certificate extension as defined in [15], it SHOULD be ignored.

If present, the wlanSSID extension implies a restriction on the

SSID of the target SSID used after enrollment, not during enrollment. The domain name associated with both the enrollment wireless LAN and the target wireless LAN SHOULD be the same and are likely to use the same AAA server. Administrators should be able to use the same certificate for both enrollment and target WLANs. Since this enrollment method is designed to enroll devices on other wireless LANs, the SSID on the enrollment WLAN could be different. If the wlanSSID extension is present, this restriction should apply to the target LAN.

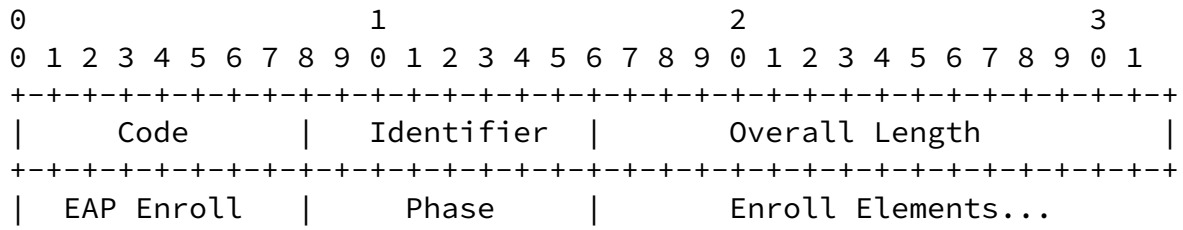
If the TLS handshake completes successfully, the EAP Enroll two-phase process begins.

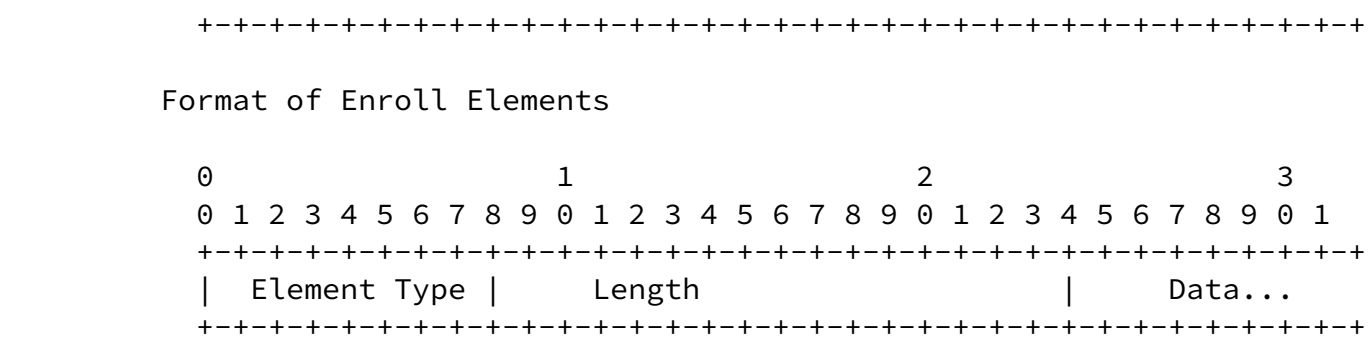
5. EAP Enroll Protocol Definition

The document defines a new EAP Method Type "EAP-Enroll". It requires a new EAP Method Type number assigned by IANA. The EAP-Enroll process consists of a Provide phase (Request/Response) followed by a Store phase (Request/Response). The Phase field is set to 0x01 during the Provide phase and 0x02 during the Store phase. Each EAP-Enroll messages is followed by zero or more Enroll Elements. Each Enroll Element has an Element Type (1 octet), a 2-octet length which counts the number of octets in the Element Data field (starting immediately after the length), and the data itself. EAP-Enroll MUST only be used inside a tunnel which provides confidentiality, message integrity, and server authentication.

A summary of the packet format is shown below. The fields are transmitted from left to right.

General Format of an EAP Enroll message





Format of Enroll Elements

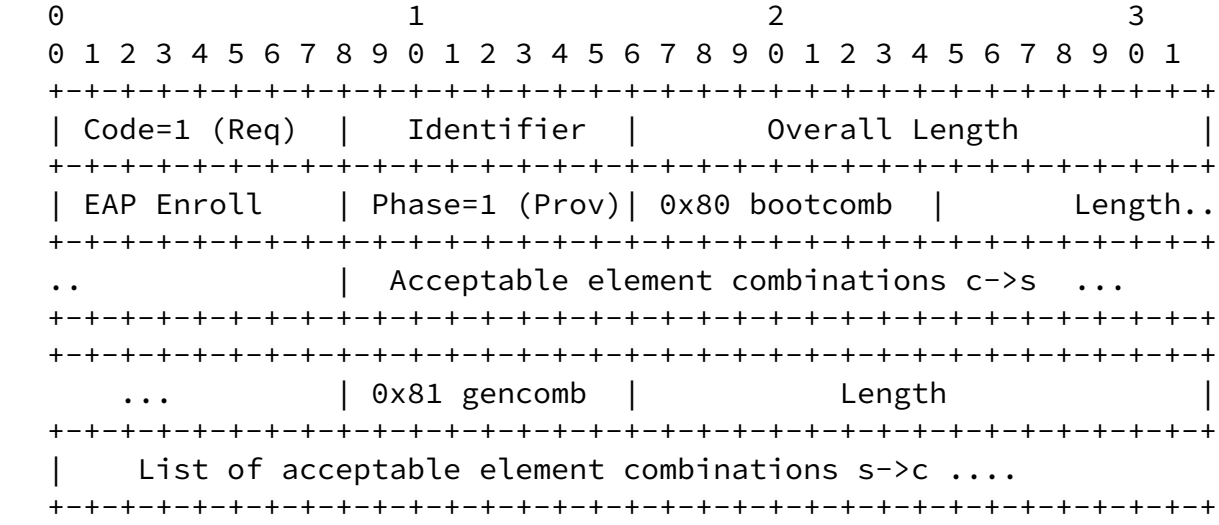
Below is a summary of Enroll Element Types. The majority of these element types are parts of the credentials that are either provided to the server to bootstrap the enrollment process, or sent to the client to store persistently.

Element Code	Semantics
0x00	RESERVED
0x01	Identity or Username
0x02	Plain-text Secret or PIN or Password
0x03	Digested Password
0x04	Nonce
0x05	Rooted Certificate
0x06	Self-Signed Certificate
0x07	Certificate Signing Request using existing private key
0x08	Certificate Signing Request using a newly generated key
0x09	Private Key in PKCS8 Format
0x0A	Signed Digest rsa
0x0B	CA Certificate
0x0C	CSR Signed Digest rsa
0x80	BootstrapCombinations: List of element combinations acceptable to server abrreviated the "bootcomb" in the diagrams.
0x81	GeneratedCombinations: Element combinations provided by the server or requested by the client from the server. abbreviated the "gencomb" in the diagrams.

0xFF	RESERVED

The EAP-Enroll process consists of a Provide request and response and a Store request and response. The server provides a Nonce element (0x04) with at least 16 octets of data in the Provide request. The Provide request MUST also contain the BootstrapCombinations Element (0x80) and the GeneratedCombinations Element (0x81).

EAP Request/EAP-Enroll (Provide Phase)



Both element lists have the same format. The data field in each element list contains a list of element types (1 octet each) or the octets 0x00 or 0xFF which have special semantics. Each list is organized as an ordered list of alternative combinations in preference order (the left-most combinations are more preferred). Each alternative is separated by the octet 0xFF. The octet 0x00 MAY be offered among alternatives, but it MUST NOT be combined with other elements inside a single alternative. Combinations are simply an unordered list of elements that MUST appear together in the same EAP message. The octet 0x00 indicates that satisfactory authentication was already performed and no additional credentials need to be provided.

Meaning of non-element octects in element lists

0x00	No additional credentials are required
0xFF	Indicates an alternative in an element list

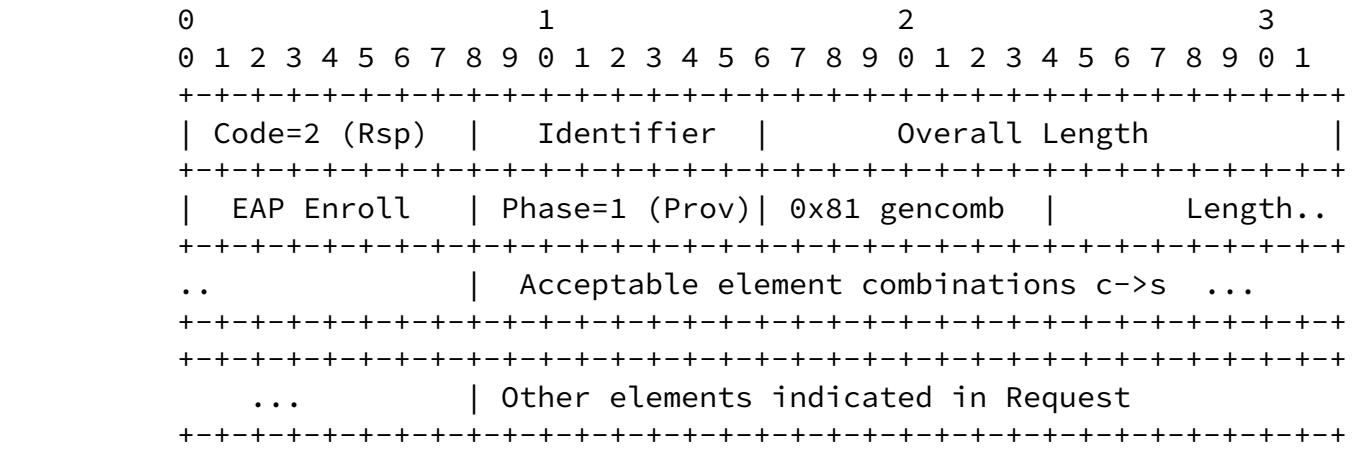
For example, the following BootstrapCombinations means that the client can provide the server either with a password and a self-signed certificate, OR a username and password.

pass word	+ self cert	OR	user name	+ digested password
0x02	0x06	0xFF	0x01	0x03

The client uses the BootstrapCombinations so it knows what type of elements it needs to include in its response (the bootstrap credentials). The client also examines the GeneratedCombinations to select a combination of elements the client would like to receive from the server in the Store phase (the persistent credentials).

After receiving an EAP Enroll Provide phase request, the client selects one combination of elements from the BootstrapCombinations that it can provide to bootstrap the enrollment process. The client responds by sending each element from that combination plus a Nonce element of its own. This client-side nonce MUST be different from the server-side nonce and must be at least 16 octets in length. The client also includes an GeneratedCombinations. The GeneratedCombinations includes a subset of the GeneratedCombinations from the request which MUST consist of only one of the originally provided combinations with no alternatives. If the client cannot respond with a set of credentials or a GeneratedCombinations list which is compatible with the server, the client MUST send an EAP Nak to terminate the enrollment.

EAP Response/EAP-Enroll (Provide Phase)

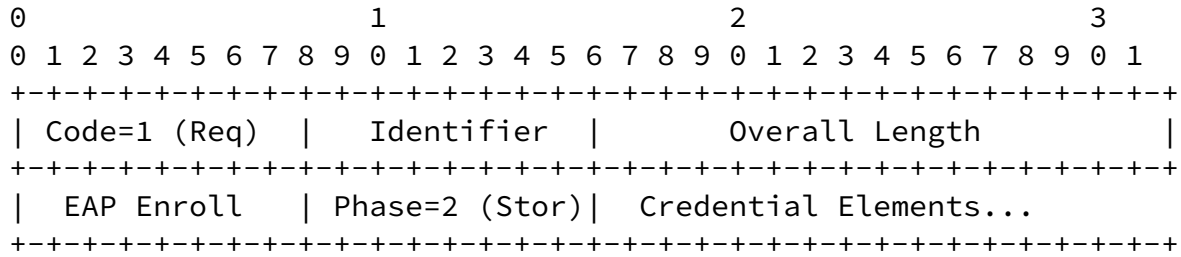


After receiving an EAP Enroll Provide response, the server tries to authenticate and authorize the client using the elements it provided. If the client is authorized, the server examines the GeneratedCombinations provided by the client and verifies that it is willing to generate the requested elements for the client. If so, the server generates these elements and returns them to the client in an EAP Enroll Store request (shown below).

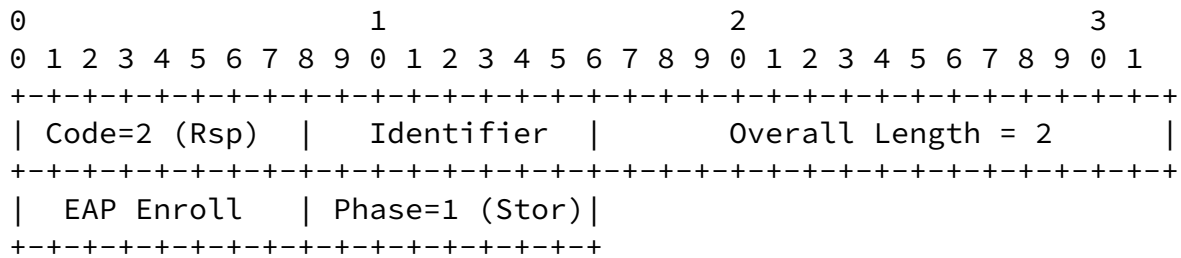
If the client did not authenticate or authorize or asked for a list of elements the server was unwilling to provide, the server SHOULD terminate the EAP exchange with an EAP-Failure message. The server MAY instead attempt another round of EAP requests if it believes this

will fix the error.

The EAP Enroll Store request contains the elements which contain the credentials the client needs to store for later access to the enrolled network.



If the client sucessfully receives and saves the credentials in the EAP Enroll Store request, it send an EAP Enroll Store response. The EAP Enroll Store response contains no elements.



6. Description of Specific Elements

Element Code	Semantics
0x00	RESERVED
0x01	Identity or Username
0x02	Plain-text Secret or PIN or Password
0x03	Digested Password
0x04	Nonce
0x05	Rooted Certificate
0x06	Self-Signed Certificate
0x07	Certificate Signing Request using existing private key
0x08	Certificate Signing Request using a newly generated key
0x09	Private Key in PKCS8 Format
0x0A	Signed Digest rsa
0x0B	CA Certificate
0x0C	CSR Signed Digest rs
0x80	BootstrapCombinations: List of element combinations acceptable to server abrreviated the "bootcomb" in the diagrams.
0x81	GeneratedCombinations: Element combinations provided by the server or requested by the client from the server. abbreviated the "gencomb" in the diagrams.
0xFF	RESERVED

0x01 is an identity or username. The data is a UTF-8 string which is not null-terminated and MUST NOT contain any NULL characters. In a Provide response, this element MUST be accompanied by a Digested

Password Element (0x03). In a Store request, this element MUST be accompanied by a Plain-Text Password element (0x02).

0x02 is a raw binary password, PIN, or other shared secret.

0x03 is a SHA-1 [6] digest of the username and password mixed with client and server nonces. The digest is 20 octets of raw data which is formed using the function: SHA(nonce + cnonce + user + SHA(password)). This element can only be included in a Provide response.

0x04 is a cryptographically secure nonce of at least 16 octets.

0x05 is a DER-encoded X.509v3 certificate which is rooted in a well-known authority. If the client wants to receive a rooted certificate from the EAP server that uses a private key generated by the client,

the client needs to provide a Certificate Signing Request (CSR) in either Element 0x07 or 0x08 in the EAP Enroll Provide response. When included in a Provide response it MUST be accompanied by a Signed Digest Element (0x0A). When included in a Store request it MUST be accompanied by a CA Certificate element (0x0B).

0x06 is a DER-encoded X.509v3 certificate which is self-signed. When this element is a listed in a BootstrapCombinations element, the client can safely imply that a rooted certificate would also be acceptable. This element can only be included in a Provide response and MUST be accompanied by a Signed Digest Element (0x0A).

0x07 is a DER-encoded PKCS#10 Certificate Signing Request [8] which uses the same public key as a certificate presented in the EAP Enroll Provide response. This element can only be included in a Provide response.

0x08 is a DER-encoded PKCS#10 Certificate Signing Request which uses a new key pair freshly generated specifically for the requested certificate. This element can only be included in a Provide response.

0x09 is a DER-encoded private key in PKCS#8 [14] format. This element can only be included in a Store request and MUST be accompanied by a rooted certificate (0x05).

0x0A is a signed digest used to insure the peer has possesion of a private key. It is formed by signing (nonce + cnonce) with sha1WithRSAEncryption as described in RFC 3370 [4] using the private key from the corresponding provided certificate (0x05 or 0x06). This element can only be included in a Provide response.

0x0B is a DER-encoded X.509v3 certificate of the authority which issued the accompanying rooted certificate. This element can only be included in a Store request and MUST be accompanied by a rooted certificate (0x05).

0x0C is a signed digest used to insure the peer has possesion of a newly generated private key. It is formed in the same way as the Signed Digest (0x0A) except that it is signed with the private key generated for a new CSR. This element can only be included in a Provide response and MUST be accompanied by a new key CSR (0x08).

7. Usage

This EAP method offer a large array of elements. The purpose of this section is to explain valid combinations of these elements and to motivate the need for these combinations.

EAP-Enroll can be used to generate three types of credentials which are of practical use: a certificate rooted in the local domain, a username/password combination, and a pre-shared key. These correspond respectively to EAP-TLS, WPA Enterprise, and WPA Personal in 802.11 networks. While using a certificate infrastructure is desirable from a security perspective, many organizations do not have a deployed certificate authority. Username/password pairs are still the norm for the majority of large organizations. When passwords are machine-generated, this style of authentication is still quite reasonable. Unfortunately, there is still also a need for EAP authentication using a pre-shared key, although this usage is discouraged. However, until a mechanism for secure fast roaming is standardized, implemented, and widely deployed, there will still be a number of deployments in wireless networks which authenticate using a pre-shared key. Also, shared-key authentication is still very popular for home access points.

When using a pre-shared key, the key is likely to be shared by a

large community, so this key is naturally provided by the server. When using a username/password combination, this document provides a mechanism to deliver a server-derived password. (The potential for using a mutually-derived password is discussed in the To Do Section.) When generating a certificate, there are a number of administrative policy issues around how the key was generated for the client. The client could generate a new private key for this certificate, the client could reuse an existing private (possibly provided during manufacturing), or the server could generate the corresponding private key. Client generation of a new key is generally RECOMMENDED, but there are some circumstances that justify using one of the other two methods. For example, a wireless client can associate with potentially dozens of wireless LANs. If the client enrolls in each network using a separate certificate and key pair, the storage requirements on the wireless device become non-trivial. In addition, some devices may not generate sufficient entropy to generate a good quality private key or may generate entropy very slowly. One solution is to use an existing key or to generate the key on the server. These two choices have different tradeoffs, so this protocol allows the server to choose.

Valid combinations of optional elements returned in a Store Request are listed below. Note (*) that the CA Certificate element (0x0B) MUST always accompany a rooted certificate in the Store request.

Valid Credentials in an EAP-Enroll Store Request

	Code(s)	Element Names	Example Usage
	0x02	Plaintext Password	WPA Personal
	0x01 + 0x03	Username + Digest Password	WPA Enterprise
*	0x05	Rooted Certificate	EAP-TLS
*	0x05 + 0x09	Rooted Certificate + Private Key	EAP-TLS

In order to authenticate the client, the server can accept several different types of bootstrap credentials. The following is a list of

valid combinations of the elements defined in this document for use in the Provide response. The client could include a CSR using a new key, and its signed digest (0x08 + 0x0C) in any of these listed combinations, or a CSR using an existing key (0x07) in any of the starred combinations.

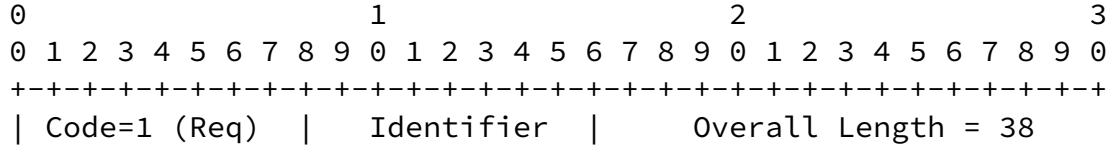
Valid Boostrap Credential Combos in EAP-Enroll Provide Response

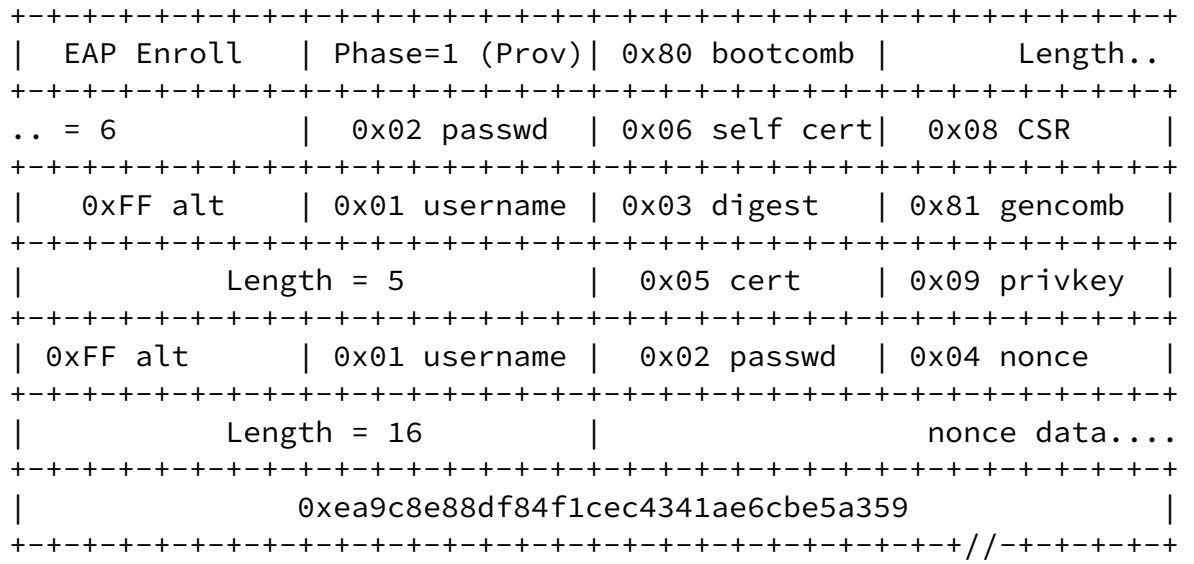
	Code(s)	Elements
	none	No Further Authentication Needed
	0x02	Plaintext Password
	0x01 + 0x03	Username + Password
*	0x05 + 0x0A	Rooted Cert
*	0x06 + 0x0A	Self-Signed Cert
*	0x05 + 0x0A + 0x02	Rooted Cert + Password
*	0x06 + 0x0A + 0x02	Self-Signed Cert + Password
*	0x05 + 0x0A + 0x01 + 0x03	Rooted Cert + Username + Password
*	0x06 + 0x0A + 0x01 + 0x03	Self-Signed Cert + Username + Password

Note that new elements could be defined (for example an element for authentication partially via credit card information) and these elements will need to define valid combinations with respect to the elements defined in this document.

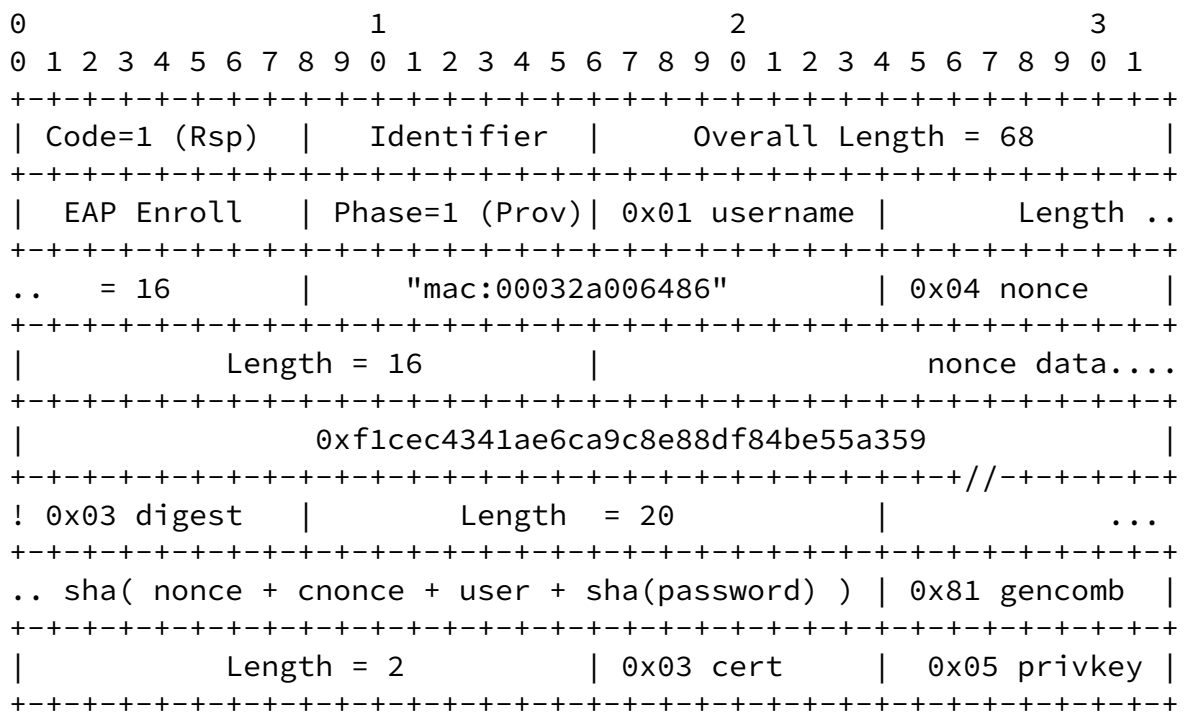
8. Example

In the example below, the Provide Request indicates that the server will accept either a username and password; or a password, self-signed certificate, and a certificate request as the credentials used to bootstrap the enrollment process. The server is prepared to offer either a rooted certificate and private key, or a machine generated username and password.

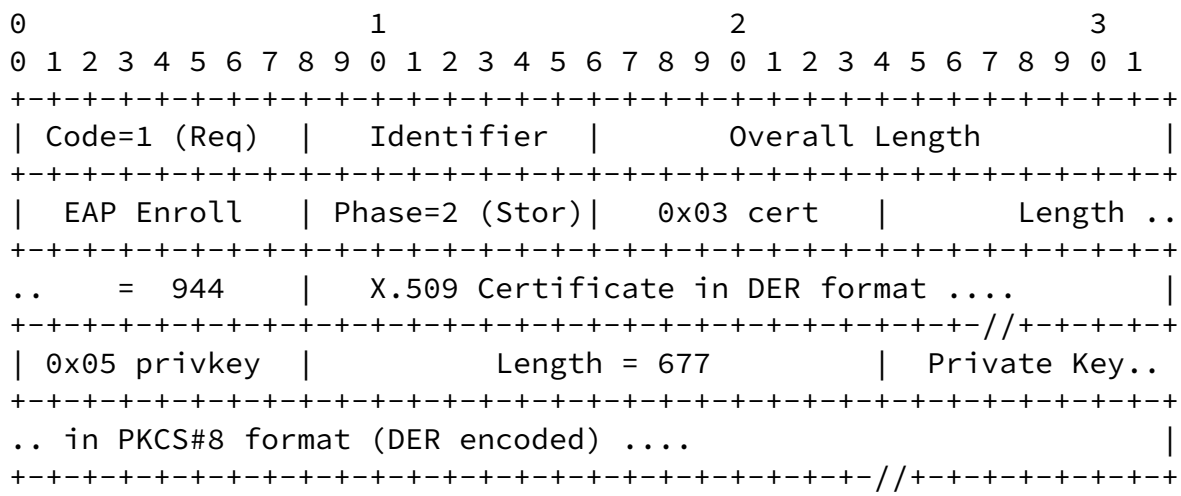




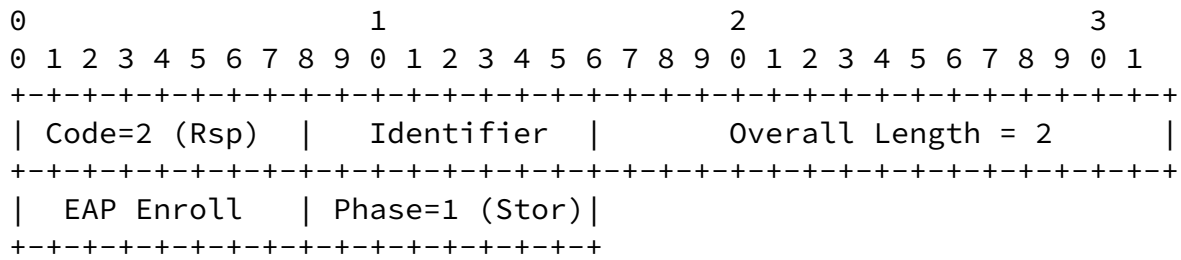
In the Provide Response, the client sends a username "bob" and a digest which uses the password "sh!", and asks for a rooted certificate and the corresponding private key.



The server authenticates and authorizes the client and returns an X.509 certificate and a private key in PKCS#8 format in an EAP Enroll Store request.



The EAP Enroll Store response just acknowledges the successful receipt of the persistent credentials



9. Security Considerations

This mechanism relies on a TLS channel to provide confidentiality, message integrity, and server authentication. If the TLS server is not authenticated, this mechanism is open to a variety of active attacks and can reveal cleartext passwords. The mechanism currently includes only a very basic way to perform mutual authentication using a shared secret. Since the passwords used during the bootstrapping process are likely to be weak, this authentication is vulnerable to dictionary attacks, but due to the incorporation of nonces, it is not vulnerable to dictionary pre-computation.

Similarly, this mechanism assumes transitive trust between the AAA server (the TLS server) and the certificate authority. It does not attempt to prevent active attacks by an attacker introduced between a AAA server and a certificate authority for example.

This mechanism allows the server to generate a username/password pair for the client. It is desirable to mutually derive this password. A possible approach to mutually deriving passwords is dicsussed in the To Do section. Likewise, the server can generate a private key for

the client in one mode. However, the server can refuse to offer a

private key and the client can refuse to enroll with a server that provides a private key.

EAP servers which implement this mechanism MUST implement Rooted Certificate and Username/Plaintext Password storage. EAP servers which implement this mechanism MUST implement Self-Signed Certificate plus Plaintext Password and Username/Digested Password for bootstrapping. EAP servers which implement this mechanism MUST implement EAP-TTLS and the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite defined in RFC 3268 [9].

Much more to do here.

10. IANA Considerations

Need a new EAP Method assignment. Need a new URN assignment following rules in RFC3688 [10]. Should probably setup a registry of Enroll Element Types.

11. To Do

This draft has several layering violations. EAP Enroll mechanism should be as independent as possible of other layers.

This EAP method defines a way to configure an EAP client with a username and password using a server generated key. It is desirable to develop a mechanism for the server to generate a username and provide a mutually derived key. One possible solution is to start the EAP-Enroll Provide phase, then use EAP-PAX [22] to derive a mutual key, finally ending with an EAP-Enroll Store phase which references the derived key. This may aggravate the layering problems with the proposal however.

Need to describe how this relates to the SACRED framework [23].

Missing a normative discussion about how to verify that the client is in possession of the private key that corresponds to a certificate or CSR. (how to use the Signed Digest and CSR Signed Digest elements).

12. Acknowledgments

Many thanks to Max Pritikin for his discussions about this idea, and to Charles Clancy who provided a security review and mmany helpful comments. Thanks also to Russ Housley and Benard Aboba for their support of this idea.

13. References

13.1 Normative References

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

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

[3] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[4] Housley, R., "Cryptographic Message Syntax (CMS) Algorithms", RFC 3370, August 2002.

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

[6] Eastlake, D. and P. Jones, "US Secure Hash Algorithm 1 (SHA1)", RFC 3174, September 2001.

[7] Myers, M., Adams, C., Solo, D., and D. Kemp, "Internet X.509 Certificate Request Message Format", RFC 2511, March 1999.

[8] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, November 2000.

[9] Chown, P., "Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)", RFC 3268, June 2002.

[10] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[11] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.

[12] International Telecommunications Union, "Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks", ITU-T Recommendation X.509, ISO Standard 9594-8, March 2000.

[13] International Telecommunications Union, "Information Technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T Recommendation X.690, 1994.

[14] RSA Laboratories, "Private-Key Information Syntax Standard,

Version 1.2", PKCS 8, November 1993.

[15] Housley, R. and T. Moore, "Certificate Extensions and Attributes Supporting Authentication in Point-to-Point Protocol (PPP) and Wireless Local Area Networks (WLAN)", draft-ietf-pkix-rfc3770bis-02 (work in progress), April 2005.

[16] Funk, P. and S. Blake-Wilson, "EAP Tunneled TLS Authentication Protocol Version 1 (EAP-TTLSv1)", draft-funk-eap-ttls-v1-00 (work in progress), February 2005.

13.2 Informational References

[17] "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 Standard 802.11, 1999, <http://standards.ieee.org/getieee802/download/802.11-1999.pdf>.

[18] "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 Amendment 6: Medium Access Control (MAC) Security Enhancements",

IEEE Standard 802.11i, July 2004, <http://standards.ieee.org/getieee802/download/802.11i-2004.pdf>.

[19] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[20] Veizades, J., Guttman, E., Perkins, C., and S. Kaplan, "Service Location Protocol", RFC 2165, June 1997.

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

[22] Clancy, C. and W. Arbaugh, "EAP Password Authenticated Exchange", draft-clancy-eap-pax-04 (work in progress), June 2005.

[23] Gustafson, D., Just, M., and M. Nystrom, "Securely Available Credentials (SACRED) - Credential Server Framework", RFC 3760, April 2004.

URIs

[24] <https://scm.sipfoundry.org/rep/ietf-drafts/rohan/eap-enroll/ap-selection.doc>

Author's Address

Rohan Mahy
SIP Edge LLC

Email: rohan@ekabal.com

Intellectual Property Statement

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

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

Disclaimer of Validity

Copyright Statement

Acknowledgment

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