PPP EAP KEA Public Key Authentication Protocol
<draft-ietf-pppext-eapkea-00.txt>

Status of this Memo

This document is a submission to the Point-to-Point Protocol Extensions Working Group of the Internet Engineering Task Force (IETF). Comments should be submitted to the ietf-ppp@merit.edu mailing list.

Distribution of this memo is unlimited.

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet-Drafts as reference material or to cite them other than as 'work in progress.'

To learn the current status of any Internet-Draft, please check the 'id-abstracts.txt' listing contained in the Internet-Drafts Shadow Directories on ds.internic.net (US East Coast), nic.nordu.net (Europe), ftp.isi.edu (US West Coast), or munnari.oz.au (Pacific Rim).

Abstract

The Point-to-Point Protocol (PPP) [1] provides a standard method for transporting multi-protocol datagrams over point-to-point links.

PPP also defines an extensible Link Control Protocol, which allows negotiation of an Authentication Protocol for authentication of its peer before allowing Network Layer protocols to transmit over the link.
PPP Extensible Authentication Protocol (EAP) [2] provides for a number of authentication mechanisms. This document specifies yet another authentication mechanism that may be used within the EAP framework. This document defines the KEA Public Key Authentication Protocol within PPP EAP. A side effect of KEA public key authentication is the creation of a session key for encryption of data on the PPP link.

1. Introduction

In order to establish communications over a point-to-point link, each end of the PPP link must first send LCP packets to configure the data link during Link Establishment phase. After the link has been established, PPP provides for an optional Authentication phase before proceeding to the Network-Layer Protocol phase.

By default, authentication is not mandatory. If authentication of the link is desired, an implementation MUST specify the Authentication-Protocol Configuration Option during Link Establishment phase.

PPP Extensible Authentication Protocol (EAP) [2] allows for a number of authentication protocols including KEA Public Key Authentication Protocol.

This document defines the PPP EAP KEA Public Key Authentication Protocol. The Link Establishment and Authentication phases, and the Authentication-Protocol Configuration Option are defined in The Point-to-Point Protocol (PPP) [1]. The Extensible Authentication protocol is defined in PPP Extensible Authentication Protocol (EAP) [2].

1.1. Specification of Requirements

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

**MUST** This word, or the adjective required, means that the definition is an absolute requirement of the specification.

**MUST NOT** This phrase means that the definition is an absolute prohibition of the specification.

**SHOULD** This word, or the adjective recommended, means that there may exist valid reasons in particular
circumstances to ignore this item, but the full implications must be understood and carefully weighed before choosing a different course.

MAY

This word, or the adjective optional, means that this item is one of an allowed set of alternatives. An implementation which does not include this option MUST be prepared to interoperate with another implementation which does include the option.

1.2. Terminology

This document frequently uses the following terms:

authenticator  The end of the link requiring the authentication. The authenticator specifies use of DSS Authentication in the EAP-Request during Authentication phase.

certificate    A certificate consists of the binding together of one or more public key values and an identity. This binding is effected through a digital signature which is computed over the data containing both the public key and the identity. This signature is applied by a "certification authority" who is recognized as issuing this certificate on behalf of the entity identified in the certificate. In this manner a recipient of this certificate can determine the recognized public key of the particular entity identified in the certificate. This requires the recipient to, either directly or indirectly, trust the authority who has issued this certificate.

certification authority (CA)  
An authority trusted by one or more users to create and assign certificates. [3].

digital signature  
In the DSS, a digital signature is produced by performing the DSA signing operation with a private key on the SHA-1 Hash value computed over the original data to be signed. The verification of this digital signature requires the verifier to obtain the original message, and the signature value, and the proper public key value that is associated with the signer (see certificates below). The verifier then also computes the SHA-1 Hash of the message data, and then perform a computation whose inputs include this hash value, the
public key, and the signature value. If the output of this computation matches a particular part of the signature value produced by the signer, then the signature is verified.

distinguished name
A unique hierarchical name. Used in the certificate’s "subject" field to denote the entity associated with the public key value(s) in the certificate[2]. Also used in the certificate’s "issuer" field to denote the entity that issued this certificate.

KEA key pair
A pair of related keys, used in the agreement of session encryption key. The two keys in the pair are known as the public and private keys. Knowledge of the public key does not necessarily imply knowledge of the private key of the pair. This document frequently uses the following terms:

peer
The other end of the point-to-point link; the end which is being authenticated by the authenticator.

private key
That key of a key pair which is known only by that user [3].

public key
That key of a key pair which is publicly known [3].

2. PPP EAP KEA Public Key Authentication

The PPP Extensible Authentication Protocol is a general protocol for PPP authentication which supports multiple authentication mechanisms. EAP MAY be negotiated at Link Control Phase. EAP MAY then be used to select the KEA Public Key Authentication mechanisms at the Authentication Phase.

The KEA Public Key Authentication Protocol is a four pass request-response protocol involving both authentication and key derivation. Since a side effect of KEA authentication is the derivation of a session encryption key, only one party need initiate the KEA Public Key Authentication Protocol. Liveness testing of the derived key indicates proper completion of the protocol.

In order to ensure that the authentication protocol is performed only once, rules are introduced to handle the case where both parties initiate the protocol.

If one party transmits a KEA Request and the other a DSS Unilateral
Request (or any other Request for that matter), then the second party may refuse the KEA Request by transmitting an LCP Terminate-Request. Should it be willing to honor the KEA Request, it will not terminate the link. Rather it shall no longer expect a Response to its DSS Unilateral Request and shall transmit a KEA Response.

If both parties transmit KEA Requests, the party transmitting the lesser value for the KEA Ra value shall no longer expect a Response to its Request, but shall instead generate a Response to the KEA Request containing the higher Ra value.

The initiating party starts the protocol with a EAP KEA Request packet. The peer MUST formulate a Response packet based on information in the Request packet as well as information only the peer knows (the peer’s private key). The peer MUST also provide in its response a reference (i.e. the Distinguished Name in the Certificate) to its own certificate (the certificate containing the peer’s public key), as well as proof that it knows the corresponding private key. The peer’s certificate is assumed to have been obtained through other means. One such means is the use of the Certificate Exchange Protocol. The Certificate Exchange Protocol is defined as an extension to the PPP protocol suite. It is suggested as occurring during a new phase in between Link Control and Authentication. The Certificate Exchange Protocol is defined in [5].

After the initial request and response, a second request/response exchange is used to test the liveness of the derived key. Successful completion of this exchange is signaled to the peer by the sending of the EAP Success Packet.

In detail, the steps in EAP KEA are:

After the Link Establishment phase is complete and Extensible Authentication Protocol is negotiated, the authenticator sends a Request packet to authenticate the peer. The Request packet has a type field specifying KEA Public Key Authentication plus the requester’s certificate reference and Ra value.

2. The peer sends a Response packet in reply to the Request. The Response packet contains the peer’s certificate reference, Rb value, a wrapped Message Encryption Key, an initialization vector, and a nonce, The nonce is encrypted using the MEK. Underlying the transmission of this Response is the calculation by the peer of a Token Encryption Key, generation and wrapping of a Message Encryption Key, and encryption of a random nonce using the Message Encryption Key.
3. The initiator then begins the KEA-Validate EAP exchange by sending a Request containing another initialization vector, and, encrypted in the MEK, the original nonce, incremented by one, and a new, random nonce. Again, underlying the transmission of this Request is the calculation by the initiator of the Token Encryption Key, the successful unwrapping of the Message Encryption Key, and its use to decrypt the peer’s nonce.

4. The peer checks the KEA-Validate Request by decrypting both nonces. The original is checked to see that it has been incremented, while the initiator’s nonce is incremented and encrypted and sent back in the KEA-Validate Response.

5. The initiator checks the KEA Validate Response by decrypting the nonce. This returned nonce should equal the nonce sent by the originator plus one. If the nonce matches as expected the initiator ends the authentication phase by sending the peer a Success packet. Otherwise the peer is sent a Failure packet. These packets are defined in PPP Extensible Authentication Protocol (EAP) [2].

3. PPP KEA DSS Public Key Authentication Packet Format

KEA authentication is performed using a derivative of the mechanism introduced in draft-ietf-cat-ftpkeasj-00.txt.

The initiating party is identified as "A"; its peer as "B". Four packets are exchanged in order to perform the authentication, first from A to B, and then from B to A, then repeating that sequence.

Both the EAP Response and Request packets for the KEA and KEA-Validate Type have the following format:

```
<table>
<thead>
<tr>
<th>Code</th>
<th>Identifier</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Type Data ...</td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 3.0-1 - The PPP EAP packet format

Code
Identifier

The identifier field is one octet and aids in matching responses with requests. The identifier field MUST be changed on each Request packet containing a different ChallengeVal.

Length

The Length field is two octets and indicates the length of the EAP Request and Response packets including the Code, Identifier, Length, Type, and Type Data fields. Octets in the packet outside the range of the Length field should be treated as Data Link Layer padding and should be ignored on reception.

Type

The Type field in the Request will carry the value 11 (KEA) or 12 (KEA-VALIDATE). The Type field in the Response SHOULD carry the corresponding value unless the peer wishes to send a Nak Type to indicate that it is incapable of handling KEA authentication.

Type Data

The contents and formats of the remainder of the packet differ for each of the four packet types: 1) KEA Request; 2) KEA Response; 3) KEA-VALIDATE Request; and 4) KEA VALIDATE Response.

The following sections define the format of the request and response.

3.1. EAP KEA Request Packet

The KEA Request packet is formatted as follows:

This information is formatted in a length-value format. No explicit type field is necessary because all fields are required and are in a determinate order. The last element does not include a length field because its length can be determined from the overall length. The EAP KEA Request packet has the following overall format:
Figure 3.0-2 - EAP KEA Request Packet format

Code

1    (Request)

Identifier

The identifier field is one octet and aids in matching responses with requests. The identifier field MUST be changed on each Request packet containing a different RanA value.

Length

The Length field is two octets and indicates the length of the EAP Request and Response packets including the Code, Identifier, Length, Type, RanA length field, RanA field, and DName field. Octets in the packet outside the range of the Length field should be treated as Data Link Layer padding and should be ignored on reception.

Type

The Type field in the Request will carry the value 11 (KEA).

ranA Length
The Length of the ranA field in bytes is specified here. A single byte is used to represent this length. For KEA this value is 128.

ranA

The value of the random number from the initiator to the responder. For KEA this field is 128 bytes in length.

DName

The DER-encoded form of the subject field in the X.509 certificate whose public key corresponds to the private key used in the KEA operation.

3.2. EAP KEA Response Packet

The KEA Response packet is formatted as follows:

This information is formatted in a length-value format. No explicit type field is necessary because all fields are required and are in a determinate order. The last element does not include a length field because its length can be determined from the overall length. The EAP KEA Response packet has the following overall format:
Figure 3.0-2 - EAP KEA Response Packet format

Code

2  (Response)

Identifier
The identifier field is one octet and MUST match the Identifier field from the corresponding request.

Length

The Length field is two octets and indicates the length of the EAP Request and Response packets including the Code, Identifier, Length, Type, RanB length field, RanB field, wMEK Length field, wMEK field, IV Length field, IV field, Encrypted Nonce Length field, Encrypted Nonce field, and DName field. Octets in the packet outside the range of the Length field should be treated as Data Link Layer padding and should be ignored on reception.

Type

The Type field in the Response will carry the value 11 (KEA) unless the peer wishes to send a Nak Type to indicate that it is incapable of handling KEA authentication.

ranB Length

The Length of the ranB field in bytes is specified here. A single byte is used to represent this length. For KEA this value is 128.

ranB

The value of the random number from the responder to the initiator. For KEA this field is 128 bytes in length.

wMEK Length

The Length of the wMEK field in bytes is specified here. A single byte is used to represent this length. For KEA and the default encryption algorithm this value is 12.

wMEK

The value of the TEK-wrapped MEK. This MEK is a truly random key generated by the responder. It is wrapped in the TEK created during the KEA computation on the responder’s side. This means this MEK can only be unwrapped by the entity that initiated this KEA exchange. For KEA and the default encryption algorithm this field is 12 bytes in length.

IV Length
The Length of the IV field in bytes is specified here. A single byte is used to represent this length. For KEA and the default encryption algorithm this value is 24.

**IV**

The value of the IV. This value is required to support the subsequent use of encryption algorithms that will use the MEK generated during this exchange. This is the IV value to be fed into the decryptor on the Requestor’s side. For KEA and the default encryption algorithm this field is 24 bytes in length.

**Encrypted Nonce Length**

The Length of the Encrypted Nonce field in bytes is specified here. A single byte is used to represent this length. For KEA and the default encryption algorithm this value is 24.

**Encrypted Nonce**

The value of the Encrypted Nonce. This value is used in the first of two liveness checks that are performed in each direction on the link. To support this liveness check the KEA Responder chooses a Nonce, encrypts it using the MEK (the one whose wrapped value is in the field wMEK) with the default encryption algorithm, and places this value here in this field. Upon receiving this value the KEA Requestor sends back in a KEA-VALIDATE Request the encrypted value of this Nonce value plus one. The liveness check for Responder is then performed by decrypting this last value and checking if it equals the original Nonce plus one.

**DName**

The value of the subject field in the X.509 certificate whose public key corresponds with the private key used in the KEA computation by the peer.

### 3.3. EAP KEA-VALIDATE Request Packet

The KEA-VALIDATE Request packet is formatted as follows:

This information is formatted in a length-value format. No explicit type field is necessary because all fields are required and are in a...
determinate order. The last element does not include a length field because its length can be determined from the overall length. The EAP KEA-VALIDATE Request packet has the following overall format:

```
+---------------+---------------+-----------------+---------------+---------------+---------------+---------------+
|     Code      |  Identifier   |            Length             |
+---------------+---------------+-----------------+---------------+---------------+---------------+
|     Type      | IVPrime Len   |           IVPrime...          |
+---------------+---------------+-----------------+---------------+---------------+---------------+
|       ...IVPrime...                                           |
+---------------+---------------+-----------------+---------------+---------------+---------------+
|                               .                               |
+---------------+---------------+-----------------+---------------+---------------+---------------+
|    ...Encrypted Nonces...     |
+---------------+---------------+-----------------+---------------+---------------+---------------+
```

Figure 3.0-3 - EAP KEA-VALIDATE Request Packet format

**Code**

1    (Request)

**Identifier**

The identifier field is one octet and aids in matching responses with requests. The identifier field MUST be changed on each Request packet containing a different EncryptedNonces value.

**Length**

The Length field is two octets and indicates the length of the EAP KEA-VALIDATE Request packet including the Code, Identifier, Length, Type, IVPrime length field, IVPrime field, and Encrypted Nonce fields. Octets in the packet outside the range of the Length field should be treated as Data Link Layer padding and should be ignored on reception.
Type

The Type field in the Request will carry the value 12 (KEA-VALIDATE).

IVPrime Length

The Length of the IVPrime field in bytes is specified here. A single byte is used to represent this length. For KEA and the default encryption algorithm this value is 24.

IVPrime

The value of the IV from the Initiator to the peer. This value is required to support the subsequent use of encryption algorithms that will use the MEK generated during this exchange. This is the IV value to be fed into the decryptor on the Responder’s side. For KEA and the default encryption algorithm this field is 24 bytes in length.

Encrypted Nonces Length

The Length of the Encrypted Nonces field in bytes is specified here. A single byte is used to represent this length. For KEA and the default encryption algorithm this value is 40.

Encrypted Nonces

The value of the Encrypted Nonces. There are two values which are concatenated together and encrypted (with the current MEK) here. The first is the number the Requestor forms when he takes the Nonce value that the Responder sent over in the previous KEA Response and adds one to it. The second value in this field is the NoncePrime value, or the random value chosen by the Requestor. The Requestor expects the Responder to increment this value by one and send this new value of NoncePrime+1 back (encrypted in the MEK) in the KEA-VALIDATE Response. For KEA and the default encryption algorithm these two values together are 40 bytes in length.

3.4. EAP KEA-VALIDATE Response Packet

The KEA-VALIDATE Response packet is formatted as follows:
This information is formatted in a length-value format. No explicit type field is necessary because all fields are required and are in a determinate order. The last element does not include a length field because its length can be determined from the overall length. The EAP KEA-VALIDATE Response packet has the following overall format:

```
 0                   1                   2                   3
+----------------+----------------+----------------+
|     Code      |  Identifier   |            Length             |
+----------------+----------------+----------------+
|     Type      |          Encrypted NoncePrime                 |
+----------------+----------------+----------------+
|               ...Encrypted NoncePrime...                      |
+----------------+----------------+----------------+
|                               .                               |
+----------------+----------------+----------------+
|              ...Encrypted NoncePrime...                       |
+----------------+----------------+----------------+
|      ...      |               |               |
+----------------+----------------+----------------+
```

Figure 3.0-3 - EAP KEA-VALIDATE Response Packet format

**Code**

2 (Response)

**Identifier**

The identifier field is one octet and MUST match the Identifier field from the corresponding request.

**Length**

The Length field is two octets and indicates the length of the EAP KEA-VALIDATE Request packet including the Code, Identifier, Length, Type, and Encrypted NoncePrime fields. Octets in the packet outside the range of the Length field should be treated as Data Link Layer padding and should be ignored on reception.

**Type**

The Type field in the Response will carry the value 12 (KEA-VALIDATE).
Encrypted NoncePrime

The value of the Encrypted NoncePrime. This value is formed by the Responder by taking the NoncePrime value which he received in the KEA-VALIDATE request and sending it back encrypted in the same MEK, after being incremented by one. This completes the two-way key liveness checks and the Requestor will upon checking this value, proceed to a successful authentication state and sending over a EAP success packet to the peer.

4. PPP EAP KEA and KEA-VALIDATE Protocol Processing

The operation of the complete PPP KEA protocol, including the two steps of Authentication/Key Generation and Liveness Checking are both shown. The first is performed by the PPP KEA protocol and the second by the EAP KEA-VALIDATE protocol.

Figure 4.0-1 depicts the operation of the EAP KEA and KEA-VALIDATE protocol. In this and the following figures depicting PDU exchanges, the curly braces ({, }) denote items in Length-Value representation.

<table>
<thead>
<tr>
<th>Side:</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>Recipient</td>
<td></td>
</tr>
</tbody>
</table>

=> EAP Request (ID1, KEA, {certA, ra})

<= EAP Response (ID1, KEA, {certB, rb, wMEK, iv, ENCRYPTMEK(eNonce)})

=> EAP Request (ID2, KEA-Validate, {ivPrime, ENCRYPTMEK(eNonce+1, eNoncePrime)})

<= EAP Response (ID2, KEA-Validate, {ENCRYPTMEK(eNoncePrime+1)})

=> EAP Success Packet (ID3)

Figure 4.0-1 PPP EAP KEA and KEA-VALIDATE Protocol processing
Security Considerations

This memo defines a method for using EAP to perform Strong authentication and key generation of/with a peer using the KEA Key Exchange and authentication algorithm.

References:


Acknowledgements:

This work is based largely on EAP. The authors would like to thank John Vollbrecht of Merit specifically for his help in understanding the intention of the EAP Internet Draft. The authors would also like to thank Paul Amaranth of Oakland University for his EAP implementation. Thanks also are due to Bill Whelan of Network Express for his Internet Draft showing a worked example of the use of EAP for public key based authentication. And thanks go to Russ Housley for his helpful comments on earlier versions of this memo. And thanks finally to Bill Simpson for the standard PPP spec boilerplate from which we have borrowed heavily.

Chair’s Address:

The working group can be contacted via the current chair:

Karl Fox
Ascend Communications, Inc.

Email: karl@ascend.com
Author's Address:

Questions about this memo can also be directed to:

DIRNSA
Attn: X22 (W. Nace)
9800 Savage Road
Fort Meade, MD 20755-6000
USA

Phone: +1 410 859-4464
Email: WANace@missi.ncsc.mil

Peter Yee
SPYRUS
2460 N. First Street
Suite 100
San Jose, CA 95131-1023
USA

Phone: +1 408 432-8180
Email: pyee@spyrus.com

James E. Zmuda
SPYRUS
2460 N. First Street
Suite 100
San Jose, CA 95131-1023
USA

Phone: +1 408 432-8180
Email: jzmuda@spyrus.com