Internet-Draft
Expires: April, 2003

Yoshihiro Ohba/Shinichi Baba
Toshiba America Research, Inc.
Subir Das
Telcordia Technologies

October 27, 2002

PANA over TLS

<draft-ohba-pana-potls-01.txt>

Status of This Memo

Abstract

This draft specifies a method to carry authentication information over TLS between PANA Client (PaC) and PANA Authentication Agent (PAA). PANA over TLS uses existing TLS protocol over a reliable transport in order to perform authentication information exchange in a secure and reliable manner. The purpose of this document is not only to provide a mechanism for carrying the authentication parameters but also to address some outstanding issues such as, multiple access routers, reauthentication, security threats, etc.

Table of Contents

1 Introduction

This protocol, PANA over TLS (Protocol for carrying Authentication for Network Access over Transport Layer Security), is designed for authentication message exchange between PaC and PAA, both of which are on the same subnet [PANAREQ].

PANA over TLS uses TLS [TLS] over a reliable transport in order to perform authentication information exchange in a secure and reliable manner. In particular, the security features provided with TLS are important for providing encryption and/or integrity protection for

the entire authentication protocol exchange including the identity of the client as well as authentication result (e.g., EAP-Success/Failure) that is not protected in some authentication protocol such as EAP [EAP]. Without protecting those information it is difficult to distinguish the case where authentication is failed due to invalid credentials from other errors that might have happened as a result of some active attack.

There are a number of protocols such as IKE (Internet Key Exchange) [IKE] and PIC (Pre-IKE Credential provisioning) [PIC] that could be used for protecting authentication message exchange over a secure communication channel. However, TLS is selected in this protocol for the following reasons.

First, unlike IKE, TLS does not require mutual authentication for establishing a secure communication channel between peer entities. It would not be a realistic requirement for assuming mutual authentication especially in roaming environments. Second, unlike PIC, TLS supports a session resumption functionality that can be used for making re-authentication faster than that is performed without session resumption.

PANA over TLS is designed for carrying any authentication protocol information including EAP messages. It is also possible to use a TLS certificate for authenticating a PaC without using any other authentication protocol. PANA over TLS supports combining multiple types of authentication to authenticate a PaC. For example, it is possible to use a TLS client certificate for authenticating an IP address of the PaC and then use EAP for authenticating the user of the PaC.

Man-in-The-Middle (MiTM) MiTM attacks against contents carried over TLS connections are protected by TLS. Other kinds of MiTM attacks are also taken into account in this protocol.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [Keywords].

3. Protocol Overview

In this protocol authentication information is carried between a PaC and a PAA over a secure TLS connection on top of a reliable transport protocol such as TCP and SCTP. The PaC and the PAA are the client and the server of the TLS connection, respectively. The TLS connection is established based on the TLS Handshake Protocol defined in [TLS].

When authentication information is carried over a TLS connection, confidentiality and integrity for the authentication information exchange between the PaC and PAA are provided by the TLS connection. Reliability, congestion control and fragmentation free communication are provided by the reliable transport (though TLS also handles

fragmentation).

To establish a TLS connection a PaC needs to find (an) IP address(es) of the PAA(s) on the link. PAA Discovery described in this document is used for this purpose.

It is possible to use a TLS client certificate that is optionally carried during a TLS handshake as the credential of PaC. In this case, additional authentication MAY be performed over the TLS connection.

MiTM attacks against contents carried over TLS connections are protected by TLS. Other kinds of MiTM attacks are also taken into account in this protocol. In order to avoid MiTM attacks for a Device Identifier [PANAREQ], PANA over TLS also supports creation of cryptographic binding between the Device Identifier and TLS session. Similarly, In order to avoid MiTM attacks for an authentication session (e.g., an EAP authentication session), PANA over TLS supports creation of cryptographic binding between the authentication session and TLS session.

PANA over TLS supports fast authentication based on the session resumption functionality of TLS [TLS].

The security association corresponding to the master secret of the TLS session established between PaC and PAA is considered to be a Local Security Association (LSA) from which other security association can be derived. The TLS master secret can be used for deriving any kind of security association.

PANA over TLS is designed to work over both multi-access links and point-to-point links (this does not necessary mean PPP links, an IP-in-IP tunnel and a GRE tunnel are also point-to-point). The only requirement is the PaC and PAA is on the same link in order to discover PAAs based on link-local multicast.

4. Protocol Specification

4.1. TLS Session, TLS Connection and Transport Connection

A TLS connection is a secure data channel over which application data is securely exchanged between TLS peers. Most of the messages defined in this protocol are carried over TLS connections.

A TLS session is a signaling channel used for establishing a master secret shared between TLS peers and establishing a TLS connection by negotiating cipher suites based on the master secret. The lifetime of a TLS session can be longer than that of a TLS connection, i.e., after terminating a TLS connection a new TLS connection can be established or an existing suspended TLS connection can be reused within the same TLS session (TLS session resumption). As specified in [TLS], TLS Alert/close_notify messages need to be exchanged before terminating the TLS connection in order for a session to be able to be resumed later.

A transport connection that is associated with a TLS connection MUST be terminated when the current TLS connection is terminated. Similarly, the current TLS connection that is associated with a transport connection MUST be terminated when the transport connection is terminated. After successful authentication, the transport connection MUST be kept open as long as the PaC is authorized for network access when Authenticated Heartbeat Protocol (AHP) is used for detecting implicit disconnection of a PaC. Otherwise, the transport connection MAY be terminated.

4.2. Transport Layer Protocol

PANA over TLS uses TCP or SCTP as the TLS transport. SCTP is preferable since it has a cookie-based 4-way handshake mechanism to protect against masquerade attacks (e.g., TCP SYN attacks) for transport connections.

UDP is also used for carrying messages used for finding (an) IP address(es) of PAA(s) and requesting (re-)authentication from PAA(s).

Both PaC and PAA need to configure an IP address before running this protocol. In IPv6, a link-local address can be used for this protocol. In IPv4, the IP address that is currently assigned to the interface is used.

The same port number PortNumber is used for TCP, SCTP and UDP.

4.3. PAA Discovery

/* Authors' note:

For ease of understanding and completeness of this document the following section has been described here. However, the authors recognize that PAA discovery is a problem by itself and need further discussions.

*/

Assuming that PAA is not co-located with an access router, a discovery mechanism is necessary for determining an IP address of the PAA. In addition, there may be multiple enforcement points and all of them may or may not be controlled by a single PAA. Methods that are used for a PaC to choose one or more PAAs when there are multiple PAAs in the same subnet are for further study. In any situation, a PAC needs to be aware of at least an IP address of each PAA and such information can be obtained by using at least one of the following ways:

1. Manual configuration

2. Notification from PAA

3. DHCP

4. Multicast query

Each method is explained below.

4.3.1. Manual Configuration

This is entirely specific to implementation and not described in this document.

4.3.2. Notification from PAA

When a PAA detects that a new PaC device is connected to the subnet, it MAY send an AuthRequest message to the PaC. The AuthRequest message is unicast over UDP. The PaC that receives an AuthRequest message will start establishing a TLS connection with the PAA. The PaC SHOULD NOT start establishing a TLS connection when it receives an from a PAA if it is in mid of performing authentication with the PAA. How a PAA detects the presence of a new PaC is out of the scope of this document.

4.3.3. DHCP

A new DHCP configuration option needs to be defined to carry the information described above.

4.3.4. Multicast Query

When PAA Discovery is performed via multicast, a PaC sends a PAADiscover message over a specific link-local multicast address "All-PAA-Nodes." Each PAA that received the message responds with an AuthRequest message. The AuthRequest message is unicast over UDP. A PAA SHOULD silently discard a PAADiscovery message received from a PaC without responding with a AuthRequest if it is mid of performing authentication with the PaC.

In the case of IPv4, All-PAA-Nodes is the same as "all-hosts" group (224.0.0.1). In the case of IPv6, All-PAA-Nodes is a link-local scoped multicast address to be assigned by IANA.

All PAAs MUST support this method.

4.4. Authentication Modes

There are two modes of authentication: Full authentication and Fast Authentication. Full Authentication is performed when a new TLS session is established with a full TLS handshake. A new session ID is allocated for the session. For Full Authentication, a server certificate MUST be used in TLS handshake to avoid a MiTM attack where TLS connections are spliced at an intermediate eavesdropper.

Fast Authentication is performed based on an existing TLS session by using the session resumption functionality of TLS. A PaC can always propose Fast Authentication whenever it has a TLS session with the PAA. Fast Authentication is performed by specifying the session ID

of the existing TLS session. On the other hand, The PAA can always choose to either perform Fast Authentication or force Full Authentication for the PaC that is proposing Fast Authentication with an existing session ID.

4.5. Authentication Types

There are two types of authentication defined for Full Authentication mode: One-Way TLS Authentication and Mutual TLS Authentication. In One-Way TLS Authentication, a PaC is authenticated without using a TLS client certificate. In One-Way TLS Authentication, a PAA MUST NOT request a TLS client certificate during a TLS handshake, and AuthInfo message exchange MUST be performed over the TLS connection. In TLS Authentication, a TLS client certificate MUST be used during TLS handshake for authenticating a PaC and additional AuthInfo message exchange MAY be performed over the TLS connection. There is no distinction in authentication type for Fast Authentication.

4.6. Multi-level Authentication

PANA over TLS supports multi-level authentication in which multiple legs of "sub-authentication" may be performed one by one until a PaC is finally authenticated by a PAA. For example, in the case of TLS authentication with AuthInfo message exchange, authentication during TLS session negotiation with a TLS client certificate can be a sub-authentication leg and authentication based on AuthInfo message exchange can be another sub-authentication leg. Or when EAP message is carried in AuthInfo message, multi-level authentication can be performed within EAP [EAPBIS].

When multi-level authentication occurs, it is the matter of authorization policy whether the entire sub-authentication legs need to be successful in order for a PaC to be finally authenticated or a restricted level of authorization may be applied when only a portion of the entire sub-authentication legs is successful, except that Mutual TLS authentication always requires successful TLS client certificate authentication to establish a TLS session. Such an authorization policy issue is out of the scope of this document.

4.7. Authentication Information Retransmission Policy

Since AuthInfo messages are carried over reliable transport, retransmission in authentication protocols (e.g., EAP) carried in AuthInfo SHOULD be disabled, except for retransmission of specific messages that require a response based on user input (e.g., username) [EAPBIS].

4.8. Creating Cryptographic Bindings

In order to prevent various kinds of MiTM attacks, it is necessary to create a binding between the security association established between PAA and PaC (i.e., TLS session) and any state that is established based on the information carried inside or outside of TLS.

4.8.1. MiTM Protection for Device Identifiers

Although TLS server certificates used for TLS handshake prevents MiTM attacks against TLS connections, another type of MiTM attack is still possible against transport connections. That is, an intermediate attacker may splice two transport connections that carry the contents of a single TLS connection between a PaC and a PAA without any modification. As a result, the attacker can successfully make its own IP address authorized for network access instead of the IP address of the PaC. Also, if a TLS server certificate is not directly associated with an IP address of the PAA, it is also possible for the intermediate attacker to be a rogue PAA (the TLS server certificate may be associated with a FQDN, but the FQDN may result in being mapped to a different IP address if DNS is not secured).

To deal with the MiTM attack against transport connections, PANA over TLS defines a Device Identifier exchange mechanism over TLS connections. DeviceID message is used for this purpose.

If a received Device Identifier contained in a DeviceID message sent from the peer is different from that is actually specified in the IP and/or MAC header(s) of the underlying transport connection, the PAA MUST return an Error message and immediately terminate the TLS connection and transport connection.

4.8.2 MiTM Protection for Authentication Sessions

As the Device Identifier needs to be cryptographically bound to the TLS session, it is also necessary for an authentication session (e.g., an EAP session) created as a result of successful AuthInfo message exchange to be cryptographically bound to the TLS session.

Consider the case in which an EAP session is created via PANA over TLS. If there is no cryptographic binding between the EAP session and TLS session, a class of MiTM attack is possible by which an attacker PaC establishes a TLS session with a legitimate PAA and then induces an unwary client for EAP authentication via e.g., 802.1X. The attacker just passes through the EAP messages and finally can gain access to the network on behalf of the unwary client.

In order to prevent this kind of MiTM attack, PANA over TLS supports explicit message exchange for establishing the cryptographic binding between the TLS session and the authentication session. This is achieved by the following way:

- o The PAA sends a AuthBind/Request message to the PaC when AuthInfo message exchange has been completed with success.

- o When the PaC receives the AuthBind/Request message, it verifies the payload of the message. If the received payload is valid, it returns a AuthBind/Response message. Otherwise, in returns an Error message with Subtype "AuthBindVerificationFailure".

- o When the PAA receives the AuthBind/Response message, it verifies

the payload of the message. If the received payload is valid, it returns a Success message. Otherwise, in returns an Error message with Subtype "AuthBindVerificationFailure".

- o If the PAA does not receive an AuthBind/Response message in a timeout period, it returns an Error message with Subtype "AuthBindVerificationFailure".

DefaultAuthBindTimeout is the default timeout value for AuthBind/Response.

AuthBind messages SHOULD be sent if a session key (i.e., a phase-2 key) is established for the authentication session. In this case, the PAA MUST be able to obtain a copy of the session key from the authentication server for the authentication session. AuthBind messages MUST NOT be sent if no session key is established for the authentication session.

4.9. Disconnect Indication

There are two types of disconnect indication supported by PANA over TLS: implicit disconnect indication and explicit disconnect indication.

Implicit disconnect indication is based on asynchronously performing re-authentication between a PaC and a PAA whenever one of the peers needs to know whether the other peer is still connected. When the transport connection remains established after successful Full or Fast Authentication, Authenticated Heartbeat Protocol can be used for implicit disconnect indication (see section "Authenticated Heartbeat Protocol"). On the other hand, when the transport connection is closed after successful Full or Fast Authentication, asynchronously performing Fast Authentication can be used for this purpose, and if the asynchronous Fast Authentication fails the initiator of the re-authentication regards that the peer has been disconnected. In both cases, if there is an established TLS connection and a transport connection, the node that detected the disconnect event MUST immediately terminate the TLS connection, transport connection and TLS session.

Explicit disconnect indication is based on explicit termination of the TLS connection and transport connection. The termination is initiated by either a PaC or a PAA that is going to be disconnected. When the transport connection remains established after successful Full or Fast Authentication, explicit disconnect indication is done by terminating the TLS connection and the transport connection. On the other hand, when the transport connection is closed after successful Full or Fast Authentication, explicit disconnect indication is done by performing Fast Authentication, and then immediately terminating the TLS connection and the transport connection, without sending DeviceID message. In both cases, the TLS session MAY remain established for future Fast Authentication based on session resumption. This is achieved by exchanging TLS Alert/close_notify messages between peers before terminating the TLS connection and the transport connection.

When a disconnect event of a PaC is detected by a PAA via either implicit disconnect indication or explicit disconnect indication, an appropriate action SHOULD be taken by the PAA for the device, e.g., deleting packet enforcement states for the PaC device at the enforcement point(s) controlled by the PAA.

4.10. Authenticated Heartbeat Protocol

PANA over TLS defines Authenticated Heartbeat Protocol (AHP) in which short messages (i.e., Heartbeat) are exchanged over the TLS connections in order to detect an inactive PaC without allowing an attacker to be able to gain authorized network access by spoofing the IP address of the legitimate PaC. Both Fast Authentication and AHP can be used for local re-authentication in which re-authentication is performed locally between PaC and PAA based on an established local security association (i.e., the TLS session) between them. Re-authentication can be performed faster by AHP than by Fast Authentication at the cost of holding the transport connection.

A PaC or PAA can send a Heartbeat/Request message whenever it needs to check whether the peer is still connected or not. When a PaC or a PAA receives a Heartbeat/Request message, it MUST respond with a Heartbeat/Response message. Heartbeat/Request message SHOULD NOT be sent too frequently (i.e., it SHOULD NOT be sent more than once per minute).

When AHP is used, the transport connection MUST remain established. The peer is considered to be disconnected when no Heartbeat/Response message is received within a timeout period after sending a Heartbeat/Request message, and further action described in "Disconnect Indication" is taken.

DefaultHeartbeatTimeout is the default timeout value for Heartbeat/Response.

4.11. Deriving Purpose-Specific Keys

The master secret of an established TLS session can be used for deriving a cryptographic key that can be used for a specific purpose. This purpose-specific key can be used by some other protocols for their client/server authentication. This document describes a generic rule for deriving such a purpose-specific key.

```
purpose_specific_key(S) = PRF(SecurityParameters.master_secret, S,
                              SecurityParameters.server_random +
                              SecurityParameters.client_random).
```

The definition of the PRF function and the structure of SecurityParameters are specified in [TLS].

S is a purpose-specific string which is a non-null terminated ASCII string defined for each purpose and is never carried in this protocol. Actual string values for S and detailed key derivation and usage rules depend on each purpose and thus are not defined in this document.

The PAA MUST NOT derive any purpose specific keys for the PaC that is not authorized for network access.

The PAA MUST immediately invalidate all derived purpose specific keys for the PaC that was once authorized for network access but has been unauthorized for any reason (e.g., re-authentication failure, disconnect indication, etc.)

4.12. Message Flows

Message flows for possible combination of authentication types and modes are illustrated in Figures 1 through 4. In these figures, messages marked with * are defined in this protocol and carried over the secure TLS connection as application data. Messages not marked with * are TLS signaling messages defined in [TLS], except for PAADiscover and AuthRequest messages that are carried over UDP. Messages surrounded in a pair of square brackets are optional. Although this protocol can carry any authentication protocol information, EAP messages are typically carried in AuthInfo messages.

```
PaC                                                  PAA

[PAADiscover]                 -------->
                              <--------        [AuthRequest]

ClientHello                   -------->
                                                 ServerHello
                                                 Certificate
                                           ServerKeyExchange
                              <--------      ServerHelloDone
ClientKeyExchange
CertificateVerify
ChangeCipherSpec
Finished                      -------->
                                            ChangeCipherSpec
                              <--------             Finished
DeviceID*                     <------->            DeviceID*
AuthInfo*                     <------->            AuthInfo*
                                  .
                                  .
```

```
      [AuthBind*]                  <------->          [AuthBind*]
                                   <--------      Success/Failure*
      [Heartbeat*]                 <------->         [Heartbeat*]
                                       .
                                       .
```

Figure 1: Message Flow for Full One-Way TLS Authentication


```
      PaC                                                     PAA

      [PAADiscover]                -------->
                                   <--------        [AuthRequest]
      ClientHello                  -------->
                                                      ServerHello
                                                      Certificate
                                                ServerKeyExchange
                                               CertificateRequest
                                   <--------      ServerHelloDone
      Certificate
      ClientKeyExchange
      CertificateVerify
      ChangeCipherSpec
      Finished                     -------->
                                                 ChangeCipherSpec
                                   <--------             Finished
      DeviceID*                    <------->            DeviceID*
      [AuthInfo*]                  <------->          [AuthInfo*]
                                       .
                                       .
      [AuthBind*]                  <------->          [AuthBind*]
                                   <--------      Success/Failure*
      [Heartbeat*]                 <------->         [Heartbeat*]
                                       .
                                       .
```

Figure 2: Message Flow for Full Mutual TLS Authentication

```
PaC                                                     PAA

                             <--------        [AuthRequest]
ClientHello                  -------->
                                                ServerHello
                                           ChangeCipherSpec
                             <--------             Finished
ChangeCipherSpec
Finished                     -------->
DeviceID*                    <------->            DeviceID*
                             <--------     Success/Failure*
[Heartbeat*]                 <------->         [Heartbeat*]
                                 .
                                 .
```

Figure 3: Message Flow for Fast Authentication


```
PaC                                                     PAA

                             <--------        [AuthRequest]
ClientHello                  -------->
                                                ServerHello
                                           ChangeCipherSpec
                             <--------             Finished
ChangeCipherSpec
Finished                     -------->
                             <--------     Success/Failure*
```

```
      [Alert/close_notify]         <-------> [Alert/close_notify]

               Figure 4: Message Flow for Fast Authentication
                         (for explicit disconnect indication)
```

4.13. Message Formats and Processing Rules

In this specification, all multi-octet fields are encoded in network byte order.

All messages defined in this document start with the following header.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Subtype    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

A type of the message.

Subtype

A type-specific information needed for decoding the message payload. The Subtype name "NoSubtype" indicates that there is no subtype for that Type.

Length

An unsigned 2-octet integer that contains the length of the message in octets, including the header and payload of the message.

4.13.1. PAADiscover Message

When this message will be sent:

This message is multicast over UDP by a PaC needs to know (an) IP address(es) of PAA(s) to perform Full Authentication.

Meaning of this message:

This message means that the sender PaC is searching PAA(s). When a

PAA receives this message from a PaC, it SHOULD return a AuthRequest message to the PaC unless it is in the middle of authentication process with regard to the PaC.

Structure of this message:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Subtype    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

Type value	Type name
0x01	PAADiscover

Subtype

Subtype value	Subtype name
0x00	NoSubtype

Length

The Length value is 4.

Payload

The payload is null.

4.13.2. AuthRequest Message

When this message will be sent:

This message is sent over UDP by a PAA when Full or Fast Authentication is needed.

Meaning of this message:

When a PaC receives this message, it starts Full or Fast Authentication if the PaC device needs to be authorized for network access. The PaC SHOULD NOT start establishing a TLS connection when it receives an AuthRequest message from a PAA if it is in the middle of authentication process with regard to the PAA.

Structure of this message:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Subtype    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```


Type

Type value	Type name
0x02	AuthRequest

Subtype

Subtype value	Subtype name
0x00	NoSubtype

Length

The Length value is 4.

Payload

The payload is null.

4.13.3. DeviceID Message

When this message will be sent:

This message MUST be sent by both PaC and PAA over an TLS connection right after a TLS handshake is finished.

Meaning of this message:

When a PaC or a PAA receives this message, it checks whether the Device Identifier contained in the message is the same as that is included in the MAC and/or IP header(s) encapsulating this message. If those two Device Identifiers are different, the receiver MUST return an Error message with Subtype "InvalidDeviceID" and immediately terminate the TLS session and the transport connection.

Structure of this message:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Subtype    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                      Device Identifier                        ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

Type value	Type name
0x03	DeviceID

Subtype

Subtype value	Subtype name
0x01	IPAddress
0x02	MACAddress
0x03	IPAndMACAddresses

IPAddress

This Subtype is used when the sender has no MAC address associated with the transport connection.

MACAddress

This Subtype is used when the sender has a MAC address associated with the transport connection.

IPAndMACAddresses

This Subtype is used when the sender has an IP address and a MAC address associated with the transport connection.

/* Authors' note:

In this version of draft, it is assumed that an appropriate Subtype is selected by configuration.

*/

Length

Variable (8, 12, 16, 20 or 28).

Payload

When the Subtype value is 0x01 (IPAddress), either IPv4 or IPv6 address is included depending on whether the transport connection is carried over IPv4 or IPv6, respectively. When the Subtype value is 0x02 (IPAndMACAddresses), either IPv4 or IPv6 address is included depending on whether the transport connection is carried over IPv4 or IPv6, respectively, immediately followed by an IEEE EUI-64 address [EUI64].

4.13.4. AuthInfo Message

When this message will be sent:

This message MUST be sent during Full One-Way TLS Authentication. This message MAY be sent during Full Mutual TLS Authentication. When this message is sent, it MUST be sent right after DeviceID message exchange. Multiple rounds of AuthInfo message exchange can occur.

Meaning of this message:

The contents and processing rules of the payload depend on the type of the authentication protocol.

Structure of this message:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Subtype    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                Authentication Protocol Message                ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

Type value	Type name
0x04	AuthInfo

Subtype

Subtype value	Subtype name
0x01	EAP

Length

Variable.

Payload

The contents depends on Subtype. If Subtype is 0x01 (EAP), an EAP PDU [EAP] is included.

4.13.5. Success Message

When this message will be sent:

This message is sent by a PAA when a PaC is finally authenticated.

Meaning of this message:

This message means that the PaC that receives this message is finally authenticated and the device associated with the Device Identifier of the PaC is authorized for network access.

Structure of this message:


```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Subtype    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

Type value	Type name
0x05	Success

Subtype

Subtype value	Subtype name
0x01	NothingToDo
0x02	HoldTransportConnection

NothingToDo

This Subtype is used when there is no further processing required for the PaC device to be fully authorized for network access.

HoldTransportConnection

This Subtype is used when the transport connection needs to be held after successful authentication. The TLS connection and the transport connection MUST remain open as long as the PaC continues to be authorized for network access.

Length

The Length value is 4.

Payload

The payload is null.

4.13.6. Failure Message

When this message will be sent:

This message is sent by a PAA when it finally fails to authenticate a PaC.

Meaning of this message:

This message means that authentication for the PaC that receives this message finally is not successful. When this message is sent, the PAA MUST immediately terminate the TLS connection, the transport connection, and the TLS session.

Structure of this message:


```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Subtype    |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

Type value	Type name
0x06	Failure

Subtype

Subtype value	Subtype name
0x00	NoSubtype

Length

The Length value is 4.

Payload

The payload is null.

4.13.7. Error Message

When this message will be sent:

This message is sent by a PAA or a PaC when it detects an error except for authentication failure. This message can occur at any time during PANA message exchange over a TLS connection.

Meaning of this message:

This message means that the sender detected an error except for authentication failure. The Subtype indicates the reason of the error. When this message is sent, the PAA MUST immediately terminate the TLS connection, the transport connection and the TLS session.

When the receiver of this message finds an error for this message, it MUST NOT return an Error message.

Structure of this message:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Subtype    |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

```
Type value    Type name
```



```
0x07          Error
```

Subtype

Subtype value	Subtype name
0x01	InvalidDeviceID
0x02	UnexpectedMessageType
0x03	UnsupportedMessageType
0x04	UnsupportedMessageSubtype
0x05	InvalidMessageLength
0x06	InvalidPayloadContents
0x07	HeartbeatResponseTimeout
0x08	AuthBindVerificationFailure

InvalidDeviceID

This Subtype is used when an invalid Device Identifier is detected in a DeviceID message sent from the peer.

UnexpectedMessageType

This Subtype is used when a message is received with a Type value that is supported by the node but the type is different from that is expected at this specific protocol phase.

UnsupportedMessageType

This Subtype is used when a message is received with a Type value that is not supported.

UnsupportedMessageSubtype

This Subtype is used when a message is received with an appropriate Type value but with a Subtype value that is not supported.

InvalidMessageLength

This Subtype is used when inconsistency is detected between the value of the Length field and the actual message length.

InvalidPayloadContents

This Subtype is used when the payload cannot be decoded based on the expected format defined for the specified message Type and Subtype.

HeartbeatResponseTimeout

This Subtype is used when a Heartbeat/Response timer for the PaC expires.

AuthBindVerificationFailure

This Subtype is used when verification of an AuthBind message fails.

Length

The Length value is 4.

Payload

The payload is null.

4.13.8. **Heartbeat Message**

When this message will be sent:

This message MAY be exchanged between a PaC to a PAA after a Success message is sent by the PAA with a Subtype value 0x02 (HoldTransportConnection). A PAA or PaC sends a Heartbeat/Request message whenever it wants to check whether the peer is still connected or not. When a PaC or a PAA receives a Heartbeat/Request message, it is requested to send Heartbeat/Response message back to the sender of the Heartbeat/Request message. Heartbeat/Request messages MAY be sent periodically, but SHOULD NOT be more than once per minute.

Meaning of this message:

If the sender of a Heartbeat/Request message does not receive a Heartbeat/Response from the peer in a Heartbeat/Response timeout period, it MUST return an Error message to the peer with the Subtype "HeartbeatResponseTimeout" and immediately terminate the TLS connection, the transport connection and the TLS session.

Structure of this message:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Subtype    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

Type value	Type name
0x08	Heartbeat

Subtype

Subtype value	Subtype name
0x01	Request
0x02	Response

Length

The Length value is 4.

Payload

The payload is null.

4.13.9. AuthBind Message

When this message will be sent:

This message will be exchanged between a PaC to a PAA after completion of AuthInfo message exchange with success and before Success message is returned. The PAA first sends an AuthBind/Request message to the PaC. The PaC sends an AuthBind/Response message in response to the AuthBind/Request message, if the received message is verified successfully.

Meaning of this message:

This message is used for creating cryptographic binding between the TLS session and an authentication session (e.g., an EAP session). Both AuthBind/Request and AuthBind/Response messages contains a field to verify the sender. If the verification fails, an Error message MUST be returned to the sender. See section "MiTM Protection for Authentication Sessions" for details.

Structure of this message:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Subtype    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                       Verification Data                       ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

Type value	Type name
0x09	AuthBind

Subtype

Subtype value	Subtype name
0x01	Request
0x02	Response

Length

The Length value is 16.

Payload

The payload contains Verification Data, which is defined as:

Verification Data = PRF(Phase2_key, "AuthBind Request") [0..11];
  [for Subtype = 0x01 (Request)]

Verification Data = PRF(Phase2_key, "AuthBind Response") [0..11];
  [for Subtype = 0x02 (Response)]

Phase2_key is a session key that is established for the authentication session (e.g., EAP session). The method for deriving a Phase2_key from an authentication session should be specified in each authentication protocol.

5. Protocol Parameters

PortNumber

The destination port number to be used for the messages defined in this document. The same PortNumber is used for UDP, TCP and SCTP. The value of the PortNumber is TBD.

DefaultHeartbeatTimeout

The default timeout value for Heartbeat/Response message. The value of DefaultHeartbeatTimeout is 5 seconds.

DefaultAuthBindTimeout

The default timeout value for AuthBind/Response message. The value of DefaultAuthBindTimeout is 5 seconds.

All-PAA-Nodes

A link-local multicast address used for sending PAADiscover messages. In the case of IPv4, All-PAA-Nodes is the same as "all-hosts" group (224.0.0.1). In the case of IPv6, All-PAA-Nodes is a link-local scoped multicast address to be assigned by IANA.

6. Security Consideration

Potential security threats for PANA over TLS are discussed in this section.

6.1 Security on PAA Discovery

Since PAADiscover and AuthRequest messages are not authenticated, it is possible for an attacker to send those messages with bogus information.

The PAA that receives a bogus PAADiscover message will respond with a AuthRequest message. Since sending an AuthRequest message does not involve in any cryptographic computation or create any state at PAA, there is little impact on PAA for sending AuthRequest messages.

If the PAADiscover message has a bogus source address, then a PaC that is not the originator of the PAA Discover message may receive an

AuthRequest message, which may trigger Full Authentication or Fast Authentication. To reduce the impact of such a false authentication trigger, a PAA MAY have a policy for not accepting a new transport connection from a PaC device that has been authorized for network access until re-authentication becomes necessary for the PaC or that attempts to establish transport connections at a rate higher than the threshold value.

6.2 Security on Transport Connection for TLS

When TCP is used for the TLS transport, it is vulnerable to a blind masquerade attack (i.e., TCP SYN attack), which could let PAAs spend memory resources for creating states for TCP connections. SCTP does not have such vulnerability due to the cookie-based four-way handshake mechanism.

Since transport protocol headers that envelop TLS PDUs are not protected, the headers are vulnerable to deliberate integrity attacks, which may incur data corruption for the transport protocol payload (blind attack is not possible). This kind of attacks are always detected by TLS anyway.

6.3 Security on TLS Handshake

The same security consideration as described in Appendix F of [TLS] is applied to this part. Since this protocol mandates the use of a server certificate for Full Authentication, MiTM attacks against contents carried over TLS connections are protected by TLS.

However, TLS handshake itself does not protect MiTM attacks in which two transport connections are spliced by an attacker in the middle, unless client and server certificates are used for TLS handshake and each certificate is associated with either the IP address used for the transport connection or a DNS entry that is mapped to the transport IP address via secure DNS. DeviceID message exchange is used for protection against MiTM attacks based on transport connection splicing (see next section).

6.4 Security on PANA Message Exchange over TLS Connection

MiTM attacks against transport connections is prevented by using protected DeviceId message exchange over TLS.

EAP or other authentication protocol exchange encapsulated in AuthInfo messages are cryptographically protected by TLS. The authentication session that is established as a result of successful EAP or other authentication protocol exchange is protected from another class of MiTM attack which is described in section "MiTM Protection for Authentication Sessions", by using the cryptographic binding mechanism based on AuthBind message exchange.

All EAP messages including EAP-Response/Identity, EAP-Success and EAP-Failure messages can be encrypted and/or integrity protected. If the contents of EAP messages processed at a PAA need to be protected

from being read in cleartext by the PAA, an appropriate EAP mechanism that supports EAP payload protection (i.e., EAP-SRP, EAP-TLS, EAP-TTLS, etc.) SHOULD be used.

PANA Success/Failure and Heartbeat messages are also cryptographically protected by TLS.

However, deliberate integrity attacks are possible at transport layer for these messages carried over TLS, as described in section "Security on Transport Connection for TLS".

7. Possible Future Direction

This section describes a possible future direction considering the ongoing work on EAP.

There are several EAP methods that use TLS for securing the payload of EAP messages. When those EAP methods are used, it might be possible to carry some of those messages in cleartext without compromising security at all. However, this requires a work in the EAP WG on security analysis as well as appropriate state machine definitions to make sure that only securing EAP payload is enough.

Once it is proven that only securing EAP payload is sufficient or the EAP specification is enhanced to have a method to protect both EAP header and payload, the PANA over TLS protocol will define an optional method that allows carrying AuthInfo messages without protection while other messages are still protected in order to strike a better balance between the required level of security and processing overhead.

Note that whatever EAP-based protection mechanism is applied to EAP header and/or payload, AuthInfo messages that carry EAP messages should be at least integrity protected if the PAA acts as a pass-through EAP authenticator so that an attacker cannot propagate "integrity broken" EAP messages all the way to the authentication server. If those messages are integrity protected by the PANA over TLS protocol, the PAA can immediately reject them before injecting into the backend authentication infrastructure.

8. Acknowledgments

The authors would like to thank Paal Engelstad, Bernard Aboba, and Alper Yegin and Hannes Tschofenig for their valuable comments.

9. References

[DHCPAUTH] R. Droms, et. al., "Authentication for DHCP Messages", RFC 3118, June 2001.

[EAP] L. Blunk, et. al., "PPP Extensible Authentication Protocol (EAP)", RFC 2284, March 1998.

[EAPBIS] L. Blunk, et. al., "PPP Extensible Authentication Protocol (EAP)", Internet-Draft, draft-ietf-pppext-rfc2284bis-06.txt, Work in Progress, September 2002.

[EUI64] IEEE, "Guidelines for 64-bit Global Identifier (EUI-64) Registration Authority", http://standards.ieee.org/regauth/oui/tutorials/EUI64.html.

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

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

[PANAREQ] A. Yegin, et al., "Protocol for Carrying Authentication for Network Access (PANA) Requirements and Terminology", Internet-Draft, Work in Progress, June 2002.

[PIC] Y. Sheffer, et al., "PIC, A Pre-IKE Credential Provisioning Protocol", Internet-Draft, Work in Progress, February 2001.

[TLS] T. Dierks, et al., "The TLS Protocol Version 1.0", RFC 2246, January 1999.

10. Authors' Information

Yoshihiro Ohba
Toshiba America Research, Inc.
P.O. Box 136
Convent Station, NJ 07961-0136
USA
Phone: +1 973 829 5174
Email: yohba@tari.toshiba.com

Shinichi Baba
Toshiba America Research, Inc.
P.O. Box 136
Convent Station, NJ 07961-0136
USA
Phone: +1 973 829 4759
Email: sbaba@tari.toshiba.com

Subir Das
MCC 1D210R, Telcordia Technologies
445 South Street, Morristown, NJ 07960
Phone: +1 973 829 4959
email: subir@research.telcordia.com