Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 8, 2011

N. Cam-Winget
J. Salowey
H. Zhou
Cisco Systems
March 7, 2011

# Transport Layer Security (TLS) Based Transports for Network Endpoint Assessment (NEA) Protocol Exchanges
draft-cam-winget-eap-tlv-03

## Abstract

This document describes how Network Endpoint Assessment (NEA) data can be carried under the protection of a Transport Layer Security (TLS) secured tunnel. This document defines NEA transports for TLS-based Extensible Authentication Protocol (EAP) tunnel methods and for TLS used over TCP.

## Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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 inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 8, 2011.

Table of Contents

## 1. Introduction

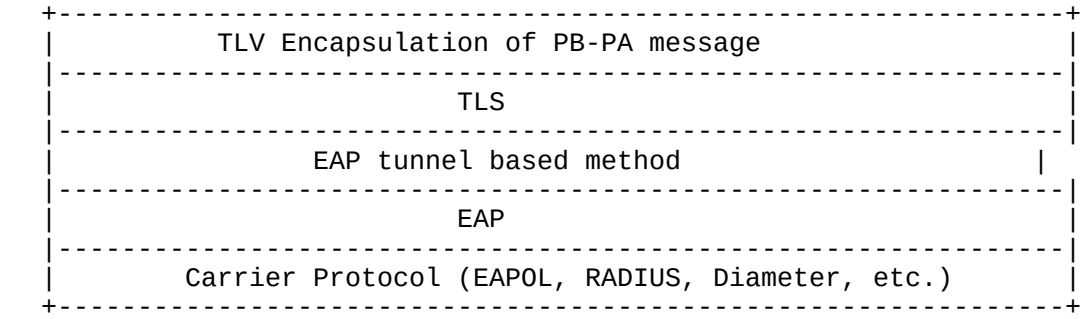
NEA has standardized a transport agnostic Posture Broker (PB) protocol defined in [RFC5793] to effect a network endpoint assessment between a Posture Broker Client and a Posture Broker Server. These PB messages can be transported inside the already defined Type-Length-Value containers in existing TLS-based tunne EAP methods such as PEAP, EAP-FAST and TTLS. Similarly, this document also defines a TCP based transport, PT-TCP, that uses TLVs encapsulated within TLS.

## 2. Specification Requirements

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 [RFC2119] .

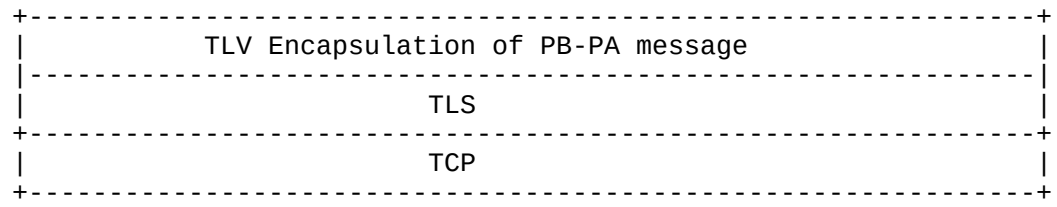
## 3. Protocol Layering Model

When using EAP as the transport, the PB messages can be encapsulated in the TLVs defined by the tunnel EAP methods. For TLS a new TLV container is defined to facilitate the PB transport over TCP. The following diagram demonstrates the relationship between protocols when an EAP tunnel method is used:




EAP based Protocol Layering Model

The following diagram demonstrates the protocol relationship of PB when PT-TCP is used:




PT-TCP based Protocol Layering Model

## 4. Protocol Flows

There are two distinct phases in TLS based transport operation:

1. TLS Setup Phase: are the messages used to establish TLS channel protection for the posture messages. The TLS Setup Phase begins with either the Posture Transport Client or Posture Transport Server initiating the TLS Handshake protocol to establish the protected TLS channel.

2. Data Transport Phase: are the messages that are protected by the TLS Record encapsulation. This phase is usually broken up into an optional entity authentication phase followed by the exchange of TLVs carrying NEA data.

### 4.1. PT-TCP Protocol Flow

This section describes the general flow of messages between the NEA Posture Transport Client and the NEA Posture Transport Server.

#### 4.1.1. Initiating a PT-TCP session

With the use of TLS as the transport, it is possible for either the Posture Transport Client or the Posture Transport Server to initiate a PT-TCP session.

#### 4.1.2. TCP Port Usage

IANA is requested to allocate a TCP Port number for the use of PT-TCP so that both the Posture Transport Client and Posture Transport Server can communicate on a known allocated port.

#### 4.1.3. TLS Setup Phase

Typically, it is the NEA Client (e.g. the Posture Transport Client) that initiates the TLS Setup Phase. However, either party, e.g. the

Posture Transport Client or the Posture Transport Server may establish a TCP connection and initiate the TLS Handshake protocol. Furthermore, the TLS Handshake protocol is also used to establish the cryptographic protections used to secure the data carried within TLS Records.

In typical deployments, it is expected for the initiator of a NEA exchange to initiate the TLS Setup. However, this specification allows for multiple NEA data transactions and as such, each transaction may originate from either the NEA client or the NEA server. Furthermore, through the use of the TLS session capabilities, PT-TCP also allows for the re-use of the TLS based (PT-TCP) session to allow either the NEA Client or the NEA Server to trigger subsequent NEA exchanges.

#### 4.1.4. NEA Data Transport Phase in PT-TCP

Once the PT-TCP session has been established, either the NEA Client or the NEA Server can trigger a NEA data transaction (typically a posture assessment). The initiator for the NEA data transaction encapsulates the PB messages in a TLV as described in Section 5.1.

As PT-TCP is full-duplex (by the TLS design), it supports the full capabilities of the PB-TNC state machine.

#### 4.1.5. Entity Authentication using SASL in PT-TCP

Implementations may support entity authentication through the use of SASL [RFC4422]. This section details the SASL profile for PT-TCP.

Typically, the PT-TCP initiator will also initiate the SASL exchange. The responder presents a list of SASL mechanism it supports through the use of the SASL-AUTH-MECH TLV. The initiator may request a list of SASL authentication mechanisms by sending an empty list of mechanisms in the SASL-AUTH-MECH TLV.

The initiator starts the authentication by sending a SASL-AUTH TLV with the mech field containing the name of the mechanism it selects. If the selected mechanism has an initial response then the client includes that response in the auth-data field. If necessary the responder sends a SASL-AUTH TLV with the auth-data field containing a SASL challenge for the selected mechanism. The SASL-AUTH exchange continues in this manner until the authentication completes upon completion the responder sends a SASL-RESULT TLV. If the authentication is successful the SASL-RESULT TLV contains an result code of success. If the authentication fails the SASL-RESULT TLV contains a result code indicating the reason for the failure. The initiator may abort the exchange by sending a SASL-RESULT TLV with an

ABORT result code.

Implementations MUST provide the EXTERNAL SASL mechanism if the initiator is authenticated during the TLS establishment. Implementations MUST also support the PLAIN SASL mechanism.

#### 4.1.5.1. Service Name

The service name for PT-TCP is "nea-pt-tcp".

#### 4.1.5.2. Mechanism Negotiation

Mechanism Negotiation is performed using the SASL-AUTH-MECH TLV. The SASL-AUTH-MECH TLV contains the list of mechanisms supported by the responder. The initiator may send a SASL-AUTH-MECH TLV with an emptily list to request a list format from the responder.

#### 4.1.5.3. Message Definition

The initiator starts authentication by sending a SASL-AUTH TLV indicating the sleeted mechanism. The initial message may contain an initial response if required by the selected mechanism. Subsequent challenges and response are carried within SASL-AUTH TLVs between the initiator and responder carrying the authentication data for the mechanism. The authentication outcome is communicated in a SASL-RESULT TLV containing a status code.

#### 4.1.5.4. Authorization Identity

The nea-pt-tcp protocol does not make use of an authorization identity.

#### 4.1.5.5. Aborting Authentication

The initiator may abort the authentication exchange by sending the SASL-AUTH-RESULT TLV with a status code of ABORT.

#### 4.1.5.6. Security Layers

The NEA PT-TCP protocol always runs under the protection of TLS. SASL security layers are not used.

#### 4.1.5.7. Multiple Authentications

Only one authentication may be in progress at any one time. Once an authentication completes, successfully on unsuccessfully, a new authentication may be initiated.

### 4.2. Tunnel EAP Message Flow

This section discusses the general flow of messages between the NEA Client's Posture Transport Client and the NEA Server's Posture Transport Server in order to perform NEA assessments when using a tunnel EAP method.

When NEA data exchange is conducted in a tunnel EAP method, it typically consists of four phases:

1. Establishment of EAP tunnel method: a secure and protected TLS channel is established between the Transport Client and Transport Server, after the Transport Server's identity has been authenticated and a shared secret encryption key is established between them.

2. Entity authentication: during this phase, the NEA Client's Posture Transport Client's identity might be optionally authenticated, so appropriate posture assessment policy could be applied according to the authenticated entity. Typically, it is done via an inner EAP method or authentication exchanges within the protected tunnel. In addition, the identity could also be authenticated as part of the tunnel establishment instead (e.g., the client sends a client certificate as part of the tunnel establishment).

3. Posture assessment: the posture data are exchanged between the NEA Client's Posture Transport client and NEA Server's Posture Transport Server. The posture data is encapsulated in a TLV or TLV like type object, as described in Section 5.2.

4. Conclusion phase: the result of the authentication and/or posture assessment is exchanged between the client and server, so they will have synchronized state. Optional cryptographic binding might be done to ensure both peers are involved in both the tunnel establishment and the inner method exchanges. Both sides are ready to tear down the tunnel and finish the EAP method.

At the end of the tunnel EAP method, an EAP-Success or EAP-Failure will be sent by the EAP server to indicate the end of the EAP authentication, and the NAS will apply appropriate authorization policy based on the authentication and posture assessment result.

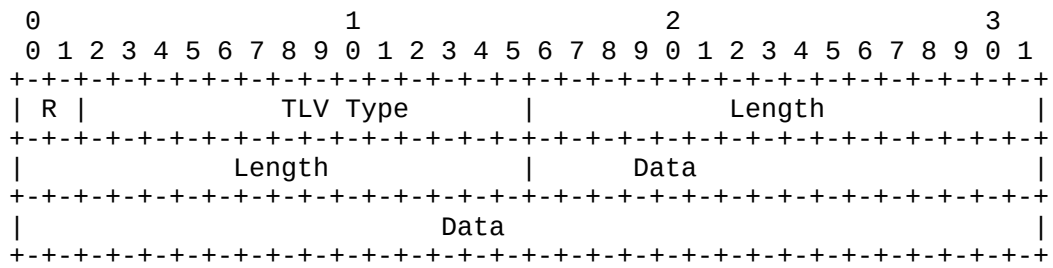
## 5. Packet Formats

As there is no explicit authentication expected in the PB-PA message exchanges, no new inner EAP method is required; rather, the TLV

formats defined in existing EAP tunnel methods can be used to encapsulate and transport PB-PA messages. Similarly, when using TLS a TLV format can be defined to carry NEA data. This section describes how NEA data can be carried in either a tunnel EAP method or TLS.

## 5.1. PT-TCP transport Format

This section defines a Type-Length-Value (TLV) encapsulation for carrying NEA data in a TLS channel. The TLS channel MUST be protected to carry NEA data using the encapsulation defined below. The fields are transmitted from left to right.




R

Reserved, set to zero (0)

TLV Type

TLV Type Code. Allocated Types include:

0 Reserved

1 NEA TLV

2 SASL-MECH TLV

3 SASL-AUTH TLV

4 SASL-RESULT TLV

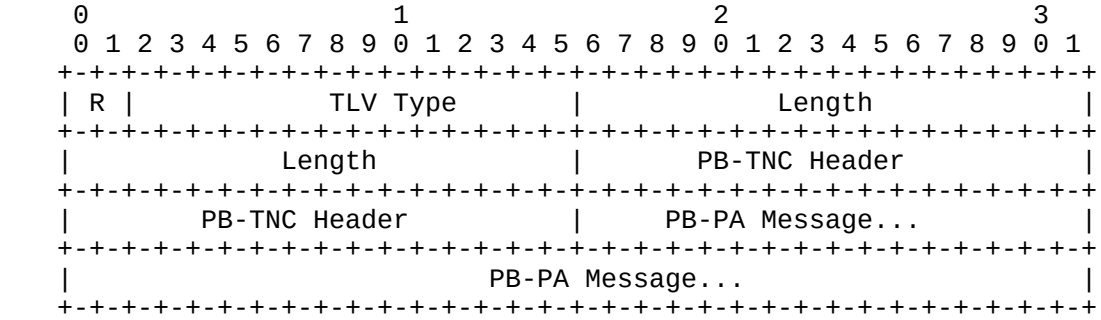
Length

The length of the Data field in octets.

Data

Data according to the TLV type.

#### 5.1.1. NEA TLV




R

Reserved, set to zero (0)

TLV Type

1 for NEA TLV

Length

The length of the Value field in octets.

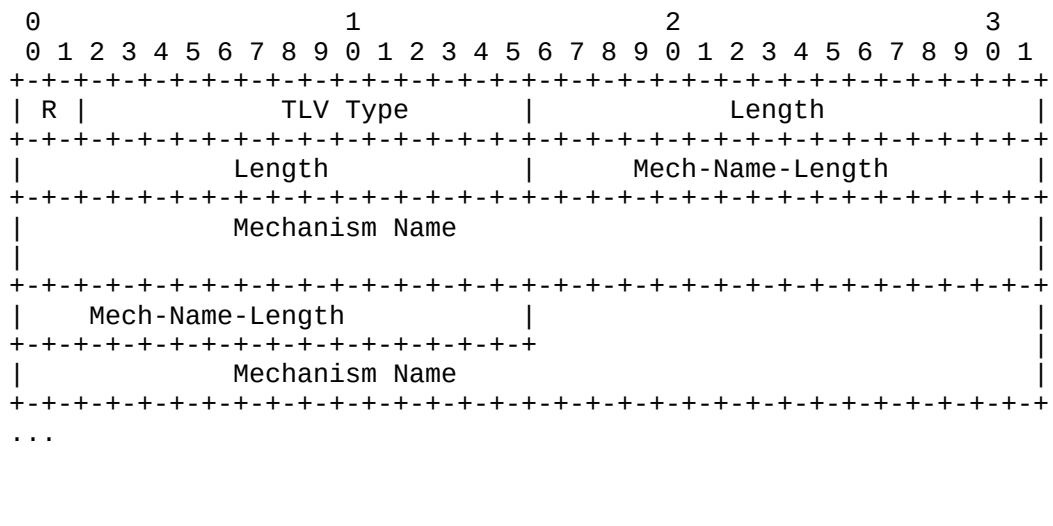
PB-TNC Header

The PB-TNC encapsulation header as described in [RFC5793].

PB-PA Message

The message between the Posture Broker Client and Posture Broker Server as described in [RFC5793].

### 5.1.2. SASL-MECH TLV




The SASL-MECH TLV contains a list of supported SASL mechanisms. Each mechanism name consists of a name length followed by the name. The total length of the list is determined by the TLV length field.

R

Reserved, set to zero (0)

TLV Type

2 for SASL-MECH TLV

Length

The length of the Value field in octets. The value field contains the list of mechanism names.

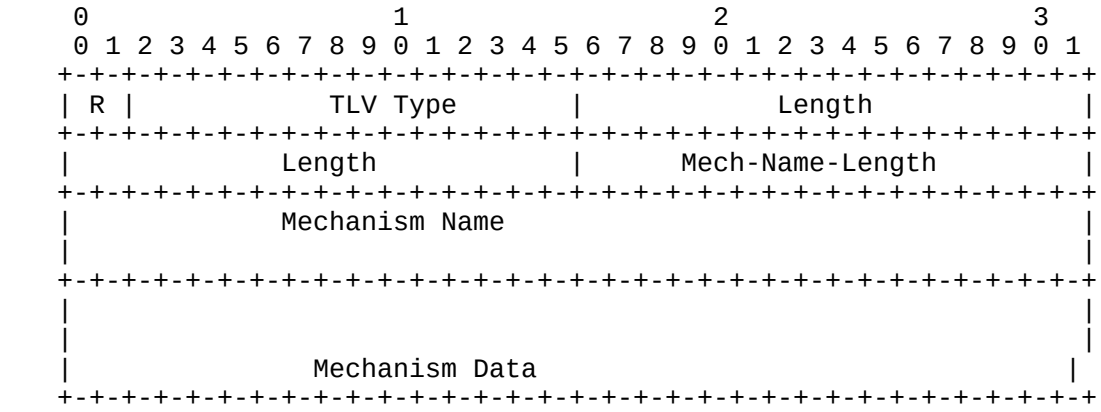
Mech-Name-Length

Length of the mechanism name in bytes.

Mech-Name

SASL mechanism Name adhering to the rules defined in [RFC4422].

### 5.1.3. SASL-AUTH TLV




The SASL-AUTH TLV contains data pertaining a SASL mechanism. The mechanism name is included in each SASL-AUTH TLV. The TLV is used by the initiator to select from a list of supported mechanisms provided by the responder. The initial response from the initiator may contain Mechanism Data containing the initial response. If the mechanism selected does not use an initial response then the mechanism data field is not included. The SASL-AUTH TLV is also used to communicate SASL mechanism data from the responder to the initiator.

R

Reserved, set to zero (0)

TLV Type

3 for SASL-AUTH TLV

Length

The length of the Value field in octets. The value field contains a mechanism name and optional mechanism data..

Mech-Name-Length

Length of the mechanism name in bytes.

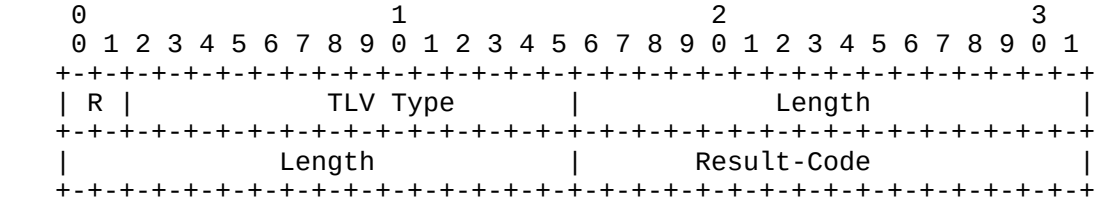
Mech-Name

SASL mechanism Name adhering to the rules defined in [RFC4422]. This is the mechanism selected for use by the initiator.

Mech-Name

SASL mechanism data for named mechanism. This field may be omitted in the initial response from the initiator if the selected mechanism does not use an initial response.

### 5.1.4. SASL-RESULT TLV




The SASL-RESULT TLV contains the result of the SASL Exchange. A result code of 0 indicates success. Other result codes indicate some sort of failure. A result code of 1 indicates the exchange was aborted by the sender. A result code of 2 indicates a failure within the mechanism. Only the responder side of the conversation may send a successful result code. Either side may send a failure result code which terminates the current authentication conversation.

R

Reserved, set to zero (0)

TLV Type

4 for SASL-Result TLV

Length

The length of the Value field in octets. This field is set to 2.

Result Code

The value of the result code.

0 Success

1 Abort

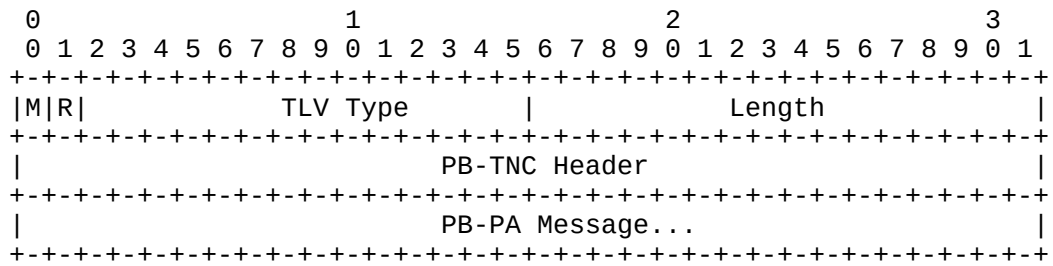
2 Mechanism Failure

## 5.2. Using tunnel EAP to transport NEA data

This section describes the TLV encapsulation used in three predominant tunnel EAP methods deployed today: PEAP, EAP-FAST and TTLS. When using EAP tunnel methods, the tunnel MUST be protected.

### 5.2.1. Carrying NEA data in PEAP or EAP-FAST

As TLV format for PEAP and EAP-FAST are the same, the diagram below shows how PB-PA messages can be encapsulated in the TLVs. Note however that the type assignments when using PEAP versus EAP-FAST may be different. The fields are transmitted from left to right.




M

0 Optional TLV

1 Mandatory TLV

R

Reserved, set to zero (0)

TLV Type

The EAP-FAST NEA TLV type:

TBD

Length

The length of the Value field in octets.

PB-TNC Header

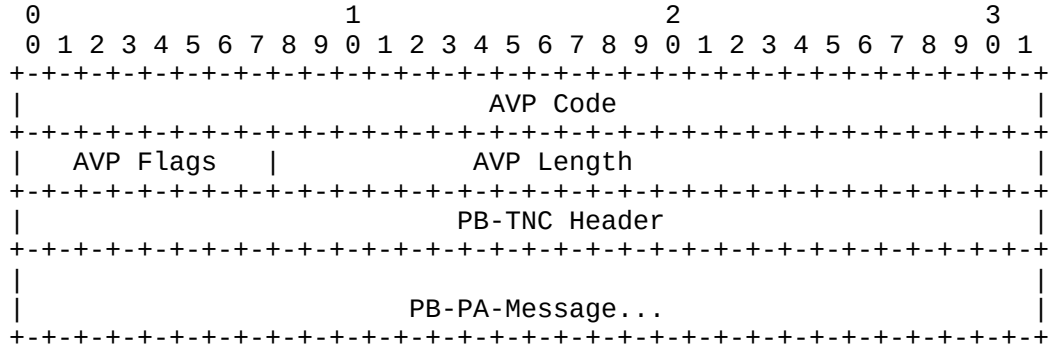
The PB-TNC encapsulation header as described in [RFC5793].

PB-PA Message

The message between the Posture Broker Client and Posture Broker Server as described in [RFC5793].

### 5.2.2. Carrying NEA data in TTLS

The TTLS AVP Format to carry PB-PA messages is defined and described below. The fields are transmitted from left to right.




AVP Code

The TTLS NEA AVP type code:

TBD

AVP Flags

The AVP flags are set to 0.

AVP Length

The length of the AVP in octets.

PB-TNC Header

The PB-TNC encapsulation header as described in [RFC5793].

PB-PA Message

The message between the Posture Broker Client and Posture Broker Server as described in [RFC5793].

## 6. Binding the PA exchange to the TLS Tunnel

Some implementations of the NEA system allow for the external validation of the data collected and sent by the posture collector. In these cases, an external measurement agent (EMA) signs the data sent by the collector. In order to prevent posture data of the endpoint from being used on another machine, the TLS tunnel and the posture data signed by the EMA must be bound together. This is done using the "tls-unique" channel binding defined in RFC 5929 [RFC5929]. The data from the first TLS Finished message sent on the most recent TLS connection handshake is included in the data signed by the EMA. The PA attributes used are specific to the EMA used by the posture collector.

The "tls-unique" channel-binding data can be used whenever a TLS transport is provided, including TLS over TCP and TLS used in tunnel EAP methods. It is RECOMMENDED that posture collectors that support an EMA provide a PA attribute to carry the "tls-unique" channel binding data. The channel binding data MAY be combined with other data using a cryptographic hash or similar technique. The channel binding attribute MUST be signed by the EMA. Posture validators that receive channel binding data MUST verify that it is consistent with the channel binding data obtained from the server-side of the TLS connection.

## 7. Security Considerations

The NEA TLV container carries network endpoint assessment information between the Posture Broker Client and the Posture Broker Server. As some of this data can be sensitive, TLS cipher suites that provide encryption are RECOMMENDED.

To address the potential man-in-the-middle attack similar to the Asokan attack described in [I-D.salowey-nea-asokan] the channel binding mechanism defined in Section 6 SHOULD be used whenever an EMA is available to sign the posture data.

## 8. IANA Considerations

IANA is requested to assign a TCP port number in the "Registered Port" range with the keyword "pt-tcp". This port will be the default port for PT-TCP defined in this document.

IANA is requested to allocate a TLV type from the EAP-FAST TLV Type registry for carrying posture data as specified in Section 5.2.1.

IANA is requested to allocate a Diameter AVP code from the Diameter AVP code registry for carrying posture data as specified in Section 5.2.2.

This document defines a registry for TLV types to be carried within PT-TCP, which may be assigned by Specification Required as defined in [RFC2434]

## 9. Acknowledgements

The authors would like to recognize Susan Thomson, Syam Appala and Subbu Srinivasan for providing input into this draft.

## 10. References

### 10.1. Normative References

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

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

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

[RFC4422] Melnikov, A. and K. Zeilenga, "Simple Authentication and Security Layer (SASL)", RFC 4422, June 2006.

[RFC4493] Song, JH., Poovendran, R., Lee, J., and T. Iwata, "The AES-CMAC Algorithm", RFC 4493, June 2006.

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

[RFC5929] Altman, J., Williams, N., and L. Zhu, "Channel Bindings for TLS", RFC 5929, July 2010.

### 10.2. Informative References

[I-D.salowey-nea-asokan]
Salowey, J. and S. Hanna, "NEA Asokan Attack Analysis", draft-salowey-nea-asokan-00 (work in progress),

October 2010.

## Appendix A. Evaluation Against NEA Requirements

This section evaluates both the PT-TCP and EAP based protocols against the PT requirements defined in the NEA Overview and Requirements and PB-TNC specifications.

### A.1. Evaluation Against Requirement C-1

Requirement C-1 states:

C-1 NEA protocols MUST support multiple round trips between the NEA Client and the NEA Server in a single assessment.

PT-TCP meets this requirement. By using the TLS protocol over TCP, multiple roundtrips of TLS records and thus PT-TCP messages are allowed.

Tunnel EAP meets this requirement. All available Tunnel EAP methods are based on the TLS design which allows for multiple round trips.

### A.2. Evaluation Against Requirement C-2

Requirement C-2 states:

C-2 NEA protocols SHOULD provide a way for both the NEA Client and the NEA Server to initiate a posture assessment or reassessment as needed.

PT-TCP meets this requirement. PT-TCP allows either the NEA Client or the NEA Server to initiate an assessment or reassessment.

Tunnel EAP does not meet this requirement. The typical use case scenario for using a Tunnel EAP method is to service the layer 2 network stack. In this use case, the endpoint would not have an IP address yet as it is requesting network access and thus would not be able to accept requests from the NEA Server. However, once network access has been granted, then yes, the NEA Client could receive (re)assessment requests from the NEA Server.

### A.3. Evaluation Against Requirement C-3

Requirement C-3 states:

C-3 NEA protocols including security capabilities MUST be capable of protecting against active and passive attacks by intermediaries and

endpoints including prevention from replay based attacks.

PT-TCP meets this requirement. TLS includes mechanisms that provide strong cryptographic authentication, message integrity and confidentiality for NEA. In addition, to further mitigate man-in-the middle attacks, the use of channel binding at the PA layer must be used.

Tunnel EAP meets this requirement. All available Tunnel EAP methods are based on the TLS design which provide strong cryptographic authentication, message integrity and confidentiality for NEA. In addition, to further mitigate man-in-the middle attacks, the use of channel binding at the PA layer must be used.

### A.4. Evaluation Against Requirement C-4

Requirement C-4 states:

C-4 The PA and PB protocols MUST be capable of operating over any PT protocol.

This requirement is not applicable to PT, though the PT-TCP protocol is independent of both the PA and PB layer.

This requirement is not applicable to PT, though the Tunnel EAP protocols are independent of both the PA and PB layer.

### A.5. Evaluation Against Requirement C-5

Requirement C-5 states:

C-5 The selection process for NEA protocols MUST evaluate and prefer the reuse of existing open standards that meet the requirements before defining new ones. The goal of NEA is not to create additional alternative protocols where acceptable solutions already exist.

As TLS is a widely used open standard, it should meet this requirement.

As EMU is still in the early stages of standardizing a Tunnel EAP method, this specification reuses already widely deployed, published Tunnel EAP methods. Rather than defining a new Tunnel EAP method, this specification proposes to adopt already used ones and provides guidance for how new Tunnel EAP methods can meet this criteria to allow for NEA to use the method standardized by EMU at some future date.

### A.6. Evaluation Against Requirement C-6

Requirement C-6 states:

C-6 NEA protocols MUST be highly scalable; the protocols MUST support many Posture Collectors on a large number of NEA Clients to be assessed by numerous Posture Validators residing on multiple NEA Servers.

PT-TCP meets this requirement. As PT-TCP is a protocol to establish a protected channel by which NEA data can be transported, it is independent of the content of the data it is transporting and thus can allow for carrying batches of data to multiple Posture Validators or Posture Collectors.

Tunnel EAP methods meet this requirement. As the Tunnel EAP methods define a protected transport channel that is independent of the content it transports, it can carry batches of data from and to multiple Posture Collectors and Posture Validators.

### A.7. Evaluation Against Requirement C-7

Requirement C-7 states:

C-7 The protocols MUST support efficient transport of a large number of attribute messages between the NEA Client and the NEA Server.

PT-TCP meets this requirement. The PT-TCP usurps 6 octets of overhead per PT-TCP message; a small overhead to the ability to carry very many PA-TNC attributes within a PB-TNC batch.

The Tunnel EAP methods meet this requirements subject to the limitations of the underlying EAP protocol and encapsulation mechanisms. Note that a typical use case for the Tunnel EAP methods is that the assessments are brief and used for enabling network access; as such, it is not recommended to use Tunnel EAP methods to carry large amounts of attributes.

### A.8. Evaluation Against Requirement C-8

Requirement C-8 states:

C-8 NEA protocols MUST operate efficiently over low bandwidth or high latency links.

PT-TCP meets this requirement. As TLS was originally designed to work at the TCP layer, it has been proven to work well over either low bandwidth or high latency links.

EAP Tunnel methods meet this requirement. The underlying EAP framework was designed and proven to work under constrained and low latency links.

### A.9. Evaluation Against Requirement C-9

Requirement C-9 states:

C-9 For any strings intended for display to a user, the protocols MUST support adapting these strings to the user's language preferences.

PT-TCP meets this requirement. The PT-TCP protocol does not define messages intended for display to the user.

EAP Tunnel methods meet this requirement. The EAP Tunnel methods do not define messages intended for display to the user.

### A.10. Evaluation Against Requirement C-10

Requirement C-10 states:

C-10 NEA protocols MUST support encoding of strings in UTF-8 format.

PT-TCP meets this requirement. The PT-TCP protocol does not define messages intended for display to the user.

EAP Tunnel methods meet this requirement. The EAP Tunnel methods do not define messages intended for display to the user.

### A.11. Evaluation Against Requirement C-11

Requirement C-11 states:

C-11 Due to the potentially different transport characteristics provided by the underlying candidate PT protocols, the NEA Client and the NEA Server MUST be capable of becoming aware of and adapting to the limitations of the available PT protocol.

PT-TCP meets this requirement. The PT-TCP protocol uses TLS which is designed to provide reliable transport that can adapt to constrained or low bandwidth links.

EAP Tunnel methods meet this requirement. The EAP Tunnel methods are based on TLS which is designed to provide reliable transport and have been proven to adapt and work well under high latency or low bandwidth conditions.

### A.12. Evaluation Against Requirement PT-1

Requirement PT-1 states:

PT-1 The PT protocol MUST NOT interpret the contents of PB messages being transported, i.e., the data it is carrying must be opaque to it.

PT-TCP meets this requirement. The PT-TCP protocol encapsulates PB messages in a TLV container without interpreting their contents.

EAP Tunnel methods meet this requirement. The EAP Tunnel methods define encapsulations for carrying arbitrary data without interpreting their contents.

### A.13. Evaluation Against Requirement PT-2

Requirement PT-2 states:

PT-2 The PT protocol MUST be capable of supporting mutual authentication, integrity, confidentiality, and replay protection of the PB messages between the Posture Transport Client and the Posture Transport Server.

PT-TCP meets this requirement. The PT-TCP protocol uses TLS to provide mutual authentication, integrity, confidentiality, and replay protection.

EAP Tunnel methods meet this requirement. The EAP Tunnel methods are based on TLS which is designed to provide mutual authentication, integrity, confidentiality, and replay protection.

### A.14. Evaluation Against Requirement PT-3

Requirement PT-3 states:

PT-3 The PT protocol MUST provide reliable delivery for the PB protocol. This includes the ability to perform fragmentation and reassembly, detect duplicates, and reorder to provide in-sequence delivery, as required.

PT-TCP meets this requirement. The PT-TCP protocol is designed to work over TCP which provides the fragmentation and reassembly services, detect duplicates and reorder messages if they arrive out of order.

EAP Tunnel methods meet this requirement. The EAP Tunnel methods are based on the EAP framework which provides retransmissions, while

reordering and fragmentation are handled by the individual EAP Tunnel methods.

### A.15. Evaluation Against Requirement PT-4

Requirement PT-4 states:

PT-4 The PT protocol SHOULD be able to run over existing network access protocols such as 802.1X and IKEv2.

PT-TCP does NOT meet this requirement as it is designed to operate over TCP.

EAP Tunnel methods meet this requirement. The EAP Tunnel methods are based on EAP which has been enabled on both 802.1X and IKEv2.

### A.16. Evaluation Against Requirement PT-5

Requirement PT-5 states:

PT-5 The PT protocol SHOULD be able to run between a NEA Client and NEA Server over TCP or UDP (similar to Lightweight Directory Access Protocol (LDAP))

PT-TCP meets this requirement. The PT-TCP protocol is designed to operate over a TCP connection.

EAP Tunnel methods do NOT meet this requirement. The EAP Tunnel methods are designed to work pre-network admission and thus are not able to communicate at the IP layer.

### A.17. Evaluation Against Requirement PT-6

Requirement PT-6 states:

PT-6 The PT protocol MUST be connection oriented; it MUST support confirmed initiation and close down.

PT-TCP meets this requirement. The PT-TCP protocol is designed to operate over a TCP connection which is connection oriented and supports initiation and tear down of the connection.

EAP Tunnel methods meet this requirement. The EAP Tunnel methods are based on EAP which provides both initiation and shutdown.

### A.18. Evaluation Against Requirement PT-7

Requirement PT-7 states:

PT-7 The PT protocol MUST be able to carry binary data.

PT-TCP meets this requirement. The PT-TCP protocol is capable of carrying binary data.

EAP Tunnel methods meet this requirement. The EAP Tunnel methods are capable of carrying binary data.

### A.19. Evaluation Against Requirement PT-8

Requirement PT-8 states:

PT-8 The PT protocol MUST provide mechanisms for flow control and congestion control.

PT-TCP meets this requirement. The PT-TCP protocol operates over TCP which provides flow control and congestion control.

EAP Tunnel methods meet this requirement. The EAP Tunnel methods are based on EAP which, by use of the half-duplex, round-robin message exchange, flow and congestion control are provided.

### A.20. Evaluation Against Requirement PT-9

Requirement PT-9 states:

PT-9 The PT protocol specifications MUST describe the capabilities that they provide for and limitations that they impose on the PB protocol (e.g. half/full duplex, maximum message size).

PT-TCP meets this requirement. This specification has provided the required information.

EAP Tunnel methods meet this requirement. This specification has provided the required information.

Authors' Addresses

Nancy Cam-Winget
Cisco Systems
80 West Tasman Drive
San Jose, CA 95134
US

Email: ncamwing@cisco.com

Joseph Salowey
Cisco Systems
2901 Third Avenue
Seattle, WA 98121
US

Email: jsalowey@cisco.com

Hao Zhou
Cisco Systems
4125 Highlander Parkway
Richfield, OH 44286
US

Email: hzhou@cisco.com

Network Working Group
Internet Draft
Intended status: Proposed Standard
Expires: May 2010

S. Hanna
Juniper Networks
P. Sangster
Symantec Corporation

March 3, 2011

# PT-EAP: Posture Transport (PT) Protocol For EAP Tunnel Methods
# draft-hanna-nea-pt-eap-01.txt

## Abstract

This document specifies PT-EAP, a Posture Broker Protocol compatible with the Trusted Computing Group's IF-T Protocol Bindings for Tunneled EAP Methods (also known as EAP-TNC). The document then evaluates PT-EAP against the requirements defined in the NEA Requirements and PB-TNC specifications.

## Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

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 inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on September 3, 2011.

## Copyright Notice

Table of Contents

## 1. Introduction

This document specifies PT-EAP, a Posture Transport Protocol (PT) compatible with the Trusted Computing Group's IF-T Protocol Bindings for Tunneled EAP Methods (also known as EAP-TNC) [10]. The document then evaluates PT-EAP against the requirements defined in the NEA Requirements [7] and PB-TNC specifications [4].

The PT protocol in the NEA architecture is responsible for transporting PB-TNC batches (often containing PA-TNC [3] attributes) across the network between the NEA Client and NEA Server. The PT protocol also offers strong security protections to ensure the exchanged messages are protected from a variety of threats from hostile intermediaries.

NEA protocols are intended to be used both for pre-admission assessment of endpoints joining the network and to assess endpoints already present on the network. In order to support both usage models, two types of PT protocols are needed. One type of PT operates after the endpoint has an assigned IP address, layering on top of the IP protocol to carry a NEA exchange. The other type of PT operates before the endpoint gains any access to the IP network. This specification defines PT-EAP, the PT protocol used to assess endpoints before they gain access to the network.

PT-EAP is comprised of two related protocols, an outer EAP tunnel method (not defined in this specification) and an inner EAP method that carries the NEA assessment inside the protections of the outer EAP tunnel method. This specification uses the term PT-EAP to refer to both collectively. The inner EAP method is based upon a method called EAP-TNC, which is part of the Trusted Computing Group's TNC architecture and standards. This specification defines the EAP-TNC inner EAP method, while allowing the outer EAP tunnel method to be specified in another specification (possibly defined by another IETF WG). The reason to define PT-EAP as including both the outer EAP tunnel method and the inner EAP method is because both are required to meet the PT requirements.

EAP-TNC is designed to operate as an inner EAP [8] method over an EAP tunnel method that meets the Requirements for a Tunnel Based EAP Method [15]. PT-EAP therefore can operate over a number of existing access protocols that support EAP for authentication. Some examples of such access protocols include 802.1X [5] for wired and wireless networks and IKEv2 [13] for establishing VPNs over IP networks.

This document defines a standard EAP inner method called EAP-TNC. It also shows how EAP-TNC may be carried over two existing EAP tunnel EAP methods: EAP-FAST [12] and EAP-TTLS [14].

### 1.1. Prerequisites

This document does not define an architecture or reference model. Instead, it defines a protocol that works within the reference model described in the NEA Requirements specification [7]. The reader is assumed to be thoroughly familiar with that document. No familiarity with Trusted Computing Group (TCG) specifications is assumed.

### 1.2. Message Diagram Conventions

This specification defines the syntax of EAP-TNC messages using diagrams. Each diagram depicts the format and size of each field in bits. Implementations MUST send the bits in each diagram as they are shown, traversing the diagram from top to bottom and then from left to right within each line (which represents a 32-bit quantity). Multi-byte fields representing numeric values MUST be sent in network (big endian) byte order.

Descriptions of bit field (e.g. flag) values are described referring to the position of the bit within the field. These bit positions are numbered from the most significant bit through the least significant bit so a one octet field with only bit 0 set has the value 0x80.

### 1.3. Terminology

This document reuses many terms defined in the NEA Requirements document [7], such as Posture Transport Client and Posture Transport Server. The reader is assumed to have read that document and understood it.

When defining the EAP-TNC method, this specification does not use the terms "EAP peer" and "EAP authenticator". Instead, it uses the terms "NEA Client" and "NEA Server" since those are considered to be more familiar to NEA WG participants. However, these terms are equivalent for the purposes of these specifications. The part of the NEA Client that terminates EAP-TNC (generally in the Posture Transport Client) is the EAP peer for EAP-TNC. The part of the NEA Server that terminates EAP-TNC (generally in the Posture Transport Server) is the EAP authenticator for EAP-TNC.

### 1.4. Conventions used in this document

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 [1].

## 2. Use of EAP-TNC

EAP-TNC is designed to encapsulate PB-TNC batches in a simple EAP method that can be carried within EAP tunnel methods. The EAP tunnel methods provide confidentiality and message integrity, so EAP-TNC does not have to do so. Therefore, EAP-TNC MUST only be used inside an EAP tunnel method that provides strong cryptographic authentication (possibly server only), message integrity and confidentiality services.

## 3. Definition of EAP-TNC

The EAP-TNC protocol operates between a Posture Transport Client and a Posture Transport Server, allowing them to send PB-TNC batches to each other over an EAP tunnel method. When EAP-TNC is used, the Posture Transport Client in the NEA reference model acts as an EAP peer (terminating the EAP-TNC method on the endpoint) and the Posture Transport Server acts as an EAP authenticator (terminating the EAP-TNC method on the NEA Server).

This section describes and defines the EAP-TNC method. First, it provides a protocol overview and a flow diagram. Second, it describes specific features like version negotiation and fragmentation. Third, it gives a detailed packet description. Finally, it describes how the

tls-unique channel binding [18] may be used to PA-TNC exchanges to the EAP tunnel method, defeating MITM attacks such as the Asokan attack [11].

### 3.1. Protocol Overview

EAP-TNC has two phases that follow each other in strict sequence: negotiation and data transport.

The EAP-TNC method begins with the negotiation phase. The NEA Server starts this phase by sending an EAP-TNC Start message: an EAP Request message of type EAP-TNC with the S (Start) flag set. The NEA Server also sets the Version field as described in section 3.2. This is the only message in the negotiation phase.

The data transport phase is the only phase of EAP-TNC where PB-TNC batches are allowed to be exchanged. This phase always starts with the NEA Client sending a PB-TNC batch to the NEA Server. The NEA Client and NEA Server then engage in a round-robin exchange with one PB-TNC batch in flight at a time. The data transport phase always ends with an EAP Response message from the NEA Client to the NEA Server. This message may be empty (not contain any data) if the NEA Server has just sent the last PB-TNC batch in the PB-TNC exchange.

At the end of the EAP-TNC method, the NEA Server will indicate success or failure to the EAP tunnel method. Some EAP tunnel methods may provide explicit confirmation of inner method success; others may not. This is out of scope for the EAP-TNC method. Successful completion of EAP-TNC does not imply successful completion of the overall authentication nor does EAP-TNC failure imply overall failure. This depends on the administrative policy in place.

The NEA Server and NEA Client may engage in an abnormal termination of the EAP-TNC exchange at any time by simply stopping the exchange. This may also require terminating the EAP tunnel method, depending on the capabilities of the EAP tunnel method.

The NEA Server and NEA Client MUST follow the protocol sequence described in this section.

### 3.2. Version Negotiation

EAP-TNC version negotiation takes place in the first EAP-TNC message sent by the NEA Server (the Start message) and the first EAP-TNC message sent by the NEA Client (the response to the Start message). The NEA Server MUST set the Version field in the Start message to the

maximum EAP-TNC version that the NEA Server supports and is willing to accept.

The NEA Client chooses the EAP-TNC version to be used for the exchange and places this value in the Version field in its response to the Start message. The NEA Client SHOULD choose the value sent by the NEA Server if the NEA Client supports it. However, the NEA Client MAY set the Version field to a value less than the value sent by the NEA Server (for example, if the NEA Client only supports lesser EAP-TNC versions). If the NEA Client only supports EAP-TNC versions greater than the value sent by the NEA Server, the EAP client MUST abnormally terminate the EAP negotiation.

If the version sent by the NEA Client is not acceptable to the NEA Server, the NEA Server MUST terminate the EAP-TNC session immediately. Otherwise, the version sent by the NEA Client is the version of EAP-TNC that MUST be used. Both the NEA Client and the NEA Server MUST set the Version field to the chosen version number in all subsequent EAP-TNC messages in this exchange.

This specification defines version 1 of EAP-TNC. Version 0 is reserved and MUST never be sent. New versions of EAP-TNC (values 2-7) may be defined by Standards Action, as defined in RFC 5226 [6].

### 3.3. Fragmentation

In most cases, EAP-TNC fragmentation will not be required. But PB-TNC batches can be very long and EAP message length is sometimes tightly constrained so EAP-TNC includes a fragmentation mechanism to be used when a particular PB-TNC batch is too long to fit into a single EAP-TNC message.

The fragmentation mechanism used in EAP-TNC is quite similar to the mechanism used by EAP-TLS [17], EAP-TTLS [14], and EAP-FAST [12]. It uses the L flag (length included) and the M flag (more fragments) as well as the Data Length field.

A party (NEA Client or NEA Server) that needs to fragment a long PB-TNC batch SHOULD break the batch into pieces (called "fragments") that will fit into EAP-TNC messages. Then this party sends the fragments in proper sequence, one fragment per EAP-TNC message. The receiving party recognizes the fragments and holds them for reassembly, sending an acknowledgment for each fragment so that the next fragment can be sent (since EAP only allows one message in flight and is half duplex).

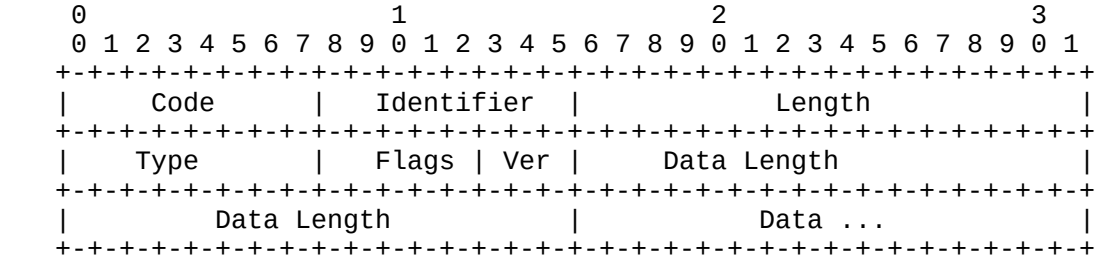
The EAP-TNC message that contains the first fragment MUST have the L flag set to indicate that fragmentation is being initiated. This packet also MUST contain the Data Length field, indicating the total octet length of the unfragmented batch and allowing the party receiving the fragments to know how much data will eventually be coming. The L flag MUST NOT be set and the Data Length field MUST NOT be present in any EAP-TNC message unless that message contains the first fragment of a fragmented PB-TNC batch. The M flag MUST be set on all but the last fragment and MUST NOT be set on the last fragment.

A party that receives an EAP-TNC message with the M flag set MUST respond with an EAP-TNC Acknowledgement message: an EAP-TNC message with no Data and with the L, M, and S flags set to 0. The party that sent an EAP-TNC message with the M flag set MUST wait for the EAP-TNC Acknowledgement packet before sending the next fragment.

EAP-TNC authenticators and NEA Clients MUST include support for EAP-TNC fragmentation with Data Lengths up to 100,000 octets. However, a NEA Server or peer still MAY decide to terminate an EAP-TNC exchange at any time for a variety of reasons.

## 3.4. EAP-TNC Message Format

This section provides a detailed description of the fields in an EAP-TNC message. For a description of the diagram conventions used here, see section 1.2. Since EAP-TNC is an EAP method, the first four fields in each message are mandated by and defined in EAP.




Code

The Code field is one octet and identifies the type of the EAP message. The only values used for EAP-TNC are:

1 - Request

2 - Response

Identifier

The Identifier field is one octet and aids in matching Responses with Requests.

Length

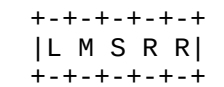
The Length field is two octets and indicates the length in octets of this EAP-TNC message, starting from the Code field. If an EAP-TNC message has been fragmented, the Length field will cover only this fragment and thus doesn't reflect the overall length of the entire unfragmented EAP-TNC message.

Type

38

[IANA Note: This value was previously reserved for another purpose but has been used for EAP-TNC for some time and never used for the other purpose so please assign this value to EAP-TNC.]

Flags




L: Length included

Indicates the presence of the Data Length field in the EAP-TNC message. This flag MUST be set for an EAP-TNC message that contains the first fragment of a fragmented EAP-TNC message and only for such a message. This flag MUST NOT be set for non-fragmented messages.

M: More fragments

Indicates that more fragments are to follow. This flag MUST be set for all EAP-TNC messages that contain a fragmented EAP-TNC message except that this bit MUST NOT be set for EAP-TNC messages that contain the last fragment of a fragmented message. This flag MUST NOT be set for EAP-TNC messages that contain unfragmented Data.

S: Start

Indicates the beginning of an EAP-TNC exchange. This flag MUST be set only for the first message from the NEA Server. If the S flag is set, the EAP message MUST NOT contain Data or have the L or M flags set.

R: Reserved

This flag MUST be set to 0 and ignored upon receipt.

Version

This field is used for version negotiation, as described in section 3.2.

Data Length

Data Length is an optional field four octets in length. It MUST be present if and only if the L flag is set. When present, it indicates the total length, before fragmentation, of a fragmented PB-TNC batch. The Data Length field MUST be set in the EAP-TNC message that contains the first in a series of fragments and MUST NOT be set in subsequent fragments.

Data

Variable length data. The length of the Data field in a particular EAP-TNC message may be determined by subtracting the length of the EAP-TNC header fields from the value of the two octet Length field. Note, however, that this data may be just one part of a longer fragmented PB-TNC batch conveyed in multiple EAP-TNC messages.

### 3.5. Preventing MITM Attacks with Channel Bindings

As described in the NEA Asokan Attack Analysis [16], a sophisticated MITM attack can be mounted against NEA systems. The attacker forwards PA-TNC messages from a healthy machine through an unhealthy one so that the unhealthy machine can gain network access. Because there are easier attacks on NEA systems, like having the unhealthy machine lie about its configuration, this attack is generally only mounted against machines with an External Measurement Agent (EMA). The EMA is a separate entity, difficult to compromise, which measures and attests to the configuration of the endpoint.

To protect against NEA Asokan attacks, the Posture Broker on an EMA-equipped endpoint SHOULD pass the tls-unique channel binding [18] for PT-EAP’s tunnel method to the EMA. This value can then be included

in the EMA's attestation and the Posture Validator responsible for communicating with the EMA may then confirm that the value matches the tls-unique channel binding for its end of the tunnel. If the values match and the integrity of the endpoint is good, the posture sent by the EMA and NEA Client is from the same endpoint as the client side of the TLS connection (since the endpoint knows the tls-unique value) so no man-in-the-middle is forwarding posture. If they differ, an attack has been detected and the Posture Validator SHOULD fail its verification.

# 4. Security Considerations

This section discusses the major threats and countermeasures provided by the EAP-TNC inner EAP method. As discussed throughout the document, the EAP-TNC method is designed to run inside an EAP tunnel method which is capable of protecting the EAP-TNC protocol from many threats. Since the EAP tunnel method will be specified separately, these security considerations specify requirements on the tunnel method but do not evaluate its ability to meet those requirements.

## 4.1. Trust Relationships

In order to understand where security countermeasures are necessary, this section starts with a discussion of where the NEA architecture envisions some trust relationships between the processing elements of the PT-EAP protocol. The following sub-sections discuss the trust properties associated with each portion of the NEA reference model directly involved with the processing of the PT-TNC protocol.

### 4.1.1. Posture Transport Client

The Posture Transport Client is trusted by the Posture Broker Client to:

- Not to observe, fabricate or alter the contents of the PB-TNC batches received from the network
- Not to observe, fabricate or alter the PB-TNC batches passed down from the Posture Broker Client for transmission on the network
- Transmit on the network any PB-TNC batches passed down from the Posture Broker Client
- Deliver properly security protected messages received from the network that are destined for the Posture Broker Client

- Provide configured security protections (e.g. authentication, integrity and confidentiality) for the Posture Broker Client's PB-TNC batches sent on the network
- Expose the authenticated identity of the Posture Transport Server
- Verify the security protections placed upon messages received from the network to ensure the messages are authentic and protected from attacks on the network
- Provide a secure, reliable, in order delivery, full duplex transport for the Posture Broker Client's messages

The Posture Transport Client is trusted by the Posture Transport Server to:

- Not send malicious traffic intending to harm (e.g. denial of service) the Posture Transport Server
- Not to intentionally send malformed messages to cause processing problems for the Posture Transport Server
- Not to send invalid or incorrect responses to messages (e.g. errors when no error is warranted)
- Not to ignore or drop messages causing issues for the protocol processing
- Verify the security protections placed upon messages received from the network to ensure the messages are authentic and protected from attacks on the network

#### 4.1.2. Posture Transport Server

The Posture Transport Server is trusted by the Posture Broker Server to:

- Not to observe, fabricate or alter the contents of the PB-TNC batches received from the network
- Not to observe, fabricate or alter the PB-TNC batches passed down from the Posture Broker Server for transmission on the network
- Transmit on the network any PB-TNC batches passed down from the Posture Broker Server

- Deliver properly security protected messages received from the network that are destined for the Posture Broker Server
- Provide configured security protections (e.g. authentication, integrity and confidentiality) for the Posture Broker Server's messages sent on the network
- Expose the authenticated identity of the Posture Transport Client
- Verify the security protections placed upon messages received from the network to ensure the messages are authentic and protected from attacks on the network

The Posture Transport Server is trusted by the Posture Transport Client to:

- Not send malicious traffic intending to harm (e.g. denial of service) the Posture Transport Server
- Not to send malformed messages
- Not to send invalid or incorrect responses to messages (e.g. errors when no error is warranted)
- Not to ignore or drop messages causing issues for the protocol processing
- Verify the security protections placed upon messages received from the network to ensure the messages are authentic and protected from attacks on the network

### 4.2. Security Threats and Countermeasures

Beyond the trusted relationships assumed in section 4.1. the PT-EAP EAP method faces a number of potential security attacks that could require security countermeasures.

Generally, the PT protocol is responsible for providing strong security protections for all of the NEA protocols so any threats to PT's ability to protect NEA protocol messages could be very damaging to deployments. For the PT-EAP method, most of the cryptographic security is provided by the outer EAP tunnel method and EAP-TNC is encapsulated within the protected tunnel. Therefore, this section highlights the cryptographic requirements that need to be met by the EAP tunnel method carrying EAP-TNC in order to meet the NEA PT requirements.

Once the message is delivered to the Posture Broker Client or Posture Broker Server, the posture brokers are trusted to properly safely process the messages.

#### 4.2.1. Message Theft

When EAP-TNC messages are sent over unprotected network links or spanning local software stacks that are not trusted, the contents of the messages may be subject to information theft by an intermediary party. This theft could result in information being recorded for future use or analysis by the adversary. Messages observed by eavesdroppers could contain information that exposes potential weaknesses in the security of the endpoint, or system fingerprinting information easing the ability of the attacker to employ attacks more likely to be successful against the endpoint. The eavesdropper might also learn information about the endpoint or network policies that either singularly or collectively is considered sensitive information. For example, if EAP-TNC is housed in an EAP tunnel method that does not provide confidentiality protection, an adversary could observe the PA-TNC attributes included in the PB-TNC batch and determine that the endpoint is lacking patches, or particular sub-networks have more lenient policies.

In order to protect again NEA assessment message theft, the EAP tunnel method carrying EAP-TNC MUST provide strong cryptographic authentication, integrity and confidentiality protection. The use of bi-directional authentication in the EAP tunnel method carrying EAP-TNC ensures that only properly authenticated and authorized parties may be involved in an assessment message exchange. When EAP-TNC is carried within a cryptographically protected EAP tunnel method like EAP-TTLS, all of the PB-TNC and PA-TNC protocol messages contents are hidden from potential theft by intermediaries lurking on the network.

#### 4.2.2. Message Fabrication

Attackers on the network or present within the NEA system could introduce fabricated PT-EAP messages intending to trick or create a denial of service against aspects of an assessment. For example, an adversary could attempt to insert into the message exchange fake PT-EAP error codes in order to disrupt communications.

The EAP tunnel method carrying an EAP-TNC method needs to provide strong security protections for the complete message exchange over the network. These security protections prevent an intermediary from being able to insert fake messages into the assessment. For example, the EAP-TTLS method’s use of hashing algorithms provides strong integrity protections that allow for detection of any changes in the

content of the message exchange. Additionally, adversaries are unable to observe the EAP-TNC method housed inside of an encrypting EAP tunnel method (e.g. EAP-TTLS) because the messages are encrypted by the TLS [2] ciphers, so an attacker would have difficulty in determining where to insert the falsified message, since the attacker is unable to determine where the message boundaries exist.

#### 4.2.3. Message Modification

This attack could allow an active attacker capable of intercepting a message to modify a PT-EAP message or transported PA-TNC attribute to a desired value to ease the compromise of an endpoint. Without the ability for message recipients to detect whether a received message contains the same content as what was originally sent, active attackers can stealthily modify the attribute exchange.

The EAP-TNC method leverages the EAP tunnel method (e.g. EAP-TTLS) to provide strong authentication and integrity protections as a countermeasure to this threat. The bi-directional authentication prevents the attacker from acting as an active man-in-the-middle to the protocol that could be used to modify the message exchange. The strong integrity protections (hashing) offered by EAP-TTLS allows the EAP-TNC message recipients to detect message alterations by other types of network based adversaries. Because EAP-TNC does not itself provide explicit integrity protection for the EAP-TNC payload, an EAP tunnel method that offers strong integrity protection is required to mitigate this threat.

#### 4.2.4. Denial of Service

A variety of types of denial of service attacks are possible against the PT-EAP if the message exchange are left unprotected while traveling over the network. The Posture Transport Client and Posture Transport Server are trusted not to participate in the denial of service of the assessment session, leaving the threats to come from the network.

The EAP-TNC method primarily relies on the outer EAP tunnel method to provide strong authentication (at least of one party) and deployers are expected to leverage other EAP methods to authenticate the other party (typically the client) within the protected tunnel. The use of a protected bi-directional authentication will prevent unauthorized parties from participating in a PT-EAP exchange.

After the cryptographic authentication by the EAP tunnel method, the session can be encrypted and hashed to prevent undetected modification that could create a denial of service situation.

However it is possible for an adversary to alter the message flows causing each message to be rejected by the recipient because it fails the integrity checking.

#### 4.2.5. NEA Asokan Attacks

As described in section 3.5. and in the NEA Asokan Attack Analysis [16], a sophisticated MITM attack can be mounted against NEA systems. The attacker forwards PA-TNC messages from a healthy machine through an unhealthy one so that the unhealthy machine can gain network access. Section 3.5. and the NEA Asokan Attack Analysis provide a detailed description of this attack and of the countermeasures that can be employed against it.

Because lying endpoint attacks are much easier than Asokan attacks and the only known effective countermeasure against lying endpoint attacks is the use of an External Measurement Agent (EMA), countermeasures against an Asokan attack are not necessary unless an EMA is in use. However, PT-EAP implementers may not know whether an EMA will be used with their implementation. Therefore, PT-EAP implementers SHOULD support these countermeasures by providing the value of the tls-unique channel binding to higher layers in the NEA reference model: Posture Broker Clients, Posture Broker Servers, Posture Collectors, and Posture Validators.

### 4.3. Requirements for EAP Tunnel Methods

Because the PT-EAP inner method described in this specification relies on the outer EAP tunnel method for a majority of its security protections, this section reiterates the PT requirements that MUST be met by the IETF standard EAP tunnel method for use with PT-EAP.

The security requirements described in this specification MUST be implemented in any product claiming to be PT-EAP compliant. The decision of whether a particular deployment chooses to use these protections is a deployment issue. A customer may choose to avoid potential deployment issues or performance penalties associated with the use of cryptography when the required protection has been achieved through other mechanisms (e.g. physical isolation). If security mechanisms may be deactivated by policy, an implementation SHOULD offer an interface to query how a message will be (or was) protected by PT so higher layer NEA protocols can factor this into their decisions.

RFC 5209 includes the following requirement that is to be applied during the selection of the EAP tunnel method(s) used in conjunction with EAP-TNC:

PT-2 The PT protocol MUST be capable of supporting mutual authentication, integrity, confidentiality, and replay protection of the PB messages between the Posture Transport Client and the Posture Transport Server.

Note that mutual authentication could be achieved by a combination of a strong authentication of one party (e.g. TLS server when EAP-TTLS is used) by the EAP tunnel method in conjunction with a second authentication of the other party (e.g. client authentication inside the protected tunnel) by another EAP method running prior to EAP-TNC.

Having the Posture Transport Client always authenticate the Posture Transport Server provides assurance to the NEA Client that the NEA Server is authentic (not a rogue or MiTM) prior to disclosing secret or potentially privacy sensitive information about what is running or configured on the endpoint. However the NEA Server's policy may allow for the delay of the authentication of the NEA Client until a suitable protected channel has been established allowing for non-cryptographic NEA Client credentials (e.g. username/password) to be used. Whether the communication channel is established with both or one party performing a cryptographic authentication, the resulting channel needs to provide strong integrity and confidentiality protection to its contents. These protections are to be bound to at least the authentication of the NEA Client, so the session is cryptographically bound to a particular authentication event.

To support countermeasures against NEA Asokan attacks as described in section 3.5. the EAP Tunnel Method used with EAP-TNC will need to support the tls-unique channel binding. This should not be a high bar since all EAP tunnel methods currently support this but not all implementations of those methods may do so.

## 4.4. Candidate EAP Tunnel Method Protections

This section discusses how EAP-TNC is used within various EAP tunnel methods to meet the PT requirements from section 4.3.

EAP-FAST and EAP-TTLS make use of TLS [2] to protect the transport of information between the NEA Client and NEA Server. Each of these EAP tunnel methods has two phases. In the first phase, a TLS tunnel is established between NEA Client and NEA Server. In the second phase, the tunnel is used to pass other information. PT-EAP requires that establishing this tunnel include at least an authentication of the NEA Server by the NEA Client.

The phase two dialog may include authentication of the user by doing other EAP methods or in the case of TTLS by using non-EAP

authentication dialogs. EAP-TNC is also carried by the phase two tunnel allowing the NEA assessment to be within an encrypted and integrity protected transport.

With all these methods, a cryptographic key is derived from the authentication that may be used to secure later transmissions. Each of these methods employs at least a NEA Server authentication using an X.509 certificates. Within each EAP tunnel method will exist a set of inner EAP method (or an equivalent using TLVs if inner methods aren't directly supported.) These inner methods may perform additional security handshakes including more granular authentications or exchanges of integrity information (such as EAP-TNC.) At some point after the conclusion of each inner EAP method, some of the methods will export the established secret keys to the outer tunnel method. It's expected that the outer method will cryptographically mix these keys into any keys it is currently using to protect the session and perform a final operation to determine whether both parties have arrived at the same mixed key. This cryptographic binding of the inner method results to the outer methods keys is essential for detection of conventional (non-NEA) Asokan attacks.

### 4.5. Security Claims for EAP-TNC as per RFC3748

This section summarizes the security claims as required by RFC3748 Section 7.2:

| | |
|---|---|
| Auth. mechanism: | None |
| Ciphersuite negotiation: | No |
| Mutual authentication: | No |
| Integrity protection: | No |
| Replay protection: | No |
| Confidentiality: | No |
| Key derivation: | No |
| Key strength: | N/A |
| Dictionary attack resistant: | N/A |
| Fast reconnect: | No |
| Crypt. binding: | N/A |
| Session independence: | N/A |
| Fragmentation: | Yes |
| Channel binding: | No |

## 5. Privacy Considerations

The role of PT-EAP is to act as a secure transport for PB-TNC over a network before the endpoint has been admitted to the network. As a transport protocol, PT-EAP does not directly utilize or require

direct knowledge of any personally identifiable information (PII). PT-EAP will typically be used in conjunction with other EAP methods that provide for the user authentication (if bi-directional authentication is used), so the user's credentials are not directly seen by the EAP-TNC inner method. Therefore, the Posture Transport Client and Posture Transport Server's implementation of EAP-TNC MUST NOT observe the contents of the carried PB-TNC batches that could contain PII carried by PA-TNC or PB-TNC.

While EAP-TNC does not provide cryptographic protection for the PB-TNC batches, it is designed to operate within an EAP tunnel method that provides strong authentication, integrity and confidentiality services. Therefore, it is important for deployers to leverage these protections in order to prevent disclosure of PII potentially contained within PA-TNC or PB-TNC within the EAP-TNC payload.

## 6. IANA Considerations

This document defines an EAP method type named EAP-TNC with the value 38.

[IANA Note: This value was previously reserved for another purpose but has been used for EAP-TNC for some time and never used for another purpose so please assign this value to EAP-TNC.]

This document also defines one new IANA registry: EAP-TNC Versions. This section explains how this registry works.

Because only eight (8) values are available in this registry, a high bar is set for new assignments. The only way to register new values in this registry is through Standards Action (via an approved Standards Track RFC).

### 6.1. Registry for EAP-TNC Versions

The name for this registry is "EAP-TNC Versions". Each entry in this registry includes a decimal integer value between 1 and 7 identifying the version, and a reference to the RFC where the version is defined.

The following entries for this registry are defined in this document. Once this document becomes an RFC, they will become the initial entries in the registry for EAP-TNC Versions. Additional entries to this registry are added by Standards Action, as defined in RFC 5226 [6].

| Value | Defining Specification |
|---|---|
| 1 | RFC # Assigned to this I-D |

## 7. References

### 7.1. Normative References

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

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

[3] Sangster P., Narayan K., "PA-TNC: A Posture Attribute Protocol (PA) Compatible with TNC", RFC 5792, March 2010.

[4] Sahita, R., Hanna, S., and R. Hurst, "PB-TNC: A Posture Broker Protocol (PB) Compatible with TNC", RFC 5793, March 2010.

[5] LAN/MAN Standards Committee of the IEEE Computer Society, Standard for Local and Metropolitan Area Networks - Port Based Network Access Control, IEEE Std. 802.1X-2004, December 2004.

[6] T. Narten, H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, May 2008.

### 7.2. Informative References

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

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

[9] Sangster, P., "PT-TLS: A Posture Transport Protocol (PT) Compatible with TNC Using Transport Layer Security (TLS)", draft-sangster-nea-pt-tls-02.txt, work in progress, March 2011.

[10] Trusted Computing Group, "TNC IF-T: Binding to TLS", http://www.trustedcomputinggroup.org/files/resource_files/51F0757E-1D09-3519-AD63B6FD099658A6/TNC_IFT_TLS_v1_0_r16.pdf, May 2009.

[11] N. Asokan, Valtteri Niemi, Kaisa Nyberg, "Man in the Middle Attacks in Tunneled Authentication Protocols", Nokia Research Center, Finland, Nov. 11, 2002, http://eprint.iacr.org/2002/163.pdf

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

[13] C. Kaufman, P. Hoffman, Y. Nir, P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

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

[15] K. Hoeper, S. Hanna, H. Zhou, J. Salowey, "Requirements for a Tunnel Based EAP Method", draft-ietf-emu-eaptunnel-req-09.txt (work in progress), December 2010.

[16] J. Salowey, S. Hanna, "NEA Asokan Attack Analysis", draft-salowey-nea-asokan-00.txt (work in progress), October 2010.

[17] D. Simon, B. Aboba, R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, March 2008.

[18] J. Altman, N. Williams, L. Zhu, "Channel Bindings for TLS", RFC 5929, July 2010.

## 8. Acknowledgments

Thanks to the Trusted Computing Group for contributing the initial text upon which this document was based.

The authors of this draft would like to acknowledge the following people who have contributed to or provided substantial input on the preparation of this document or predecessors to it: Amit Agarwal, Morteza Ansari, Diana Arroyo, Stuart Bailey, Boris Balacheff, Uri Blumenthal, Gene Chang, Scott Cochrane, Pasi Eronen, Aman Garg, Sandilya Garimella, David Grawrock, Thomas Hardjono, Chris Hessing, Ryan Hurst, Hidenobu Ito, John Jerrim, Meenakshi Kaushik, Greg Kazmierczak, Scott Kelly, Bryan Kingsford, PJ Kirner, Sung Lee, Lisa Lorenzin, Mahalingam Mani, Bipin Mistry, Seiji Munetoh, Rod Murchison, Barbara Nelson, Kazuaki Nimura, Ron Pon, Ivan Pulleyn, Alex Romanyuk, Ravi Sahita, Chris Salter, Mauricio Sanchez, Paul Sangster, Dean Sheffield, Curtis Simonson, Jeff Six, Ned Smith,

Michelle Sommerstad, Joseph Tardo, Lee Terrell, Chris Trytten, and John Vollbrecht.

This document was prepared using 2-Word-v2.0.template.dot.

# Appendix A. Evaluation Against NEA Requirements

This section evaluates the PT-EAP protocol against the PT requirements defined in the NEA Overview and Requirements and PB-TNC specifications. Each subsection considers a separate requirement and highlights how PT-EAP meets the requirement.

## A.1. Evaluation Against Requirement C-1

Requirement C-1 says:

C-1 NEA protocols MUST support multiple round trips between the NEA Client and NEA Server in a single assessment.

PT-EAP meets this requirement. Use of the EAP protocol along with EAP-TNC and suitable EAP tunnel methods will allow for multiple roundtrips.

## A.2. Evaluation Against Requirements C-2

Requirement C-2 says:

C-2 NEA protocols SHOULD provide a way for both the NEA Client and the NEA Server to initiate a posture assessment or reassessment as needed.

PT-EAP does NOT meet this requirement. Generally EAP is used by the endpoint during the joining of the network. At that time, the endpoint lacks an IP address so is unable to accept inbound posture assessment requests from the NEA Server. Subsequent reassessments of the endpoint after it has been given access to a portion of the IP network can use the PT-TLS protocol that supports the NEA Client and NEA Server to initiate an assessment.

## A.3. Evaluation Against Requirements C-3

Requirement C-3 says:

C-3 NEA protocols including security capabilities MUST be capable of protecting against active and passive attacks by intermediaries and endpoints including prevention from replay based attacks.

PT-EAP meets this requirement by leveraging the security capabilities of the underlying EAP tunnel method. EAP-TNC itself does not provide protection against a variety of

potential attacksso it relies on cryptographic support by the EAP tunnel method.

### A.4. Evaluation Against Requirements C-4

Requirement C-4 says:

C-4 The PA and PB protocols MUST be capable of operating over any PT protocol. For example, the PB protocol must provide a transport independent interface allowing the PA protocol to operate without change across a variety of network protocol environments (e.g. EAP/802.1X, PANA, TLS and IKE/IPsec).

Not applicable to PT, but PT-EAP is independent of PA and PB allowing those protocols to operate over other PT protocols.

### A.5. Evaluation Against Requirements C-5

Requirement C-5 says:

C-5 The selection process for NEA protocols MUST evaluate and prefer the reuse of existing open standards that meet the requirements before defining new ones. The goal of NEA is not to create additional alternative protocols where acceptable solutions already exist.

Based on this requirement, PT-EAP should receive a strong preference. PT-EAP is compatible with IF-T Binding to Tunneled EAP Methods 1.1, an open TCG specification that has been widely implemented.

### A.6. Evaluation Against Requirements C-6

Requirement C-6 says:

C-6 NEA protocols MUST be highly scalable; the protocols MUST support many Posture Collectors on a large number of NEA Clients to be assessed by numerous Posture Validators residing on multiple NEA Servers.

PT-EAP meets this requirement. The PT-EAP protocol is independent of the number of Posture Collectors and Posture Validators.

### A.7. Evaluation Against Requirements C-7

Requirement C-7 says:

C-7 The protocols MUST support efficient transport of a large number of attribute messages between the NEA Client and the NEA Server.

PT-EAP meets this requirement, subject to the limitations of the underlying EAP protocol. PT-EAP allows for the transport of a very large number of attributes, up to 2^32 - 1 octets per PB-TNC batch. Furthermore, the PT-EAP protocol transports data efficiently, only adding 10 octets of overhead per PT-EAP message, which is small considering that a single PT-EAP message may carry multiple PA-TNC attributes.

However, it is important to note that the EAP protocol that underlies PT-EAP is not a good choice for transporting large amounts of data. EAP only supports one packet in flight at a time, which severely limits throughput. Further, some network equipment imposes timeout restrictions on EAP exchanges. Therefore, PT-EAP should not be used to transport large amounts of attributes.

### A.8. Evaluation Against Requirements C-8

Requirement C-8 says:

C-8 NEA protocols MUST operate efficiently over low bandwidth or high latency links.

PT-EAP protocols meet this requirement. PT-EAP was designed to minimize the amount of overhead included in the protocol to allow for efficient use over bandwidth or latency constrained network links.

### A.9. Evaluation Against Requirements C-9

Requirement C-9 says:

C-9 For any strings intended for display to a user, the protocols MUST support adapting these strings to the user’s language preferences.

PT-EAP meets this requirement. PT-EAP does not include messages intended for display to the user.

### A.10. Evaluation Against Requirements C-10

Requirement C-10 says:

C-10 NEA protocols MUST support encoding of strings in UTF-8 format.

PT-EAP meets this requirement. The PT-EAP protocol does not include any strings in its fields but it allows higher-layer protocols to encode their strings in UTF-8 format. This allows the protocol to support a wide range of languages efficiently.

### A.11. Evaluation Against Requirements C-11

Requirement C-11 says:

C-11 Due to the potentially different transport characteristics provided by the underlying candidate PT protocols, the NEA Client and NEA Server MUST be capable of becoming aware of and adapting to the limitations of the available PT protocol. For example, some PT protocol characteristics that might impact the operation of PA and PB include restrictions on: which end can initiate a NEA connection, maximum data size in a message or full assessment, upper bound on number of roundtrips, and ordering (duplex) of messages exchanged. The selection process for the PT protocols MUST consider the limitations the candidate PT protocol would impose upon the PA and PB protocols.

PT-EAP meets this requirement. The PT-EAP implementations may be limited in number of roundtrips, assessment overall time, or data transmission. These constraints will be exposed up the protocol stack so the Posture Broker Client and Posture Broker Server can optimize and make most efficient use of the available resources during the assessment.

### A.12. Evaluation Against Requirements PT-1

Requirement PT-1 says:

PT-1 The PT protocol MUST NOT interpret the contents of PB messages being transported, i.e., the data it is carrying must be opaque to it.

PT-EAP meets this requirement. The PT-EAP encapsulates PB-TNC batches without interpreting their contents.

### A.13. Evaluation Against Requirements PT-2

Requirement PT-2 says:

PT-2 The PT protocol MUST be capable of supporting mutual authentication, integrity, confidentiality, and replay protection of the PB messages between the Posture Transport Client and the Posture Transport Server.

PT-EAP meets this requirement. The PT-EAP protocol leverages an EAP tunnel method to provide mutual authentication, integrity protection and confidentiality as well as replay protection. For more information see the Security Considerations in section 4.

### A.14. Evaluation Against Requirements PT-3

Requirement PT-3 says:

PT-3 The PT protocol MUST provide reliable delivery for the PB protocol. This includes the ability to perform fragmentation and reassembly, detect duplicates, and reorder to provide in-sequence delivery, as required.

EAP-TNC includes support for fragmentation and the underlying EAP tunnel methods include support for duplicate detection and reordering to provide in-sequence delivery.

### A.15. Evaluation Against Requirements PT-4

Requirement PT-4 says:

PT-4 The PT protocol SHOULD be able to run over existing network access protocols such as 802.1X and IKEv2.

PT-EAP meets this requirement. The PT-EAP operates on top of the 802.1X and IKEv2 protocols.

### A.16. Evaluation Against Requirements PT-5

Requirement PT-5 says:

PT-5 The PT protocol SHOULD be able to run between a NEA Client and NEA Server over TCP or UDP (similar to Lightweight Directory Access Protocol (LDAP)).

PT-EAP does NOT meet this requirement. PT-EAP is intended for a different usage. PT-EAP is intended to be used for pre-network admission before the endpoint has been given an IP address and routes on the network. This means that network layer protocols such as IP are not yet able to communicate with

the system. The PT-TLS (PT Binding to TLS) [9] meets this requirement.

### A.17. Evaluation Against Requirements PT-6 (from PB-TNC specification)

Requirement PT-6 says:

PT-6 The PT protocol MUST be connection oriented; it MUST support confirmed initiation and close down.

PT-EAP meets this requirement. The PT-EAP fits into the EAP framework which provides for orderly initiation and shutdown.

### A.18. Evaluation Against Requirements PT-7 (from PB-TNC specification)

Requirement PT-7 says:

PT-7 The PT protocol MUST be able to carry binary data.

PT-EAP meets this requirement. The PT-EAP is capable of carrying binary data.

### A.19. Evaluation Against Requirements PT-8 (from PB-TNC specification)

Requirement PT-8 says:

PT-8 The PT protocol MUST provide mechanisms for flow control and congestion control.

PT-EAP meets this requirement. The PT-EAP utilizes EAP’s half duplex, round robin message exchange to provide flow and congestion control.

### A.20. Evaluation Against Requirements PT-9 (from PB-TNC specification)

Requirement PT-9 says:

PT-9 PT protocol specifications MUST describe the capabilities that they provide for and limitations that they impose on the PB protocol (e.g. half/full duplex, maximum message size).

PT-EAP specification meets this requirement. This specification discusses the level of transport service provided to the Posture Broker Client and Posture Broker Server. Generally, the PT-EAP method supports the pre-network admission usages discussed in RFC 5209. The maximum message size for PT-EAP is 2^16-10 octets. EAP by its nature is half duplex and

simple which allows it to be used in a wide variety of settings including over link layer protocols during the entrance to the network.

## Authors' Addresses

Steve Hanna
Juniper Networks, Inc.
79 Parsons Street
Brighton, MA 02135 USA
Email: shanna@juniper.net

Paul Sangster
Symantec Corporation
6825 Citrine Drive
Carlsbad, CA 92009 USA
Email: paul_sangster@symantec.com

Network Working Group
Internet Draft
Intended status: Proposed Standard
Expires: September 2011

P. Sangster
Symantec Corporation
N. Cam-Winget
J. Salowey
Cisco Systems

June 13, 2011

# PT-TLS: A Posture Transport (PT) Protocol Based on Transport Layer Security (TLS) draft-ietf-nea-pt-tls-00.txt

## Abstract

This document specifies PT-TLS, a Posture Transport (PT) protocol that carries the Network Endpoint Assessment (NEA) message exchange under the protection of a Transport Layer Security (TLS) secured tunnel.

## Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

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 inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on September 13, 2011.

## Copyright Notice

Table of Contents

## 1. Introduction

This document specifies PT-TLS, a Posture Transport (PT) protocol protected by a Transport Layer Security (TLS) tunnel. The document then evaluates PT-TLS against the applicable requirements defined in the NEA Overview and Requirements [RFC5209] and PB-TNC [RFC5793] specifications.

NEA protocols are intended to be used for pre-admission assessment of endpoints joining the network and to assess endpoints already present on the network. In order to support both usage models, two different types (or bindings) of PT protocols are necessary to operate before and after the endpoint has an assigned IP address and other network layer information. This specification focuses on the PT protocol used to assess endpoints already present on the network and thus is able to use TCP/IP based transport protocols.

The PT protocol in the NEA architecture is responsible for transporting PB-TNC batches (often containing PA-TNC [RFC5792] attributes) over the network between the Posture Transport Client component of the NEA Client and the Posture Transport Server component of the NEA Server. The PT protocol also offers strong security protections to ensure the exchanged messages are protected from a variety of threats from hostile intermediaries.

### 1.1. Prerequisites

This document does not define an architecture or reference model. Instead, it defines one binding of the PT protocol that works within the reference model described in the NEA Overview and Requirements specification. The reader is assumed to be thoroughly familiar with the NEA Overview and Requirements specification. No familiarity with TCG specifications is assumed.

### 1.2. Message Diagram Conventions

This specification defines the syntax of PT-TLS messages using diagrams. Each diagram depicts the format and size of each field in

bits. Implementations MUST send the bits in each diagram as they are shown, traversing the diagram from top to bottom and then from left to right within each line (which represents a 32-bit quantity). Multi-byte fields representing numeric values must be sent in network (big endian) byte order.

Descriptions of bit field (e.g. flag) values are described referring to the position of the bit within the field. These bit positions are numbered from the most significant bit through the least significant bit so a one octet field with only bit 0 set has the value 0x80.

### 1.3. Conventions used in this document

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 [RFC2119].

## 2. Design Considerations

This section discusses some of the key design considerations for the PT protocol. This document specifies the PT binding for use when performing an assessment or reassessment after the endpoint has been admitted to the network and is capable of using TCP/IP to communicate with the NEA Server. If the endpoint does not yet have TCP/IP layer access to the NEA Server (and vice versa), the endpoint should use the PT-EAP (Posture Transport (PT) Protocol for EAP Tunnel Methods) [PT-EAP] protocol when performing an assessment.

Because the endpoint has TCP/IP access to the NEA Server (potentially on a restricted portion of the network), the NEA Client and NEA Server have the ability to establish (or re-use) a reliable TCP/IP connection in order to perform the assessment. The TCP/IP connection enables the assessment to occur over a relatively high performance, reliable channel capable of supporting multiple roundtrip message exchanges in full duplex manner. These connection properties are very different from what is available when the endpoint is initially joining the network (e.g. during an 802.1X based assessment), therefore the design described in this specification follows a different path to maximize the benefits of the underlying TCP/IP connection.

### 2.1. Benefits of TCP/IP Connectivity

The PT protocol is typically able to offer to the NEA Client and NEA Server significantly higher quality of service and flexibility of operation than link layer oriented bindings such as PT-EAP (Posture Transport (PT) Protocol for EAP Tunnel Methods). However, there may

be some added risks when the endpoint is on the network prior to its initial assessment (if no admission time assessment had been performed). Because of these risks, the combined use of an EAP-based assessment during admission followed by reassessment using TCP/IP may be appropriate in some environments.
Some of the benefits to having a TCP/IP based transport during an assessment include:

- Full Duplex connectivity - can send multiple assessment messages prior to receiving a response including sending of asynchronous messages (e.g. alerts of posture or policy changes)

- High Bandwidth - potentially much higher bandwidth than other transports (e.g. EAP) allowing more in-band data (e.g. remediation, verbose posture information)

- Large Messages - ability to send very large PA messages without directly fragmenting them (underlying carrier protocol may introduce fragmentation)

- Bi-directional - NEA Client and NEA Server can initiate an assessment or reassessment

- Multiple Roundtrips - NEA Client and NEA Server can exchange numerous messages without fear of infrastructure timeouts. However, the entire exchange should be kept as brief as possible if the user has to wait for its completion.

## 2.2. Leveraging Proven TLS Security

All PT protocol bindings must be capable of providing strong authentication, integrity and confidentiality protection for the PB-TNC batches. Rather than define a new protocol over TCP/IP to provide adequate protection, this specification requires the use of Transport Layer Security [RFC5246] to secure the connection. TLS was selected because it's a widely deployed protocol with parallel protections to a number of the EAP tunnel methods, and it meets all of the security requirements.

## 2.3. TLV-Oriented Based Message Encapsulation

The design of the PT-TLS protocol is based upon the use of type-length-value (TLV) oriented protocol message that identifies the type of message, the message's length and a potentially variable length payload value. The use of a TLV orientated encoding was chosen to match the Internet standard PA-TNC and PB-TNC protocols. Because the PA-TNC, PB-TNC and PT-TLS protocols are typically implemented inside

the same process space, this allows a common set of message parsing code to be used. Similarly creation of debugging tools is simplified by the common encoding methodologies. TLV-based encoding was used in each of the NEA protocols in part because it enables a very space efficient representation on the network and is simpler to parse than some other encodings to benefit lower powered (or battery constrained) devices.

### 2.4. No Change to Base TLS Protocol

During the design of the PT-TLS protocol, several approaches were considered with different costs and benefits. Several considered approaches involved integrating the PT protocol into the TLS handshake protocol. Because the PT protocol requires the underlying TLS carrier to provide security protections, the PT protocol couldn't operate before the cipher suites were negotiated and in use. One option was to integrate into the TLS handshake protocol after the ChangeCipherSpec phase allowing the PT message to be protected. The benefit of this approach is that the assessment protocol could operate below the application protocols allowing for easier integration into applications. However, making this change would require some extensions to the TLS handshake protocol standards and existing widely deployed TLS implementations, so it wasn't clear that the cost was warranted, particularly because the application independence can also be offered by a shim library between the application and TLS library that provides the PT protocol encapsulation/decapsulation.

The other general approach considered was to have PT-TLS layer on top of TLS as an application protocol (using the standard application_data ContentType). This has the advantage that existing TLS software could be used. However, the PB-TNC traffic would need to be encapsulated/decapsulated by a new PT-TLS protocol layer before being passed to the TLS library. This didn't seem like a significant issue as PB-TNC is architected to layer on PT anyway.

After considering the different options, it was determined that layering the PT protocol on top of the TLS protocol without requiring current TLS protocol implementations to change met all the requirements and offered the best path toward rapid adoption and deployment. Therefore the following sections describe a PT protocol that is carried on top of TLS.

## 3. PT-TLS Protocol

This section specifies the PT-TLS protocol, a Posture Transport (PT) protocol carried by the Transport Layer Security (TLS) protocol over

a TCP/IP network. As shown in Figure 1, this protocol runs directly on top of TLS as an application. This means PT-TLS is encapsulated within the TLS Record Layer protocol using the standard ContentType for applications (application_data).

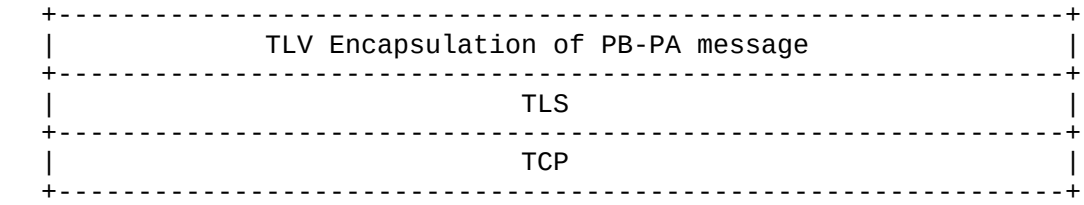



Figure 1: PT-TLS Layering Model

### 3.1. Initiating a PT-TLS Session

The PT-TLS protocol may be initiated by a Posture Transport Client or a Posture Transport Server. This flexibility supports different use cases. For example, a Posture Transport Client that wishes to trigger a NEA assessment to determine whether its security posture is good can start up a PT-TLS session and request a posture assessment. On the other hand, when an endpoint requests access to a protected network or resource, a Posture Transport Server can start up a PT-TLS session and perform a posture assessment before deciding whether to grant access.

The party that initiates a PT-TLS session is known as the "PT-TLS Initiator". The other party in the session (which receives the request to open a PT-TLS session) is known as the "PT-TLS session responder".

#### 3.1.1. Issues with Server Initiated PT-TLS Sessions

In order for a NEA Server to establish a PT-TLS session, the NEA Client needs to be listening for a connection request on a TCP port known by the NEA Server. In many deployments, the security policies (e.g. firewall software) of an endpoint are designed to minimize the number of open inbound TCP/UDP ports that are available to the network to reduce the potential attack footprint. This is one issue that makes it difficult for a NEA Server to initiate a PT-TLS session.

Another issue with this scenario involves X.509 certificates. When the NEA Server creates a TLS session to the NEA Client, the NEA Client is effectively acting as the TLS server during the TLS protocol exchange. This means the NEA Client would typically need to possess an X.509 certificate to protect the initial portion of the

TLS handshake. In situations where the NEA Server initiates the creation of the TLS session, both the NEA Client and NEA Server MUST possess X.509 certificates to fully authenticate the session. For many deployments, provisioning X.509 certificates to all NEA Clients has scalability and cost issues; therefore, it is recommended that the NEA Client not listen for connection requests from the NEA Server but instead establish and maintain a TLS session to the NEA Server proactively, so either party can initiate an assessment using the preexisting TLS session as required.
Therefore, NEA Clients SHOULD be capable of establishing and holding open a TLS session with the NEA Server immediately after obtaining network access. A NEA Client MAY listen for connection requests from the NEA Server and establish a new PT-TLS session when one does not already exist. Having an existing PT-TLS session allows either party to initiate an assessment without requiring the NEA Client to be listening for new connection requests.

#### 3.1.2. Establish or Re-Use Existing PT-TLS Session

A single PT-TLS session can support multiple NEA assessments, which can be started by either party (the PT-TLS Initiator or the PT-TLS Responder). The party that starts a NEA assessment is known as the "assessment initiator" and the other party is known as the "assessment responder".

If the assessment initiator already has a PT-TLS session to the assessment responder, the initiator can re-use this session; otherwise, a new PT-TLS session must be established.

### 3.2. TCP Port Usage

In order for a PT-TLS Initiator to establish a TCP connection to a PT-TLS Responder, the initiator needs to know the TCP port number on which the responder is listening for assessment requests. Therefore, this specification requests the IANA reserve a TCP port number for use with the PT-TLS protocol upon publication of this specification as an Internet standard RFC.

### 3.3. Preventing MITM Attacks with Channel Bindings

As described in the NEA Asokan Attack Analysis [ASOKAN], a sophisticated MITM attack can be mounted against NEA systems. The attacker forwards PA-TNC messages from a healthy machine through an unhealthy one so that the unhealthy machine can gain network access. Because there are easier attacks on NEA systems, like having the unhealthy machine lie about its configuration, this attack is generally only mounted against machines with an External Measurement

Agent (EMA). The EMA is a separate entity, difficult to compromise, which measures and attests to the configuration of the endpoint.

To protect against NEA Asokan attacks, the Posture Broker on an EMA-equipped endpoint should pass the tls-unique channel binding [RFC5929] for PT-TLS's underlying TLS session to the EMA. This value can then be included in the EMA's attestation and the Posture Validator responsible for communicating with the EMA may then confirm that the value matches the tls-unique channel binding for its end of the connection. If the values match, the posture sent by the EMA and NEA Client is from the same endpoint as the client side of the TLS connection (since the endpoint knows the tls-unique value), so no man-in-the-middle is forwarding posture. If they differ, an attack has been detected. The Posture Validator MUST fail its verification of the endpoint if an attack has been detected.

### 3.4. PT-TLS Message Flow

This section discusses the general flow of messages between the NEA Client's Posture Transport Client and the NEA Server's Posture Transport Server in order to perform NEA assessments using the PT-TLS protocol.

#### 3.4.1. Assessment Triggers

Initially, the NEA Client or NEA Server will decide that an assessment is needed. What stimulates the decision to perform an assessment is outside the scope of this specification, but some examples include:

- NEA Server becoming aware of suspicious behavior on an endpoint
- NEA Server receiving new policies requiring immediate action
- NEA Client noticing a change in local security posture
- NEA Client wishing to access a protected network or resource

Because either the NEA Client or NEA Server can trigger the establishment of the TLS session and initiate the assessment, this document will use the terms "assessment initiator" and the "assessment responder". This nomenclature allows either NEA component to fill either of the PT-TLS roles.

#### 3.4.2. PT-TLS Message Exchange Phases

The PT-TLS message exchange occurs in three distinct phases:

- TLS Setup (including TLS Handshake protocol)
- PT-TLS Negotiation
- PT-TLS Data Transport

The TLS Setup phase is responsible for the establishment of the TCP connection and the TLS protections for the PT-TLS messages. The TLS Setup phase normally starts with the establishment of a TCP connection between the Posture Transport Client and Posture Transport Server. The new connection triggers the TLS Handshake protocol to establish the cryptographic protections for the TLS session. The TLS Setup phase SHOULD NOT be repeated after the PT-TLS Data Transport phase has been reached unless a change of TLS cipher suite or keying material is required to properly protect the session. This phase also enables the establishment of the tls-unique shared secret that can be used in a later phase to bind the posture sent with this TLS connection.

The PT-TLS Negotiation phase is only performed at the start of the first assessment on a TLS session. During this phase, the NEA Client and NEA Server discover each other's PT-TLS capabilities and establish a context that will apply to all future PT-TLS messages sent over the TLS session. The PT-TLS Negotiation phase MUST NOT be repeated after the session has entered the Data Transport phase. NEA assessment messages (PB-TNC batches) MUST NOT be sent by the NEA Client or NEA Server prior to the completion of the PT-TLS Negotiation phase to ensure that the security protections for the session are properly established and applied to the NEA assessment messages.

Finally the Data Transport phase allows the NEA Client and NEA Server to exchange PT messages under the protection of the TLS session consistent with the capabilities established in earlier phases. The exchanged messages can be a PT-TLS protected NEA assessment as described in this specification or other vendor-defined PT-TLS exchanged messages.

#### 3.4.2.1. TLS Setup Phase

After a new TCP connection is established between the Posture Transport Client and Posture Transport Server, a standard TLS exchange is performed to negotiate a common security context for protecting subsequent communications. As discussed in section 3.4.1. , the TCP connection establishment and/or the TLS handshake protocol could be initiated by either the NEA Client or NEA Server. The most common situation would be for the assessment initiator to trigger the creation of the TCP connection and TLS handshake, so an assessment

could begin when no session already exists. When the NEA Server has initiated the TLS Setup, the NEA Server is acting as a TLS client and the NEA Client is the TLS server (accepting the inbound TLS session request). The expected normal case is that the NEA Client initiates this phase, so that the NEA Server is acting as the TLS server and therefore the bootstrapping of the security of the TLS session is using the NEA Server's certificate. Having the NEA Client initiate the TLS session avoids the need for the NEA Client to also possess a certificate.
During the TLS Setup phase of PT-TLS, the PT-TLS Initiator contacts the listening port of the PT-TLS Responder and performs a TLS handshake. The PT-TLS Responder MUST possess a trustworthy X.509 certificate used to authenticate to the TLS initiator and used to bootstrap the security protections of the TLS session. The PT-TLS Initiator MAY also use an X.509 certificate to authenticate to the PT-TLS Responder providing for a bi-directional authentication of the PT-TLS session.
Due to deployment issues with issuing and distributing certificates to a potentially large number of NEA Clients, this specification allows the NEA Client to be authenticated during the PT-TLS Negotiation phase using other more cost effective methods. At the conclusion of a successful initial TLS Setup phase, the NEA Client and NEA Server have a protected session to exchange messages. This allows the protocol to transition to the PT-TLS Negotiation phase.

#### 3.4.2.2. PT-TLS Negotiation Phase

Once a TLS session has been established between Posture Transport Client and Posture Transport Server, the PT-TLS Initiator sends a Version Request Message indicating its supported PT-TLS protocol version range. Next, the PT-TLS Responder sends a Version Response Message which selects a protocol version from within the range offered. The PT-TLS Responder SHOULD select the preferred version offered if supported; otherwise, the highest version that the responder is able to support from the received Version Request Message. If the PT-TLS Responder is unable or unwilling to support any of the versions included in the Version Request Message, the responder SHOULD send a Version Not Supported error message.

If no client side authentication occurred during the TLS Setup phase, the Posture Transport Client can authenticate the client using PT-TLS entity authentication messages. If the Posture Transport Client wishes to trigger an authentication of the client, the Posture Transport Client SHOULD send a Request SASL Mechanisms and/or SASL Mechanism Selection message (see section 3.8.3. for details). The Posture Transport Server MAY also chose to send a SASL Mechanisms

message without receiving a Request SASL Mechanisms to kick start the exchange.

When the Posture Transport Client receives the SASL Mechanisms list, the Posture Transport Client responds with a SASL Mechanism Selection indicating the method of authentication to be used. Upon selecting an appropriate SASL mechanism, the Posture Transport Client and Server exchange SASL mechanism specific messages in order to authenticate the client identity on the PT-TLS Initiator. When the client authentication successfully completes and no additional authentications are required, the PT-TLS session transitions into the Data Transport phase, where it will remain for the duration of the session. Note that the PT-TLS Responder could choose to not authenticate the client or to continue performing a posture assessment even if the authentication did not complete successfully.

#### 3.4.2.3. PT-TLS Data Transport Phase

Once a PT-TLS session is available to carry NEA assessments, either the Posture Transport Client or Server can start an assessment when provided a PB-TNC batch for transmission. The assessment initiator first envelopes the PB-TNC batch in a PT-TLS message, then assigns a message identifier to the message and finally transmits it over the session. The assessment responder validates the PT-TLS message and delivers the encapsulated PB-TNC batch to its upstream component (Posture Broker Client or Server).

Most PT-TLS messages contain PB-TNC batches that house PA-TNC requests for posture information or a response containing the requested posture information. The Posture Transport Client and Posture Transport Server may also exchange messages between them, such as a PT-TLS Error Message indicating that a problem occurred processing a message. During an assessment, the Posture Transport Client and Server merely encapsulate and exchange the PB-TNC batches and are unaware of the state of the assessment.

The PT-TLS protocol allows either party to send a PT-TLS message at any time, reflecting the full duplex nature of the underlying TLS session. For example, an assessment initiator may send several PT-TLS messages prior to receiving any responses from the assessment responder. All implementations of PT-TLS MUST support full duplex PT-TLS message exchange. However, some NEA protocols may not be able to make use of the full-duplex message exchange.

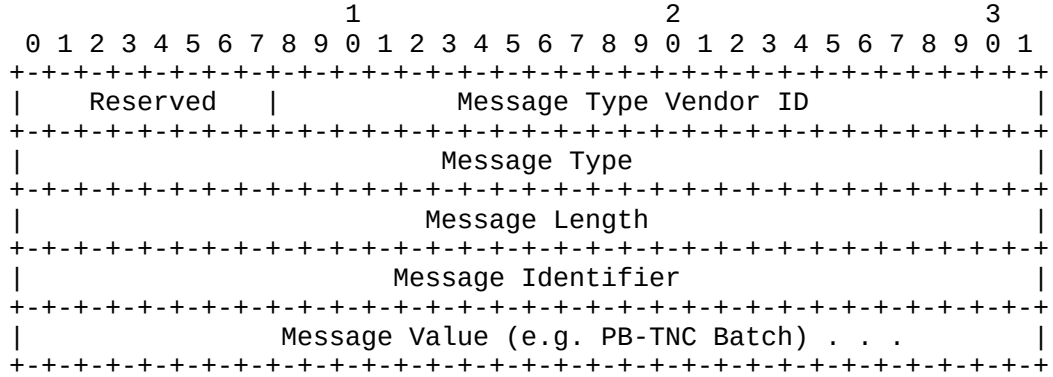
### 3.4.3. TLS Requirements

In order to ensure that strong security is always available for deployers and to improve interoperability, this section discusses some requirements on the underlying TLS transport used by PT-TLS.

Implementations of PT-TLS MUST support use of TLS 1.1 [RFC4346] and SHOULD also include support for TLS 1.2 [RFC5246]. For each TLS version supported, implementations of the PT-TLS MUST at least support the TLS_RSA_WITH_AES_128_CBC_SHA cipher suite. This cipher suite requires the server to provide a certificate that can be used during the key exchange. Implementations SHOULD NOT include support for cipher suites that do not minimally offer PT-TLS Responder (typically Posture Transport Server) authentication, such as the anonymous Diffie-Hellman cipher suites (e.g. TLS_DH_anon_WITH_AES_128_CBC_SHA).

### 3.5. PT-TLS Message Format

This section describes the format and semantics of the PT-TLS message. Every message sent over a PT-TLS session MUST start with the PT-TLS header described in this section.
The following is the PT-TLS header:




Reserved

Reserved for future use. This field MUST be set to 0 on transmission and ignored upon reception.

Message Type Vendor ID

This field indicates the owner of the name space associated with the Message Type. This is accomplished by specifying the 24 bit SMI Private Enterprise Number (Vendor ID) of the party who owns the Message Type name space. IETF Standard PT-TLS Message Types MUST use zero (0) in this field.

The PT-TLS Message Type Vendor ID 0xffffff is reserved. Posture Transport Clients and Servers MUST NOT send PT-TLS messages in which the PT-TLS Message Type Vendor ID has this reserved value (0xffffff). If a Posture Transport Client or Posture Transport Server receives a message containing this reserved value (0xffffff) in the PT-TLS Message Type Vendor ID, the recipient SHOULD respond with an Invalid Parameter error code in a PT-TLS Error message.

Message Type

This field defines the type of the PT-TLS message within the scope of the specified Message Type Vendor ID that is included in the Message Value field. The specific IETF standard values allowable in this field when the Message Type Vendor ID is the IETF SMI Private Enterprise Number value (0) are defined in section 3.6. Recipients of a message containing a Message Type Vendor ID and Message Type that is unrecognized SHOULD respond with a Type Not Supported error code in a PT-TLS Error message.

Posture Transport Clients and Posture Transport Servers MUST NOT require support for particular vendor-defined PT-TLS Message Types and MUST interoperate with other parties despite any differences in the set of vendor-defined PT-TLS Message Types supported (although they MAY permit administrators to configure them to require support for specific vendor-defined PT-TLS message types).

If the PT-TLS Message Type Vendor ID field has the value zero (0), then the PT-TLS Message Type field contains an IETF Standard PT-TLS Message Type, as listed in the IANA registry. IANA maintains a registry of PT-TLS Message Types. Entries in this registry are added by Expert Review with Specification Required, following the guidelines in section 6.1. Section 3.6. of this specification defines the initial set of IETF Standard PT-TLS Message Types.

The PT-TLS Message Type 0xffffffff is reserved. Posture Transport Clients and Posture Transport Servers MUST NOT send PT-TLS messages in which the PT-TLS Message Type has this reserved value (0xffffffff). If a Posture Transport Client or Posture Transport Server receives a message in which the PT-TLS Message Type has this reserved value (0xffffffff), it SHOULD respond with an Invalid Parameter error code in a PT-TLS Error message.

Message Length

This field contains the length in octets of the entire PT-TLS message (including the entire header). Therefore, this value MUST always be at least 16. Any Posture Transport Client or Posture Transport Server that receives a message with a PT-TLS Message Length field whose value is less than 16 SHOULD respond with an Invalid Parameter PT-TLS error code. Similarly, if a Posture Transport Client or Posture Transport Server receives a PT-TLS message for a Message Type that has a known Message Length and the Message Length indicates a different value (greater or less than the expected value), the recipient SHOULD respond with an Invalid Parameter PT-TLS error code.

Message Identifier

This field contains a value that uniquely identifies the PT-TLS message on a per message sender (Posture Transport Client or Server) basis. This value can be copied into the body of a response message to indicate which message was received and caused the response. For example, this field is included in the PT-TLS Error Message so the recipient can determine which message sent caused the error.

The Message Identifier MUST be a monotonically increasing counter starting at zero indicating the number of the messages the sender has transmitted over the TLS session. It is possible that a busy or long lived session might exceed 2^32-1 messages sent, so the message sender MUST roll over to zero upon reaching the 2^32nd message, thus restarting the increasing counter. During a rollover, it is feasible that the message recipient could be confused if it keeps track of every previously received Message Identifier, so recipients MUST be able to handle roll over situations without generating errors.

Message Value

The contents of this field vary depending on the particular Message Type Vendor ID and Message Type given in the PT-TLS header for this PT-TLS message. This field most frequently contains a PB-TNC batch. The contents of this field for each of the IETF Standard PT-TLS Message Types are defined in this specification.

## 3.6. IETF Standard PT-TLS Message Types

This section defines the NEA standard PT-TLS Message Types used to carry PT-TLS messages and PB-TNC batches between the Posture Transport Client and Posture Transport Server.

The following table summarizes the initial set of IETF standard message type values, which are used with the PT-TLS Message Type Vendor ID field set to the IETF SMI PEN (0).

| Value (Name) | Definition |
|---|---|
| 0 (Experimental) | Reserved for experimental use. This type will not offer interoperability but allows for experimentation. This message type MUST only be sent when the NEA Client and NEA Server are in the Data Transport phase and only on a restricted, experimental network. Production code MUST send an Invalid Message error code in a PT-TLS Error message if an Experimental message is received. |
| 1 (Version Request) | Version negotiation request including the range of versions supported by the sender. This message type MUST only be sent by the TLS session initiator as the first PT-TLS message in the PT-TLS Negotiation phase. Recipients MUST send an Invalid Message error code in a PT-TLS Error message if a Version Request is received at another time. |
| 2 (Version Response) | PT-TLS protocol version selected by the responder. This message type MUST only be sent by the TLS session responder as the second message in the PT-TLS Negotiation phase. Recipients MUST send an Invalid Message error code in a PT-TLS Error message if a Version Response is received at another time. |
| 3 (Request SASL Mechanisms) | Sent by the PT-TLS Initiator to retrieve the set of SASL mechanisms supported by the PT-TLS Responder. This message type MUST only be sent by the PT-TLS Initiator in the PT-TLS Negotiation phase. Recipients MUST send an Invalid Message error code in a PT-TLS Error message if a Request |

SASL Mechanisms message is received at another time.

4 (SASL Mechanisms) Sent by the PT-TLS Responder to indicate what SASL mechanisms it is willing to use for authentication on this session. Recipients MUST send an Invalid Message error code in a PT-TLS Error message if a SASL Mechanisms message is received at another time.

5 (SASL Mechanism Selection) Sent by the PT-TLS Initiator to select a SASL mechanism from the list offered by the PT-TLS Responder or optimistically (see 3.8.9. for details). This message type MUST only be sent by the PT-TLS Initiator in the PT-TLS Negotiation phase. Recipients MUST send an Invalid Message error code in a PT-TLS Error message if a SASL Mechanism Selection is received after the PT-TLS Negotiation phase. Once a SASL mechanism has been selected, it may not change until the mechanism completes either successfully or as a failure.

6 (SASL Authentication Data) Opaque octets exchanged between the PT-TLS Initiator and Responder's SASL mechanisms to perform the client authentication. This message type MUST only be sent during the PT-TLS Negotiation phase. Recipients MUST send an Invalid Message error code in a PT-TLS Error message if a SASL Authentication Data message is received after the PT-TLS Negotiation phase.

7 (SASL Result) Indicates the result code of the SASL mechanism authentication. A success result indicates that the PT-TLS Initiator and Responder may transition to the Data Transport phase, thus allowing the assessment to start. Note that the PT-TLS Responder may choose to allow the transition to Data

| | |
|---|---|
| | Transport phase even if authentication is unsuccessful before making its access control decision. This message type MUST only be sent by the PT-TLS Responder when the NEA Client and NEA Server are in the PT-TLS Negotiation phase. Recipients MUST send an Invalid Message error code in a PT-TLS Error message if a SASL Result is received after the PT-TLS Negotiation phase. |
| 8 (PB-TNC Batch) | Contains a PB-TNC batch. For more information on PB-TNC batches see section 4 of the PB-TNC specification. This message type MUST only be sent when the NEA Client and NEA Server are in the PT-TLS Data Transport phase. Recipients SHOULD send an Invalid Message error code in a PT-TLS Error message if a PB-TNC Batch is received outside of the Data Transport phase. |
| 9 (PT-TLS Error) | PT-TLS Error message as described in section 3.9. This message type may be used during any PT-TLS phase. |
| 10+ (Reserved) | These values are reserved for future allocation following guidelines defined in the IANA Considerations section 6.1. Recipients of messages of type 13 or higher that do not support the PT-TLS Message Type Vendor ID and PT-TLS Message Type of a received PT-TLS message MUST respond with a Type Not Supported PT-TLS error code in a PT-TLS Error message. |

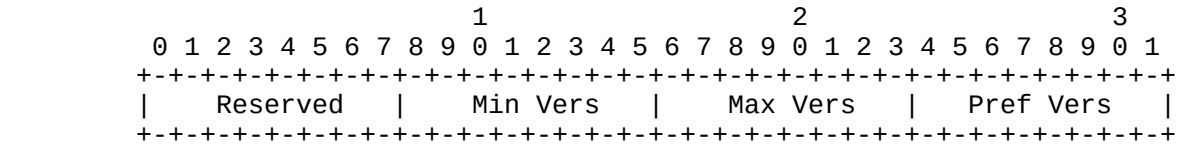
### 3.7. PT-TLS Version Negotiation

This section describes the message format and semantics for the PT-TLS protocol version negotiation. This exchange is used by the PT-TLS Initiator to trigger a version negotiation at the start of an assessment. The PT-TLS Initiator MUST send a Version Request message as its first PT-TLS message and MUST NOT send any other PT-TLS messages on this connection until it receives a Version Response

message or an Error message. The PT-TLS Responder MUST complete the version negotiation (or cause an error) prior to sending or accepting reception of any additional messages. After the successful completion of the version negotiation, both the Posture Transport Client and Posture Transport Server MUST only send messages compliant with the negotiated protocol version. Subsequent assessments on the same session MUST use the negotiated version number and therefore SHOULD NOT send additional version negotiation messages.

### 3.7.1. Version Request Message

This message is sent by a PT-TLS Initiator as the first PT-TLS message in a PT-TLS session. This message discloses the sender's supported versions of the PT-TLS protocol. To ensure compatibility, this message MUST always be sent using version 1 of the PT-TLS protocol. Recipients of this message MUST respond with a Version Response, or a PT-TLS Error message (Version Not Supported or Invalid Message). The following diagram shows the format of the Version Request Message:




Reserved

Reserved for future use. This field MUST be set to 0 on transmission and ignored upon reception.

Min Vers

This field contains the minimum version of the PT-TLS protocol supported by the sender. This field MUST be set to 1 indicating support for the first version of PT-TLS. However, future versions of this specification will probably remove this requirement so PT-TLS Responders MUST be prepared to receive other values.

Max Vers

This field contains the maximum version of the PT-TLS protocol supported by the sender. This field MUST be set to 1 indicating support for the first version of PT-TLS. However, future versions of this specification will probably

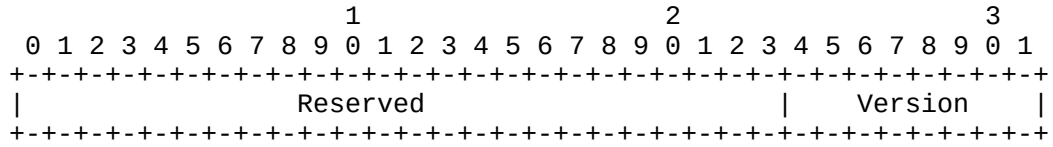
remove this requirement so PT-TLS Responders MUST be prepared to receive other values.

Pref Vers

This field contains the sender's preferred version of the PT-TLS protocol. This is a hint to the recipient that the sender would like this version selected if supported. The value of this field MUST fall within the range of Min Vers to Max Vers. This field MUST be set to 1 indicating support for the first version of PT-TLS. However, future versions of this specification will probably remove this requirement so PT-TLS Responders MUST be prepared to receive other values.

### 3.7.2. Version Response Message

This message is sent in response to receiving a Version Request Message at the start of a new assessment session. If a recipient receives a Version Request after a successful version negotiation has occurred on the session, the recipient SHOULD send an Invalid Message error code in a PT-TLS Error message and have TLS close the session. This message MUST be sent using the syntax, semantics, and requirements of the protocol version specified in this message.




Reserved

Reserved for future use. This field MUST be set to 0 on transmission and ignored upon reception.

Version

This field contains the version selected by the sender of this message. The version selected MUST be within the Min Vers to Max Vers inclusive range sent in the Version Request Message. If a PT-TLS Initiator receives a message with an invalid Version selected, the PT-TLS Initiator MUST respond with a Version Not Supported PT-TLS error message.

### 3.8. Entity Authentication using SASL

This section includes a description of the message format and semantics necessary to perform client authentication (authentication of the PT-TLS Initiator) over PT-TLS. The general model used for providing an authentication using PT-TLS messages over TLS is to integrate the Simple Authentication and Security Layer (SASL) [RFC4422] framework. SASL provides a number of standards-based authentication mechanisms capable of authenticating the PT-TLS Initiator using a variety of base technologies.

Because either the NEA Client or NEA Server can initiate the TLS session used for the assessment, either could act as the TLS server and be authenticated as part of the TLS exchange. Therefore, either the NEA Client or NEA Server could also be the party not authenticated during the TLS handshake (assuming that TLS mutual authentication is not used) and be required to authenticate using the PT-TLS entity authentication. Therefore we call this an entity authentication since the entity being authenticated could be either the NEA Client or NEA Server. Note that normally the NEA Client is being authenticated.

#### 3.8.1. SASL Entity Authentication Requirements

Implementations compliant with the PT-TLS specification MUST implement the SASL authentication messages described in this section. In order to ensure interoperability, all PT-TLS implementations compliant with this specification MUST at least support the PLAIN SASL mechanism [RFC4616]. Similarly, implementations MUST provide the EXTERNAL SASL mechanism if both parties are authenticated during the TLS establishment.

#### 3.8.2. SASL in PT-TLS Overview

Mechanism negotiation is performed using the Request SASL Mechanisms and SASL Mechanisms TLVs. The PT-TLS Initiator sends a Request SASL Mechanisms TLV to request a preference-ordered list of SASL mechanisms that the PT-TLS Responder is willing to use for this session. The PT-TLS Initiator selects one SASL mechanism from the list and sends a SASL Mechanism Selection TLV completing the negotiation. Subsequent challenges and responses are carried within the SASL Authentication Data TLV carrying the authentication data for the selected mechanism. The authentication outcome is communicated in a SASL Result TLV containing a status code.

### 3.8.3. Initiating the SASL Authentication

Typically the NEA Client would setup the PT-TLS session (see section 3.1. ), so the NEA Client would be considered the PT-TLS Initiator and thus is the party being authenticated. The PT-TLS Initiator may initiate the entity authentication by sending a Request SASL Mechanisms TLV to retrieve the mechanism set offered by the PT-TLS Responder (NEA Server) or wait for the PT-TLS Responder to send the mechanism set if authentication is required by its policy. The PT-TLS Initiator may also proactively send a SASL Mechanism Selection TLV proposing a SASL mechanism if it believes is acceptable by the PT-TLS Responder without requesting the supported set. If the NEA Server receives a SASL Mechanism Selection TLV that contains an unacceptable SASL mechanism, the NEA Server would respond with a SASL Mechanism Error in a PT-TLS Error TLV.

In situations where the PT-TLS Responder does not require an entity authentication (either authentication isn’t necessary or was performed during the TLS Setup phase), the PT-TLS Responder would send a SASL Mechanisms TLV with no mechanisms included (only the PT-TLS header) when requested.

In order to avoid race conditions caused by the PT-TLS Responder sending an unrequested SASL Mechanisms TLV, the PT-TLS Initiator MUST be able to ignore unsolicited SASL Mechanisms TLV received immediately after sending a SASL Mechanism Selection TLV or the reception of two SASL Mechanisms TLVs immediately after sending one SASL Mechanism Request.

### 3.8.4. SASL Authentication Flow

If an authentication is required by the PT-TLS Responder and not requested by the PT-TLS Initiator, the PT-TLS Responder (typically the NEA Server) SHOULD initiate the entity authentication exchange by sending a SASL Mechanisms TLV. If the PT-TLS Responder receives a NEA assessment message before the completion of any required entity authentication, the PT-TLS Responder MUST send an Authentication Required PT-TLS Error indicating to the PT-TLS Initiator that an authentication exchange is required prior to entering the PT-TLS Data Transport phase. The SASL Mechanisms TLV includes a preference-ordered list of the SASL mechanisms that the PT-TLS Responder is willing to use and allows for the selection of one by the PT-TLS Initiator for use with this session.

#### 3.8.5. Aborting SASL Authentication

The initiator may abort the authentication exchange by sending the SASL Result TLV with a status code of ABORT.

#### 3.8.6. Linkages to SASL Framework

##### 3.8.6.1. SASL Service Name

The service name for PT-TLS is "nea-pt-tls".

##### 3.8.6.2. SASL Authorization Identity

The nea-pt-tls protocol does not make use of a SASL authorization identity string as described in RFC4422.

##### 3.8.6.3. SASL Security Layer

The NEA PT-TLS protocol always runs under the protection of TLS. SASL security layers are not used.

##### 3.8.6.4. Multiple Authentications

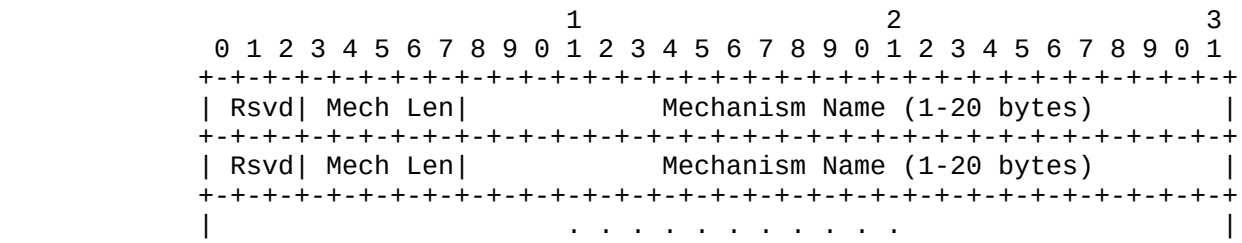
Only one SASL mechanism authentication may be in progress at any one time. Once a SASL mechanism completes (successfully or unsuccessfully) a new authentication may be initiated.

#### 3.8.7. Request SASL Mechanisms

This TLV is sent by the PT-TLS Initiator (typically the NEA Client) to request that the PT-TLS Responder return the list of SASL mechanisms that it is willing and able to use to authenticate. This TLV has no payload (Value field) since the type itself indicates the purpose of the TLV to the recipient. When the PT-TLS Responder receives this TLV, it will respond with the SASL Mechanisms TLV.

#### 3.8.8. SASL Mechanisms

This TLV is sent by the PT-TLS Responder (typically NEA Server) to indicate the list of SASL mechanisms that it is willing and able to use to authenticate the PT-TLS Initiator. Each mechanism name consists of a length followed by a name. The total length of the list is determined by the TLV Length field. If the PT-TLS Responder receives a Request SASL Mechanisms TLV and the PT-TLS Responder does not require an authentication to occur, it MUST respond with a SASL Mechanisms TLV containing no mechanisms (TLV only includes the PT-TLS header).




Rsvd (Reserved)

Reserved for future use. This field MUST be set to 0 on transmission and ignored upon reception.

Mech Len (Mechanism Name Length)

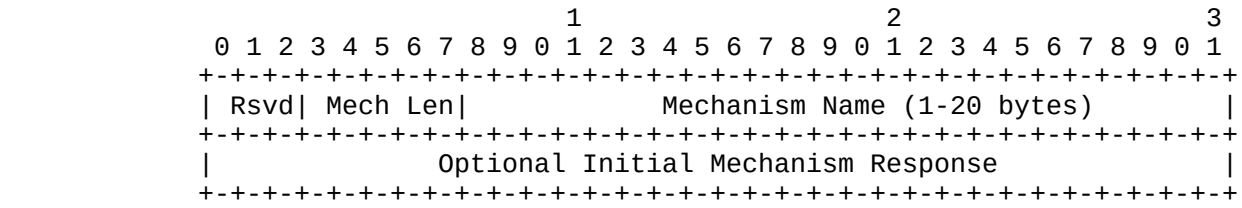
The length of the Mechanism-Name field in octets.

Mechanism Name

SASL mechanism name adhering to the rules defined in RFC4422.

### 3.8.9. SASL Mechanism Selection

This TLV is sent by the PT-TLS Initiator (typically the NEA Client) in order to select a SASL mechanism for use on this session. Because the PT-TLS Initiator sends first for the SASL negotiation, it may start with Request Mechanisms TLV or just optimistically send a SASL Mechanism Selection TLV in hopes it is acceptable to the PT-TLS Responder (to avoid a roundtrip).




Rsvd (Reserved)

Reserved for future use. This field MUST be set to 0 on transmission and ignored upon reception.

Mech Len (Mechanism Name Length)

The length of the Mechanism-Name field in octets.

Mechanism Name

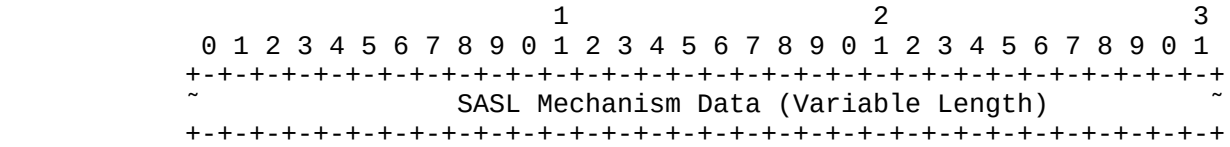
SASL mechanism name adhering to the rules defined in RFC4422.

Optional Initial Mechanism Response

Initial set of authentication information required from the PT-TLS Initiator to kick start the authentication. This data is optional and if not provided would be solicited by the PT-TLS Responder in the first SASL Authentication Data TLV request.

### 3.8.10. SASL Authentication Data

This TLV carries an opaque (to PT-TLS) blob of octets being exchanged between the PT-TLS Initiator and the PT-TLS Responder. This TLV facilitates their communications without interpreting any of the bytes. The SASL Authentication Data TLV MUST NOT be sent until a SASL mechanism has been established for a session. The SASL Authentication Data TLV MUST NOT be sent after a SASL Result is sent with a Successful status.



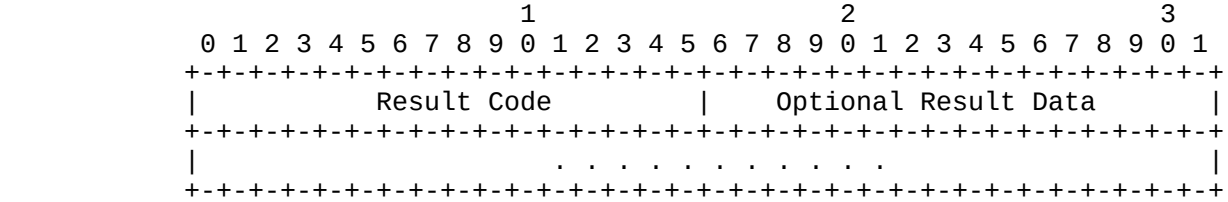

SASL Mechanism Data

Opaque, variable length set of bytes exchanged between the PT-TLS Initiator's SASL mechanism and its peer PT-TLS Responder's SASL mechanism. These bytes MUST NOT be interpreted by the PT-TLS layer.

### 3.8.11. SASL Result

This TLV is sent by the PT-TLS Responder at the conclusion of the SASL exchange to indicate the authentication result. This TLV may also be sent by the PT-TLS Initiator to indicate a client side failure. The PT-TLS Initiator MUST NOT send other Result-Code values besides Failure. Upon reception of a SASL Result TLV indicating an

Abort, the recipient MUST terminate the current authentication conversation.




Result Code

This field contains the result of the SASL authentication exchange.

| Value (Name) | Definition |
|---|---|
| 0 (Success) | SASL authentication was successful and identity was confirmed. |
| 1 (Abort) | SASL authentication exchange was aborted by the sender |
| 2 (Mechanism Failure) | SASL mechanism failure |

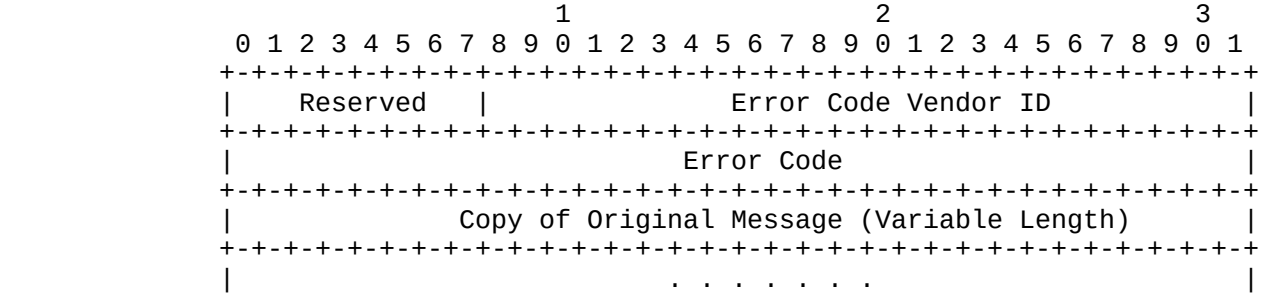
Optional Result Data

This field contains a variable length set of additional data for a successful result. This field MUST be zero length unless the PT-TLS Responder is returning a Result Code of Success and has more data to return. For more information on the additional data, see RFC 4422.

## 3.9. Error Message

This section describes the format and contents of the PT-TLS Error Message sent by the NEA Client or NEA Server when it detects a PT-TLS level protocol error. Each error message contains an error code indicating the error that occurred, followed by a copy of the message that caused the error.
When a PT-TLS error is received, the recipient MUST NOT respond with a PT-TLS error because this could result in an infinite loop of error messages being sent. Instead, the recipient MAY log the error, modify its behavior to avoid future errors, ignore the error, terminate the assessment, or take other action as appropriate (as

long as it is consistent with the requirements of this specification).
The Message Value portion of a PT-TLS Error Message contains the following information:




Reserved

Reserved for future use. This field MUST be set to 0 on transmission and ignored upon reception.

Error Code Vendor ID

This field contains the IANA assigned SMI Private Enterprise Number for the vendor whose Error Code name space is being used in the message. For IETF standard Error Code values this field MUST be set to zero (0). For other vendor-defined Error Code name spaces this field MUST be set to the SMI Private Enterprise Number of the vendor.

Error Code

This field contains the error code. This error code exists within the scope of Error Code Vendor ID in this message. Posture Transport Clients and Posture Transport Servers MUST NOT require support for particular vendor-specific PT-TLS Error Codes and MUST interoperate with other parties despite any differences in the set of vendor-specific PT-TLS Error Codes supported (although they MAY permit administrators to configure them to require support for specific PT-TLS error codes).

When the Error Code Vendor ID is set to the IETF Private Enterprise Number, the following table lists the supported IETF standard numeric error codes:

| Value (Name) | Definition |
|---|---|
| 0 (Reserved) | Reserved value indicates that the PT-TLS Error Message SHOULD be ignored by all recipients. This MAY be used for debugging purposes to allow a sender to see a copy of the message that was received while a receiver is operating on its contents. |
| 1 (Malformed Message) | PT-TLS message unrecognized or unsupported. This error code SHOULD be sent when the basic message content sanity test fails. The sender of this error code MUST consider it a fatal error and abort the assessment. |
| 2 (Version Not Supported) | This error SHOULD be sent when a PT-TLS Responder receives a PT-TLS Version Request message containing a range of version numbers that doesn't include any version numbers that the recipient is willing and able to support on the session. All PT-TLS messages carrying the Version Not Supported error code MUST use a Version number of 1. All parties that receive or send PT-TLS messages MUST be able to properly process an error message that meets this description, even if they cannot process any other aspect of PT-TLS version 1. The sender and receiver of this error code MUST consider this a fatal error and close the TLS session after sending or receiving this PT-TLS message. |
| 3 (Type Not Supported) | PT-TLS message type unknown or not supported. When a recipient receives a PT-TLS message type that it does not support, it MUST send back this error, ignore the message and proceed. For example, this could occur if the sender used a Vendor ID for the Message Type that is not supported by the recipient. This error message |

| | |
|---|---|
| | does not indicate a fatal error has occurred, so the assessment is allowed to continue. |
| 4 (Failed Authentication) | The authentication of the identity of the client failed. This could occur if the SASL mechanism was unable to authenticate the claimed identity of the PT-TLS Initiator. This error message does not indicate a fatal error has occurred, so the authentication is allowed to be re-started. |
| 5 (Invalid Message) | PT-TLS message received was invalid based on the protocol state. For example, this error would be sent if a recipient receives a message associated with the PT-TLS Negotiation Phase (such as Version messages) after the protocol has reached the PT-TLS Data Transport Phase. The sender and receiver of this error code MUST consider it a fatal error and close the TLS session after sending or receiving this PT-TLS message. |
| 6 (SASL Mechanism Error) | A fatal error occurred while trying to perform the entity authentication. For example, the NEA Client is unable to support any of the offered SASL mechanisms. The sender and receiver of this error code MUST consider it a fatal error and close the TLS session after sending or receiving this PT-TLS message. |
| 6 (Authentication Needed) | The PT-TLS Responder has received a NEA assessment message before the completion of the entity authentication. This could occur if the NEA Client initiated the PT-TLS session and then started sending PB messages before a required (by NEA Server policy) entity authentication was performed. When the PT-TLS Initiator (typically NEA Client) |

receives this error, it should initiate an entity authentication as discussed in 3.8.3.

Copy of Original Message

This variable length value contains a copy (up to 1024 bytes) of the original PT-TLS message that caused the error. If the original message is longer than 1024 bytes, only the initial 1024 bytes will be included in this field. This field is included so the error recipient can determine which message sent caused the error. In particular, the recipient can use the Message Identifier field from the Copy of Original Message to determine which message caused the error.

# 4. Security Considerations

This section discusses the major threats potentially faced by each binding of the PT protocol and countermeasures provided by the PT-TLS protocol.

## 4.1. Trust Relationships

In order to understand where security countermeasures are necessary, this section starts with a discussion of where the NEA architecture envisions some trust relationships between the processing elements of the PT-TLS protocol. The following sub-sections discuss the trust properties associated with each portion of the NEA reference model directly involved with the processing of the PT-TLS protocol.

### 4.1.1. Posture Transport Client

The Posture Transport Client is trusted by the Posture Broker Client to:

- Not observe, fabricate or alter the contents of the PB-TNC batches received from the network
- Not observe, fabricate or alter the PB-TNC batches passed down from the Posture Broker Client for transmission on the network
- Transmit on the network any PB-TNC batches passed down from the Posture Broker Client

- Deliver properly security protected messages received from the network that are destined for the Posture Broker Client
- Provide configured security protections (e.g. authentication, integrity and confidentiality) for the Posture Broker Client's PB-TNC batches sent on the network
- Expose the authenticated identity of the Posture Transport Server
- Verify the security protections placed upon messages received from the network to ensure the messages are authentic and protected from attacks on the network
- Provide a secure, reliable, in order delivery, full duplex transport for the Posture Broker Client's messages

The Posture Transport Client is trusted by the Posture Transport Server to:

- Not send malicious traffic intending to harm (e.g. denial of service) the Posture Transport Server
- Not send malformed messages (e.g. messages lacking PT-TLS header)
- Not send invalid or incorrect responses to messages (e.g. errors when no error is warranted)
- Not ignore or drop messages causing issues for the protocol processing (e.g. dropping PT-TLS SASL Authentication Data messages)
- Verify the security protections placed upon messages received from the network to ensure the messages are authentic and protected from attacks on the network

#### 4.1.2. Posture Transport Server

The Posture Transport Server is trusted by the Posture Broker Server to:

- Not observe, fabricate or alter the contents of the PB-TNC batches received from the network
- Not observe, fabricate or alter the PB-TNC batches passed down from the Posture Broker Server for transmission on the network

- Transmit on the network any PB-TNC batches passed down from the Posture Broker Server
- Deliver properly security protected messages received from the network that are destined for the Posture Broker Server
- Provide configured security protections (e.g. authentication, integrity and confidentiality) for the Posture Broker Server's messages sent on the network
- Expose the authenticated identity of the Posture Transport Client
- Verify the security protections placed upon messages received from the network to ensure the messages are authentic and protected from attacks on the network
- Provide a secure, reliable, in order delivery, full duplex transport for the Posture Broker Server's messages

The Posture Transport Server is trusted by the Posture Transport Client to:

- Not send malicious traffic intending to harm (e.g. denial of service) the Posture Transport Server
- Not send malformed messages (e.g. messages lacking PT-TLS header)
- Not send invalid or incorrect responses to messages (e.g. errors when no error is warranted)
- Not ignore or drop messages causing issues for the protocol processing (e.g. dropping PT-TLS SASL Result messages)
- Verify the security protections placed upon messages received from the network to ensure the messages are authentic and protected from attacks on the network

### 4.2. Security Threats and Countermeasures

Beyond the trusted relationships assumed in section 4.1. the PT-TLS protocol faces a number of potential security attacks that could require security countermeasures.

Generally, the PT-TLS protocol is responsible for offering strong security protections for all of the NEA protocols so any threats to its ability to protect NEA protocol messages could be very damaging to deployments. Once the message is delivered to the Posture Broker

Client or Posture Broker Server, the posture brokers are trusted to properly and safely process the messages.

#### 4.2.1. Message Theft

When PT-TLS messages are sent over unprotected network links or spanning local software stacks that are not trusted, the contents of the messages may be subject to information theft by an intermediary party. This theft could result in information being recorded for future use or analysis by the adversary. Messages observed by eavesdroppers could contain information that exposes potential weaknesses in the security of the endpoint, or system fingerprinting information easing the ability of the attacker to employ attacks more likely to be successful against the endpoint. The eavesdropper might also learn information about the endpoint or network policies that either singularly or collectively is considered sensitive information. For example, if PT-TLS does not provide confidentiality protection, an adversary could observe the PA-TNC attributes included in the PT-TLS message and determine that the endpoint is lacking patches, or particular sub-networks have more lenient policies.

In order to protect against NEA assessment message theft, the PT-TLS protocol provides strong cryptographic authentication, integrity and confidentiality protection. Deployers are strongly encouraged to employ best practice of the day TLS ciphers to ensure the information remains safe despite advances in technology and discovered cipher weaknesses. The use of bi-directional authentication of the assessment transport session ensures that only properly authenticated and authorized parties may be involved in an assessment dialog. The PT-TLS protocol also provides strong cryptography for all of the PB-TNC and PA-TNC protocol messages traveling over the network allowing the message contents to be hidden from potential theft by the adversary even if the attacker is able to observe the encrypted PT-TLS session.

#### 4.2.2. Message Fabrication

Attackers on the network or present within the NEA system could introduce fabricated PT-TLS messages intending to trick or create a denial of service against aspects of an assessment. For example, an adversary could attempt to insert into the message exchange fake PT-TLS error codes in order to disrupt communications.

The PT-TLS protocol provides strong security protections for the complete message exchange over the network. These security protections prevent an intermediary from being able to insert fake messages into the assessment. In particular, the TLS’s protocol use

of hashing algorithms provides strong integrity protections that allow for detection of any changes in the content of the message stream. Additionally, adversaries are unable to observe the PT-TLS protocol exchanges because they are encrypted by the TLS ciphers, so would have difficulty in determining where to insert the falsified message, since the attacker is unable to determine where the message boundaries exist. Even a successful message insertion did occur; the recipient would be able to detect it due to the TLS cipher suite's integrity checking failing.

#### 4.2.3. Message Modification

This attack could allow an active attacker capable of intercepting a message to modify a PT-TLS message or transported PA-TNC attribute to a desired value to ease the compromise of an endpoint. Without the ability for message recipients to detect whether a received message contains the same content as what was originally sent, active attackers can stealthily modify the attribute exchange.

The PT-TLS protocol leverages the TLS protocol to provide strong authentication and integrity protections as a countermeasure to this theat. The bi-directional authentication prevents the attacker from acting as an active man-in-the-middle to the protocol that could be used to modify the message exchange. The strong integrity protections (e.g. hashing) offered by TLS allows PT-TLS message recipients to detect message alterations by other types of network based adversaries.

#### 4.2.4. Denial of Service

A variety of types of denial of service attacks are possible against the PT-TLS protocol if the message exchanges are left unprotected while traveling over the network. The Posture Transport Client and Posture Transport Server are trusted not to participate in the denial of service of the assessment session, leaving the threats to come from the network.

The PT-TLS protocol provides bi-directional authentication capabilities in order to prevent a man-in-the-middle on the network from becoming an undetected active proxy of PT-TLS messages. Because the PT-TLS protocol runs after the TLS handshake and thus cipher establishment/use, all of the PT-TLC messages are protected from undetected modification that could create a denial of service situation. However it is possible for an adversary to alter the message flows causing each message to be rejected by the recipient because it fails the integrity checking.

The PT-TLS protocol operates as an application protocol on top of TLS and thus TCP/IP protocols, so is subject to denial of service attacks against the TLS, TCP and IP protocols.

#### 4.2.5. NEA Asokan Attacks

As described in section 3.3. and in the NEA Asokan Attack Analysis [ASOKAN], a sophisticated MITM attack can be mounted against NEA systems. The attacker forwards PA-TNC messages from a healthy machine through an unhealthy one so that the unhealthy machine can gain network access. Section 3.3. and the NEA Asokan Attack Analysis provide a detailed description of this attack and of the countermeasures that can be employed against it.

Because lying endpoint attacks are much easier than Asokan attacks and the only known effective countermeasure against lying endpoint attacks is the use of an External Measurement Agent (EMA), countermeasures against an Asokan attack are not necessary unless an EMA is in use. However, PT-TLS implementers may not know whether an EMA will be used with their implementation. Therefore, PT-TLS implementers SHOULD support the Asokan attack countermeasures by providing the value of the tls-unique channel binding to higher layers in the NEA reference model: Posture Broker Clients, Posture Broker Servers, Posture Collectors, and Posture Validators.

## 5. Privacy Considerations

The role of PT-TLS is to act as a secure transport for PB-TNC and other higher layer protocols. As such, PT-TLS does not directly utilize personally identifiable information (PII) except when client authentication is enabled. When client authentication is being used, the NEA Client will be asked to use SASL which may disclose a local identifier (e.g. username) associated with the endpoint and an authenticator (e.g. password) to authenticate that identity. Because the identity and authenticator are potentially privacy sensitive information, the NEA Client MUST offer a mechanism to restrict which NEA Servers will be sent this information. Similarly, the NEA Client should provide an indication to the person being identified that a request for their identity has been made in case they choose to opt out of the authentication to remain anonymous.
PT-TLS provides cryptographic peer authentication, message integrity and data confidentiality protections to higher layer NEA protocols that may exchange data potentially including PII. These security services can be used to protect any PII involved in an assessment from passive and active attackers on the network. Endpoints sending potentially privacy sensitive information should ensure that the PT-

TLS security protections (TLS cipher suites) negotiated for an assessment of the endpoint are adequate to avoid interception and off-line attacks of any long term privacy sensitive information.

## 6. IANA Considerations

This specification requests the creation of two new IANA registries and the assignment of a TCP port number. First, this specification requests the IANA reserve a registered TCP port number for use with the PT-TLS protocol upon publication of this specification as an Internet standard RFC.

This section also defines the contents of two new IANA registries: PT-TLS Message Types, and PT-TLS Error Codes. This section explains how these registries work.

All of the registries defined in this document support IETF standard values and vendor-defined values. To explain this phenomenon, we will use the PT-TLS Message Type as an example but the other registries work the same way.

Whenever a PT-TLS Message Type appears on a network, it is always accompanied by an SMI Private Enterprise Number (PEN), also known as a vendor ID. If this vendor ID is zero, the accompanying PT-TLS Message Type is an IETF standard value listed in the IANA registry for PT-TLS Message Types and its meaning is defined in the specification listed for that PT-TLS Message Type in that registry. If the vendor ID is not zero, the meaning of the PT-TLS Message Type is defined by the vendor identified by the vendor ID (as listed in the IANA registry for SMI PENs). The identified vendor is encouraged but not required to register with IANA some or all of the PT-TLS Message Types used with their vendor ID and publish a specification for each of these values.

This delegation of namespace is analogous to the technique used for OIDs. It can result in interoperability problems if vendors require support for particular vendor-specific values. However, such behavior is explicitly prohibited by this specification, which dictates that "Posture Transport Clients and Posture Transport Servers MUST NOT require support for particular vendor-specific PT-TLS Error Codes and MUST interoperate with other parties despite any differences in the set of vendor-specific PT-TLS Error Codes supported (although they MAY permit administrators to configure them to require support for specific PT-TLS error codes)." Similar requirements are included for PT-TLS Message Types and PT-TLS Auth Types.

### 6.1. Designated Expert Guidelines

For all of the IANA registries defined by this specification, new values are added to the registry by Expert Review with Specification Required, using the Designated Expert process defined in RFC 5226 [RFC5226].

This section provides guidance to designated experts so that they may make decisions using a philosophy appropriate for these registries.

The registries defined in this document have plenty of values. In most cases, the IETF has approximately 2^32 values available for it to define and each vendor has the same number of values for its use. Because there are so many values available, designated experts should not be terribly concerned about exhausting the set of values.

Instead, designated experts should focus on the following requirements. All values in these IANA registries MUST be documented in a specification that is permanently and publicly available. IETF standard values MUST also be useful, not harmful to the Internet, and defined in a manner that is clear and likely to ensure interoperability.

Designated experts should encourage vendors to avoid defining similar but incompatible values and instead agree on a single IETF standard value. However, it is beneficial to document existing practice.

There are several ways to ensure that a specification is permanently and publicly available. It may be published as an RFC. Alternatively, it may be published in another manner that makes it freely available to anyone. However, in this latter case, the vendor MUST supply a copy to the IANA and authorize the IANA to archive this copy and make it freely available to all if at some point the document becomes no longer freely available to all through other channels.

The following three sections provide guidance to the IANA in creating and managing the new IANA registries defined by this specification.

### 6.2. Registry for PT-TLS Message Types

The name for this registry is "PT-TLS Message Types". Each entry in this registry should include a human-readable name, an

SMI Private Enterprise Number, a decimal integer value between 0 and 2^32-1, and a reference to the specification where the contents of this message type are defined. This specification must define the meaning of the PT-TLS message type and the format and semantics of the PT-TLS Message Value field that include the designated Private Enterprise Number in the PT-TLS Message Type Vendor ID field and the designated numeric value in the PT-TLS Message Type field.

The following entries for this registry are defined in this document. Once this document becomes an RFC, they should become the initial entries in the registry for PT-TLS Message Types. Additional entries to this registry are added by Expert Review with Specification Required, following the guidelines in section 6.1.

| PEN | Value | Name | Defining Specification |
|---|---|---|---|
| 0 | 0 | Experimental | RFC # Assigned to this I-D |
| 0 | 1 | Version Request | RFC # Assigned to this I-D |
| 0 | 2 | Version Response | RFC # Assigned to this I-D |
| 0 | 3 | Request SASL Mechanisms | RFC # Assigned to this I-D |
| 0 | 4 | SASL Mechanisms | RFC # Assigned to this I-D |
| 0 | 5 | SASL Mechanism Selection | RFC # Assigned to this I-D |
| 0 | 6 | SASL Authentication Data | RFC # Assigned to this I-D |
| 0 | 7 | SASL Result | RFC # Assigned to this I-D |
| 0 | 8 | PT-TLS Batch | RFC # Assigned to this I-D |
| 0 | 9 | Reserved | RFC # Assigned to this I-D |
| 0 | 10 | Reserved | RFC # Assigned to this I-D |
| 0 | 11 | PT-TLS Error | RFC # Assigned to this I-D |
| 0 | 12 | Reserved | RFC # Assigned to this I-D |
| 0 | 0xffffffff | Reserved | RFC # Assigned to this I-D |

### 6.3. Registry for PT-TLS Error Codes

The name for this registry is "PT-TLS Error Codes". Each entry in this registry should include a human-readable name, an SMI Private Enterprise Number, a decimal integer value between 0 and 2^32-1, and a reference to the specification where this error code is defined. This specification must define the meaning of this error code and the format and semantics of the Error Information field for PT-TLS messages that have a PT-TLS Vendor ID of 0, a PT-TLS Message Type of PT-TLS Error, the designated Private Enterprise Number in the PT-TLS Error Code Vendor ID field, and the designated numeric value in the PT-TLS Error Code field.

The following entries for this registry are defined in this document. Once this document becomes an RFC, they should become the initial entries in the registry for PT-TLS Error Codes. Additional entries to this registry are added by Expert Review with Specification Required, following the guidelines in section 6.1.

| PEN | Value | Name | Defining Specification |
|---|---|---|---|
| 0 | 0 | Reserved | RFC # Assigned to this I-D |
| 0 | 1 | Malformed Message | RFC # Assigned to this I-D |
| 0 | 2 | Version Not Supported | RFC # Assigned to this I-D |
| 0 | 3 | Type Not Supported | RFC # Assigned to this I-D |
| 0 | 4 | Failed Authentication | RFC # Assigned to this I-D |
| 0 | 5 | Invalid Message Error | RFC # Assigned to this I-D |
| 0 | 6 | SASL Mechanism Error | RFC # Assigned to this I-D |
| 0 | 7 | Authentication Needed | RFC # Assigned to this I-D |

## 7. Acknowledgments

The authors of this draft would also like to acknowledge the following people who have contributed to or provided substantial input on the preparation of this document or predecessors to it: Syam Appala, Stuart Bailey, Lauren Giroux, Steve Hanna, Josh Howlett, Scott Kelly, Sung Lee, Lisa Lorenzin, Ravi Sahita, Subbu Srinivasan, Susan Thomson and Mark Townsend.

This document was prepared using 2-Word-v2.0.template.dot.

## 8. References

### 8.1. Normative References

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

[RFC4422] Melnikov A., Zeilenga K., "Simple Authentication and Security Layer (SASL)", RFC 4422, June 2006.

[RFC4616] Zeilenga K., "The PLAIN Simple Authentication and Security Layer (SASL) Mechanism", RFC 4616, August 2006.

[RFC5226] Narten T., Alvestrand H., "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, May 2008.

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

[RFC5792] Sangster P., Narayan K., "PA-TNC: A Posture Attribute Protocol (PA) Compatible with TNC", RFC 5792, March 2010.

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

## 8.2. Informative References

[ASOKAN] Salowey, J., Hanna, S., "NEA Asokan Attack Analysis", draft-salowey-nea-asokan-00.txt (work in progress), October 2010.

[IFT-TLS] Trusted Computing Group, "TNC IF-T: Binding to TLS", http://www.trustedcomputinggroup.org/files/resource_files/51F0757E-1D09-3519-AD63B6FD099658A6/TNC_IFT_TLS_v1_0_r16.pdf, May 2009.

[PT-EAP] Hanna, S., Sangster, P., "PT-EAP: Posture Transport (PT) Protocol For EAP Tunnel Methods", draft-hanna-nea-pt-eap-01.txt (work in progress), March 2011.

[RFC4346] Dierks T., Rescorla E., "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.

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

[RFC5929] Altman, J., Williams, N., Zhu L., "Channel Bindings for TLS", RFC 5929, July 2010.

# Appendix A. Evaluation Against NEA Requirements

This section evaluates the PT-TLS protocol against the PT requirements defined in the NEA Overview and Requirements and PB-TNC specifications. Each subsection considers a separate requirement and highlights how PT-TLS meets the requirement.

## A.1. Evaluation Against Requirement C-1

Requirement C-1 says:

C-1 NEA protocols MUST support multiple round trips between the NEA Client and NEA Server in a single assessment.

PT-TLS meets this requirement. Use of the TLS protocol over TCP/IP allows for multiple round trips of PT-TLS messages, which can carry multiple round trips of PB-TNC batches.

## A.2. Evaluation Against Requirements C-2

Requirement C-2 says:

C-2 NEA protocols SHOULD provide a way for both the NEA Client and the NEA Server to initiate a posture assessment or reassessment as needed.

PT-TLS meets this requirement. PT-TLS allows the NEA Client or the NEA Server to initiate a posture assessment or reassessment.

## A.3. Evaluation Against Requirements C-3

Requirement C-3 says:

C-3 NEA protocols including security capabilities MUST be capable of protecting against active and passive attacks by intermediaries and endpoints including prevention from replay based attacks.

PT-TLS meets this requirement. The use of TLS provides strong cryptographic authentication, integrity and confidentiality services for the NEA protocols.

## A.4. Evaluation Against Requirements C-4

Requirement C-4 says:

C-4 The PA and PB protocols MUST be capable of operating over any PT protocol. For example, the PB protocol must provide a transport independent interface allowing the PA protocol to operate without change across a variety of network protocol environments (e.g. EAP/802.1X, PANA, TLS and IKE/IPsec).

While this requirement is not applicable to PT, the PT-TLS protocol is independent of PA and PB allowing those protocols to operate over other PT protocols.

### A.5. Evaluation Against Requirements C-5

Requirement C-5 says:

C-5 The selection process for NEA protocols MUST evaluate and prefer the reuse of existing open standards that meet the requirements before defining new ones. The goal of NEA is not to create additional alternative protocols where acceptable solutions already exist.

Based on this requirement, PT-TLS should receive a strong preference. PT-TLS is equivalent with IF-T Binding to TLS 1.0, an open TCG specification. Selecting PT-TLS as the basis for the PT protocol will ensure compatibility with IF-T Binding to TLS, and with its implementations.

### A.6. Evaluation Against Requirements C-6

Requirement C-6 says:

C-6 NEA protocols MUST be highly scalable; the protocols MUST support many Posture Collectors on a large number of NEA Clients to be assessed by numerous Posture Validators residing on multiple NEA Servers.

PT-TLS meets this requirement. The PT-TLS protocol is independent of the quantity or size of the PA-TNC messages and the number of Posture Collectors and Posture Validators. PT-TLS provides the Posture Broker Client and Posture Broker Server a transport capable of carrying PT-TNC batches up to 2^32-16 octets in length. Posture Broker Clients and Posture Broker Servers wishing to send a PB-TNC batch longer than 2^32-16 octets could opt to split up set of attributes into multiple PB-TNC batches and send them sequentially since PT-TLS is full duplex.

The fields present in the PT-TLS protocol are also very scalable, allowing for the definition of a large (2^32) number of IETF standard and vendor-defined PT-TLS message types and message identifiers.

### A.7. Evaluation Against Requirements C-7

Requirement C-7 says:

C-7 The protocols MUST support efficient transport of a large number of attribute messages between the NEA Client and the NEA Server.

PT-TLS meets this requirement. PT-TLS will allow for transport of a very large number of attributes leveraging the underlying TCP/IP network access. The PT-TLS protocol only adds 16 octets of overhead per PT-TLS message, which is negligible since a single PT-TLS message might carry very many PA-TNC attributes within a single PB-TNC batch.

### A.8. Evaluation Against Requirements C-8

Requirement C-8 says:

C-8 NEA protocols MUST operate efficiently over low bandwidth or high latency links.

PT-TLS protocols meet this requirement. TLS will operate well over high latency or low bandwidth links leveraging TCP's ability to adjust to the underlying network carrier. The NEA protocols encapsulated by the PT-TLS protocol are designed to be able to operate over EAP with long RADIUS proxy chains so they can adapt to high latency or low bandwidth links. With the small amount of overhead added by PT-TLS, TLS, and TCP/IP, these protocols should still be efficient over high latency or low bandwidth networks.

### A.9. Evaluation Against Requirements C-9

Requirement C-9 says:

C-9 For any strings intended for display to a user, the protocols MUST support adapting these strings to the user's language preferences.

PT-TLS meets this requirement. The PT-TLS protocol does not include messages intended for display to the user.

### A.10. Evaluation Against Requirements C-10

Requirement C-10 says:

C-10 NEA protocols MUST support encoding of strings in UTF-8 format.

PT-TLS meets this requirement. All strings in the PT-TLS protocol are encoded in UTF-8 format. This allows the protocol to support a wide range of languages efficiently.

### A.11. Evaluation Against Requirements C-11

Requirement C-11 says:

C-11 Due to the potentially different transport characteristics provided by the underlying candidate PT protocols, the NEA Client and NEA Server MUST be capable of becoming aware of and adapting to the limitations of the available PT protocol. For example, some PT protocol characteristics that might impact the operation of PA and PB include restrictions on: which end can initiate a NEA connection, maximum data size in a message or full assessment, upper bound on number of roundtrips, and ordering (duplex) of messages exchanged. The selection process for the PT protocols MUST consider the limitations the candidate PT protocol would impose upon the PA and PB protocols.

PT-TLS meets this requirement. The PT-TLS protocol leverages the underlying TLS connection to offer a reliable, full duplex session capable of being initiated by the NEA Client or NEA Server. This TLS session allows for transmission of large PB-TNC batches with many roundtrips with very low overhead (only 16 octets of protocol overhead per PT-TLS message).

### A.12. Evaluation Against Requirements PT-1

Requirement PT-1 says:

PT-1 The PT protocol MUST NOT interpret the contents of PB messages being transported, i.e., the data it is carrying must be opaque to it.

PT-TLS meets this requirement. The PT-TLS protocol encapsulates PB-TNC batches without interpreting their contents.

### A.13. Evaluation Against Requirements PT-2

Requirement PT-2 says:

PT-2 The PT protocol MUST be capable of supporting mutual authentication, integrity, confidentiality, and replay protection of the PB messages between the Posture Transport Client and the Posture Transport Server.

PT-TLS meets this requirement. The PT-TLS protocol leverages TLS to provide mutual authentication, integrity protection and confidentiality as well as replay protection. For more information see the Security Considerations section 4.

### A.14. Evaluation Against Requirements PT-3

Requirement PT-3 says:

PT-3 The PT protocol MUST provide reliable delivery for the PB protocol. This includes the ability to perform fragmentation and reassembly, detect duplicates, and reorder to provide in-sequence delivery, as required.

PT-TLS meets this requirement. The PT-TLS protocol operates over TCP/IP which provides fragmentation/reassembly services and can detect/discard duplicate message and re-order messages if they arrive out of order over the network. PT-TLS provides a reliable, in-order delivery NEA message transport to the Posture Broker Client and Posture Broker Server components.

### A.15. Evaluation Against Requirements PT-4

Requirement PT-4 says:

PT-4 The PT protocol SHOULD be able to run over existing network access protocols such as 802.1X and IKEv2.

PT-TLS does NOT meet this requirement as it’s intended for a different usage. PT-TLS protocol requires the use of a TCP/IP connection to the network. PT-EAP (PT Binding to EAP Tunnel Methods) meets this requirement. PT-TLS is intended to be used after the endpoint has been admitted to the network.

### A.16. Evaluation Against Requirements PT-5

Requirement PT-5 says:

PT-5 The PT protocol SHOULD be able to run between a NEA Client and NEA Server over TCP or UDP (similar to Lightweight Directory Access Protocol (LDAP)).

PT-TLS meets this requirement. The PT-TLS protocol operates on top of an existing TCP/IP connection using TLS for network security.

### A.17. Evaluation Against Requirements PT-6 (from PB-TNC specification)

Requirement PT-6 says:

PT-6 The PT protocol MUST be connection oriented; it MUST support confirmed initiation and close down.

PT-TLS meets this requirement. The PT-TLS protocol operates on top of an existing TCP/IP connection which is connection oriented and supports confirmed initiation and tear down of the connection.

### A.18. Evaluation Against Requirements PT-7 (from PB-TNC specification)

Requirement PT-7 says:

PT-7 The PT protocol MUST be able to carry binary data.

PT-TLS meets this requirement. The PT-TLS protocol is capable of carrying binary data.

### A.19. Evaluation Against Requirements PT-8 (from PB-TNC specification)

Requirement PT-8 says:

PT-8 The PT protocol MUST provide mechanisms for flow control and congestion control.

PT-TLS meets this requirement. The PT-TLS protocol operates on top of TCP/IP which provides flow and congestion control.

### A.20. Evaluation Against Requirements PT-9 (from PB-TNC specification)

Requirement PT-9 says:

PT-9 PT protocol specifications MUST describe the capabilities that they provide for and limitations that they impose on the PB protocol (e.g. half/full duplex, maximum message size).

PT-TLS meets this requirement. This specification discusses the level of transport service provided to the Posture Broker Client and Posture Broker Server. Generally, the PT-TLS protocol supports the post network admission usages discussed in RFC 5209. The maximum message size for PT-TLS is only 16 octets less then the maximum message size allowable by PB-TNC.

Authors' Addresses

Paul Sangster
Symantec Corporation
6825 Citrine Dr
Carlsbad, CA 92009

Email: paul_sangster@symantec.com

Nancy Cam-Winget
Cisco Systems
80 West Tasman Drive
San Jose, CA 95134
US

Email: ncamwing@cisco.com

Joseph Salowey
Cisco Systems
2901 Third Avenue
Seattle, WA 98121
US

Email: jsalowey@cisco.com