Internet Engineering Task Force (IETF)
Request for Comments: 5793
Category: Standards Track
ISSN: 2070-1721

R. Sahita
Intel
S. Hanna
Juniper
R. Hurst
Microsoft
K. Narayan
Cisco Systems

March 2010

PB-TNC: A Posture Broker (PB) Protocol Compatible with Trusted Network Connect (TNC)

Abstract

This document specifies PB-TNC, a Posture Broker protocol identical to the Trusted Computing Group's IF-TNCCS 2.0 protocol. The document then evaluates PB-TNC against the requirements defined in the NEA Requirements specification.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc5793.

Copyright Notice

Table of Contents

1. Introduction

This document specifies PB-TNC, a Posture Broker (PB) protocol identical to the Trusted Computing Group's IF-TNCCS 2.0 protocol [7]. The document then evaluates PB-TNC against the requirements defined in the Network Endpoint Assessment (NEA) Requirements specification [8].

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 [8]. The reader is assumed to be thoroughly familiar with that document. No familiarity with TCG specifications is assumed.

1.2. Message Diagram Conventions

This specification defines the syntax of PB-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 1-octet field with only bit 0 set has the value 0x80.

1.3. Terminology

This document reuses the terminology defined in the NEA Requirements document. One new term is defined in this section.

Batch - A group of PB-TNC messages sent over a Posture Transport (PT) protocol at one time. Since the PB-TNC protocol needs to be able to work over a half-duplex PT protocol, PB-TNC messages are grouped into batches. The Posture Broker Client sends one batch to the Posture Broker Server, which responds with a batch.

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. PB-TNC Design Considerations

The primary purpose of the PB-TNC protocol is to carry Posture Attribute (PA) messages between Posture Collectors and Posture Validators. Also, PB-TNC must carry messages between the Posture Broker Client and the Posture Broker Server (known as PB-TNC messages) and manage the state of the assessment.

2.1. Message Addressing

The NEA Overview and Requirements document [8] describes in section 5.1.1.1 several ways that messages can be addressed and delivered to the proper Posture Collector(s) and Posture Validator(s). Of the techniques described in that section, PB-TNC supports dynamic identifiers and message types.

2.1.1. Message Types

Message types are the simplest and most common way to handle message delivery. Each PA message sent via PB-TNC has an associated PA message type, composed of a PA Message Vendor ID and a PA subtype.

The PA-TNC specification [10] provides a list of IETF Standard PA Subtypes, which are used with a PA Message Vendor ID of 0. These include values such as Operating System and Anti-Virus, which are used for messages relating to operating system and anti-virus posture.

Vendor-specific PA message types may be indicated by placing the defining vendor’s Structure of Management Information (SMI) Private Enterprise Number into the PA Message Vendor ID field and a PA Subtype value assigned by that vendor in the PA Subtype field. This allows each vendor to define its own set of PA Subtype values without worrying about collisions with other vendors or with standard values.

The PA message type is somewhat analogous to a MIME type in that it indicates the type of the PA message. Posture Collectors and Posture Validators can use local APIs to indicate to the Posture Broker Client and Posture Broker Server which PA message types they are interested in receiving. For instance, a Posture Validator that evaluates anti-virus posture might indicate that it would like to receive PA messages with a PA Message Vendor ID of 0 and a PA Subtype that matches the IETF Standard PA Subtype for Anti-Virus. It might also indicate interest in some vendor-specific PA message types to get additional vendor-specific information on anti-virus posture.

This type-based subscription model allows great flexibility in design and implementation. One Posture Validator may be responsible for evaluating several functions: anti-virus and host-based firewall, for instance. Posture Collectors do not need to know which Posture Validators are installed on the Posture Broker Server or what they handle. The Posture Collector simply sends PA messages with message types and the Posture Broker Server delivers them to the right Posture Validators.

Because the Posture Broker Client and Posture Broker Server must have access to the PA Message Vendor ID and PA Subtype fields and because these are routing identifiers independent of the contents of the PA messages, these fields are located in PB-TNC not inside the PA messages themselves.

A similar type-based system is used to tag PB-TNC messages. In this case, the extensibility benefits are not as essential as with PA-TNC messages, but the ability to define IETF Standard PB-TNC Message Types and vendor-specific PB-TNC message types is still valuable.

2.1.2. Dynamic Identifiers

The type-based message delivery model described above is not ideal for all circumstances. Sometimes it is important for a Posture Collector to deliver a message to a particular Posture Validator. For example, a particular Posture Validator might send a remediation message and the Posture Collector might need to send a response only to that one Posture Validator. To handle this circumstance, PB-TNC provides delivery based on dynamic identifiers.

When a Posture Broker Server loads a Posture Validator, it assigns it a Posture Validator ID. Any PA messages sent by a Posture Validator include that Posture Validator’s Posture Validator ID in the Posture Validator ID field of the PB-PA message. A Posture Collector that receives such a message can send a message in response and request exclusive delivery to the Posture Validator identified by that Posture Validator ID.

Dynamic identifiers avoid problems caused by the multicast nature of message types. Multiple Posture Collectors or Posture Validators may be registered for the same message type, and this can cause confusion if they all respond and the software designer did not consider that possibility. The dynamic identifier system allows more directed responses, but it does not work until at least one message has been received (so that the dynamic identifiers can be received). Static identifiers were considered as another alternative but rejected because they result in a brittle system that only works with a particular set of Posture Collectors and Posture Validators and causes problems if two Posture Collectors or Posture Validators with the same static identifier are installed.

2.2. Vendor IDs

In several places, PB-TNC needs to define a set of standard values but also allow vendor-specific extensions. In each of these places (PB-TNC Message Types, PA Subtypes, Remediation Parameters Types, and Error Codes), the solution chosen was to preface the values with a vendor ID. If a vendor ID is 0, the values in the next field are registered in an IANA registry and their meanings defined in an RFC. If a vendor ID is non-zero, the values in the next field are vendor specific and defined by the vendor whose SMI Private Enterprise Number matches the vendor ID. Vendor-specific messages that are not understood by the recipient are ignored and skipped unless they have the NOSKIP flag set, in which case an error code is returned.

2.3. Efficiency

PB-TNC needs to work with low bandwidth transports and low power devices. Therefore, a simple, compact format was chosen for the PB-TNC protocol: binary messages with a Type-Length-Value structure.

3. PB-TNC Protocol Description

3.1. Protocol Overview

The PB-TNC protocol carries batches of PB messages between a Posture Broker Client and a Posture Broker Server. It encapsulates PA messages and manages the NEA session. It runs over a PT protocol.

In order to work well over half-duplex PT protocols (such as those based on EAP [9]), PB-TNC supports half-duplex protocol operation. In this mode, the Posture Broker Client and Posture Broker Server take turns sending a single batch of messages to each other. While the half-duplex nature of PB-TNC could slow exchanges that require many round trips or bidirectional multimedia exchanges, this is not a problem in practice because endpoint assessments do not typically involve multimedia or a large number of round trips. The benefit of working over half-duplex transports outweighs any limitations imposed.

PB-TNC also supports full-duplex protocol operation so that PB-TNC exchanges can be re-initialized immediately when needed (e.g., if the Posture Broker Server policy changes or if the Posture Broker Client detects a suspicious event).

Each PB-TNC batch consists of a header followed by a sequence of PB-TNC messages. Each PB-TNC message has a Type-Length-Value (TLV) format with a few flags. The TLV format allows a recipient to skip messages that it does not understand. The TLV format also provides a standard way to mark messages as mandatory to ensure interoperability between a Posture Broker Client and a Posture Broker Server.

This specification defines certain standard PB-TNC message types. It also permits vendors to define their own vendor-specific message types. One of the most important standard PB-TNC message types is PB-PA. A message with this type contains a PA message and various message routing information. A Posture Broker Client or Posture Broker Server that receives such a message does not interpret the PA message within. Instead, it delivers the PA message to the appropriate set of Posture Collectors or Posture Validators, as determined using the message routing information contained in the PB-PA message.

A Posture Broker Server will often need to communicate with several Posture Broker Clients at once. The reverse may also be true, as when an endpoint has multiple network interfaces connected to different networks. Each connection between a Posture Broker Server and a Posture Broker Client is instantiated as a separate PB-TNC session. There may be several simultaneous sessions between a single Posture Broker Server and Posture Broker Client, but this is unusual.

3.2. PB-TNC State Machine

Figure 1 illustrates the PB-TNC state machine, showing the set of states that a PB-TNC session can have and the possible transitions among these states. The following paragraphs describe this state machine in more detail.

```
               Receive CRETRY        SRETRY
                    or SRETRY   +----------------+
                         +--+   |                |
                         v  |   v                |
                        +---------+  CRETRY  +---------+
              CDATA     | Server  |<---------| Decided | CLOSE
           +----------->| Working |--------->|         |-------+
           |            +---------+  RESULT  +---------+       |
           |                ^ |  |                             v
           |                | |  +---------------------->=======
         ========           | |            CLOSE        " End "
         " Init "      CDATA| |SDATA                    =======
         ========           | |                          ^    ^
           |  |             | v                          |    |
           |  | SDATA   +---------+          CLOSE       |    |
           |  +-------->| Client  |----------------------+    |
           |            | Working |                           |
           |            +---------+                           |
           |                |  ^                              |
           |                +--+                              |
           |            Receive CRETRY                        |
           |   CLOSE                                          |
           +--------------------------------------------------+
```

Figure 1: PB-TNC state machine

In this diagram, states are indicated by rectangular boxes. The initial and terminal states have double outlines (with = and "). State transitions are indicated by unidirectional arrows marked with the cause of the transition.

Many transitions (CDATA, SDATA, CRETRY, SRETRY, and RESULT) are triggered by the transmission or reception of a PB-TNC batch of a particular type. The type of a PB-TNC batch is indicated by the contents of the Batch Type field in the PB-TNC header for that batch. For brevity, this document says "a FOO batch" instead of "a PB-TNC batch whose Batch Type field contains FOO". Other transitions are triggered by receiving a PB-TNC batch of a particular type (e.g., Receive CRETRY). The CLOSE transition may be triggered by sending or receiving a CLOSE batch but may also be triggered by termination of the underlying PT connection.

A PB-TNC session starts in the Init state when the underlying transport protocol (PT) establishes a connection between a Posture Broker Client and a Posture Broker Server. If the Posture Broker Client initiated the underlying transport session, it starts by sending a CDATA batch to the Posture Broker Server, thus causing a transition to the Server Working state. If the Posture Broker Server initiated the transport session, it starts by sending a PB-TNC batch of type SDATA to the Posture Broker Client, thus causing a transition to the Client Working state.

The Posture Broker Client and Posture Broker Server may now alternate sending CDATA and SDATA batches to each other. Only the Posture Broker Client can send a data batch when the session is in the Client Working state, and only the Posture Broker Server can send a data batch when the session is in the Server Working state.

The most common way to end an exchange is for the Posture Broker Server to send a RESULT batch. This causes a transition into the Decided state. This is not a terminal state. The PT session can remain open and another exchange can be initiated by having the Posture Broker Client send a CRETRY batch. This can be useful when the Posture Broker Client (or more likely a Posture Collector) discovers a suspicious condition on the endpoint, for example. If the underlying transport protocol (PT) supports full-duplex operation, the Posture Broker Server can also initiate another exchange from this state by sending a SRETRY batch. This can be useful when the policy changes on the server, for example.

Whether an SRETRY or CRETRY message or both are sent, the next state is the Server Working State. From this state, the Posture Broker Server sends an SDATA batch and the new exchange begins. The state transitions marked Receive CRETRY and Receive CRETRY or SRETRY indicate that it is permissible to receive such messages in the indicated states, generally when the Posture Broker Client sent a CRETRY message at roughly the same time as the Posture Broker Server decided to send an SRETRY. In that case, a CRETRY message may be received while in the Server Working or Client Working state. Also, an SRETRY message may be received while in the Server Working state. These messages are redundant and therefore ignored, as indicated by the relevant transitions, which don’t cause a state change.

The only terminal state is the End state. This state is reached if the underlying PT connection closes. This can be caused by an action of the Posture Broker Client or Posture Broker Server or it can be caused by some external factor, such as pulling the network plug. When possible, a CLOSE batch SHOULD be sent before the underlying PT connection is terminated. However, there may be cases where the PT connection is closed without notice. For example, a plug may be pulled, a software program may fail, or a Posture Broker Client or Posture Broker Server may be unable to send a CLOSE message due to half-duplex limitations in the underlying PT protocol. In these cases, the Posture Broker Client and Posture Broker Server will generally receive some form of notification from the Posture Transport Client and Posture Transport Server that the PT connection has been closed. This notification can trigger the CLOSE transition. However, the notification interaction is not standardized since the vertical interfaces in the NEA Reference Model are not standardized. In any case, the reception of the CLOSE batch or notification of termination of the transport causes the transition to the End state.

Note that a Posture Broker Client and Posture Broker Server may not always have exactly the same state for a given PB-TNC session. For example, say that a session is in the Client Working state and the Posture Broker Client transmits a CDATA batch. While this batch is in transit (transmitted by the Posture Broker Client but not yet received by the Posture Broker Server), the Posture Broker Client will think that the session is in Server Working state but the Posture Broker Server will think that the session is in Client Working state. However, this is a temporary condition and does not cause problems in practice. The only possible issue is that a Posture Broker Client or Posture Broker Server does not know whether the other party has received its message until it receives a response from the other party.

If a half-duplex transport is used, note that the Posture Broker Server cannot send a SRETRY batch when the session is in the Decided state because the Posture Broker Server sent the most recent batch (the RESULT batch) and this would violate the half-duplex nature of the transport protocol. Instead, a server that wishes to initiate a new exchange in the Decided state when a half-duplex transport is in use should close the PT connection without sending a CLOSE batch and start a new PB-TNC session. This limitation does not exist when a full-duplex transport is used.

The Posture Broker Server and Posture Broker Client MUST follow the state machine described in this section.

3.3. Layering on PT

PB-TNC batches are carried over protocol bindings of the PT protocol, which provides the interaction between a Posture Transport Client and a Posture Transport Server. PB-TNC counts on PT to provide a secure transport. In particular, PT MUST support mutual authentication of the Posture Transport Client and the Posture Transport Server, confidentiality and integrity protection for PB-TNC batches, and protection against replay attacks. PB-TNC is unaware of the underlying transport protocols being used. PB-TNC operates directly on PT; no further layer of PB-TNC is expected.

3.3.1. Posture Transport (PT) Protocol Requirements Addendum

RFC 5209 [8] describes normative requirements for the Posture Transport protocol. This section specifies additional requirements for the Posture Transport protocol. Candidate Posture Transport protocols must indicate conformance to requirements specified in this section as well as section 7.4 of RFC 5209.

The additional requirements for candidate PT protocols are:

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

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

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

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).

3.4. Example of PB-TNC Encapsulation

This section shows how PA messages can be carried inside a PB-TNC batch that is inside a PT protocol.

Within the PT protocol, the PB-TNC header is packaged next, followed by two PB-PA messages that contain PA messages meant for the Posture Collectors and Posture Validators on the platform.

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           PT Protocol                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          PB-TNC Header                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           PB-PA Message                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           PB-PA Message                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Example of PB-TNC message encapsulation

This figure is conceptual, of course, and not an exact byte-for-byte replica.

4. PB-TNC Protocol Specification

This section defines the syntax and semantics of the PB-TNC protocol fields. If a Posture Broker Client or Posture Broker Server receives a batch that violates the requirements of this specification, it MUST respond by sending a fatal Invalid Parameter error in a CLOSE batch unless this document specifies otherwise.

4.1. PB-TNC Header

Every PB-TNC batch MUST start with the following header. A PB-TNC batch MUST contain only one instance of this header followed by zero or more PB-TNC messages. The PB-TNC messages are defined in subsequent sections of this specification.

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |    Version    |D|     Reserved                        | B-Type|
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                       Batch Length                            |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Version (8 bits)

This field indicates the version of the format for the PB-TNC message. This version is intended to allow for evolution of the PB-TNC protocol in a manner that can easily be detected by message recipients.

This field MUST be set to 2 when the batch conforms to this specification. Later versions of PB-TNC may define other values for this field. The values 0x00, 0x09, 0x0a, 0x0d, 0x20, and 0x3c are reserved and cannot be used for any version of PB-TNC to ensure that PB-TNC can be easily distinguished from earlier posture broker protocols already in use.

If a Posture Broker Client or Posture Broker Server receives a Version value that it does not support, it MUST respond with a PB-TNC batch with batch type CLOSE that contains only a fatal Version Not Supported error code and whose Version header field has the value 2. Implementations responding to a PB-TNC message containing a supported version MUST use the same Version number to minimize the risk of version incompatibility. PB-TNC message initiators that support multiple PB-TNC protocol versions SHOULD be able to alter which version of PB-TNC message they send based on prior message exchanges with a particular peer Posture Broker Client or Posture Broker Server.

Directionality (D) (1 bit)

When a Posture Broker Client is sending this message, the Directionality bit MUST be set to 0. When a Posture Broker Server is sending this message, the Directionality bit MUST be set to 1. This helps avoid any situation where two Posture Broker Clients or two Posture Broker Servers engage in a dialog. It also helps with debugging.

Reserved (19 bits)

This field is reserved. For this version of this specification, it MUST be set to 0 on transmission and ignored on reception. Future versions of this specification may allow senders to set some of these bits and recipients to interpret them.

B-Type (Batch Type) (4 bits)

This field is used to drive the state machine described in section 3.2. This field MUST have one of the values from the following table. If any other value is received, the recipient MUST ignore the contents of the batch and send a fatal Invalid Parameter error code in a CLOSE batch. If the value received is not permitted for the current state, according to the state machine in section 3.2., the recipient MUST ignore the contents of the batch and send a fatal Unexpected Batch Type error code in a CLOSE batch.

Number	Name	Definition
1	CDATA	The Posture Broker Client may send a batch with this Batch Type to convey messages to the Posture Broker Server. A Posture Broker Server MUST NOT send this Batch Type. A CDATA batch may be empty (contain no messages) if the Posture Broker Client has nothing to send.
2	SDATA	The Posture Broker Server may send a batch with this Batch Type to convey messages to the Posture Broker Client. A Posture Broker Client MUST NOT send this Batch Type. An SDATA batch may be empty (contain no messages) if the Posture Broker Server has nothing to send.
3	RESULT	The Posture Broker Server may send a batch with this Batch Type to indicate that it has completed its evaluation. The batch MUST include a PB-Assessment-Result message and MAY include a PB-Access-Recommendation message.
4	CRETRY	The Posture Broker Client may send a batch with this Batch Type to indicate that it wishes to restart an exchange. A Posture Broker Server MUST NOT send this Batch Type. A CRETRY batch may be empty (contain no messages) if the Posture Broker Client has nothing else to send.
5	SRETRY	The Posture Broker Server may send a batch with this Batch Type to indicate that it wishes to restart the exchange. A Posture Broker Client MUST NOT send this Batch Type. A SRETRY batch may be empty (contain no messages) if the Posture Broker Server has nothing else to send.
6	CLOSE	The Posture Broker Server or Posture Broker Client may send a batch with this Batch Type to indicate that it is about to terminate the underlying PT connection. A CLOSE batch may be empty (contain no messages) if there is nothing to send. However, if the termination is due to a fatal error, then the CLOSE batch MUST contain a PB-Error message.

Batch Length (32 bits)

This length field contains the size of the full PB-TNC batch in octets. This length includes the PB-TNC header and all the PB-TNC messages in the batch. In other words, it includes the entire contents of the batch. This field MUST contain at least the value 8 for the fixed-length fields in this header. Any Posture Broker Client or Posture Broker Server that receives a PB-TNC message with a PB-TNC Message Length field whose value is less than 8 MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

4.2. PB-TNC Message

All PB-TNC messages have the same overall structure, which is described in this section. Of course, the format and semantics of the PB-TNC Message Value field will vary, depending on the values of the PB-TNC Vendor ID and PB-TNC Message Type fields.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |     Flags     |               PB-TNC Vendor ID                |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                       PB-TNC Message Type                     |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                      PB-TNC Message Length                    |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |               PB-TNC Message Value (Variable Length)          |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Flags (8 bits)

This field defines flags impacting the processing of this message.

Bit 0 of this Flags field (the most significant bit) is known as the NOSKIP flag. If this flag is cleared (value 0), then the recipient (a Posture Broker Client or Posture Broker Server) may skip (ignore) this message if the message type is not understood or the recipient cannot or will not process the message as required in the definition of that message. If this flag is set (value 1), then recipients MUST NOT skip this attribute.

This flag does not mean that all recipients must support this message. Instead, any recipient that receives a message with this flag set to 1 but cannot or will not process it as required MUST NOT act on any part of the PB-TNC batch. Instead, the recipient MUST respond with a fatal Unsupported Mandatory Message error code in a CLOSE batch. In order to avoid taking action on some messages in a batch only to later find an unsupported NOSKIP flagged message, recipients of a PB-TNC batch might choose to scan all of the messages in the batch prior to acting upon any of the messages, checking to determine whether one of them is an unsupported message with the NOSKIP flag set.

The other bits in this Flags field are reserved. For this version of PB-TNC, they MUST be set to 0 on transmission and ignored on reception.

PB-TNC Vendor ID (24 bits)

The PB-TNC Vendor ID field identifies a vendor by using the SMI Private Enterprise Number (PEN). Any organization can receive its own unique PEN from IANA, the Internet Assigned Numbers Authority. This Vendor ID qualifies the PB-TNC Message Type field so that each vendor has 2^32-1 separate message types available for their use.

Message types standardized by the IETF use zero (0) in this field. The Vendor ID 0xffffff is reserved. Posture Broker Clients and Posture Broker Servers MUST NOT send messages in which the Vendor ID has this reserved value (0xffffff). If a Posture Broker Client or Posture Broker Server receives a message in which the PB-TNC Vendor ID has this reserved value (0xffffff), it MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

PB-TNC Message Type (32 bits)

The PB-TNC Message Type field identifies the type of the PB-TNC message contained in the PB-TNC Message Value field. The PB-TNC message type 0xffffffff is reserved. Posture Broker Clients and Posture Broker Servers MUST NOT send messages in which the PB-TNC Message Type field has this reserved value (0xffffffff). If a Posture Broker Client or Posture Broker Server receives a message in which the PB-TNC Message Type field has this reserved value (0xffffffff), it MUST respond with a fatal Invalid Parameter error code in a CLOSE batch. Unless otherwise prohibited in the definition of a particular PB-TNC message type (e.g., PB-Language-Preference), a single PB-TNC batch may contain multiple messages with the same message type and/or vendor ID.

The IETF and any other organization with a PEN can define 2^32-1 unique PB-TNC message types, as long as the organization's PEN is placed in the PB-TNC Vendor ID field of the message. Since the PB-TNC message type is qualified by the vendor ID, there is no risk of conflicts as long as each organization uses its own PEN for the vendor ID and manages its own set of 2^32-1 message type values.

This document defines certain PB-TNC message types that, when used with the IETF SMI PEN (0), have standard meanings. These are known as IETF Standard PB-TNC Message Types. Some of these PB-TNC message types are mandatory and therefore MUST be implemented by all Posture Broker Client and Posture Broker Server implementations that claim compliance with this specification. For details on which PB-TNC message types are mandatory, see the description of these message types later in section 4.

IANA maintains a registry of PB-TNC message types. Entries in this registry are added by Expert Review with Specification Required, following the guidelines in section 6.1.

New vendor-specific PB-TNC message types (those used with a non-zero PB-TNC vendor ID) may be defined and employed by vendors without IETF or IANA involvement. However, Posture Broker Clients and Posture Broker Servers MUST NOT require support for particular vendor-specific PB-TNC message types and MUST interoperate with other parties despite any differences in the set of vendor-specific PB-TNC message types supported (although they MAY permit administrators to configure them to require support for specific PB-TNC message types).

Note that the PB-TNC Message Type field is completely separate from the PA Subtype field. The same value (e.g., 0) may have different meanings as a PB-TNC message type and as a PA subtype.

PB-TNC Message Length (32 bits)

This field specifies the length of this PB-TNC message in octets. It includes this header (the fields Flags, PB-TNC Vendor ID, PB-TNC Message Type, and PB-TNC Message Length). Therefore, this value MUST always be at least 12. Any Posture Broker Client or Posture Broker Server that receives a message with a PB-TNC Message Length field whose value is less than 12 MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

PB-TNC Message Value (variable length)

The syntax and semantics of this field vary, depending on the values in the PB-TNC Vendor ID and PB-TNC Message Type fields. The syntax and semantics of several standard messages are defined in subsequent sections of this specification.

4.3. IETF Standard PB-TNC Message Types

The following table provides a reference list with brief descriptions of the IETF Standard PB-TNC Message Types defined in this specification. These PB-TNC message types must be used with a PB-TNC vendor ID of zero (0). If these PB-TNC message type values are used with a different PB-TNC vendor ID, they have a completely different meaning that is not defined in this specification.

For more details on these message types, see the remainder of section 4. For IETF Standard PA Subtypes (which are completely different from PB-TNC message types), please refer to the PA-TNC specification [10].

Message Type	Definition
0	PB-Experimental - reserved for experimental use
1	PB-PA - contains a PA message
2	PB-Assessment-Result - the overall assessment result computed by the Posture Broker Server
3	PB-Access-Recommendation - includes Posture Broker Server access recommendation
4	PB-Remediation-Parameters - includes Posture Broker Server remediation parameters
5	PB-Error - error indicator
6	PB-Language-Preference - sender’s preferred language(s) for human-readable strings
7	PB-Reason-String - string explaining reason for Posture Broker Server access recommendation

4.4. PB-Experimental

The PB-Experimental PB-TNC message type is a PB-TNC message type (value 0) that has been set aside for experimental purposes. It may be used to test code or for other experimental purposes. It MUST NOT be used in a production environment or in a product. This meaning for this PB-TNC message type only applies if the PB-TNC Vendor ID field in the PB-TNC Message Header contains the value zero (0). If a different Vendor ID is contained in that field, the PB-TNC message type 0 has a completely different meaning not defined in this specification.

The contents of the PB-TNC Message Length and PB-TNC Message Value fields for this PB-TNC message type are not specified. They may have almost any value, depending on what experiments are being conducted. Similarly, the Flags field for this message may have the NOSKIP bit set or cleared, depending on what experiments are being conducted. However, note that the PB-TNC Message Length field must have a value of at least 12 since that is the total of the length of the fixed-length fields at the start of the PB-TNC message (the fields Flags, PB-TNC Vendor ID, PB-TNC Message Type, and PB-TNC Message Length). Any Posture Broker Client or Posture Broker Server that receives a message with a PB-TNC Message Length field whose value is invalid MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

A Posture Broker Client or Posture Broker Server implementation intended for production use MUST NOT send a message with this Message Type with the value zero (0) as the vendor ID. If it receives a message with this message type and with the value zero (0) as the vendor ID, it MUST ignore the message unless the NOSKIP bit is set, in which case it MUST respond with a fatal Unsupported Mandatory Message error code in a CLOSE batch.

4.5. PB-PA

The PB-TNC message type named PB-PA (value 1) contains one PA message. Many batches will contain several PB-PA messages, but some batches may not contain any messages of this type.

All Posture Broker Client and Posture Broker Server implementations MUST implement support for this PB-TNC message type. Generally, this support will consist of forwarding the enclosed PA message to the appropriate Posture Collectors and Posture Validators. Specific requirements are contained later in the description of this message type.

The type of the PA message contained in a PB-PA message is indicated by the PA Message Vendor ID and PA Subtype fields, as described later in this section. The PA-TNC specification [10] describes several standard PA message types that can be identified by the PA Message Vendor ID and PA Subtype values listed in the PA-TNC specification. Other PA message types may also be defined, as described in the description of the PA Subtype field later in this section.

The NOSKIP flag in the PB-TNC Message Header MUST be set for this message type. Any Posture Broker Client or Posture Broker Server that receives a PB-PA message with the NOSKIP flag not set MUST ignore the message and MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

For the PB-PA message type, the PB-TNC Vendor ID field MUST contain the value zero (0) and the PB-TNC Message Type field MUST contain 1. If a non-zero value is contained in the PB-TNC Vendor ID field, message type 1 has a completely different meaning not defined in this specification.

The PB-TNC Message Length field MUST contain the length of the entire PB-TNC message, including the fixed-length fields at the start of the PB-TNC message (the fields Flags, PB-TNC Vendor ID, PB-TNC Message Type, and PB-TNC Message Length), the fixed-length fields listed below (Flags, PA Message Vendor ID, PA Subtype, Posture Collector Identifier, and Posture Validator Identifier), and the PA Message Body. Since the PA Message Body is variable length, the value in the PB-TNC Message Length field will vary also. However, it MUST always be at least 24 to cover the fixed-length fields listed in the preceding sentences. Any Posture Broker Client or Posture Broker Server that receives a PB-PA message with a PB-TNC Message Length field that has an invalid value MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

The following diagram illustrates the format and contents of the PB-TNC Message Value field for this message type. The text after this diagram describes the fields shown here.

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |    Flags      |               PA Message Vendor ID            |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                           PA Subtype                          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |  Posture Collector Identifier | Posture Validator Identifier  |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                 PA Message Body (Variable Length)             |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Flags (8 bits)

This field contains flags relating to the PA message.

Bit 0 of this flags field (the most significant bit) is known as the EXCL flag (for exclusive). If the EXCL bit is cleared (value 0), the Posture Broker Client or Posture Broker Server that receives this PB-TNC message SHOULD deliver the PA message contained in this PB-TNC message to all Posture Collectors or Posture Validators that have expressed an interest in PA messages with this PA Message Vendor ID and PA subtype. If a Posture Broker Client receives a message with the EXCL flag set (value 1), the Posture Broker Client SHOULD deliver the PA message contained in this PB-TNC message only to the Posture Collector identified by the Posture Collector Identifier field. However, if the identified Posture Collector has not expressed an interest in PA messages with this PA Message Vendor ID and PA subtype, the PA message should be silently discarded. Analogous requirements apply to a Posture Broker Server that receives a message with the EXCL flag set.

The EXCL bit allows, for example, a Posture Validator to handle the circumstance where there are two Posture Collectors on the endpoint that are interested in a particular kind of PA messages and the Posture Validator has remediation instructions that only apply to one of those Posture Collectors.

The other bits in this Flags field are reserved. For this version of PB-TNC, they MUST be set to 0 on transmission and ignored on reception.

PA Message Vendor ID (24 bits)

The PA Message Vendor ID field identifies a vendor by using the SMI Private Enterprise Number (PEN). Any organization can receive its own unique PEN from IANA, the Internet Assigned Numbers Authority. The PA Message Vendor ID qualifies the PA Subtype field so that each vendor has 2^32-1 separate PA subtypes available for its use. PA subtypes standardized by the IETF are always used with a PA Message Vendor ID of the value zero (0) in this field. The PA Message Vendor ID 0xffffff is reserved. A Posture Broker Client or Posture Broker Server MUST NOT send messages in which the PA Message Vendor ID field has this reserved value (0xffffff). If a Posture Broker Client or Posture Broker Server receives a message in which the PA Message Vendor ID has this reserved value (0xffffff), it MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

PA Subtype (32 bits)

The PA Subtype field identifies the type of the PA message contained in the PA Message Body field. The PA subtype 0xffffffff is reserved. A Posture Broker Client or Posture Broker Server MUST NOT send messages in which the PA Subtype field has this reserved value (0xffffffff). If a Posture Broker Client or Posture Broker Server receives a message in which the PA Subtype has this reserved value (0xffffffff), it MUST respond with a fatal Invalid Parameter error code in a CLOSE batch. A Posture Broker Client or Posture Broker Server MUST support having multiple PA messages in a single PB-TNC batch that have the same PA subtype and/or PA Message Vendor ID.

IANA maintains a registry of PA subtypes. Entries in this registry are added by Expert Review with Specification Required, following the guidelines in section 6.1. No PA subtypes are defined in this specification. Definitions of IETF Standard PA Subtypes are contained in the PA-TNC specification [10] and other specifications. IETF Standard PA Subtypes are always used with a PA Message Vendor ID of zero (0).

New vendor-specific PA subtypes (those used with a non-zero PA Message Vendor ID) may be defined and employed by vendors without IETF or IANA involvement. However, Posture Broker Clients and Posture Broker Servers MUST NOT require support for particular vendor-specific PA subtypes and MUST interoperate with other parties despite any differences in the set of vendor-specific PA subtypes supported (although they MAY permit administrators to configure them to require support for specific PA subtypes).

Note that the PB-TNC Message Type field is completely separate from the PA Subtype field. The same value (e.g., 0) may have different meanings as a PB-TNC message type and as a PA subtype.

Posture Collector Identifier (16 bits)

The Posture Collector Identifier field contains the identifier of the Posture Collector associated with this PA message.

The Posture Broker Client is responsible for assigning one or more Posture Collector Identifier values (but not 0xffff) to each Posture Collector involved in a message exchange. Multiple Posture Collector identifiers are required for appropriate correlation in cases where there are multiple components of the same type handled by a single Posture Collector, e.g., an endpoint with two VPN client implementations handled by a single VPN Posture Collector. Please refer to section 3.3 of the PA-TNC specification for an example that illustrates the use of multiple Posture Collector Identifiers. The Posture Collector Identifier value(s) assigned to a Posture Collector by a Posture Broker Client MUST NOT change during the course of a PT session. This identifier is used to identify a unique Posture Collector communicating with the Posture Broker Client on the endpoint during a NEA exchange, and is used by the Posture Validator to send response attributes to a specific Posture Collector component if required.

When a Posture Broker Server sets the EXCL flag for a PA message, the Posture Broker Server MUST set the Posture Collector Identifier field to the identifier of the Posture Collector that should receive the PA message. If the EXCL flag is not set, a Posture Broker Server MAY still set the Posture Collector Identifier value for PA messages that it sends to indicate that the PA message is intended as a response to a message sent by the

Posture Collector associated with the specified Posture Collector Identifier. If the Posture Broker Server does not wish to indicate any Posture Collector in this manner, it SHOULD set this field to the reserved value 0xffff.

Posture Validator Identifier (16 bits)

The Posture Validator Identifier field contains the identifier of the Posture Validator associated with this PA message.

The Posture Broker Server MUST assign a unique Posture Validator Identifier value (but not 0xffff) to each Posture Validator involved in a message exchange and include this Posture Validator identifier in this field for any PA messages sent by that Posture Validator. The Posture Validator Identifier value assigned to a Posture Validator by a Posture Broker Server MUST NOT change during the course of a PT session. This identifier is used to identify a unique Posture Validator communicating with the Posture Broker Server endpoint during a NEA exchange, and is used by the Posture Collector to send attributes to a specific Posture Validator if required.

When a Posture Broker Client sets the EXCL flag for a PA message, the Posture Broker Client MUST set the Posture Validator Identifier field to the identifier of the Posture Validator that should receive the PA message. If the EXCL flag is not set, a Posture Broker Client MAY still set the Posture Validator Identifier value for PA messages that it sends to indicate that the PA message is intended as a response to a message sent by the Posture Validator associated with the specified Posture Validator Identifier. If the Posture Broker Client does not wish to indicate any Posture Validator in this manner, it SHOULD set this field to the reserved value 0xffff.

PA Message Body (variable length)

The PA Message Body field contains the body of the PA message that is being carried in this PB-TNC message. The length of this field can be determined by subtracting the length of the fixed-length fields at the start of the PB-TNC message (the fields Flags, PB-TNC Vendor ID, PB-TNC Message Type, and PB-TNC Message Length) and the fixed-length fields at the start of the PB-PA message (Flags, PA Message Vendor ID, PA Subtype, Posture Collector Identifier, and Posture Validator Identifier) from the message length contained in the PB-TNC Message Length field. The length of these fixed-length fields is 24 octets. Therefore, any Posture Broker Client or Posture Broker Server that receives a PB-PA message with a PB-TNC Message Length field whose value is less than 24 MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

4.6. PB-Assessment-Result

The PB-TNC message type named PB-Assessment-Result (value 2) is used by the Posture Broker Server to provide the assessment result after the Posture Broker Server has completed the assessment of the endpoint. The Posture Broker Server will typically compute the assessment result as a cumulative of the individual assessment results received from the various Posture Validators; the algorithm for computation of assessment result at the Posture Broker layer is implementation specific and can also change based on policies in a specific deployment. The Posture Broker Server MUST include one message of this type in any batch of type RESULT and MUST NOT include a message of this type in any other type of batch. The Posture Broker Client MUST NOT send a PB-TNC message with this message type. If a Posture Broker Server receives a PB-TNC message with this message type, it MUST respond with a fatal Invalid Parameter error in a CLOSE batch. The Posture Broker Client MUST implement and process this message and MUST ignore any message with this message type that is not part of a batch of type RESULT.

The NOSKIP flag in the PB-TNC Message Header MUST be set for this message type. The PB-TNC Vendor ID field MUST contain the value zero (0) and the PB-TNC Message Type field MUST contain 2. If a non-zero value is contained in the PB-TNC Vendor ID field, message type 2 has a completely different meaning not defined in this specification. The PB-TNC Message Length field MUST contain the value 16 since that is the total of the length of the fixed-length fields at the start of the PB-TNC message (the fields Flags, PB-TNC Vendor ID, PB-TNC Message Type, and PB-TNC Message Length) along with the Assessment Result field described below. Any Posture Broker Client or Posture Broker Server that receives a PB-Assessment-Result message with a PB-TNC Message Length field that does not have a value of 16 MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

The following diagram illustrates the format and contents of the PB-TNC Message Value field for this message type. The text after this diagram describes the fields shown here.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Assessment Result                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Assessment Result

This 32-bit field MUST contain one of the following values

Value Description

0 Posture Broker Server assessed the endpoint to be compliant with policy.

1 Posture Broker Server assessed the endpoint to be non-compliant with policy but the difference from compliance was minor.

2 Posture Broker Server assessed the endpoint to be non-compliant with policy and the assessed difference from compliance was very significant.

3 Posture Broker Server was unable to determine policy compliance due to an error.

4 Posture Broker Server was unable to determine whether the assessed endpoint is compliant with policy based on the attributes provided by endpoint.

If a Posture Broker Client receives an Assessment Result value other than the five values described above, it MUST respond with a fatal Invalid Parameter error in a CLOSE batch. Other values may be defined in future versions of PB-TNC but only if the PB-TNC version number is changed. Therefore, there is no need for an IANA registry for Assessment Result values.

4.7. PB-Access-Recommendation

The PB-TNC message type named PB-Access-Recommendation (value 3) is used by the Posture Broker Server to provide an access recommendation after the Posture Broker Server has completed some assessment of the endpoint. The PB-Assessment-Result and the PB-Access-Recommendation attribute together constitute the global assessment decision for an endpoint. The PB-Access-Recommendation is not authoritative, and the network and host-based access control systems would typically use additional information to determine the network access that is granted to the endpoint. The Posture Broker Server MAY include one message of this type in any batch of type RESULT and MUST NOT include a message of this type in any other type of batch. Posture Broker Clients MUST NOT send a PB-TNC message with this message type. If a Posture Broker Server receives a PB-TNC message with this message type, it MUST respond with a fatal Invalid Parameter error in a CLOSE batch. The Posture Broker Client MUST implement and process this message and MUST ignore any message with this message type that is not part of a batch of type RESULT.

The NOSKIP flag in the PB-TNC Message Header MUST NOT be set for this message type. Any Posture Broker Client or Posture Broker Server that receives a PB-Access-Recommendation message with the NOSKIP flag set MUST ignore the message and MUST respond with a fatal Invalid Parameter error code in a CLOSE batch. The PB-TNC Vendor ID field MUST contain the value zero (0) and the PB-TNC Message Type field MUST contain 3. If a non-zero value is contained in the PB-TNC Vendor ID field, message type 3 has a completely different meaning not defined in this specification. The PB-TNC Message Length field MUST contain the value 16 since that is the total of the length of the fixed-length fields at the start of the PB-TNC message (the fields Flags, PB-TNC Vendor ID, PB-TNC Message Type, and PB-TNC Message Length) along with the Access Recommendation field described below. Any Posture Broker Client or Posture Broker Server that receives a PB-Access-Recommendation message with a PB-TNC Message Length field that does not have a value of 16 MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

The following diagram illustrates the format and contents of the PB-TNC Message Value field for this message type. The text after this diagram describes the fields shown here.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          Reserved             |   Access Recommendation Code  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Reserved (16 bits)

These Reserved bits MUST be set to 0 on transmission and ignored on reception.

Access Recommendation Code (16 bits)

The Access Recommendation Code field identifies the Access Recommendation that the Posture Broker Server has made for this Posture Broker Client at this time. This field MUST have one of these three values: 1 for Access Allowed (full access), 2 for Access Denied (no access), or 3 for Quarantined (partial access). If a Posture Broker Client receives an Access Recommendation Code value other than these three values, it MUST respond with a fatal Invalid Parameter error code in a CLOSE batch. Other values may be defined in future versions of PB-TNC but only if the PB-TNC version number is changed. Therefore, there is no need for an IANA registry for Access Recommendation Codes.

4.8. PB-Remediation-Parameters

The PB-TNC message type named PB-Remediation-Parameters (value 4) is used by the Posture Broker Server to provide global (not Posture Validator-specific) remediation parameters after the Posture Broker Server has completed some assessment of the endpoint. The Posture Broker Server MAY include one or more messages of this type in any batch of any type, but this message type is most useful in batches of type RESULT.

The Posture Broker Client MUST NOT send a PB-TNC message with this message type. If a Posture Broker Server receives a PB-TNC message with this message type, it MUST respond with a fatal Invalid Parameter error in a CLOSE batch. The Posture Broker Client may implement and process this message but is not required to do so. It may skip this message. Even if the Posture Broker Client implements this message type, it is not obligated to act on it.

The NOSKIP flag in the PB-TNC Message Header MUST NOT be set for this message type. The PB-TNC Vendor ID field MUST contain the value zero (0) and the PB-TNC Message Type field MUST contain 4. If a non-zero value is contained in the PB-TNC Vendor ID field, message type 4 has a completely different meaning not defined in this specification.

The PB-TNC Message Length field MUST contain the length of the entire PB-TNC message, including the fixed-length fields at the start of the PB-TNC message (the fields Flags, PB-TNC Vendor ID, PB-TNC Message Type, and PB-TNC Message Length), the fixed-length fields listed below (Reserved, Remediation Parameters Vendor ID, and Remediation Parameters Type), and the Remediation Parameters. Since the Remediation Parameters field is variable length, the value in the PB-TNC Message Length field will vary also. However, it MUST always be at least 20 to cover the fixed-length fields listed in the preceding sentences. Any Posture Broker Client that receives a PB-Remediation-Parameters message with a PB-TNC Message Length field that contains an invalid value (e.g., less than 20) MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

The following diagram illustrates the format and contents of the PB-TNC Message Value field for this message type. The text after this diagram describes the fields shown here.

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |    Reserved   |       Remediation Parameters Vendor ID        |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                  Remediation Parameters Type                  |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |            Remediation Parameters (Variable Length)           |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Reserved (8 bits)

These Reserved bits MUST be set to 0 on transmission and ignored on reception.

Remediation Parameters Vendor ID (24 bits)

The Remediation Parameters Vendor ID field identifies a vendor by using the SMI Private Enterprise Number (PEN). Any organization can receive its own unique PEN from IANA, the Internet Assigned Numbers Authority. The Remediation Parameters Vendor ID qualifies the Remediation Parameters Type field so that each vendor has 2^32 separate Remediation Parameters Types available for its use. Remediation Parameters Types standardized by the IETF are always used with the value zero (0) in this field.

Remediation Parameters Type (32 bits)

The Remediation Parameters Type field identifies the type of remediation parameters contained in the Remediation Parameters field. A Posture Broker Client or Posture Broker Server MUST support having multiple Remediation Parameters messages contained in a single PB-TNC batch that have the same Remediation Parameters Type and/or Remediation Parameters Vendor ID.

IANA maintains a registry of PB-TNC Remediation Parameters Types. Entries in this registry are added by Expert Review with Specification Required, following the guidelines in section 6.1. A list of IETF Standard PB-TNC Remediation Parameters Types defined in this specification appears later in this section.

New vendor-specific Remediation Parameters Types (those used with a non-zero Remediation Parameters vendor ID) may be defined and employed by vendors without IETF or IANA involvement. However, Posture Broker Clients and Posture Broker Servers MUST NOT require support for particular vendor-specific Remediation Parameters Types and MUST interoperate with other parties despite any differences in the set of vendor-specific Remediation Parameters

Types supported (although they MAY permit administrators to configure them to require support for specific Remediation Parameters Types).

Note that the Remediation Parameters Type is completely separate from the PB-TNC Message Type and the PA Subtype fields. The same value (e.g., 0) may have different meanings in each of these fields.

Remediation Parameters (variable length)

The Remediation Parameters field contains the actual remediation parameters carried in this PB-TNC message. The length of this field can be determined by subtracting the length of the fixed-length fields at the start of the PB-TNC message (the fields Flags, PB-TNC Vendor ID, PB-TNC Message Type, and PB-TNC Message Length) and the fixed-length fields at the start of the PB-Remediation-Parameters message (Reserved, Remediation Parameters Vendor ID, and Remediation Parameters Type) from the message length contained in the PB-TNC Message Length field. The length of these fixed-length fields is 20 octets. Therefore, any Posture Broker Client that receives a PB-Remediation-Parameters message with a PB-TNC Message Length field whose value is less than 20 MUST consider this a malformed message. The Posture Broker Client MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

4.8.1. IETF Standard PB-TNC Remediation Parameters Types

This subsection defines several Remediation Parameters Types that have been standardized by the IETF.

Remediation-URI

This Remediation Parameters Type is employed by creating a PB-Remediation-Parameters message with a Remediation Parameters Vendor ID equal to the value zero (0) and a Remediation Parameters Type of 1. The Remediation Parameters field in the PB-Remediation-Parameters message MUST contain a URI, as described in RFC 3986 [2]. This URI contains instructions and resources for remediation. The Posture Broker Client MAY load the URI and display the resulting web page to the user. The Posture Broker Client MAY also ignore the URI or take another action with it. The Posture Broker Server and any other parties involved in configuring this remediation URI should consider the likely capabilities of the Posture Broker Client when creating the URI and the content referenced by the URI. For example, they should consider the Posture Broker Client’s language preferences as expressed in the PB-Language-Preference message.

Remediation-String

This Remediation Parameters Type is employed by creating a PB-Remediation-Parameters message with a Remediation Parameters Vendor ID equal to the value zero (0) and a Remediation Parameters Type of 2. The Remediation Parameters field in the PB-Remediation-Parameters message MUST contain the structure defined below, which contains human-readable instructions for remediation.

The Posture Broker Client MAY display the instructions to the user. The Posture Broker Client MAY also ignore the instructions or take another action with them. The Posture Broker Server and any other parties involved in configuring these instructions should consider the likely capabilities of the Posture Broker Client when creating the instructions. For example, they should consider the Posture Broker Client’s language preferences as expressed in the PB-Language-Preference message.

The following diagram illustrates the format and contents of the Remediation Parameters field when carrying a Remediation-String parameter. The text after this diagram describes the fields shown here.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Remediation String Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Remediation String (Variable Length)           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Lang Code Len |  Remediation String Lang Code (Variable Len)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Remediation String Length (32 bits)

The Remediation String Length contains the length of the Remediation String field in octets.

Remediation String (variable length)

The Remediation String field MUST contain a UTF-8 [6] encoded string. This string contains human-readable instructions for remediation that MAY be displayed to the user by the Posture Broker Client. NUL termination MUST NOT be included. If a

Posture Broker Client receives a Reason String that does contain a NUL termination, it MUST respond with a fatal Invalid Parameter error in a CLOSE batch.

Lang Code Len (8 bits)

The Lang Code Len field contains the length of the Remediation String Lang Code field in octets. This value may be set to zero to indicate that the language code for the Remediation String field is not known.

Remediation String Lang Code (variable length)

The Remediation String Lang Code field contains a US-ASCII string composed of a well-formed RFC 4646 [3] language tag that indicates the language(s) used in the Remediation String in the Remediation Parameters field. A zero-length string may be sent for this field (essentially omitting this field) to indicate that the language code for the Remediation String field is not known.

4.9. PB-Error

The PB-TNC message type named PB-Error (value 5) is used by the Posture Broker Client or Posture Broker Server to indicate that an error has occurred. The Posture Broker Client or Posture Broker Server MAY include one or more messages of this type in any batch of any type. Other messages may also be included in the same batch. The party that receives a PB-Error message MAY log it or take other action as deemed appropriate. If the FATAL flag is set (value 1), the recipient MUST terminate the PB-TNC session after processing the batch without sending any messages in response. Every Posture Broker Client and Posture Broker Server MUST implement this message type.

The NOSKIP flag in the PB-TNC Message Header MUST be set for this message type. The PB-TNC Vendor ID field MUST contain the value zero (0) and the PB-TNC Message Type field MUST contain 5. If a non-zero value is contained in the PB-TNC Vendor ID field, message type 5 has a completely different meaning not defined in this specification.

The PB-TNC Message Length field MUST contain the length of the entire PB-TNC message, including the fixed-length fields at the start of the PB-TNC message (the fields Flags, PB-TNC Vendor ID, PB-TNC Message Type, and PB-TNC Message Length), the fixed-length fields listed below (Flags, Error Code Vendor ID, Error Code, and Reserved), and the Error Parameters. Since the Error Parameters field is variable length, the value in the PB-TNC Message Length field will vary also.

However, it MUST always be at least 20 to cover the fixed-length fields listed in the preceding sentences. Any Posture Broker Client or Posture Broker Server that receives a PB-Error message with a PB-TNC Message Length field that contains an invalid value (e.g., less than 20) MUST respond with a fatal Invalid Parameter error code in a CLOSE batch. Any PB-Error message generated while processing a PB-Error message MUST be a fatal error to avoid the chance of generating an infinite loop of errors.

The following diagram illustrates the format and contents of the PB-TNC Message Value field for this message type. The text after this diagram describes the fields shown here.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    Flags      |            Error Code Vendor ID               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |           Error Code          |           Reserved            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |              Error Parameters (Variable Length)               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Flags (8 bits)

This field defines flags relating to the error.

Bit 0 of this flags field (the most significant bit) is known as the FATAL flag. If the FATAL bit is cleared (value 0), the Posture Broker Client or Posture Broker Server that receives this PB-TNC message SHOULD process this error and then continue with the exchange. If the FATAL flag is set (value 1), the Posture Broker Client or Posture Broker Server that receives this PB-TNC message MUST terminate the exchange after processing the error. In addition, any Posture Broker Client or Posture Broker Server that sends a fatal error MUST NOT process the batch that caused the error and MUST terminate the exchange after sending the batch containing the error report. A PB-Error message with the FATAL flag set MUST always be sent in a CLOSE batch since the sender will be terminating the exchange immediately after sending the batch.

The FATAL bit allows a Posture Broker Client or Posture Broker Server to signal a fatal error (like an invalid batch type) and/or a non-fatal error (like an invalid language tag for a preferred language).

The other bits in this Flags field are reserved. For this version of PB-TNC, they MUST be set to 0 on transmission and ignored on reception.

Error Code Vendor ID (24 bits)

The Error Code Vendor ID field identifies a vendor by using the SMI Private Enterprise Number (PEN). Any organization can receive its own unique PEN from IANA, the Internet Assigned Numbers Authority. The Error Code Vendor ID qualifies the Error Code field so that each vendor has 2^16 separate Error Codes available for its use. Error codes standardized by the IETF are always used with the value zero (0) in this field. For detailed descriptions of those messages, see the next few subsections.

Error Code (16 bits)

The Error Code field identifies the type of error being signaled with this message. The format of the Error Parameters field depends on the value of the Error Code Vendor ID and the Error Code. However, any recipient that does not understand a particular error code can process the error fairly well by using the FATAL flag to determine whether the error is fatal and the PB-TNC Message Length to skip over the Error Parameters field (or log it).

IANA maintains a registry of PB-TNC Error Codes. Entries in this registry are added by Expert Review with Specification Required, following the guidelines in section 6.1. A list of IETF Standard PB-TNC Error Codes defined in this specification appears later in section 4.9.1.

New vendor-specific error codes (those used with a non-zero error code vendor ID) may be defined and employed by vendors without IETF or IANA involvement. Posture Broker Clients and Posture Broker Servers that receive an unknown error code MUST process this error code gracefully by ignoring or logging it if it is not marked as fatal and terminating the exchange if it is marked as fatal.

Reserved (16 bits)

The Reserved bits MUST be set to 0 on transmission and ignored on reception.

4.9.1. IETF Standard PB-TNC Error Codes

The following error codes are IETF Standard PB-TNC Error Codes, hence the Error Code Vendor ID MUST be the value zero (0). The following table defines the 16-bit error code. Vendor-specific error codes may be defined by setting the Error Code Vendor ID to the defining vendor’s SMI PEN and setting the Error Code field to whatever error code(s) that vendor has defined. The format, length, and meaning of the Error Parameters field varies, based on the Error Code Vendor ID and Error Code. Subsequent sections of this document define the format, length, and meaning of the Error Parameters for the IETF Standard PB-TNC Error Codes defined in this section.

Error Code	Definition
0	Unexpected Batch Type. Error Parameters are empty.
1	Invalid Parameter. Error Parameters has offset where invalid value was found.
2	Local Error. Error Parameters are empty.
3	Unsupported Mandatory Message. Error Parameters has offset of offending PB-TNC Message
4	Version Not Supported. Error Parameters has information about which versions are supported.

4.9.2. Error Parameters Structures for IETF Standard PB-TNC Error Codes

This section defines the format, length, and meaning of the Error Parameters field for the IETF Standard PB-TNC Error Codes defined in this specification.

The Error Parameters field is zero length for the IETF Standard PB-TNC Error Code 0. The FATAL flag MUST be set for this error code.

The Error Parameters field has the following structure for the IETF Standard PB-TNC Error Code 1. The Offset field is the offset in octets from the start of the PB-TNC batch to the invalid value. The FATAL flag may be either set or cleared for this error code.

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                             Offset                            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Error Parameters field is zero length for the IETF Standard PB-TNC Error Code 2. The FATAL flag MUST be set for this error code.

The Error Parameters field has the following structure for the IETF Standard PB-TNC Error Code 3. The Offset field is the offset in octets from the start of the PB-TNC batch to the PB-TNC message whose message type was not recognized (and where the NOSKIP flag was set). The FATAL flag MUST be set for this error code.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                              Offset                           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Error Parameters field has the following structure for the IETF Standard PB-TNC Error Code 4. The FATAL flag MUST be set for this error code.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Bad Version  |  Max Version  |  Min Version  |   Reserved    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Bad Version field is the version number that was received and is not supported. The Max Version and Min Version fields indicate which PB-TNC version numbers are supported by the sender of the error code. The sender MUST support all PB-TNC versions between the Min Version and the Max Version, inclusive (i.e., including the Min Version and the Max Version) but excluding the reserved versions listed in section 4.1. The Reserved field MUST be set to 0 on transmission and ignored upon reception. When possible, recipients of this error code SHOULD send future messages to the Posture Broker Server or Posture Broker Client that originated this error message with a PB-TNC version number within the stated range.

Any party that is sending the Version Not Supported error code MUST include that error code as the only PB-TNC message in a PB-TNC CLOSE batch with version number 2. All parties that send PB-TNC batches SHOULD be able to properly process a batch that meets this description, even if they cannot process any other aspect of PB-TNC version 2. This ensures that a PB-TNC version exchange can proceed properly, no matter what versions of PB-TNC the parties implement.

4.10. PB-Language-Preference

The PB-TNC message type named PB-Language-Parameters (value 6) is used by the Posture Broker Client to indicate which language or languages it would prefer for any human-readable strings that might be sent to it. This allows the Posture Broker Server and Posture Validators to adapt any messages they may send to the Posture Broker Client’s preferences (probably determined by the language preferences of the endpoint’s users).

The Posture Broker Server may also send this message type to the Posture Broker Client to indicate the Posture Broker Server’s language preferences, but this is not very useful since the Posture Broker Client rarely sends human-readable strings to the Posture Broker Server and, if it does, rarely can adapt those strings to the preferences of the Posture Broker Server.

No Posture Broker Client or Posture Broker Server is required to send or implement this message type. However, a Posture Broker Server SHOULD attempt to adapt to user language preferences by implementing this message type, passing the language preference information to Posture Validators, and allowing administrators to configure human-readable languages in whatever languages are preferred by their users.

A Posture Broker Client or Posture Broker Server may include a message of this type in any batch of any type. However, it is suggested that this message be included in the first batch sent by the Posture Broker Client or Posture Broker Server in a PB-TNC session so that the recipient can start adapting its human-readable messages as soon as possible. If one PB-Language-Parameters message is received and then another one is received in a later batch for the same PB-TNC session, the value included in the later message should be considered to replace the value in the earlier message.

A Posture Broker Client or Posture Broker Server MUST NOT include more than one message of this type in a single batch. If a Posture Broker Client or Posture Broker Server receives more than one message of this type in a single batch, it should ignore all but the last one.

The NOSKIP flag in the PB-TNC Message Header MUST NOT be set for this message type. The PB-TNC Vendor ID field MUST contain the value zero (0) and the PB-TNC Message Type field MUST contain 6. If a non-zero value is contained in the PB-TNC Vendor ID field, message type 6 has a completely different meaning not defined in this specification.

The PB-TNC Message Length field MUST contain the length of the entire PB-TNC message, including the fixed-length fields at the start of the PB-TNC message (the fields Flags, PB-TNC Vendor ID, PB-TNC Message Type, and PB-TNC Message Length) and the Language Preference field. Since the Language Preference field is variable length, the value in the PB-TNC Message Length field will vary also. However, it MUST always be at least 12 to cover the fixed-length fields listed in the preceding sentences.

The following diagram illustrates the format and contents of the PB-TNC Message Value field for this message type. The text after this diagram describes the fields shown here.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              Language Preference (Variable Length)            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Language Preference (variable length)

The Language Preference field contains an Accept-Language header, as described in RFC 3282 [4] (using the RFC 2234 ABNF definition of Accept-Language included in that RFC, US-ASCII only, no control characters allowed, no comments, no NUL termination). Any Posture Broker Client or Posture Broker Server that sends a PB-Language-Preference message MUST ensure that the Language Preference field conforms to this format. For example, one acceptable value would be "Accept-Language: fr, en" (without the quote marks).

A zero-length Language Preference field indicates that no language preference information is available. Generally, there’s no need to send a PB-Language-Preference message with a zero-length Language Preference field since this is equivalent to sending no PB-Language-Preference message at all, but it may be useful to send a zero-length Language Preference field if a PB-Language-Preference message with a non-zero-length Language Preference field was sent in an earlier batch but these preferences no longer apply.

4.11. PB-Reason-String

The PB-TNC message type named PB-Reason-String (value 7) is used by the Posture Broker Server to provide a human-readable explanation for the global assessment decision conveyed in the PB-Assessment-Result & PB-Access-Recommendation messages. Therefore, a PB-Reason-String message SHOULD only be included in the same batch as the PB-Assessment-Result and PB-Access-Recommendation message. The Posture Broker Client MUST NOT ever send a PB-Reason-String message.

The Posture Broker Client is not required to implement this message type and the Posture Broker Server is not required to send it. However, there is some benefit to doing so since users are often curious about why the endpoint was considered non-compliant. The manner in which a Posture Broker Client uses this field is up to the implementer and not specified here. The Posture Broker Client MAY display the message to the user, log it, ignore it, or take any other action that is not inconsistent with the requirements of this specification. Since the strings contained in this message are human-readable, the Posture Broker Server SHOULD adapt them to the Posture Broker Client’s language preferences as expressed in any PB-Language-Preference message sent by the Posture Broker Client in this PB-TNC session.

A Posture Broker Server MAY include more than one message of this type in any batch of any type. However, it is suggested that this message be included in the same batch as the PB-Assessment-Result and PB-Access-Recommendation message. If more than one PB-Reason-String message is included in a single batch, the Posture Broker Client SHOULD consider the strings included in these messages to be equivalent in meaning. This allows the Posture Broker Server to return multiple equivalent reason strings in different languages, which may help if the Posture Broker Server is not able to accommodate the Posture Broker Client’s language preferences.

The NOSKIP flag in the PB-TNC Message Header MUST NOT be set for this message type. The PB-TNC Vendor ID field MUST contain the value zero (0) and the PB-TNC Message Type field MUST contain 7. If a non-zero value is contained in the PB-TNC Vendor ID field, message type 7 has a completely different meaning not defined in this specification.

The PB-TNC Message Length field MUST contain the length of the entire PB-TNC message, including the fixed-length fields at the start of the PB-TNC message (the fields Flags, PB-TNC Vendor ID, PB-TNC Message Type, and PB-TNC Message Length), the fixed-length fields listed below (Reason String Length and Lang Code Len), and the Reason String and Reason String Language Code fields. Since the Reason String and Reason String Language Code fields are variable length, the value in the PB-TNC Message Length field will vary also. However, it MUST always be at least 17 to cover the fixed-length fields listed in the preceding sentences. In fact, the PB-TNC Message Length field MUST be exactly the sum of 17 (for the fixed-length fields) and the values of the Reason String Length and Lang Code Len fields. If this is not the case, the recipient MUST respond with a fatal Invalid Parameter error code in a CLOSE batch.

The following diagram illustrates the format and contents of the PB-TNC Message Value field for this message type. The text after this diagram describes the fields shown here.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                      Reason String Length                     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                Reason String (Variable Length)                |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | Lang Code Len | Reason String Language Code (Variable Length) |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Reason String Length (32 bits)

The Reason String Length field contains the length of the Reason String field in octets.

Reason String (variable length)

The Reason String field contains a UTF-8 encoded string that provides a human-readable reason for the Posture Broker Server’s assessment decision. NUL termination MUST NOT be included. If a Posture Broker Client receives a Reason String that does contain a NUL termination, it MUST respond with a fatal Invalid Parameter error code in a CLOSE batch. A zero-length string MUST NOT be sent since this is the same as sending no reason string at all, leaving the reason unspecified.

Lang Code Len (8 bits)

The Lang Code Len field contains the length of the Reason String Language Code field in octets.

Reason String Language Code (variable length)

The Reason String Language Code field contains a US-ASCII string containing a well-formed RFC 4646 [3] language tag that indicates the language(s) used in the Reason String in this message. NUL termination MUST NOT be included in this field. A zero-length string MAY be sent for this field (essentially omitting this field) to indicate that the language code for the reason string is not known.

5. Security Considerations

PT is required and assumed to provide reliable and secure transport for the PB-TNC protocol (including authentication, confidentiality, integrity protection, and replay protection). Still, it is useful to describe the possible threats to PB-TNC and the countermeasures that are or can be employed. This section does that.

5.1. Threat Model

There are several possible threats to the PB-TNC protocol.

Untrusted intermediaries on the network between the NEA Client and the NEA Server may attempt to observe data sent between the Posture Broker Client and the Posture Broker Server via PB-TNC, modify this data in transit, reorder it, or replay it. They may also attempt to mount a denial-of-service attack against either party or truncate the exchange prematurely. If successful, these attacks may result in improper assessment decisions relating to the NEA Client, failure to reassess these decisions in light of changed circumstances, improper remediation instructions sent to the NEA Client (which could lead to the compromise of the NEA Client), unauthorized access to confidential information about the NEA Client’s health and/or identity, improper reason strings or other messages that might be displayed to the user, access to reusable credentials such as posture assertions, denial of service on the NEA Client, and even complete denial of access to the network (if a denial-of-service attack against the NEA Server was successful and the network required permission from the NEA Server to grant network access).

Trusted intermediaries between the Posture Broker Client and the Posture Broker Server include the Posture Transport Client and the Posture Transport Server. These parties are considered trusted because they are responsible for properly implementing the security protections provided by PT. If they fail to do so properly, these security protections may be diminished or eliminated altogether. The possible attacks are the same as those listed in the previous paragraph. To give one fairly likely example, if a Posture Transport Client fails to properly authenticate and authorize the Posture Transport Server (whether through implementation error or through user configuration to "trust anyone"), the improperly authorized Posture Transport Server may mount any of the previously described attacks against the NEA Client.

Compromise of any of the trusted parties (the Posture Broker Client, the Posture Transport Client, the Posture Broker Server, or the Posture Transport Server) may result in failures that are equivalent to those listed in the first paragraph. These failures may be even more dangerous since they will not be detectable by observing network traffic or by examining and comparing audit logs. Failure to properly secure communications between the Posture Broker Client and the Posture Transport Client or between the Posture Broker Server and the Posture Transport Server is usually indistinguishable from compromise of those parties. Compromise of the operating system or other critical software, firmware, or hardware components on the NEA Client or NEA Server will typically result in an equivalent result. And an attacker’s ability to gain privileged access to the NEA Client or NEA Server (even for a brief time, long enough to disable or misconfigure security settings) is generally equivalent as well. If the NEA Client or NEA Server are dependent on other services for their proper operation (including Posture Collectors, Posture Validators, directories, and patch management services), compromise of those services may result in compromise or failure of the dependent parties. Of course, compromise or failure of NEA Server components is most serious since this would probably affect a large number of NEA Clients while the effects of NEA Client compromise might well be limited to a single machine.

5.2. Countermeasures

The primary countermeasure against attacks by untrusted network intermediaries is the security provided by the PT protocol. Any candidate PT protocols should be carefully examined to ensure that all the threats described above are adequately addressed.

As noted above, compromise or erroneous operation of any of the trusted parties is a serious matter with substantial security implications. This includes the Posture Broker Client, the Posture Broker Server, the Posture Transport Client, and the Posture Transport Server. These are all security-sensitive components so they should be built and managed in accordance with best practices for security devices. This is especially important for the NEA Server and its components since a compromise of this device would affect the security and availability of the entire network (similar to compromise of a AAA server). Communications between the trusted parties must also be secured. For example, if the Posture Broker Server and the Posture Transport Server are separate components, their communications must be secured.

Since the NEA Client may be a mobile device with little physical security (such as a laptop computer or even a public telephone), it should generally be assumed that some proportion of Access NEA Clients will be compromised and therefore hostile. The NEA Server should be designed to be robust against hostile NEA Clients. Once a compromised NEA Client is detected, it can be treated in a manner equivalent to an untrusted party and should pose no greater threat than any other untrusted party.

Countermeasures against a compromised NEA Server (or a component thereof such as a Posture Broker Server or a Posture Transport Server) include prevention of compromise, detection of compromise, and mitigation of the effects of compromise. For prevention, the NEA Server and its components and dependencies should be implemented using secure implementation techniques (e.g., secure coding and minimization) and managed using secure practices (e.g., strong authentication and separation of duty). For detection, the behavior of the NEA Server should be monitored (e.g., via logging especially of remediation instructions, intrusion detection systems, and probes that impersonate a valid NEA Client and record NEA Server behavior) and any anomalies analyzed. For mitigation, NEA Clients should not blindly follow remediation instructions received from a trusted NEA Server. At least for patches and other dangerous actions, they should validate these actions (e.g., via user confirmation) before proceeding. It should not be possible to configure a NEA Client to trust all NEA Servers without proper authentication and authorization.

6. IANA Considerations

Four new IANA registries are defined by this specification: PB-TNC Message Types, PA Subtypes, PB-TNC Remediation Parameters Types, and PB-TNC Error Codes. This section explains how these registries work.

All of these registries support IETF standard values and vendor-defined values. To explain this phenomenon, we will use the PB-TNC Message Type as an example but the other three registries work the same way. Whenever a PB-TNC 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 PB-TNC Message Type is an IETF standard value listed in the IANA registry for PB-TNC Message Types and its meaning is defined in the specification listed for that PB-TNC Message Type in that registry. If the vendor ID is not zero, the meaning of the PB-TNC 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 PB-TNC 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 Broker Clients and Posture Broker Servers MUST NOT require support for particular vendor-specific PB-TNC message types and MUST interoperate with other parties despite any differences in the set of vendor-specific PB-TNC message types supported (although they MAY permit administrators to configure them to require support for specific PB-TNC message types)." Similar requirements are included for PA Subtypes, Remediation Parameters Types, and PB-TNC Error Codes.

6.1. Designated Expert Guidelines

For all of the four 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 [5].

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 the same number of values for its use. The only exception is the registry for PB-TNC Error Codes where 2^16 values are available for the IETF and 2^16 values for each vendor. 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.

6.2. Registry for PB-TNC Message Types

The name for this registry is "PB-TNC 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 2^32-2, and a reference to a specification where the contents of this message type are defined. This specification must define the meaning of this PB-TNC message type and the format and semantics of the PB-TNC Message Value field for PB-TNC messages that include the designated numeric value in the PB-TNC Message Type field and the designated Private Enterprise Number in the PB-TNC Vendor ID field.

Entries to this registry are added by Expert Review with Specification Required, following the guidelines in section 6.1.

The following entries for this registry are defined in this document. They are the initial entries in the registry for PB-TNC Message Types.

PEN	Integer	Name	Defining Specification
0	0	PB-Experimental	RFC 5793
0	1	PB-PA	RFC 5793
0	2	PB-Assessment-Result	RFC 5793
0	3	PB-Access-Recommendation	RFC 5793
0	4	PB-Remediation-Parameters	RFC 5793
0	5	PB-Error	RFC 5793
0	6	PB-Language-Preference	RFC 5793
0	7	PB-Reason-String	RFC 5793
0	0xffffffff	Reserved	RFC 5793

6.3. Registry for PA Subtypes

The name for this registry is "PA Subtypes". Each entry in this registry should include a human-readable name, an SMI Private Enterprise Number, a decimal integer value between 0 and 2^32-2, and a reference to a specification where the contents of this PA subtype are defined. This specification must define the meaning of this PA subtype and the format and semantics of the PA Message Body field for PB-TNC messages that have a PB-TNC Vendor ID of 0, a PB-TNC Message Type of PB-PA, the designated numeric value in the PA Subtype field, and the designated Private Enterprise Number in the PA Message Vendor ID field.

Entries to this registry are added by Expert Review with Specification Required, following the guidelines in section 6.1.

This document does not define any initial entries for this registry. Therefore, this registry should initially be empty. Subsequent RFCs (such as PA-TNC) will define entries in this registry.

6.4. Registry for PB-TNC Remediation Parameters Types

The name for this registry is "PB-TNC Remediation Parameters 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 a specification where the contents of this remediation parameters type are defined. This specification must define the meaning of this remediation parameters type value and the format and semantics of the Remediation Parameters field for PB-TNC messages that have a PB-TNC Vendor ID of 0, a PB-TNC Message Type of PB-Remediation-Parameters, the designated numeric value in the Remediation Parameters Type field, and the designated Private Enterprise Number in the Remediation Parameters Vendor ID field.

Entries to this registry are added by Expert Review with Specification Required, following the guidelines in section 6.1.

The following entries for this registry are defined in this document. They are the initial entries in the registry for PB-TNC Remediation Parameters Types.

PEN	Integer	Name	Defining Specification
0	1	Remediation-URI	RFC 5793
0	2	Remediation-String	RFC 5793

6.5. Registry for PB-TNC Error Codes

The name for this registry is "PB-TNC 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 2^16-1, and a reference to a 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 Parameters field for PB-TNC messages that have a PB-TNC Vendor ID of 0, a PB-TNC Message Type of PB-Error, the designated numeric value in the Error Code field, and the designated Private Enterprise Number in the Error Code Vendor ID field.

Entries to this registry are added by Expert Review with Specification Required, following the guidelines in section 6.1.

The following entries for this registry are defined in this document. They are the initial entries in the registry for PB-TNC Error Codes.

PEN	Integer	Name	Defining Specification
0	0	Unexpected Batch Type	RFC 5793
0	1	Invalid Parameter	RFC 5793
0	2	Local Error	RFC 5793
0	3	Unsupported Mandatory Message	RFC 5793
0	4	Version Not Supported	RFC 5793

7. Acknowledgments

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

The authors of this document 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: Bernard Aboba, Amit Agarwal, Morteza Ansari, Diana Arroyo, Stuart Bailey, Boris Balacheff, Gene Chang, Roger Chickering, Scott Cochrane, Pasi Eronen, Aman Garg, Sandilya Garimella, Lauren Giroux, Mudit Goel, Charles Goldberg, Thomas Hardjono, Chris Hessing, Hidenobu Ito, John Jerrim, Meenakshi Kaushik, Greg Kazmierczak, Scott Kelly, Tom Kelnar, Bryan Kingsford, PJ Kirner, Houcheng Lee, Sung Lee, Lisa Lorenzin, Mahalingam Mani, Paul Mayfield, Michael McDaniels, Bipin Mistry, Rod Murchison, Barbara Nelson, Kazuaki Nimura, Ron Pon, Ivan Pulleyn, Alex Romanyuk, Chris Salter, Mauricio Sanchez, Paul Sangster, Dean Sheffield, Curtis Simonson, Jeff Six, Ned Smith, Michelle Sommerstad, Joseph Tardo, Lee Terrell, Chris Trytten, Brad Upson, Ram Vadali, Guha Prasad Venataraman, John Vollbrecht, Jun Wang, and Han Yin.

8. References

8.1. Normative References

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

[2] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

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

[4] Alvestrand, H., "Content Language Headers", RFC 3282, May 2002.

[5] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

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

8.2. Informative References

[7] Hanna, S., Hurst, R. and R. Sahita, "TNC IF-TNCCS: TLV Binding", Trusted Computing Group, February 2008.

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

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

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

Appendix A. Use Cases

A.1. Initial Client-Triggered Assessment

This scenario involves the assessment of an endpoint initiated during network join. The assessment is triggered by the Posture Broker Client (PBC) and involves collection of patch information from both Standard Operating System (OS) Posture Collector and vendor-specific Patch Posture Collector (PC). The assessment by both the vendor-specific Patch Posture Validator (PV) and Standard OS Posture Validator result in a compliant assessment decision that results in a compliant System Assessment Decision to be returned by the Posture Broker Server (PBS).

```
   +--------+ +-------+ +---------+ +--------+ +-------++--------+
   | Vndr. X| |  Std. | |   Std.  | |  Std.  | | Std.  || Vndr. X|
   |Patch PC| | OS PC | |   PBC   | |  PBS   | | OS PV ||Patch PV|
   +----+---+ +---+---+ +-----+---+ +---+----+ +---+----++---+---+
        |         |   N/W Join|         |          |         |
        |         |     ----->|         |          |         |
        |         | Req Post. |         |          |         |
        |         +<----------+         |          |         |
        |         | Req Post. |         |          |         |
        +<--------------------|         |          |         |
        |Vndr X Patch Posture |         |          |         |
        |-------------------->|         |          |         |
        |         |OS Posture |         |          |         |
        |         |---------->|         |          |         |
        |         |           | Posture |          |         |
        |         |           | Report  |          |         |
        |         |           +-------->|          |         |
        |         |           |         |  Verify  |         |
        |         |           |         |  Posture |         |
        |         |           |         |--------->          |
        |         |           |         |          | Verify  |
        |         |           |         |          | Posture |
        |         |           |         |------------------->|
        |         |           |         | OS Reslt |         |
        |         |           |         |<---------|         |
        |         |           |         | VndrX Patch Result |
        |         |           | Assess  |<-------------------|
        |         |           | Result  |                    |
        |         |           <---------|          |         |
        |         | OS PRslt  |         |          |         |
        |         |<----------|         |          |         |
        | VndrX Patch PResult |         |          |         |
        |<--------------------|         |          |         |
```

A.1.1. Message Contents

This section shows the contents of the key fields in each of the PA messages exchanged in this use case. When necessary, additional commentary is provided to explain why certain fields contain the shown values. Note that many of the flows shown are between components on the same system so no message contents are shown.

A.1.1.1. N/W Join

This flow represents the event that causes the PBC to decide to start an assessment of the endpoint in order to gain access to the network. This is merely an event and doesn’t include a message being sent.

A.1.1.2. Request Posture (Req Post.)

This flow illustrates an invocation of the OS and Patch Posture Collectors requesting particular posture attributes to be sent. Because this use case is triggered locally, NEA doesn’t specify the contents of this flow.

A.1.1.3. Vendor X Patch Posture (VndrX Patch Posture)

This flow contains the PA message from the Vendor X Patch Posture Collector; the message content is described in the PA-TNC specification.

A.1.1.4. OS Posture

This flow contains the PA message from the OS Posture Collector; the message content is described in the PA-TNC specification.

A.1.1.5. Posture Report

This flow contains the PB message containing the PA messages from the Patch and OS Posture Collectors:

```
PB Envelope {

 HDR {

  D bit=0 (Posture Broker Client is originator)

  Batch Type=CDATA

  Batch Length

  }

   PB Message 1 {

    Vendor-id=0

    Type =2 (PB-PA)

    Length

    Value = {

       PA-Msg-vendor-id=0 (Standard)

       PA-subtype=1 (OS)
```

```
          OS Posture PA Message

       }

     }

     PB Message 2 {

       Vendor-id=0

       Type =2 (PB-PA)

       Length

       Value = {

          PA-Msg-vendor-id=1 (Vendor X)

          PA-subtype=1 (Vendor X PA sub-type for patch management)

          Vendor X Patch Posture PA Message

        }

      }

   }
```

A.1.1.6. Verify Posture

This flow illustrates an invocation of the OS and Patch Posture Validators requesting verification of the posture attributes received. Because this flow happens locally within the NEA server, NEA doesn’t specify the message content.

A.1.1.7. OS Posture Result (OS Reslt)

This flow contains the PA message (Posture Assessment Result) from the OS Posture Validator; the message content is described in the PA-TNC specification.

A.1.1.8. Vendor X Patch Posture Result (VndrX Patch Result)

This flow contains the PA message (Posture Assessment Result) from the Vendor X Patch Posture Validator; the message content is described in the PA-TNC specification.

A.1.1.9. Assessment Result (Assess Result)

This flow contains the PB message containing the system assessment result computed by the Posture Broker Server and the PA messages from the Patch and OS Posture Validators:

```
   PB Envelope {

    HDR {

     D bit=1 (Posture Broker Server is originator)

     Batch Type=RESULT

     Batch Length

     }

      PB Message 1 {

       Vendor-id=0,

       Type =3 (Access-Recommendation)

       Length

       Value = {

         System-Evaluation-Result=0 (Compliant)

       }

     }

     PB Message 2 {

       Vendor-id=0,

       Type=2 (PB-PA)

       Length

       Value = {

          PA-Msg-vendor-id=0

          PA-subtype=1 (OS)
          OS Posture Result PA Message

        }

      }

     PB Message 3 {

       Vendor-id=0,

       Type=2 (PB-PA)

       Length

       Value = {

          PA-Msg-vendor-id=1 (Vendor X)

          PA-subtype=1 (Vendor X PA sub-type for patch management)

          Vendor X Patch Posture Result PA Message

        }

      }

   }
```

A.1.1.10. Posture Result (OS PRslt & Vndr X Post PResult)

These flows illustrate an invocation of the OS and Vendor X Patch Posture Collectors to receive the posture assessment results. Because this flow is triggered locally, NEA doesn’t specify the contents of this flow.

A.2. Server-Initiated Assessment with Remediation

This scenario involves the assessment of an endpoint initiated by the NEA server. The assessment is triggered by the Posture Broker Server and involves collection of Anti-Virus attributes for two Anti-Virus components running on the endpoint. The endpoint is assessed to be compliant by one of the vendor (Vendor X) anti-virus posture validators and non-compliant by the other vendor (Vendor Y) anti-virus posture validator. This results in a non-compliant System Assessment Decision to be returned by the Posture Broker Server. The Posture Broker Server also returns remediation instructions for the endpoint as part of the response.

```
   +--------+  +-------+ +---------+ +--------+ +-------+ +--------+
   | Vndr Y |  | Vndr X| |   Std.  | |  Std.  | | Vndr X| | Vndr Y |
   |  AV PC |  | AV PC | |   PBC   | |  PBS   | | AV PV | |  AV PV |
   +----+---+  +---+---+ +-----+---+ +---+----+ +---+---+ +----+---+
        |          |           | N/W Join|          |          |
        |          |           |   ----->|          |          |
        |          |           |         |  Create  |          |
        |          |           |         |Post. Req |          |
        |          |           |         |--------->|          |
        |          |           |         |Create Posture Req   |
        |          |           |         |----------+--------->|
        |          |           |         |Vndr Y AV Posture Req|
        |          |           |         |<---------+----------|
        |          |           |         |Vndr X AV |          |
        |          |           |         |Post. Req |          |
        |          |           | Posture |<---------|          |
        |          |           | Request |          |          |
        |          | Vndr X AV |<--------|          |          |
        |          | Post. Req |         |          |          |
        |          |<----------|         |          |          |
        |      Vndr Y AV       |         |          |          |
        |     Posture Req      |         |          |          |
        +<---------+-----------|         |          |          |
        |  Vndr Y AV Posture   |         |          |          |
        +----------+---------->|         |          |          |
        |          | Vndr X AV |         |          |          |
        |          |  Posture  |         |          |          |
        |          |---------->| Posture |          |          |
        |          |           |Response |          |          |
        |          |           |-------->|          |          |
        |          |           |         |  Verify  |          |
        |          |           |         |  Posture |          |
        |          |           |         |--------->|          |
        |          |           |         |     Verify Posture  |
        |          |           |         |----------+--------->|
        |          |           |         |Vndr Y Posture Result|
        |          |           |         |<---------+----------|
        |          |           |         |Vndr X AV |          |
        |          |           |         |Post Reslt|          |
        |          |           |  Assess |<---------|          |
        |          |           |  Result |          |          |
        |          | Vndr X AV |<--------|          |          |
        |          |Post Reslt |<--------|          |          |
        |          |<----------|         |          |          |
        | Vndr Y AV Post Reslt |         |          |          |
        +<---------+-----------|         |          |          |
        |          |           |         |          |          |
```

A.2.1. Message Contents

This section shows the contents of the key fields in each of the PA messages exchanged in this use case. When necessary, additional commentary is provided to explain why certain fields contain the shown values. Note that many of the flows shown are between components on the same system so no message contents are shown.

A.2.1.1. N/W Join

This flow represents the event that causes the PBS to decide to start an assessment of the endpoint in order to gain access to the network. This is merely an event and doesn’t include a message being sent.

A.2.1.2. Create Posture Request (Create Posture Req)

This flow illustrates an invocation of the Vendor X and Vendor Y Anti-Virus posture validators requesting posture requests to be created. Because this use case is triggered locally, NEA doesn’t specify the contents of this flow.

A.2.1.3. Vendor X Anti-Virus Posture Request (Vndr X AV Post. Req)

This flow contains the PA message (Posture Request) from the Vendor X Anti-Virus Posture Validator; the message content is described in the PA-TNC specification.

A.2.1.4. Vendor Y Anti-Virus Posture Request

This flow contains the PA message (Posture Request) from the Vendor Y Anti-Virus Posture Validator; the message content is described in the PA-TNC specification.

A.2.1.5. Posture Request

This flow contains the PB message containing the PA messages from the Vendor X and Vendor Y Anti-Virus Posture Validators:

```
PB Envelope {

 HDR {

  D bit=1 (Posture Broker Server is originator)

  Batch Type=SDATA

  Batch Length
```

```
    }
     PB Message 1 {
       Vendor-id=0
       Type =2 (PB-PA)
       Length
       Value = {
          PA-Msg-vendor-id=1 (Vendor X)
          PA-subtype=2 (Vendor X PA sub-type for Anti-Virus)
          Vendor X AV Posture Request PA Message
       }
     }
     PB Message 2 {
       Vendor-id=0
       Type =2 (PB-PA)
       Length
       Value = {
          PA-Msg-vendor-id=2 (Vendor Y)
          PA-subtype=1 (Vendor Y PA sub-type for Anti-Virus)
          Vendor Y AV Posture Request PA Message
        }
      }
   }
```

A.2.1.6. Process Posture Request (Vndr X AV Post Req & Vndr Y AV Posture Req)

This flow illustrates an invocation of the Vendor X and Vendor Y Anti-Virus Posture Collectors to process the Posture Request and return particular posture attributes requested. Because this use case is triggered locally, NEA doesn’t specify the contents of this flow.

A.2.1.7. Vendor Y Anti-Virus Posture (Vndr Y AV Posture)

This flow contains the PA message (response to the Posture Request) from the Vendor Y Anti-Virus Posture Collector; the message content is described in the PA-TNC specification.

A.2.1.8. Vendor X Anti-Virus Posture (Vndr X AV Posture)

This flow contains the PA message (response to the Posture Request) from the Vendor X Anti-Virus Posture Collector; the message content is described in the PA-TNC specification.

A.2.1.9. Posture Response

This flow contains the PB message containing the PA messages from the Vendor X and Vendor Y Anti-Virus Posture Collectors:

```
PB Envelope {

 HDR {

  D bit=0 (Posture Broker Client is originator)

  Batch Type=CDATA

  Batch Length

 }

  PB Message 1 {

    Vendor-id=0

    Type =2 (PB-PA)

    Length

    Value = {
        PA-Msg-vendor-id=1 (Vendor X)

        PA-subtype=2 (Vendor X PA sub-type for Anti-Virus)

        Vendor X AV Posture PA Message

    }

  }

  PB Message 2 {

    Vendor-id=0

    Type =2 (PB-PA)

    Length

    Value = {

        PA-Msg-vendor-id=2 (Vendor Y)

        PA-subtype=1 (Vendor Y PA sub-type for Anti-Virus)

        Vendor Y AV Posture PA Message

     }

   }

}
```

A.2.1.10. Verify Posture

This flow illustrates an invocation of the Vendor X and Vendor Y Anti-Virus Posture Validators requesting verification of the posture attributes received. Because this flow happens locally within the NEA server, NEA doesn’t specify the message contents.

A.2.1.11. Vendor Y Anti-Virus Posture Result (Vndr Y AV Post Result)

This flow contains the PA message (Posture Assessment Result) from the Vendor Y Anti-Virus Posture Validator; the message content is described in the PA-TNC specification.

A.2.1.12. Vendor X Anti-Virus Posture Result (Vndr Y AV Post Result)

This flow contains the PA message (Posture Assessment Result) from the Vendor X Anti-Virus Posture Validator; the message content is described in the PA-TNC specification.

A.2.1.13. Assessment Result (Assess Result)

This flow contains the PB message containing the system assessment result computed by the Posture Broker Server and the PA messages from the Patch and OS Posture Validators:

```
PB Envelope {

 HDR {

  D bit=1 (Posture Broker Server is originator)

  Batch Type=RESULT

  Batch Length

 }

  PB Message 1 {

    Vendor-id=0,

    Type=3 (Access-Recommendation)

    Length

    Value = {

      PB-Assessment-Result=1 (Non-Compliant)

    }

  }

  PB Message 2 {

    Vendor-id=0,

    Type=4 (Remediation-Parameters)

    Length
```

```
       Value = {
        Remediation-Param-Vendor-ID=0
        Remediation-Param-Type=1 (Remediation-URI)
        Remediation-Param=’’http://xyz’’
        }
      }
    PB Message 3 {
       Vendor-id=0,
       Type=4 (Remediation-Parameters)
       Length
       Value = {
        Remediation-Param-Vendor-ID=0
        Remediation-Param-Type=2 (Remediation-String)
        Remediation-Param=’’Try Step1, Step2,...’’
        }
      }
     PB Message 4 {
       Vendor-id=0,
       Type=2 (PB-PA)
       Length
       Value = {
          PA-Msg-vendor-id=1 (Vendor X)
          PA-subtype=2 (Vendor X PA sub-type for Anti-Virus)
          Vendor X AV Posture Result PA Message
        }

      }

     PB Message 5 {

       Vendor-id=0,

       Type=2 (PB-PA)

       Length

       Value = {

           PA-Msg-vendor-id=2 (Vendor Y)

           PA-subtype=1 (Vendor Y PA sub-type for Anti-Virus)

           Vendor Y AV Posture Result PA Message

        }

      }

   }
```

A.2.1.14. Posture Result (Vndr X AV Post Reslt & Vndr Y AV Post Reslt)

These flows illustrate an invocation of the Vendor X and Vendor Y Anti-Virus Posture Collectors to receive the posture assessment results. Because this flow is triggered locally, NEA doesn’t specify the contents of this flow.

A.3. Client-Triggered Reassessment

This scenario involves the reassessment of an endpoint as a result of enabling a software component on the endpoint. The endpoint has two VPN client software components, one from vendor X for the user’s home network and other from vendor Y for the network that the endpoint is currently accessing. The assessment is triggered when the user tries to use the Vendor X VPN client; this is a violation of the posture policy. The Posture Broker Client triggers the posture assessment when it receives a notification from the Standard VPN Posture Collector about the change to the operational state of the VPN component on the endpoint. Note that the VPN Posture Collector supports standard attributes and some vendor-defined attributes from vendor X’s and vendor Y’s namespaces. This use case doesn’t leverage vendor-defined attributes. The assessment involves verification of the standard VPN posture attributes by the Standard VPN Posture Validator that results in a non-compliant assessment result. This use case relies on the use of a virtual Posture Collector concept described in section 3.3 of the PA-TNC specification. As illustrated in this example, the Posture Broker Client will assign two Posture Collector IDs to a single Posture Collector (Standard VPN PC), and the Posture Collector will generate two separate PA messages to report the posture for Vendor X and Vendor Y VPN Clients. The Posture Broker Client will use the assigned IDs in the PB message sent to the NEA Server. This entire behavior will be completely opaque to the NEA Server, which will handle the PB message as if there were two VPN Posture Collectors on the NEA Client.

```
   +--------+  +-------+ +---------+ +--------+ +--------+ +--------+

   |Vndr X  |  |Vndr Y | |Standard | |Standard| |Standard| |Standard|

   |VPNClnt |  |VPNClnt| | VPN PC  | |  PBC   | |   PBS  | | VPN PV |

   +----+---+  +---+---+ +-----+---+ +---+----+ +---+----+ +----+---+
   Enble|          |           |         |          |           |

   ---->|          |           |         |          |           |

        |  VPN Status Change   |         |          |           |

        |--------------------->| Posture |          |           |

        |          |           | Change  |          |           |

        |          |           |-------->|          |           |

        |          |           |Req. Post|          |           |

        |          |           |<--------|          |           |

        |          |Ins/Rq Info|         |          |           |

        |          |<----------|         |          |           |

        | Inspect/Request Info |         |          |           |

        |<---------+-----------|VPNX Post|          |           |

        |          |           |-------->|          |           |

        |          |           |VPNY Post|          |           |
        |          |           |-------->|          |           |
        |          |           |         | Posture  |           |
        |          |           |         |  Report  |           |
        |          |           |         |--------->|           |
        |          |           |         |          |Vrfy Post. |
        |          |           |         |          |---------->|
        |          |           |         |          |VPN PRslt  |
        |          |           |         |  Assess  |<----------|
        |          |           |         |  Result  |           |
        |          |           |         |<---------|           |
        |          |           |VPN PRslt|          |           |
        |          |           |<--------|          |           |
```

A.3.1. Message Contents

This section shows the contents of the key fields in each of the PA messages exchanged in this use case. When necessary, additional commentary is provided to explain why certain fields contain the shown values. Note that many of the flows shown are between components on the same system so no message contents are shown.

A.3.1.1. Enable VPN Client (Enble)

This flow represents the end user triggered event of starting the VPN Client software from Vendor X. This is merely an event and doesn’t include a message being sent.

A.3.1.2. Notify Status Change (VPN Status Change)

This flow represents the detection of the active state of the Vendor X VPN Client software by the Standard VPN Posture Collector. This is merely an event and doesn’t include a message being sent.

A.3.1.3. Notify Posture Change (Posture Change)

This flow represents the notification of the VPN Posture change sent from the VPN Posture Collector to the Standard Posture Broker Client. This is merely an event and doesn’t include a message being sent.

A.3.1.4. Request Posture (Req. Post)

This flow illustrates an invocation of the VPN Posture Collector requesting particular posture attributes to be sent. Because this use case is triggered locally, the contents of this flow aren’t specified by NEA.

A.3.1.5. Inspect/Request Information (Ins/Rq Info)

This flow illustrates the acquisition of the posture attributes by the Standard VPN Posture Collector from the Vendor X and Vendor Y VPN Client components. Because this flow is triggered locally, NEA doesn’t specify the message contents.

A.3.1.6. Vendor X VPN Posture (VPNX Post.)

This flow contains the PA message from the VPN Posture Collector for Vendor X VPN Client posture; the message content is described in the PA-TNC specification.

A.3.1.7. Vendor Y VPN Posture (VPNY Post.)

This flow contains the PA message from the VPN Posture Collector for Vendor Y VPN Client posture; the message content is described in the PA-TNC specification.

A.3.1.8. Posture Report (Post. Rpt.)

This flow contains the PB message containing the PA message from the VPN Posture Collector:

```
PB Envelope {

 HDR {

  D bit=0 (Posture Broker Client is originator)

  Batch Type=CRETRY

  Batch Length

 }
```

```
     PB Message 1 {

       Vendor-id=0

       Type =2 (PB-PA)

       Length

       Value = {

          PA-Msg-vendor-id=0

          PA-subtype=7 (VPN)

          Posture-Collector-ID=1 //Virtual Posture Collector ID for
   Vendor X VPN Client

          Vendor X VPN Posture PA Message

       }

     }

     PB Message 2 {

       Vendor-id=0

       Type =2 (PB-PA)

       Length

       Value = {

          PA-Msg-vendor-id=0

          PA-subtype=7 (VPN)

          Posture-Collector-ID=2 //Virtual Posture Collector ID for
   Vendor Y VPN Client

          Vendor Y VPN Posture PA Message

       }

     }
```

A.3.1.9. Verify Posture (Vrfy Post.)

This flow illustrates an invocation of the VPN Posture Validator requesting verification of the posture attributes received. Because this flow happens locally within the NEA server, NEA doesn’t specify the message contents.

A.3.1.10. VPN Posture Result (VPN PRslt)

This flow contains the PA message (Posture Assessment Result) from the VPN Posture Validator; the message content is described in the PA-TNC specification.

A.3.1.11. Assessment Result (Assess Result)

This flow contains the PB message containing the system assessment result computed by the Posture Broker Server and the PA messages from the VPN Posture Validator:

```
 PB Envelope {

   HDR {

    D bit=1 (Posture Broker Server is originator)

    Batch Type=RESULT

    Batch Length

   }


  PB Message 1 {

    Vendor-id=0,

    Type =3 (Access-Recommendation)

    Length

    Value = {

      PB-Assessment-Result=1 (Non-Compliant)

    }

  }
```

```
     PB Message 2 {

       Vendor-id=0,

       Type=2 (PB-PA)

       Length

       Value = {

          PA-Msg-vendor-id=0

          PA-subtype=7 (VPN)

          VPN Posture Result PA Message

        }

      }
```

A.3.1.12. Posture Result (VPN PRslt)

This flow illustrate an invocation of the VPN Posture Collectors to receive the posture assessment result. Because this flow is triggered locally, NEA doesn’t specify the contents of this flow.

Appendix B. Evaluation against NEA Requirements

This section evaluates the PB-TNC protocol against the requirements defined in the NEA Requirements document. Each subsection considers a separate requirement from the NEA Requirements document. Only common requirements (C-1 through C-11) and PB requirements (PB-1 through PB-6) are considered, since these are the only ones that apply to PB.

B.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.

PB-TNC meets this requirement. It allows an unlimited number of round trips between the NEA Client and NEA Server.

B.2. Evaluation against Requirement 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.

PB-TNC meets this requirement. Either the NEA Client or the NEA Server can initiate a posture assessment or reassessment.

There is one limitation on this support. If a NEA Server wishes to initiate a reassessment after it has sent a RESULT batch, it must close the underlying transport session and initiate a new assessment. For half-duplex transports, this is unavoidable unless a constant exchange of messages is maintained, which would be very wasteful. For full-duplex transports, it would be possible to allow the Posture Broker Server to send an SRETRY batch even in the Decided state. If the NEA working group reaches consensus that this change should be made, it will be.

B.3. Evaluation against Requirement 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.

PB-TNC does not include any security capabilities. It depends on PT to supply a secure transport. This addresses all the necessary threats without adding an extra layer of security. Since this requirement only applies to NEA protocols that include security capabilities, PB-TNC meets this requirement.

B.4. Evaluation against Requirement 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).

PB-TNC meets this requirement. PB-TNC can operate over any PT protocol that meets the requirements for PT stated in the NEA Requirements document. Also, PB-TNC insulates the PA protocol from any specifics of the PT protocol. With PB-TNC, all PT protocols are equivalent from the perspective of the PA protocol.

B.5. Evaluation against Requirement 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, PB-TNC should receive a strong preference. PB-TNC is equivalent with IF-TNCCS 2.0, an open TCG specification. IF-TNCCS 2.0 is an extension of the existing IF-TNCCS 1.X protocols, which have been implemented by dozens of vendors and open source projects.

B.6. Evaluation against Requirement 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.

PB-TNC meets this requirement. PB-TNC supports up to 2^16-1 Posture Collectors and an equal number of Posture Validators in a given PB-TNC session. It also supports an unlimited number of NEA Clients and NEA Servers.

The scalability of PB-TNC extends into other areas as well. For example, PB-TNC supports an unlimited number of batches and each batch can contain up to 2^32-1 octets and about 2^24 PA messages. Each PA message can contain up to 2^32-1 octets. Of course, sending this much data in a NEA assessment is not generally advisable, but the point is that PB-TNC is highly scalable.

B.7. Evaluation against Requirement 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.

PB-TNC meets this requirement. Each PB-TNC batch can contain about 2^24 PA messages. Since PB-TNC supports an unlimited number of batches in a session, this number is actually unlimited (except perhaps by PT protocols, user patience, or other external factors). As for efficiency, PB-TNC adds only 24 octets of overhead per PA message. PA-TNC can include many attributes in a single PA message so this overhead is diluted further.

B.8. Evaluation against Requirement C-8

Requirement C-8 says:

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

PB-TNC meets this requirement. A minimal PB-TNC exchange can be as small as 72 octets and one round trip. Even if privacy policies or other factors require multiple round trips, PB-TNC generally imposes an overhead of only 8 octets per batch and 24 octets per PA message.

B.9. Evaluation against Requirement 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.

PB-TNC meets this requirement. It defines a standard way for the NEA Client and NEA Server to send their language preferences to each other, leveraging the widely implemented Accept-Language format defined in RFC 3282.

B.10. Evaluation against Requirement C-10

Requirement C-10 says:

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

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

B.11. Evaluation against Requirement 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 round trips, 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.

PB-TNC meets this requirement. The PB-TNC protocol is designed to be flexible enough to operate with a variety of underlying PT protocols, including those that may have limitations on message or assessment size, number of round trips, and duplex. Local APIs can allow Posture Collectors and Posture Validators to discover when they are operating in a less constrained deployment and then make use of more verbose attributes. Similarly, Posture Collectors could choose not to send or use smaller attributes (including assertions from previous assessments) when faced with a very constrained network connection.

B.12. Evaluation against Requirement PB-1

Requirement PB-1 says:

PB-1 The PB protocol MUST be capable of carrying attributes from the Posture Broker Server to the Posture Broker Client. This enables the Posture Broker Client to learn the posture assessment decision and if appropriate to aid in remediation and notification of the endpoint owner.

PB-TNC meets this requirement. It can carry attributes from the Posture Broker Client to the Posture Broker Server and back in an unlimited number of round trips. Furthermore, PB-TNC provides explicit attribute support for posture decision and remediation aid notification.

B.13. Evaluation against Requirement PB-2

Requirement PB-2 says:

PB-2 The PB protocol MUST NOT interpret the contents of PA messages being carried; i.e., the data it is carrying must be opaque to it.

PB-TNC meets this requirement. It does not parse or interpret PA messages in any way.

B.14. Evaluation against Requirement PB-3

Requirement PB-3 says:

PB-3 The PB protocol MUST carry unique identifiers that are used by the Posture Brokers to route (deliver) PA messages between Posture Collectors and Posture Validators. Such message routing should facilitate dynamic registration or deregistration of Posture Collectors and Validators. For example, a dynamically registered anti-virus Posture Validator should be able to subscribe to receive messages from its respective anti-virus Posture Collector on NEA Clients.

PB-TNC meets this requirement. PB-TNC tags each PA message with a PA subtype that the Posture Brokers can use to deliver the PA messages to the proper Posture Collectors and Posture Validators. By tagging messages according to their content, PB-TNC allows Posture Collectors and Posture Validators to be dynamically registered and deregistered, ensuring that each one receives the proper data. PB-TNC also supports exclusive delivery, which allows messages to be targeted at a particular Posture Collector or Posture Validator.

B.15. Evaluation against Requirement PB-4

Requirement PB-4 says:

PB-4 The PB protocol MUST be capable of supporting a half-duplex PT protocol. However, this does not preclude PB from operating full-duplex when running over a full-duplex PT.

PB-TNC meets this requirement. In order to insulate PA from any differences between half-duplex and full-duplex PT protocols, PB-TNC always operates in a half-duplex mode, regardless of the capabilities of the PT protocol. While this could in theory slow assessments that require many round trips or bidirectional multimedia exchanges, this is not a problem in practice because endpoint assessments do not typically involve multimedia or a large number of round trips.

B.16. Evaluation against Requirement PB-5

Requirement PB-5 says:

PB-5 The PB protocol MAY support authentication, integrity, and confidentiality protection for the attribute messages it carries between a Posture Broker Client and Posture Broker Server. This provides security protection for a message dialog of the groupings of attribute messages exchanged between the Posture Broker Client and Posture Broker Server. Such protection is orthogonal to PA protections (which are end to end) and allows for simpler Posture Collector and Validators to be implemented, and for consolidation of cryptographic operations possibly improving scalability and manageability.

PB-TNC does not address this optional requirement. It leaves security to PT (which is required to address it) and PA (which SHOULD do so). There seems to be minimal benefit in adding a third layer of security to the NEA protocol stack. However, if the NEA working group determines that PB should include support for authentication, integrity protection, and confidentiality protection, then this could be added to PB in a similar manner to the way that the PA-TNC security is done.

B.17. Evaluation against Requirement PB-6

Requirement PB-6 says:

PB-6 The PB protocol MUST support grouping of attribute messages to optimize transport of messages and minimize round trips.

PB-TNC meets this requirement. Multiple attribute messages can be conveyed in a single PA message. In fact, that’s how PA-TNC works.

Authors’ Addresses

Ravi Sahita
Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95054 USA
EMail: Ravi.Sahita@intel.com

Steve Hanna
Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, CA 94089 USA
EMail: shanna@juniper.net

Ryan Hurst
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052 USA
EMail: Ryan.Hurst@microsoft.com

Kaushik Narayan
Cisco Systems Inc.
10 West Tasman Drive
San Jose, CA 95134 USA
EMail: kaushik@cisco.com