SACM
Internet-Draft
Intended status: Standards Track
Expires: December 30, 2017

C. Schmidt
D. Haynes
C. Coffin
The MITRE Corporation
D. Waltermire
National Institute of Standards and Technology
J. Fitzgerald-McKay
Department of Defense

June 28, 2017

Software Inventory Message and Attributes (SWIMA) for PA-TNC
draft-ietf-sacm-nea-swima-patnc-00

Abstract

This document extends the PA-TNC specification [RFC5792] by providing specific attributes and message exchanges to allow endpoints to report their installed software inventory information to a NEA server (as described in [RFC5209]).

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 30, 2017.

Copyright Notice

Table of Contents

1. Introduction

Knowing the list of the software installed on endpoints is useful to understand and maintain the security state of a network. For example, if an enterprise policy requires the presence of certain software and prohibits the presence of other software, reported software installation information can be used to indicate compliance and non-compliance with these requirements. When reported endpoint software installation inventory lists, shortened to "software inventories" for the remainder of this document, contain patch level data for identified software, further comparison to vulnerability or threat alerts can be used to determine an endpoint's exposure to attack. These are some of the highly useful management use cases supported by software inventory data provided by this specification.

There is a need for a standardized method for exchanging software inventory data that includes a unique software identifier associated with a specific version of a software product. In some cases, it may also be necessary to convey observable installation information and characterization information for a software product including metadata about the product beyond its identifier. These "Messages and Attributes" enable software identification, installation, and characterization information to be provided for any software installed on any endpoint that supports this specification. Such information can come from multiple sources, including tag files (such as ISO SWID tags [SWID15]), reports from third party inventory tools, output from package managers, and other sources.

This specification is designed to only report software that is installed on an endpoint. In particular, it does not monitor or report information about what software is running on the endpoint. Likewise, it is not intended to report individual files, libraries, installation packages, or similar artifacts. While all of this information has its uses, this information requires different metadata and monitoring methods. As a result, this specification focuses solely on software inventory information, leaving reporting of other classes of endpoint information to other specifications.

This specification defines a new set of PA-TNC attributes, which are used to communicate requests for software inventory information and software installation change events. The exchange of these messages allows software inventory information to be sent to a NEA Server, which can make this information available to other applications.

1.1. Network Endpoint Assessment (NEA)

The Network Endpoint Assessment (NEA) Working Group defines an open solution architecture that enables network operators to collect and utilize information about endpoint configuration and state. This information can be used to enforce policies, monitor endpoint health, and for many other activities. Information about the software present on an endpoint is an important consideration for such activities. The attributes defined in this document are used to communicate software inventory evidence, collected from a range of possible sources, from the posture collector on the endpoint to the posture validator on a NEA Server using the PA-TNC interface, as shown in Figure 1 below.

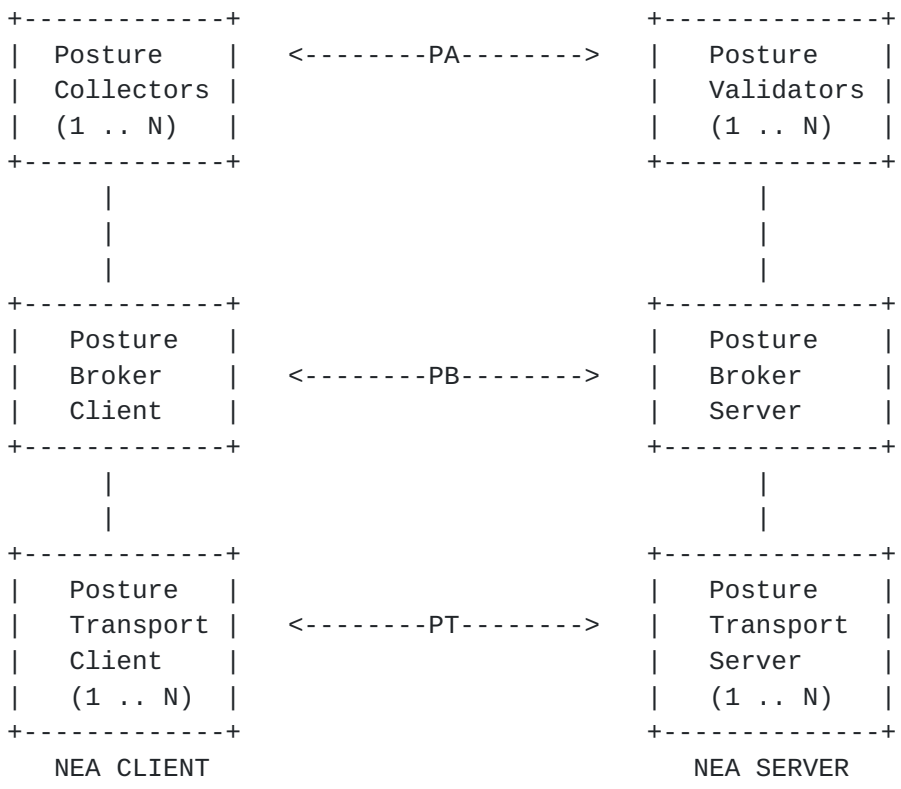


Figure 1: NEA Reference Model

To better understand this specification, the reader should review the NEA reference architecture as described in the Network Endpoint Assessment (NEA): Overview and Requirements [RFC5209]. The reader should also review the PA-TNC interfaces as defined in RFC 5792 [RFC5792].

This document is based on standards published by the Trusted Computing Group's Trusted Network Communications (TNC) workgroup. The TNC and NEA architectures are interoperable and the following components are equivalent:

TNC Component	NEA Component
Integrity Measurement Verifier (IMV)	Posture Validator
Integrity Measurement Collector (IMC)	Posture Collector
TNC Server (TNCS)	Posture Broker Server
TNC Client (TNCC)	Posture Broker Client

Table 1: Equivalent components in TNC and NEA architectures

1.2. Keywords

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in RFC 2119 [RFC2119]. This specification does not distinguish blocks of informative comments and normative requirements. Therefore, for the sake of clarity, note that lower case instances of must, should, etc. do not indicate normative requirements.

1.3. Definitions

This section defines terms with special meaning within this document.

SW-PC - A Posture Collector (PC) that conforms to this specification. Note that such a posture collector might also support other PA-TNC exchanges beyond those defined herein.

SW-PV - A Posture Validator (PV) that conforms to this specification. Note that such a posture verifier might also support other PA-TNC exchanges beyond those defined herein.

SW Attribute - This is a PA-TNC attribute (as defined in RFC 5792 [RFC5792] extension for conveying software inventory information. This specification defines several new attribute types.

Endpoint's Software Inventory Evidence Collection - The set of information regarding the set of software installed on an endpoint. An endpoint's software inventory evidence collection might include information created by or derived from multiple sources, including but not limited to SWID tag files deposited on the file system during software installation, information generated to report output from software discovery tools, and information dynamically generated by a software or package management system on an endpoint.

Software Inventory Evidence Record - The endpoint's Software Inventory Evidence Collection is composed of "records". Each record corresponds to one installed instance of a particular software product as reported by some data source. It is possible for a single installed instance to have multiple software inventory evidence records in an endpoint's Software Inventory Evidence Collection - this can happen if multiple sources all report the same software installation instance.

Software Identifier - A string associated with a specific version of a specific software product. Each supported SWIMA data model has its own rules for how a software identifier (see Section 3.4.1) is derived from the representation of the given software product using that data model, and different sources for this information might populate relevant information differently. As such, while each software identifier uniquely identifies a specific software product, the same software product might be associated with multiple software identifiers reflecting differences between different data sources and supported data models.

2. Background

2.1. Supported Use Cases

This section describes the use cases supported by this specification. The primary use of exchanging software inventory information over the PA-TNC interface is to enable a challenger (e.g. NEA Server) to obtain inventory evidence about some system in a way that conforms to NEA procedures and expressed using a standard format. Collected software information can support a range of security activities including determining whether an endpoint is permitted to connect to the enterprise, determining which endpoints contain software that requires patching, and similar activities.

2.1.1. Use Software Inventory as an Access Control Factor

Some enterprises might define security policies that require connected endpoints to have certain pieces of security software installed. By contrast, some security policies might prevent access to resources by endpoints that have certain prohibited pieces of software installed, since such applications might pose a security risk. To support such policies, the NEA Server needs to collect software inventory evidence from an endpoint that is seeking to initiate or continue connectivity to the enterprise resource.

Based on this specification, the SW-PC can provide a complete or partial inventory to the SW-PV as required to determine policy

compliance. The SW-PV can then use this as evidence of compliance or non-compliance to make a policy-based access decision.

2.1.2. Central Stores of Up-to-Date Endpoint Software Inventory Data

Many tools use information about an endpoint's software inventory to monitor and enforce the security of a network. For example, a software patching tool needs to determine if there is out-of-date software installed that needs to be updated. A vulnerability management tool needs to identify endpoints with known vulnerable software installed (patched or otherwise) to gauge an endpoint's relative exposure to attack. A license management tool needs to verify that all installed software within the enterprise is accounted for. A central repository representing an up-to-date understanding of each endpoint's software inventory facilitates these activities. Multiple tools can share such a repository ensuring that software inventory information is collected more frequently and efficiently, leading to a more complete and consistent understanding of installed software state as compared to each tool collecting the same inventory information from endpoints individually.

This specification supports these activities through a number of mechanisms. As noted above, a SW-PC can provide a complete list of software present in an endpoint's Software Inventory Evidence Collection to the SW-PV, which can then pass this information on to a central repository, such as a Configuration Management Database (CMDB) or similar application. In addition, SW-PCs are required to be able to monitor for changes to an endpoint's Software Inventory Evidence Collection in near real-time and immediately push reports of detected changes to the SW-PV. Thus any central repository fed by a SW-PV receiving inventory information can be updated quickly after a change occurs. Keeping a central repository synchronized with current software inventory information in this way allows tools to make efficient decisions based on up-to-date, consistent information.

2.1.3. PA-TNC Use Cases

Software Inventory Message and Attributes for PA-TNC are intended to operate over the PA-TNC interface and, as such, are intended to meet the use cases set out in the PA-TNC specification.

2.2. Non-supported Use Cases

Some use cases not covered by this specification include:

- Addressing how the endpoint's Software Inventory Evidence Collection is populated. In particular, NEA components are not expected to perform software discovery activities beyond compiling

information in an endpoint's Software Inventory Evidence Collection. This collection might come from multiple sources on the endpoint (e.g., information generated dynamically by package management tools or discovery tools, as well as SWID tag files discovered on the file system). While an enterprise might make use of software discovery capabilities to identify installed software, such capabilities are outside the scope of this specification.

- Converting inventory information expressed in a proprietary format into formats used in the attributes described in this specification. Instead, this specification focuses exclusively on defining interfaces for the transportation of software information expecting that reporting tools will converge around some set of standardized formats for this information.

- Mechanisms for a posture validator to request a specific list of software information based on arbitrary software properties. For example, requesting only information about software from a particular vendor is not supported. After the endpoint's Software Inventory Evidence Collection has been copied to some central location, such as the CMDB, processes there can perform queries based on any criteria present in the collected information, but this specification does not address using such queries to constrain the initial collection of this information from the endpoint.

- Use of properties of certain sources of software information that might facilitate local tests (i.e., on the endpoint) of endpoint state. For example, the optional package_footprint field of an ISO SWID tag can contain a list of files and hash values associated with the software indicated by the tag. Tools on the endpoint can use the values in this field to test for the presence of the indicated files. Successful evaluation of such tests leads to greater assurance that the indicated software is present on the endpoint. Currently, most SWID tag creators do not provide values for tag fields that support local testing. For this reason, the added complexity of supporting endpoint testing using these fields is out of scope for this specification, but may be considered in a future version.

2.3. Specification Requirements

Below are the requirements that the Software Inventory Message and Attributes for PA-TNC specification is required to meet in order to successfully play its role in the NEA architecture.

Efficient: The NEA architecture enables delay of network access until the endpoint is determined not to pose a security threat to the network based on its asserted integrity information. To minimize user frustration, the Software Inventory Message and Attributes for PA-TNC ought to minimize overhead delays and make PA-TNC communications as rapid and efficient as possible.

Efficiency is also important when one considers that some network endpoints are small and low powered, some networks are low bandwidth and/or high latency, and some transport protocols (such as PT-EAP, Posture Transport (PT) Protocol for Extensible Authentication Protocol (EAP) Tunnel Methods [RFC7171]) or their underlying carrier protocol might allow only one packet in flight at a time or only one roundtrip. However, when the underlying PT protocol imposes fewer constraints on communications, this protocol ought to be capable of taking advantage of more robust communication channels (e.g. using larger messages or multiple roundtrips).

Scalable: Software Inventory Message and Attributes for PA-TNC needs to be usable in enterprises that contain tens of thousands of endpoints or more. As such, it needs to allow a security tools to make decisions based on up-to-date information about an endpoint's software inventory without creating an excessive burden on the enterprise's network.

Interoperable: This specification defines the protocol for how PCs and PVs can exchange and use software information to provide a NEA Server with information about an endpoint's software inventory. Therefore, a key goal for this specification is ensuring that all SW PCs and PVs, regardless of the vendor who created them, are able to interoperate in their performance of these duties.

Support precise and complete historical reporting: This specification outlines capabilities that support real-time understanding of the state of endpoint in a network in a way that can be used by other tools. One means of facilitating such an outcome is for a CMDB to be able to contain information about all endpoints connected to the enterprise for all points in time between the endpoint's first connection and the present. In such a scenario, it is necessary that any PC be able to report any changes to its software inventory evidence collection in near real-time while connected and, upon reconnection to the enterprise, be able to update the NEA Server (and through it the CMDB) with regard to the state of its software inventory evidence collection throughout the entire interval when it was not connected.

2.4. Non-Requirements

There are certain requirements that the Software Inventory Message and Attributes for PA-TNC specification explicitly is not required to meet. This list is not exhaustive.

End to End Confidentiality: This specification does not define a mechanism for confidentiality, nor is this property automatically provided by using the PA-TNC interface. In the NEA architecture, confidentiality is generally provided by the underlying transport protocols, such as the PT Binding to TLS [RFC6876] or PT-EAP Posture Transport for Tunneled EAP Methods [RFC7171] - see Section 9 for more information on related standards. Should users wish to protect the confidentiality of assessment instructions or results, then an appropriate transport needs to be used.

2.5. Assumptions

The Posture Broker Client and Posture Broker Server are assumed to provide reliable delivery for PA-TNC messages and attributes sent between the SW-PCs and the SW-PVs. In the event that reliable delivery cannot be provided, the Posture Collector or Posture Validator is expected to terminate the connection.

2.6. Non-Assumptions

This specification explicitly does not assume that software inventory information exchanges reflect the software installation state of the endpoint. This specification does not attempt to detect when the endpoint is providing false information, either through malice or error, but instead focuses on correctly and reliably providing the reported Software Inventory Evidence Collection to the NEA Server. Similarly, this specification makes no assumption about the completeness of the Software Inventory Evidence Collection's coverage of the total set of software installed on the endpoint. It is possible, and even likely, that some installed software is not represented by a record in an endpoints Software Inventory Evidence Collection. See Section 7 for more on this security consideration.

3. System Requirements

The Software Inventory Message and Attributes for PA-TNC specification facilitates the exchange of software inventory and event information. Specifically, each application supporting Software Inventory Message and Attributes for PA-TNC includes a component known as the SW-PC that receives messages sent with the SW Attributes component type. The SW-PC is also responsible for sending appropriate SW Attributes back to the SW-PV in response. This

section outlines what software inventories and events are and the requirements on SW-PCs and SW-PVs in order to support the stated use cases of this specification.

3.1. Data Sources

The records in an endpoint's software inventory evidence collection come from one or more "sources". A source represents one collection of software inventory information about the endpoint. Examples of sources include, but are not limited to, ISO SWID tags deposited on the filesystem and collected therefrom, information derived from package managers (e.g., RPM or YUM), and the output of software inventory scanning tools.

There is no expectation that any one source of inventory information will have either perfect or complete software inventory information. For this reason, this specification supports the simultaneous use of multiple sources of software inventory information. Each source might have its own "sphere of expertise" and report the software within that sphere. For example, a package manager would have excellent understanding of the software that it managed, but would not necessarily have any information about software installed via other means.

A SW-PC is not required to utilize every possible source of software information on its endpoint. Some SW-PCs might be explicitly tied only to one or a handful of software inventory sources, or it could be designed to dynamically accommodate new sources. For all software inventory evidence sources that a particular SW-PC supports, it MUST completely support all requirements of this specification with regard to those sources. A potential source that cannot support some set of required functionality (e.g. it is unable to monitor the software it reports for change events, as discussed in Section 3.6) MUST NOT be used as a source of endpoint software inventory information, even if it could provide some information. In other words, a source either supports full functionality as described in this specification, or it cannot be used at all.

When sending information about installed software the SW-PC MUST include the complete set of relevant data from all supported sources of software inventory evidence. In other words, sources need to be used consistently. This is because, if a particular source is included in an initial inventory, but excluded from a later inventory, the SW-PV receiving this information might reasonably conclude that the software reported by that source was no longer installed on the endpoint. As such, it is important that all supported sources be used every time the SW-PC provides information to a SW-PV.

Note that, if a SW-PC collects data from multiple sources, it is possible that some software products might be "double counted". This can happen if both sources of inventory evidence provide a record for a single installation of a software product. When a SW-PC reports information or records events from multiple inventory evidence sources, it MUST use the information those sources provide, rather than attempting to perform some form of reduction. In other words, if multiple sources report records corresponding to a single installation of a software product, all such records from each source are required to be part of the SW- PC's processing even if this might lead to multiple reporting, and the SW-PC is not to ignore some records to avoid such multiple reporting.

All inventory records reported by a SW-PC include a Source Identifier linking them to a particular source. Source Identifiers are discussed more in Section 3.4.5.

3.2. Data Models

Software Inventory Message and Attributes supports an open set of data models by which installed software instances can be described. This reflects the fact that, as of the writing of this specification, no one software description data model has come to dominate use, and new models are being developed regularly. As a result, information available to SW-PCs might come in a variety of formats, and a SW-PC could have little control over the format of the data it is able to collect.

The format of the data model used to convey software inventory information had very little impact on the behavior of the components defined in this specification. The SW-PV has no dependency on the data model conveyed in SWIMA messages. For this reason, it MUST NOT reject a record or respond with a PA-TNC Error due to the data model used in attributes it receives. (Note that the SW-PV might serve as the front-end of other functionality that does have a dependency on the data model used to express software information, but any such dependency is beyond the scope of this specification and needs to be addressed outside the behaviors specified in this document. This specification is only concerned with collection and delivery of software inventory information; components that consume and use this information are a separate concern.)

The SW-PC does have one functional dependency on the data models used in the software records it delivers, but only insofar as it is required to deterministically create a Software Identifier (described in Section 3.4.1) based on each record it delivers. The SW-PC MUST be able to generate a Software Identifier for each record it delivers, and if the SW-PC cannot do so the record cannot be

delivered by the SW-PC. All SW-PCs MUST at least be able to generate Software Identifiers for the data model types specified in Section 6 of this document. A SW-PC MAY include the ability to generate Software Identifiers for other data model types, and thus be able to support them as well.

3.3. Basic Attribute Exchange

In the most basic exchange supported by this specification, a SW-PV sends a request to the SW-PC requesting some type of information about the endpoint's software inventory. This simple exchange is shown in Figure 2.

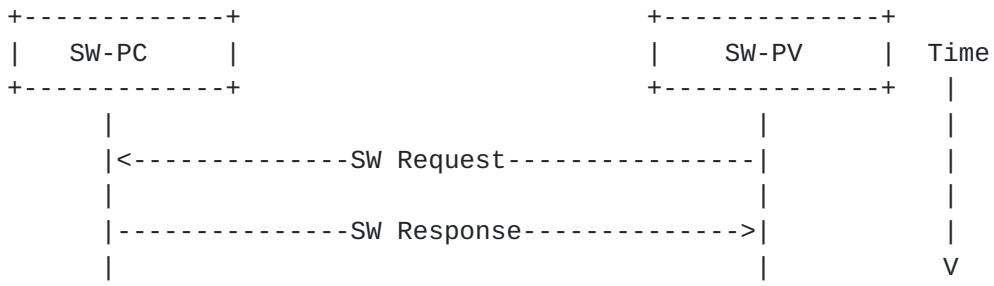


Figure 2: Basic SW Attribute Exchange

Upon receiving such a SW Request from the SW-PV, the SW-PC is expected to collect all the relevant software inventory information from the endpoint's software evidence collection and place it within its response attribute.

SW-PVs MUST discard without error any SW Response attributes that they receive for which they do not know the SW Request parameters that led to this SW Response. This is due to the fact that the SW Request includes parameters that control the nature of the response (as will be described in the following sections) and without knowing those parameters the SW Response cannot be reliably interpreted. Most often receiving an unsolicited SW Response attribute happens when a NEA Server has multiple SW-PVs; one SW-PV sends a SW Request but, unless exclusive delivery is used by the SW-PC in sending the response, both SW-PVs receive copies of the resulting SW Response. In this case, the SW-PV that didn't send the SW Request would lack the context necessary to correctly interpret the SW Response it received and would simply discard it. Note, however, that proprietary measures might allow a SW-PV to discover the SW Request parameters for a SW Response even if that SW-PV did not send the given SW Request. As such, there is no blanket requirement for a SW-PV to discard all SW Responses to SW Request the SW-PV did not

generate itself, only that SW-PVs are required to discard SW Responses for which they cannot get the necessary context to interpret.

In the case that it is possible to do so, the SW-PC SHOULD send its SW Response attribute to the SW-PV that requested it using exclusive delivery as described in section 4.5 of RFC 5793 (PB-TNC) [RFC5793]. Exclusive delivery ensures that only the sender of the SW Request receives the resulting SW Response.

3.4. Core Software Reporting Information

Different parameters in the SW Request can influence what information is returned in the SW Response. However, while each SW Response provides different additional information about this installed software, they all share a common set of fields that support reliable software identification on an endpoint. These fields include: Software Identifiers, the Data Model Type, Record Identifiers, Software Locators, and Source Identifiers. These fields are present for each reported piece of software in each type of SW Response. The following sections examine these three types of information in more detail.

3.4.1. Software Identifiers

A Software Identifier uniquely identifies a specific version of a specific software product. The Software Inventory Message and Attributes for PA-TNC specification does not dictate the structure of a Software Identifier (beyond stating that it is a string) or define how it is created. Instead, each data model described in the Software Data Model IANA table (Section 10.4) includes its own rules for how a Software Identifier is created based on a record in the Endpoint's Software Inventory Evidence Collection expressed in that data model. Other data models will have their own procedures for the creation of associated Software Identifiers. Within the Software Inventory Message and Attributes for PA-TNC specification, the Software Identifier is simply an opaque string and there is never any need to unpack any information that might be part of that identifier.

A Software Identifier is a fraction of the size of the inventory record from which it is derived. For some combinations of data models and sources, the full record might never be necessary as the identifier can be directly correlated to the contents of the full record. This is possible with authoritative SWID tags, since these tags always have the same contents and thus a Software Identifier derived from these tags can be used as a lookup to a local copy of the full tag. For other combinations of source and data model, a server might not be able to determine the specific software product

and version associated with the identifier without requesting delivery of the full record. However, even in those cases, downstream consumers of this information might never need the full record as long as the Software Identifiers they receive can be tracked reliably. A SW-PV can use Software Identifiers to track the presence of specific software products on an endpoint over time in a bandwidth-efficient manner.

There are two important limitations of Software Identifiers to keep in mind:

1. The identifiers do not necessarily change when the associated record changes. In some situations, a record in the endpoint's Software Inventory Evidence Collection will change due to new information becoming available or in order to correct prior errors in that information. Such changes might or might not result in changes to the Software Identifier, depending on the nature of the changes and the rules governing how Software Identifiers are derived from records of the appropriate data model.

2. It is possible that a single software product is installed on a single endpoint multiple times. If both of these installation instances are reported by the same source using the same data format, then this can result in identical Software Identifiers for each installation instances. In other words, Software Identifiers might not uniquely identify installation instances; they just are intended to uniquely identify software products (which might have more than one installation instance). Instead, to reliably distinguish between multiple instances of a single software product, one needs to make use of Record Identifiers, described in the following section.

3.4.2. Data Model Type

The Data Model Type consists of two fields: the Data Model Type PEN and the Data Model Type. The combination of these fields is used to identify the type of data model of the associated software inventory record. The data model is significant not only because it informs the recipient of the data model of the associated record, but because the Software Identifier of the record is derived directly from the data recorded in that data model. As a result, Software Identifiers for the same software product but expressed via different data models might be different. Clearly identifying the type of data model from with the Software Identifier was derived thus provides useful context for that value.

The PEN (or Private Enterprise Number) identifies the organization that assigns meaning to the Data Model Type field value. PENs are managed by IANA in the Private Enterprise Numbers registry. PENs allow vendors to designate their own set of data models for software inventory description. IANA reserves the PEN of 0x000000. Data Model Types associated with this PEN are defined in the Software Data Model IANA table, created in Section 10.4 of this specification. Note that this IANA table reserves all values greater than or equal to 0xC0 (192) for enterprise use. This means that local enterprises can use custom data formats and indicate them with the IANA PEN and a Data Model Type value between 0xC0 and 0XFF, inclusive. Those enterprises are responsible for configuring their SW-PCs to correctly report those custom data models.

3.4.3. Record Identifiers

A Record Identifier is a 4-byte integer generated by the SW-PC that is uniquely associated with a specific record within the Endpoint's Software Inventory Evidence Collection. The SW-PC MUST assign a unique identifier to each record when it is added to the Endpoint's Software Inventory Evidence Collection. The Record Identifier SHOULD remain unchanged if that record is modified. The SW-PC might wish to assign Record Identifiers sequentially, but any scheme is acceptable provided that no two records receive the same identifier.

Servers can use Record Identifiers to distinguish between multiple instances of a single software product installed on an endpoint. Since each installation instance of a software product is associated with a separate record, servers can use the record identifier to distinguish between instances. For example, if an event is reported (as described in Section 3.7), the record identifier will allow the server to discern which instance of a software product is involved.

3.4.4. Software Locators

In addition to the need to identify software products, many use cases of inventory information need to know where software is located on the endpoint. This information might be needed to direct remediation actions or similar processes. For this reason, every reported software product also includes a Software Locator to identify where the software is installed on the endpoint.

If the location is not provided directly by the record source the SW-PC is responsible for attempting to determine the location of the software product. The "location" of a product SHOULD be the directory in which the software products executables are kept. However, if that directory is shared by other software products, the "location" SHOULD be the location of the primary executable

associated with the software product. The source and/or SW-PC MUST be consistent in reporting the location of a software product (i.e., it cannot use the executable location in one report and the directory location in another).

The location is expressed as a URI string consisting of a scheme and path. [RFC3986] The location URI does not include an authority part. The URI schema describes the context of the described location. For example, in most cases the location of the installed software product will be expressed in terms of its path in the filesystem. For such locations, the location URI scheme MUST be "file" or the URI MUST appear without a scheme. (I.e., "file" is default scheme.) It is possible that other schemes could be used to represent other location contexts. Apart from reserving the "file" scheme, this specification does not reserve schemes. When representing software products in other location contexts, tools MUST be consistent in their use of schemes, but the exact string used in those schemes is not normatively defined here.

It is possible, that a SW-PC is unable to determine the location of a reported software product. In this case, the SW-PC MUST provide a zero-length Software Locator. However, SW-PCs SHOULD only make such an assignment as a last resort. Even a probable location for a software product is preferable to using a zero-length locator.

3.4.5. Source Identifiers

All SW-PCs MUST track the source of each piece of software information they report. Each time a SW-PC gets information to send to a given SW-PV from a new source (from the perspective of that SW-PV), the SW-PC MUST assign that source a Source Identification Number, which is an 8-bit unsigned integer. Each item reported includes the Source Identification Number that provided that information. All information that is provided by that source, MUST be delivered with this same Source Identification Number. This MUST be done for each source used. If the SW-PC ever is unclear as to whether a given source is new or not, it MUST assume that this is a new source and assign it a new Source Identification Number. Source Identification Numbers can help with (although will not completely eliminate) the challenges posed by multiple reporting of a single software instance: since a single source would only report an instance or event once, if multiple reports of a similar instance come from multiple sources, this might be an instance of multiple reporting (although it still might not be so). On the other hand, if multiple instances are reported by a single source, this almost certainly means that there are actually multiple instance that are being legitimately reported.

The SW-PC is responsible for tracking associations between Source Identifiers and the given data source. This association MUST remain consistent with regard to a given SW-PV while there is an active PB-TNC session with that SW-PV. The SW-PC MAY have a different Source Identifier association for different SW-PVs. Likewise, the SW-PC MAY change the Source Identifier association for a given SW-PV if the PB-TNC session terminates. However, implementers of SW-PCs will probably find it easier to manage associations by maintaining the same association for all SW-PVs and across multiple sessions.

Of special note, events records reported from the SW-PC's event log (discussed in Section 3.7) also need to be sent with their associated data source. The Source Identifier reported with events MUST be the current (i.e., at the time the event is sent) Source Identifier associated with the data source that produced the event, regardless of how long ago that event occurred. Event logs are likely to persist far longer than a single PB-TNC session. SW-PCs MUST ensure that each event can be linked to the appropriate data source, even if the Source Identifiers used when the event was created have since been reassigned. In other words, when sending an event, it needs to be sent with the Source Identifier currently linked to the data source that produced it, regardless of whether a different Source Identifier would have been associated with the event when the event was first created.

Note that the Source Identification Number is primarily used to support recognition, rather than identification, of sources. That is to say, a Software Identification Number can tell a recipient that two events were reported by the same source, but will not necessarily help that recipient determine which source was used. Moreover, different SW-PCs will not necessarily use the same Source Identification Numbers for the same sources. SW-PCs MUST track the assignment of Source Identification Numbers to ensure consistent application thereof. SW-PCs MUST also track which Source Identification Numbers have been used with each SW-PV with which they communicate.

3.4.6. Using Software and Record Identifiers in SW Attributes

A SW Attribute reporting an endpoint's Software Inventory Evidence Collection always contains the Software Identifiers associated with the identified software products. A SW Attribute might or might not also contain copies of software inventory evidence records. The attribute exchange is identical to the diagram shown in Figure 2 regardless of whether software inventory evidence records are included. The SW Request attribute indicates whether the response is required to include software inventory evidence records. Excluding software inventory evidence records can reduce the attribute size of

the response by multiple orders of magnitude when compared to sending the same inventory with full records.

3.5. Targeted Requests

Sometimes a SW-PV does not require information about every piece of software on an endpoint but only needs to receive updates about certain software instances. For example, enterprise endpoints might be required to have certain software products installed and to keep these updated. Instead of requesting a complete inventory just to see if these products are present, the SW-PV can make a "targeted request" for the software in question.

Targeted requests follow the same attribute exchange described in Figure 2. The SW-PV targets its request by providing one or more Software Identifiers in its SW Request attribute. The SW-PC MUST then limit its response to contain only records that match the indicated Software Identifier(s). This allows the network exchange to exclude information that is not relevant to a given policy question, thus reducing unnecessary bandwidth consumption. The SW-PC's response might or might not include software inventory evidence records, depending on the parameters of the SW Request.

Note that targeted requests identify the software relevant to the request only through Software Identifiers. This specification does not support arbitrary, parameterized querying of records. For example, one cannot request all records from a certain software publisher, or all records created by a particular record source. Targeted requests only allow a requestor to request specific software (as identified by their Software Identifiers) and receive a response that is limited to the named software.

There is no assumption that a SW-PC will recognize "synonymous records" - that is, records from different sources for the same software. Recall that different sources and data models may use different Software Identifier strings for the same software product. The SW-PC returns only records that match the Software Identifiers in the SW Request, even if there might be other records in the endpoint's Software Inventory Evidence Collection for the same software product. This is necessary because SW-PCs might not have the ability to determine that two Software Identifiers refer to the same product.

Targeted requests do not include Record Identifiers or Software Locators. The response to a targeted request MUST include all records associated with the named Software Identifiers, including the case where there are multiple records associated with a single Software Identifier.

SW-PCs MUST accept targeted requests and process them correctly as described above. SW-PVs MUST be capable of making targeted requests and processing the responses thereto.

3.6. Monitoring Changes in an Endpoint's Software Inventory Evidence Collection

The software collection on an endpoint is not static. As software is installed, uninstalled, patched, or updated, the Software Inventory Evidence Collection is expected to change to reflect the new software state on the endpoint. Different record sources might update the evidence collection at different rates. For example, a package manager might update its records in the Software Inventory Evidence Collection immediately whenever it is used to add or remove a software product. By contrast, sources that perform periodic examination of the endpoint would likely only update their records in the Software Inventory Evidence Collection after each examination.

All SW-PCs MUST be able to be able to detect changes to the Software Inventory Evidence Collection. Specifically, SW-PCs MUST be able to detect:

- The creation of records
- The deletion of records
- The alteration of records

An "alteration" is anything that modifies the contents of a record (or would modify it, if the record is dynamically generated on demand) in any way, regardless of whether the change is functionally meaningful.

SW-PCs MUST detect such changes to the endpoint's Software Inventory Evidence Collection in close to real-time (i.e., within seconds) when the Posture Collector is operating. In addition, in the case where there is a period during which the SW-PC is not operating, the SW-PC MUST be able to determine the net change to the endpoint's Software Inventory Evidence Collection over the period it was not operational. Specifically, the "net change" represents the difference between the state of the endpoint's Software Inventory Evidence Collection when the SW-PC was last operational and monitoring its state, and the state of the endpoint's software inventory evidence collection when the SW-PC resumed operation. Note that a net change might not reflect the total number of change events over this interval. For example, if a record was altered three times during a period when the SW-PC was unable to monitor for changes, the net change of this interval might only note that there was an alteration to the record,

but not how many individual alteration events occurred. It is sufficient for a SW-PC's determination of a net change to note that there was a difference between the earlier and current state rather than enumerating all the individual events that allowed the current state to be reached.

The SW-PC MUST assign a time to each detected change in the endpoint's Software Inventory Evidence Collection. These timestamps correspond to the SW-PC's best understanding as to when the detected change occurred. For changes to the endpoint's Software Inventory Evidence Collection that occur while the SW-PC is operating, the SW-PC ought to be able to assign a time to the event that is accurate to within a few seconds. For changes to the endpoint's Software Inventory Evidence Collection that occur while the SW-PC is not operational, upon becoming operational the SW-PC needs to make a best guess as to the time of the relevant events (possibly by looking at timestamps on files), but these values might be off. In the case of dynamically generated records, the time of change is the time at which the data from which the records are generate changes, not the time at which a changed record is generated. For example, if records are dynamically generated based on data in an RPM database, the time of change would be when the RPM database changed.

With regard to deletions of records, the SW-PC needs to detect the deletion and MUST retain a copy of the full deleted record along with the associated Record Identifier and Software Locator so that the record and associated information can be provided to the SW-PV upon request. This copy of the record MUST be retained for a reasonable amount of time. Vendors and administrators determine what "reasonable" means, but a copy of the record SHOULD be retained for as long as the event recording the deletion of the record remains in the SW-PC's event log (as described in Section 3.7). This is recommended because, as long as the event is in the SW-PC's change logs, the SW-PC might send an event attribute (described in Section 3.7) that references this record, and a copy of the record is needed if the SW-PV wanted a full copy of the relevant records. In the case that there a SW-PC is called upon to report a deleted record that is no longer available, the SW-PC SHOULD return a 0-length record.

With regard to alterations to a record, SW-PCs MUST detect any alterations to the contents of a record. Alterations need to be detected even if they have no functional impact on the record. A good guideline is that any alteration to a record that might change the value of a hash taken on the record's contents needs to be detected by the SW-PC. A SW-PC might be unable to distinguish modifications to the content of a record from modifications to the metadata the file system associates with the record. For example, a

SW-PC might use the "last modification" timestamp as an indication of alteration to a given record, but a record's last modification time can change for reasons other than modifications to the record contents. A SW-PC is still considered compliant with this specification if it also reports metadata change events that do not change the record itself as alterations to the record. In other words, while SW-PC authors are encouraged to exclude modifications that do not affect the bytes within the record, discriminating between modifications to file contents and changes to file metadata can be difficult and time consuming on some systems. As such, as long as the alterations detected by a SW-PC always cover all modifications to the contents of record, the SW-PC is considered compliant even if it also registers alterations that do not modify the contents of a record as well. When recording an alteration to a record, the SW-PC is only required to note that an alteration occurred. The SW-PC is not required to note or record how the record altered, nor is it possible to include such details in SW Attributes reporting the change to a SW-PV. There is no need to retain a copy of the original record.

When a record changes it SHOULD retain the same Record Identifier. The Software Locator might or might not change, depending on whether the software changed locations during the changes that led to the record change. A record change MUST retain the same Software Identifier. This means that any action that changes a software product (e.g., application of a patch that results in a change to the product's version) MUST NOT be reflected by a record change but instead MUST result in the deletion of the old record and the creation of a new record. This reflects the requirement that a record in the endpoint's Software Inventory Evidence Collection correspond directly with an instance of a specific software product.

3.7. Reporting Change Events

As noted in the preceding section, SW-PCs MUST be able to detect changes to the endpoints Software Inventory Evidence Collection (creation, deletion, and alteration) in near real-time while the SW-PC is operational, and MUST be able to account for any net change to the endpoint's Software Inventory Evidence Collection that occurs when the SW-PC is not operational. However, to be of use to the enterprise, the NEA Server needs to be able to receive these events and be able to understand how new changes relate to earlier changes. In Software Inventory Message and Attributes for PA-TNC, this is facilitated by reporting change events. All SW-PCs MUST be capable of receiving requests for change events and sending change event attributes. All SW-PVs MUST be capable of requesting and receiving change event attributes.

3.7.1. Event Identifiers

To be useful, change events need to be correctly ordered. Ordering of events is facilitated by two pieces of information: an Event Identifier (EID) value and an EID Epoch value.

An EID is a 4-byte unsigned integer that the SW-PC assigns sequentially to each observed event (whether detected in real-time or deduced by looking for net changes over a period of SW-PC inactivity). All EIDs exist within the context of some "EID Epoch", which is also represented as a 4-byte unsigned integer. EID Epochs are used to ensure synchronization between the SW-PC and any SW-PVs with which it communicates. EID Epoch values MUST be generated in such a way as to minimize the chance that an EID Epoch will be reused, even in the case where the SW-PC reverts to an earlier state. For this reason, sequential EID Epochs are discouraged, since loss of state could result in value reuse. In the case where a SW-PC needs to reset its EID counter, either because it has exhausted all available EID values or because the SW-PC's event log becomes corrupted, then a new EID Epoch MUST be selected.

Within an Epoch, EIDs MUST be assigned sequentially, so that if a particular event is assigned an EID of N, the next observed event is given an EID of N+1. In some cases, events might occur simultaneously, or the SW-PC might not otherwise be able to determine an ordering for events. In these cases, the SW-PC creates an arbitrary ordering of the events and assigns EIDs according to this ordering. Two change events MUST NOT ever be assigned the same EID within the same EID Epoch. No meaningful comparison can be made between EID values of different Epochs.

The EID value of 0 is reserved and MUST NOT be associated with any event. Specifically, an EID of 0 in a SW Request attribute indicates that a SW-PV wants an inventory response rather than an event response, while an EID of 0 in a SW Response is used to indicate the initial state of the endpoint's Software Inventory Evidence Collection prior to the observation of any events. Thus the very first recorded event in a SW-PC's records within an EID Epoch MUST be assigned a value of 1 or greater. Note that EID and EID Epoch values are assigned by the SW-PC without regard to whether events are being reported to one or more SW-PVs. The SW-PC records events and assigns EIDs during its operation. Any and all SW-PVs that request event information from the SW-PC will have those requests served from the same event records and thus will see the same EIDs and EID Epochs for the same events.

If a SW-PC uses multiple sources, a SW-PC's assignment of EIDs MUST reflect the presence and order of all events on the endpoint (at

least for supported sources) regardless of the source. This means that if source A experiences an event, and then source B experiences two events, and then source A experiences another two events, the SW-PC is required to capture five events with consecutive EID values reflecting the order in which the events occur.

The SW-PC MUST ensure there is no coverage gap (i.e., change events that are not recorded in the SW-PC's records) in its change event records. This is necessary because a coverage gap might give a SW-PV a false impression of the endpoint's state. For example, if a SW-PV saw an event indicating that a particular record had been added to the endpoint's software inventory evidence collection, and saw no subsequent events indicating that record had been deleted, it might reasonably assume that this record was still present and thus that the indicated software was still installed (assuming the Epoch has not changed). If there is a coverage gap in the SW-PC's event records, however, this assumption could be false. For this reason, the SW-PC's event records MUST NOT contain gaps. In the case where there are periods where it is possible that changes occurred without the SW-PC detecting or recording them, the SW-PC MUST either compute a net change and update its event records appropriately, or pick a new EID Epoch to indicate a discontinuity with previous event records.

Within a given Epoch, once a particular event has been assigned an EID, this association MUST NOT be changed. That is, within an Epoch, once an EID is assigned to an event, that EID cannot be reassigned to a different event, and the event cannot be assigned a different EID. When the SW-PC's Epoch changes, all of these associations between EIDs and events are cancelled, and EID values once again become free for assignment.

3.7.2. Core Event Tracking Information

Whether reporting events or full inventories it is important to know how the reported information fits into the overall timeline of change events. This is why all SW Response attributes include fields to place that response within the sequence of detected events. Specifically, all SW Responses include a Last EID and EID Epoch field. The EID Epoch field identifies the EID Epoch in which the SW Response was sent. If the SW Response is reporting events, all reported events occurred within the named EID Epoch. The Last EID (which is also always from the named EID Epoch) indicates the EID of the last recorded change event at the time that the SW Response was sent. These two fields allow any response to be placed in the context of the complete set of detected change events within a given EID Epoch.

3.7.3. Updating Inventory Knowledge Based on Events

Modern endpoints can have hundreds of software products installed, most of which are unlikely to change from one day to the next. As such, instead of exchanging a complete list of an endpoint's inventory on a regular basis, one might wish to only identify changes since some earlier known state of this inventory. This is readily facilitated by the use of EIDs to place change events in a context relative to earlier state.

As noted above, every SW Response sent by a SW-PC to a SW-PV (as described in Section 3.3 through Section 3.5) includes the EID Epoch and EID of the last event recorded prior to that response being compiled. This allows the SW-PV to place all subsequently received event records in context relative to this SW Response attribute (since the EIDs represent a total ordering of all changes to the endpoint's software inventory evidence collection). Specifically, a SW-PV (or, more likely, a database that collects and records its findings) can record an endpoint's full inventory and also the EID and Epoch of the most recent event reflected at the time of that inventory. From that point on, if change events are observed, the attribute describing these events indicates the nature of the change, the affected records, and the order in which these events occurred (as indicated by the sequential EIDs). Using this information, any remote record of the endpoint's Software Inventory Evidence Collection can be updated appropriately.

3.7.4. Using Event Records in SW Attributes

A SW-PV MUST be able to request a list of event records instead of an inventory. The attribute flow in such an exchange looks the same as the basic flow shown in Figure 2. The only difference is that, in the SW Request attribute, the SW-PV provides an EID other than 0. (A value of 0 in these fields represents a request for an inventory.) When the SW-PC receives such a request, instead of identifying records from the endpoint's Software Inventory Evidence Collection, it consults its list of detected changes. The SW-PC MUST add an event record to the SW Response attribute for each recorded change event with an EID greater than or equal to the EID in the SW Request attribute (although targeting of requests, as described in the next paragraph, might limit this list). A list of event records MUST only contain events with EIDs that all come from the current Epoch.

SW-PVs can target requests for event records by including one or more Software Identifiers, as described in Section 3.5, in the SW Request that requests an event record list. A targeted request for event records is used to indicate that only events affecting software that matches one of the provided Software Identifiers are to be returned.

Specifically, in response to a targeted request for event records, the SW-PC MUST exclude any event records that are less than the indicated EID (within the current EID Epoch) and exclude any event records where the affected software does not match one of the provided Software Identifiers. This might mean that the resulting list of event records sent in the response attribute does not provide a continuous sequence of EIDs. Both SW-PCs and SW-PVs MUST support targeted request for event records.

3.7.5. Partial and Complete Lists of Event Records in SW Attributes

Over time, a SW-PC might record a large number of change events. If a SW-PV requests all change events covering a large period of time, the resulting SW Response attribute might be extremely large, especially if the SW-PV requests inclusion of software inventory evidence records in the response. In the case that the resulting attribute is too large to send (either because it exceeds the 4GB attribute size limit imposed by the PA-TNC specification, or because it exceeds some smaller size limit imposed on the SW-PC) the SW-PC MAY send a partial list of event records back to the SW-PV.

Generation of a partial list of events in a SW Response attribute requires the SW-PC to identify a "consulted range" of EIDs. A consulted range is the set of event records that are examined for inclusion in the SW Response attribute and that are included in that attribute if applicable. Recall that, if a SW Request is targeted, only event records that involve the indicated software would be applicable. (See Section 3.5 for more on Targeted Request.) If a request is not targeted, all event records in the considered range are applicable and included in the SW Response attribute.

The lower bound of the consulted range MUST be the EID provided in the SW Request. (Recall that a SW Request indicates a request for event records by providing a non-0 EID value in the SW Request. See Section 3.7.4.) The upper bound of the consulted range is the EID of the latest event record (as ordered by EID values) that is included in the SW Response attribute if it is applicable to the request. The EID of this last event record is called the "Last Consulted EID". The SW-PC chooses this Last Consulted EID based on the size of the event record list it is willing to provide to the SW-PV.

A partial result list MUST include all applicable event records within the consulted range. This means that for any applicable event record (i.e., any event record in an un-targeted request, or any event record associated with software matching a requested Software Identifier in a targeted request) whose EID is greater than or equal to the EID provided in the SW Request and whose EID is less than or equal to the Last Consulted EID, that event record MUST be included

in the SW Response conveying this partial list of event records. This ensures that every partial list of event records is always complete within its indicated range.

All SW Response attributes that convey event records include a Last Consulted EID field. This is in addition to the EID Epoch and Last EID fields that are present in all SW Responses. Note that, if responding to a targeted SW Request, the SW Response attribute might not contain the event record whose EID matches the Last Consulted EID value. For example, the last consulted EID record might have been deemed inapplicable because it did not match the specified list of Software Identifiers in the SW Request.

If a SW-PV receives a SW Response attribute where the Last EID and Last Consulted EID fields are identical, the SW-PV knows that it has received a result list that is complete, given the parameters of the request, up to the present time.

On the other hand, if the Last EID is greater than the Last Consulted EID, the SW-PV has received a partial result list. (The Last Consulted EID MUST NOT exceed the Last EID.) In this case, if the SW-PV wishes to try to collect the rest of the partially delivered result list it then sends a new SW Request whose EID is one greater than the Last Consulted EID in the preceding response. Doing this causes the SW-PC to generate another SW Response attribute containing event records where the earliest reported event record is the one immediately after the event record with the Last Consulted EID (since EIDs are assigned sequentially). By repeating this process until it receives a SW Response where the Last EID and Last Consulted EID are equal, the SW-PV is able to collect all event records over a given range, even if the complete set of event records would be too large to deliver via a single attribute.

Implementers need to be aware that a SW Request might specify an EID that is greater than the EID of the last event recorded by a SW-PC. In accordance with the behaviors described in Section 3.7.4, a SW-PC MUST respond to such a request with a SW Response attribute that contains zero event records. This is because the SW-PC has recorded no event records with EIDs greater than or equal to the EID in the SW Request. In such a case, the Last Consulted EID field MUST be set to the same value as the Last EID field in this SW Response attribute. This case is called out because the consulted range on a SW-PC in such a situation is a negative range, where the "first" EID in the range (provided in the SW Request) is greater than the "last" EID in the range (this being the EID of the last recorded event on the SW-PC). Implementers need to ensure that SW-PCs do not experience problems in such a circumstance.

Note that this specification only supports the returning of partial results when returning event records. There is no way to return a partial inventory list under this specification.

3.7.6. Synchronizing Event Identifiers and Epochs

Since EIDs are sequential within an Epoch, if a SW-PV's list of event records contains gaps in the EID values within a single Epoch, the SW-PV knows that there are events that have not been accounted for. The SW-PV can either request a new event list to collect the missing events or request a full inventory to re-sync its understanding of the state of the endpoint's Software Inventory Evidence Collection. In either case, after the SW-PV's record of the endpoint's Software Inventory Evidence Collection has been updated, the SW-PV can record the new latest EID value and track events normally from that point on.

If the SW-PV receives any attribute from a SW-PC where the EID Epoch differs from the EID Epoch that was used previously, then SW-PV or any entity using this information to track the endpoint's Software Inventory Evidence Collection knows that there is a discontinuity in their understanding of the endpoint's state. To move past this discontinuity and reestablish a current understanding of the state of the endpoint's Software Inventory Evidence Collection, the SW-PV needs to receive a full inventory from the endpoint. The SW-PV cannot be brought in sync with the endpoint's state through the collection of any set of event records in this situation. This is because it is not possible to account for all events on the SW-PC since the previous Epoch was used, because there is no way to query for EIDs from a previous Epoch. Once the SW-PV has received a full inventory for the new Epoch, the SW-PV records the latest EID reported in this new Epoch and can track further events normally.

A SW-PC MUST NOT report events with EIDs from any Epoch other than the current EID Epoch. The SW-PC MAY choose to purge all event records from a previous Epoch from memory after an Epoch change. Alternately, the SW-PC MAY choose to retain some event records from a previous EID Epoch and assign them new EIDs in the current Epoch. However, in the case where a SW-PC chooses the latter option it MUST ensure that the order of events according to their EIDs is unchanged and that there is no coverage gap between the first retained event recorded during the previous Epoch (now reassigned with an EID in the current Epoch) and the first event recorded during the current Epoch. In particular, the SW-PC MUST ensure that all change events that occurred after the last recorded event from the previous Epoch are known and recorded. (This might not be possible if the Epoch change is due to state corruption on the SW-PC.) A SW-PC might choose to reassign EIDs to records from a preceding Epoch to create a "sliding

window" of events, where each Epoch change represents a shift in the window of available events.

In the case where a SW-PC suffers a crash and loses track of its current EID Epoch or current EID, then it MUST generate a new EID Epoch value and begin assigning EIDs within that Epoch. In this case, the SW-PC MUST purge all event records from before the crash as it cannot ensure that there is not a gap between the last of those records and the next detected event. The process for generating a new EID Epoch MUST minimize the possibility that the newly generated EID Epoch is the same as a previously used EID Epoch.

The SW-PV will normally never receive an attribute indicating that the latest EID is less than the latest EID reported in a previous attribute within the same EID Epoch. If this occurs, the SW-PC has suffered an error of some kind, possibly indicative of at least partial corruption of its event log. In this case, the SW-PV MUST treat the situation as if there was a change in Epoch and treat any local copy of the endpoint's Software Inventory Evidence Collection as out-of-sync until a full inventory can be reported by the SW-PC. In this case, the SW-PV SHOULD log the occurrence so the SW-PC can be examined to ensure it is now operating properly.

3.8. Subscriptions

Thus far, all attribute exchanges discussed assume that a SW-PV sent an SW Request attribute and the SW-PC is providing a direct response to that request. The Software Inventory Message and Attributes for PA-TNC specification also supports the ability for a SW-PC to send a SW Response to the SW-PV in response to observed changes in the endpoint's software inventory evidence collection, instead of in direct response to a SW Request. An agreement by a SW-PC to send content when certain changes are detected to the endpoint's Software Inventory Evidence Collection is referred to in this specification as a "subscription", and the SW-PV that receives this content is said to be "subscribed to" the given SW-PC. All SW-PCs and SW-PVs MUST support the use of subscriptions.

3.8.1. Establishing Subscriptions

A SW-PV establishes a subscription on a particular SW-PC by sending a SW Request attribute with the Subscription flag set. The SW Request attribute is otherwise identical to the SW Requests discussed in previous sections. Specifically, such a SW Request might or might not request inclusion of software inventory evidence records, might or might not be targeted, and might request change event records or endpoint inventory. Assuming no error is encountered, a SW-PC MUST send a SW Response attribute in direct response to this SW Request

attribute, just as if the Subscription flag was not set. As such, the attribute exchange that establishes a new subscription in a SW-PC has the same flow seen in the previous attribute exchanges, as depicted in Figure 2. If the SW-PV does not receive a PA-TNC Error attribute (as described in Section 3.9 and Section 5.16) in response to their subscription request, the subscription has been successfully established on the SW-PC. The SW Request attribute that establishes a new subscription is referred to as the "establishing request" for that subscription.

When a subscription is established it is assigned a Subscription ID value. The Subscription ID is equal to the value of the Request ID of the establishing request. (For more about Request IDs, see Section 5.6.)

A SW-PC MUST have the ability to record and support multiple simultaneous subscriptions from a single party and from multiple parties. A SW-PV MUST have the ability to record and support multiple simultaneous subscriptions to a single party and subscriptions to multiple parties.

3.8.2. Managing Subscriptions

The SW-PC MUST record each accepted subscription along with the identity of the party to whom attributes are to be pushed in compliance with the subscription. This identity includes both the NEA Server's connection ID and the Posture Validator Identifier from the PB-PA message that delivered the request.

Likewise, SW-PVs MUST record each accepted subscription for which they are the subscribing party along with the associated Subscription ID and the identity of the SW-PC that will be fulfilling the subscription. The SW-PV needs to retain this information in order to correctly interpret pushed SW Response attributes sent in fulfillment of the subscription. The identity of the SW-PC is given in the Posture Collector Identifier of the PB-PA message header in all messages from that SW-PC.

3.8.3. Terminating Subscriptions

Subscriptions MAY be terminated at any time by the subscribing SW-PV by setting the Clear Subscriptions flag in a SW Request. (See Section 5.7 for more on using this flag.) In the case that a SW Request with the Clear Subscriptions flag set is received the SW-PC MUST only clear subscriptions that match both the NEA server connection ID and the SW-PV ID for this SW Request, and MUST clear all such subscriptions.

This specification does not give the SW-PV the ability to terminate subscriptions individually - all subscriptions to the SW-PV are cleared when the Clear Subscriptions flag is set.

This specification does not give the SW-PC the ability to unilaterally terminate a subscription. However, if the SW-PC experiences a fatal error fulfilling a subscription, resulting in sending a PA-TNC Error attribute of type SW_SUBSCRIPTION_FULFILLMENT_ERROR, then the subscription whose fulfillment led to the error MUST be treated as terminated by both the SW-PC and the SW-PV. Only the subscription experiencing the error is cancelled and other subscriptions are unaffected. See Section 3.9 for more on this error condition.

Finally, a subscription is terminated if the connection between the SW-PC and SW-PV is deleted. This occurs when the connection ID used in the messages between the SW-PC and the SW-PV becomes unbound. Loss of this connection ID would prevent the SW-PC from sending messages in fulfillment of this subscription. As such, loss of the connection ID necessarily forces subscription termination between the affected parties.

3.8.4. Subscription Status

A SW-PV can request that a SW-PC report the list of active subscriptions for which the SW-PV is the subscriber. A SW-PV can use this to recover lost information about active subscriptions. A SW-PV can also use this capability to verify that a SW-PC has not forgotten any of its subscriptions. The latter is especially useful where a SW-PC does not send any attributes in fulfillment of a given subscription for a long period of time. The SW-PV can check the list of active subscriptions on the SW-PC and verify whether the inactivity is due to a lack of reportable events or due to the SW-PC forgetting its obligations to fulfill a given subscription.

A SW-PV requests a list of its subscriptions on a given SW-PC by sending that SW-PC a Subscription Status Request. The SW-PC MUST then respond with a Subscription Status Response (or a PA-TNC Error if an error condition is experienced). The Subscription Status Response MUST contain one subscription record for each of the active subscriptions for which the SW-PV is the subscribing party.

3.8.5. Fulfilling Subscriptions

As noted in Section 3.6 SW-PCs MUST have the ability to automatically detect changes to an endpoint's Software Inventory Evidence Collection in near real-time. For every active subscription, the SW-PC MUST send an attribute to the subscribed SW-PV whenever a change

is detected to relevant records within the endpoint's Software Inventory Evidence Collection. Such an attribute is said to be sent "in fulfillment of" the given subscription and any such attribute MUST include that subscription's Subscription ID. If the establishing request for that subscription was a targeted request, then only records that match the Software Identifiers provided in that establishing request are considered relevant. Otherwise, (i.e., for non-targeted requests) any record is considered relevant for this purpose. Figure 3 shows a sample attribute exchange where a subscription is established and then later attributes are sent from the SW-PC in fulfillment of the established subscription.

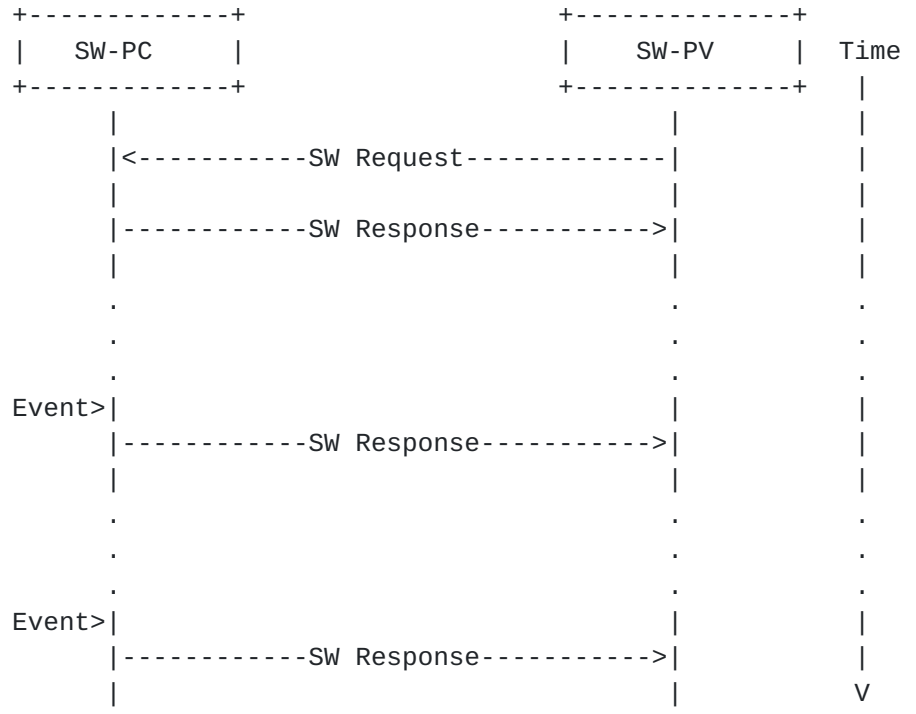


Figure 3: Subscription Establishment and Fulfillment

The contents of an attribute sent in fulfillment of a subscription depend on the parameters provided in the establishing request for that subscription. Specifically, the contents of an attribute sent in fulfillment of a subscription have the same format as would a direct response to the establishing request. For example, if the establishing request stipulated a response that contained an event record list that included software inventory evidence records, all attributes sent in fulfillment of this subscription will also consist of event record lists with software inventory evidence records. As such, all SW Responses displayed in the exchange depicted in Figure 3 have the same format. A SW Response generated in fulfillment of an

active subscription MUST be a valid SW Response attribute according to all the rules outlined in the preceding sections. In other words, an attribute constructed in fulfillment of a subscription will look the same as an attribute sent in direct response to an explicit request from a SW-PV that had the same request parameters and which arrived immediately after the given change event. There are a few special rules that expand on this guideline:

3.8.5.1. Subscriptions Reporting Inventories

In the case that a SW-PV subscribes to a SW-PC requesting an inventory attribute whenever changes are detected (i.e. the EID in the establishing request is 0), then the SW-PC MUST send the requested inventory whenever a relevant change is detected. (A "relevant change" is any change for untargeted requests, or a change to an indicated record in a targeted request.) Upon detection of a relevant change for an active subscription, the SW-PC sends the appropriate inventory information as if it had just received the establishing request. Attributes sent in fulfillment of this subscription will probably have a large amount of redundancy, as the same records are likely to be present in each of these SW Attributes. The role of an inventory subscription is not to report records just for the items that changed - that is the role of a subscription that reports events (see Section 3.8.5.2). A SW-PC MUST NOT exclude a record from an attribute sent in fulfillment of an inventory subscription simply because that record was not involved in the triggering event (although a record might be excluded for other reasons, such as if the subscription is targeted - see Section 3.8.5.3).

3.8.5.2. Subscriptions Reporting Events

The way in which a SW-PV indicates it wishes to establish a subscription requesting event records is by providing a non-zero EID in the SW Request establishing the subscription (see Section 3.7.1). However, when the SW-PC constructs an attribute in fulfillment of the subscription (other than the direct response to the establishing request), it MUST only include event records for the detected change(s) that precipitated this response attribute. In other words, it MUST NOT send a complete list of all changes starting with the indicated EID, up through the latest change, every time a new event is detected. In effect, the EID in the establishing request is treated as being updated every time an attribute is sent in fulfillment of this subscription, such that a single event is not reported twice in fulfillment of a single subscription. As such, every SW-PC MUST track the EID of the last event that triggered an attribute for the given subscription. When the next event (or set of events) is detected, the SW-PC MUST only report events with EIDs

after the last reported event. In the case that the EID Epoch of the SW-PC changes, the SW-PC MUST treat EID values in the new Epoch as being after all EIDs assigned in the previous Epoch regardless of the relative numeric values of these EIDs.

Note that while a subscription is active, the subscribing SW-PV MAY make other requests for event records that overlap with events that are reported in fulfillment of a subscription. Such requests are unaffected by the presence of the subscription, nor is the subscription affected by such requests. In other words, a given request will get the same results back whether or not there was a subscription. Likewise, an attribute sent in fulfillment of a subscription will contain the same information whether or not other requests had been received from the SW-PV.

A SW-PV needs to pay attention to the EID Epoch in these attributes, as changes in the Epoch might create discontinuities in the SW-PV's understanding of the endpoint's Software Inventory Evidence Collection state, as discussed in Section 3.7.6. In particular, once the EID Epoch changes, a SW-PV is unable have confidence that it has a correct understanding of the state of an endpoint's Software Inventory Evidence Collection until after the SW-PV collects a complete inventory.

SW-PCs MAY send partial lists of event records in fulfillment of a subscription. (See Section 3.7.5 for more on partial list of event records.) In the case that a SW-PC sends a partial list of event records in fulfillment of a subscription, it MUST immediately send the next consecutive partial list, and continue doing so until it has sent the equivalent of the complete list of event records. In other words, if the SW-PC sends a partial list it does not wait for another change event to send another SW Response, but continues sending SW Responses until it has sent all event records that would have been included in a complete fulfillment of the subscription. Note that the direct response to the establishing request is not considered to be sent in fulfillment of a subscription. However, in this case the SW-PC MUST treat the presence of unreported events as a triggering event for pushing additional messages in fulfillment of the newly established subscription. As such, the net effect is that, if the direct response to the establishing request (i.e., the Subscription Fulfillment flag is unset) is partial, the SW-PC will immediately follow this with additional attributes (with the Subscription Fulfillment flag set) until the complete set of events has been sent to the SW-PV.

3.8.5.3. Targeted Subscriptions

Subscriptions MAY be targeted to only apply to records that match a given set of Software Identifiers. In the case where changes are detected that affect multiple records, some matching the establishing request's Software Identifiers and some not, the attribute sent in fulfillment of the subscription MUST only include inventory or events (as appropriate) for records that match the establishing request's Software Identifiers. The SW-PC MUST NOT include non-matching records in the attribute, even if those non-matching records experienced change events that were co-temporal with change events on the matching records.

In addition, a SW-PC MUST send an attribute in fulfillment of a targeted subscription only when changes to the endpoint's Software Inventory Evidence Collection impact one or more records matching the subscription's establishing request's Software Identifiers. A SW-PC MUST NOT send any attribute in fulfillment of a targeted subscription based on detected change to the endpoint's Software Inventory Evidence Collection that did not involve any of the records targeted by that subscription.

3.8.5.4. No Subscription Consolidation

A SW-PV MAY establish multiple subscriptions to a given SW-PC. If this is the case, it is possible that a single change event on the endpoint might require fulfillment by multiple subscriptions, and that the information included in attributes that fulfill each of these subscriptions might overlap. The SW-PC MUST send separate attributes for each established subscription that requires a response due to the given event. Each of these attributes MUST contain all information required to fulfill that individual subscription, even if that information is also sent in other attributes sent in fulfillment of other subscriptions at the same time. In other words, SW-PCs MUST NOT attempt to combine information when fulfilling multiple subscriptions simultaneously, even if this results in some redundancy in the attributes sent to the SW-PV.

3.8.5.5. Delayed Subscription Fulfillment

A SW-PC MAY delay the fulfillment of a subscription following a change event in the interest of waiting to see if additional change events are forthcoming and, if so, conveying the relevant records back to the SW-PV in a single SW Response attribute. This can help reduce network bandwidth consumption between the SW-PC and the SW-PV. For example, consider a situation where 10 changes occur a tenth of a second apart. If the SW-PC does not delay in assembling and sending SW Response attributes, the SW-PV will receive 10 separate SW

Response attributes over a period of 1 second. However, if the SW-PC waits half a second after the initial event before assembling a SW Response, the SW-PV only receives two SW Response attributes over the same period of time.

Note that the ability to consolidate events for a single subscription over a given period of time does not contradict the rules in Section 3.8.5.4 prohibiting consolidation across multiple subscriptions. When delaying fulfillment of subscriptions, SW-PCs are still required to fulfill each individual subscription separately. Moreover, in the case that change events within the delay window cancel each other out (e.g., a record is deleted and then re-added), the SW-PC MUST still report each change event rather than just reporting the net effect of changes over the delay period. In other words, delayed fulfillment can decrease the number of attributes send by the SW-PC, but it does not reduce the total number of change events reported.

SW-PCs are not required to support delayed fulfillment of subscriptions. However, in the case that the SW-PC does support delayed subscription fulfillment, it MUST be possible to configure the SW-PC to disable delayed fulfillment. In other words, parties deploying SW-PCs need to be allowed to disable delayed subscription fulfillment in their SW-PCs. The manner in which such configuration occurs is left to the discretion of implementers, although implementers MUST protect the configuration procedure from unauthorized tampering. In other words, there needs to be some assurance that unauthorized individuals are not able to introduce long delays in subscription fulfillment.

3.9. Error Handling

In the case where the SW-PC detects an error in a SW Request attribute that it receives it MUST respond with a PA-TNC Error attribute with an error code appropriate to the nature of the error. (See Section 4.2.8 of PA-TNC [RFC5792] for more details about PA-TNC Error attributes and error codes as well as Section 5.16 in this specification for error codes specific to SW Attributes.) In the case that an error is detected in a SW Request the SW-PC MUST NOT take any action requested by this SW Request, even if partial completion of the SW is possible. In other words, a SW Request that contains an error is completely ignored by the SW-PC (beyond sending a PA-TNC Error attribute, and possibly logging the error locally) rather than being partially executed.

In the case where the SW-PC receives a valid SW Request attribute but experiences an error during the process of responding to that attribute's instructions where that error prevents the SW-PC from

properly or completely fulfilling that request, the SW-PC MUST send a PA-TNC Error attribute with an error code appropriate to the nature of the error. In the case where a PA-TNC Error attribute is sent, the SW-PC MUST NOT take any of the actions requested by the SW Request attribute which led to the detected error. This is the case even if some actions could have been completed successfully, and might even require the SW-PC to reverse some successful actions already taken before the error condition was detected. In other words, either all aspects of a SW Request complete fully and successfully (in which case the SW-PC sends a SW Response attribute), or no aspects of the SW Request occur (in which case the SW-PC sends a PA-TNC Error attribute). In the case that a SW-PC sends a PA-TNC Error attribute in response to a SW Request then the SW-PC MUST NOT also send any SW Response attribute in response to the same SW Request. For this reason, the sending of a SW Response attribute MUST be the last action taken by a SW-PC in response to a SW Request to avoid the possibility of a processing error occurring after that SW Response attribute is sent.

In the case that the SW-PC detects an error that prevents it from properly or completely fulfilling its obligations under an active subscription, the SW-PC MUST send a PA-TNC Error attribute of type SW_SUBSCRIPTION_FULFILLMENT_ERROR to the SW-PV that established this subscription. This type of PA-TNC Error attribute identifies the specific subscription that cannot be adequately honored due to the error condition as well as an error "sub-type". The error sub-type is used to indicate the type of error condition the SW-PC experienced that prevented it from honoring the given subscription. In the case that the error condition cannot be identified or does not align with any of the defined error codes, the SW_ERROR error code SHOULD be used in the sub-type. In the case that a SW_SUBSCRIPTION_FULFILLMENT_ERROR is sent, the associated subscription MUST be treated as cancelled by both the SW-PC and SW-PV.

The SW-PV MUST NOT send any PA-TNC Error attributes to SW-PCs. In the case that a SW-PV detects an error condition, it SHOULD log this error but does not inform any SW-PC's of this event. Errors might include, but are not limited to, detection of malformed SW Response attributes sent from a given SW-PC, as well as detection of error conditions when the SW-PV processes SW Responses.

Both SW-PCs and SW-PVs SHOULD log errors so that administrators can trace the causes of errors. Log entries SHOULD include the type of the error, the time it was detected, and additional descriptive information to aid in understanding the nature and cause of the error. Logs are an important debugging tool and implementers are

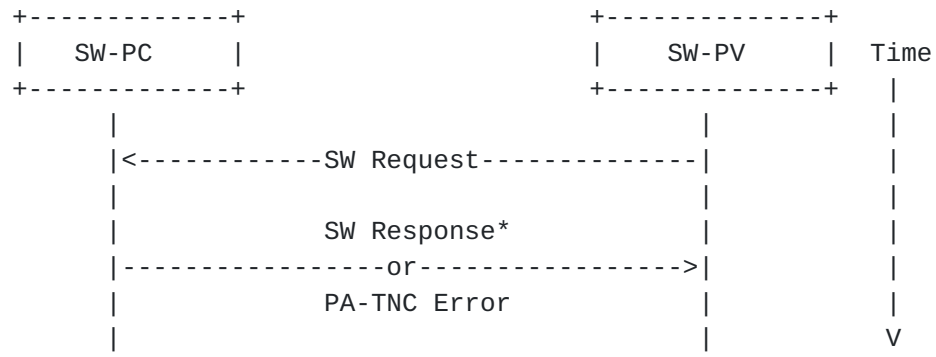
strongly advised to include comprehensive logging capabilities in their products.

4. Protocol

The software inventory protocol supports two different types of message exchanges which are described the following subsections, along with implementation requirements for supporting these exchanges.

4.1. Direct Response to a SW Request

The first type of exchange is used to provide the SW-PV with a software inventory or event collection from the queried endpoint.



*SW Response is one of the following: Software Identifier Inventory, Software Identifier Events, Software Inventory, or Software Events.

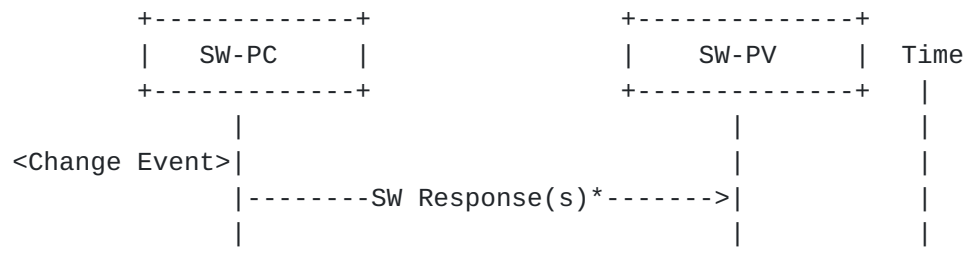
Figure 4: SW Attribute Exchange (Direct Response to SW Request)

In this exchange, the SW-PV indicates to the SW-PC, via a SW Request, the nature of the information it wishes to receive (inventory vs. events, full or targeted) and how it wishes the returned inventory to be expressed (with or without software inventory evidence records). The SW-PC responds with the requested information using the appropriate attribute type. A single SW Request MUST only lead to a single SW Response or PA-TNC Error that is in direct response to that request.

4.2. Subscription-Based Response

The second type of exchange allows change event-based reporting based on a subscription. If there is an active subscription on the endpoint, the SW-PC sends a SW Response to the SW-PV following a

change event on the endpoint in fulfillment of that subscription. Such an exchange is shown in Figure 5.



*SW Response is one of the following: Software Identifier Inventory, Software Identifier Events, Software Inventory, or Software Events.

Figure 5: SW Attribute Exchange (In Fulfillment of an Active Subscription)

Note that, unlike direct responses to a SW Request, a single change event can precipitate multiple SW Responses for a single subscription, but only if all but the last of those SW Responses convey partial lists of event records. When providing multiple SW Responses in this way, the initial responses contain partial lists of event records and the last of those SW Responses conveys the remainder of the relevant event records, completing the delivery of all relevant events in response to the change event. A single Change Event MUST NOT otherwise be followed by multiple SW Response or PA-TNC Error attributes in any combination.

4.3. Required Exchanges

All SW-PVs and SW-PCs MUST support both types of exchanges. In particular, SW-PCs MUST be capable of pushing a SW Response to a SW-PV immediately upon detection of a change to the endpoint's Software Inventory Evidence Collection in fulfillment of established SW-PV subscriptions, as described in Section 3.8.

5. Software Inventory Messages and Attributes

This section describes the format and semantics of the Software Inventory Message and Attributes for PA-TNC protocol. This protocol uses the PA-TNC message header format [RFC5792].

5.1. PA Subtype (AKA PA-TNC Component Type)

The NEA PB-TNC interface provides a general message-batching protocol capable of carrying one or more PA-TNC messages between the Posture Broker Client and Posture Broker Server. When PB-TNC is carrying a PA-TNC message, the PB-TNC message headers contain a 32 bit identifier called the PA Subtype. The PA Subtype field indicates the type of component associated with all of the PA-TNC attributes carried by the PB-TNC message. The core set of PA Subtypes is defined in the PA-TNC specification. This specification defines a new "SW Attributes" PA Subtype, which is registered in Section 10.1 of this document, which is used as a namespace for the collection of SW attributes defined in this document.

For more information on PB-TNC and PA-TNC messages and message headers, see the PB-TNC [RFC5793] and PA-TNC [RFC5792] specifications, respectively.

5.2. SW Attribute Overview

Each PA-TNC attribute described in this specification is intended to be sent between the SW-PC and SW-PV, so will be carried in a PB-TNC message indicating a PA Subtype of "SW Attributes". PB-TNC messages MUST always include the SW Attributes Subtype defined in Section 5.1 when carrying SW Attributes over PA-TNC. The attributes defined in this specification appear below along with a short summary of their purposes. Each attribute is described in greater detail in subsequent sections.

SW Request: This attribute is used to request a software inventory or software event list from an endpoint. This attribute might also establish a subscription on the recipient SW-PC. See Section 5.7 for more information.

Software Identifier Inventory: This attribute is used to convey an inventory without the inclusion of software inventory evidence records. See Section 5.8 for more information.

Software Identifier Events: This attribute is used to convey a list of events concerning changes to an endpoint's Software Inventory Evidence Collection. Reported events do not include software inventory evidence records. See Section 5.9 for more information.

Software Inventory: This attribute is used to convey an inventory expressed including software inventory evidence records. See Section 5.10 for more information.

Software Events: This attribute is used to convey a list of events concerning changes to an endpoint's inventory evidence collection. Reported events include software inventory evidence records. See Section 5.11 for more information.

Subscription Status Request: This attribute is used to request a SW-PC send a summary of all the active subscriptions it has where the requesting party is the subscriber. See Section 5.12 for more information.

Subscription Status Response: This attribute is used to convey information about the active subscriptions that a SW-PC has for a given subscriber. See Section 5.13 for more information.

Source Metadata Request: This attribute is used to request a SW-PC send metadata about the sources is uses to collect software inventory information. See Section 5.14 for more information.

Source Metadata Response: This attribute is used to convey descriptive metadata about the sources a SW-PC uses to collect software inventory information. See Section 5.15 for more information.

PA-TNC Error: This is the standard PA-TNC Error attribute as defined in PA-TNC [RFC5792] and is used to indicate that an error was encountered during a software inventory exchange. Use of the PA-TNC attribute by Software Inventory Message and Attributes for PA-TNC is described in Section 5.16.

Because one of the Software Identifier Inventory, Software Identifier Events, Software Inventory, or Software Events attributes are expected to be sent to a SW-PV in direct response to a SW Request attribute or in fulfillment of an active subscription, those four attribute types are referred to collectively in this document as "SW Response attributes".

All SW-PVs MUST be capable of sending SW Request attributes and be capable of receiving and processing all SW Response attributes as well as PA-TNC Error attributes. All SW-PCs MUST be capable of receiving and processing SW Request attributes and be capable of sending all types of SW Response attributes as well as PA-TNC Error attributes. In other words, both SW-PVs and SW-PCs are required to support their role in exchanges using any of the attribute types defined in this section. SW-PVs MUST ignore any SW Request attributes that they receive. SW-PCs MUST ignore any SW Response attributes or PA-TNC Error attributes that they receive.

5.3. Message Diagram Syntax

This specification defines the syntax of new PA-TNC messages and attributes 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 from left to right for each 32-bit quantity traversing the diagram from top to bottom. Multi-octet fields representing numeric values MUST be sent in network (big endian) byte order.

Descriptions of bit fields (e.g. flags) values refer to the position of the bit within the field. These bit positions are numbered from the most significant bit through the least significant bit. As such, an octet with only bit 0 set would have a value of 0x80 (1000 0000), an octet with only bit 1 set would have a value of 0x40 (0100 0000), and an octet with only bit 7 set would have a value of 0x01 (0000 0001).

5.4. Software Inventory Message and Attributes for PA-TNC Attribute Enumeration

PA-TNC attribute types are identified in the PA-TNC Attribute Header via the Attribute Type Vendor ID and Attribute Type fields. Table 2 identifies the appropriate values for these fields for each attribute type used within the Software Inventory Message and Attributes for PA-TNC protocol. The following is a summary of the registered attributes. All attributes have a PEN value of 0x000000. For the Integer value field, both the hexadecimal and decimal values are provided.

Attribute Name	Integer	Description
SW Request	0x00000011 (17)	Request from a SW-PV to a SW-PC for the SW-PC to provide a software inventory or event list
Software Identifier Inventory	0x00000012 (18)	An inventory sent without software inventory evidence records sent from a SW-PC.
Software Identifier Events	0x00000013 (19)	A collection of events impacting the endpoint's Software Inventory Evidence Collection, where events do not include software inventory evidence records.

Software Inventory	0x00000014 (20)	An inventory including software inventory evidence records sent from a SW-PC.
Software Events	0x00000015 (21)	A collection of events impacting the endpoint's Software Inventory Evidence Collection, where events include software inventory evidence records.
Subscription Status Request	0x00000016 (22)	A request for a list of a SW-PV's active subscription on a SW-PC.
Subscription Status Response	0x00000017 (23)	A list of a SW-PV's active subscriptions on a SW-PC.
Source Metadata Request	0x00000018 (24)	A request for information about a SW-PC's data sources.
Subscription Metadata Response	0x00000019 (25)	Descriptive metadata about a SW-PC's data sources.
Reserved	0x0000001A - 0x0000001F (26 - 31)	These attribute types are reserved for future use in revisions to Software Inventory Message and Attributes for PA-TNC.
PA-TNC Error	0x00000008 (8)	An error attribute as defined in the PA-TNC specification [RFC5792].

Table 2: SW Attribute Enumeration

5.5. Normalization of Text Encoding

In order to ensure consistency of transmitted attributes some fields require normalization of their format. When this is necessary, this is indicated in the field's description. In such cases, the field contents MUST be normalized to Network Unicode format as defined in RFC 5198 [RFC5198]. Network Unicode format defines a refinement of UTF-8 that ensures a normalized expression of characters. SW-PCs and SW-PVs MUST NOT perform conversion and normalization on any field values except those specifically identified as requiring normalization in the following sections. Note, however, that some

data models require additional normalization before source information is added to an Endpoint's Inventory Evidence Collection as a record. The references from the Software Data Model IANA table (see Section 10.4) will note where this is necessary.

5.6. Request IDs

All SW Request attributes MUST include a Request ID value. The Request ID field provides a value that identifies a given request relative to other requests between a SW-PV and the receiving SW-PC. Specifically, the SW-PV assigns each SW Request attribute a Request ID value that is intended to be unique within the lifetime of a given network connection ID as assigned by the SW-PV's Posture Broker Server.

In the case that a SW Request requests the establishment of a subscription and the receiving SW-PC agrees to that subscription, the Request ID of that SW Request (i.e., the establishing request of the subscription) becomes that subscription's Subscription ID. All attributes sent in fulfillment of this subscription include a flag indicating that the attribute fulfills a subscription and the subscription's Subscription ID. A SW-PV MUST NOT reuse a Request ID value in communicating to a given SW-PC while that Request ID is also serving as a Subscription ID for an active subscription with that SW-PC. In the case where a SW-PC receives a SW Request from a given SW-PV where that Request ID is also the Subscription ID of an active subscription with that SW-PV, the SW-PC MUST respond with a PA-TNC Error attribute with an error code of SW_SUBSCRIPTION_ID_REUSE_ERROR. Note that this error does not cancel the indicated subscription.

Subscription Status Requests and Subscription Status Responses do not include Request IDs.

In the case where all possible Request ID values have been exhausted within the lifetime of a single network connection ID, the sender MAY reuse previously used Request IDs within the same network connection if the ID is not being used as a Subscription ID. In such a case of Request ID reuse, Request IDs SHOULD be reused in the order of their original use. In other words, a SW-PC SHOULD NOT use a given Request ID value more than once within a persistent connection between a given Posture Broker Client-Posture Broker Server pair. In the case where reuse is necessary due to exhaustion of possible ID values, the SW-PC SHOULD structure the reuse to maximize the time between original and subsequent use. The Request ID value is included in a SW Response attribute directly responding to this SW Request to indicate which SW Request was received and caused the response. Request IDs can be randomly generated or sequential, as long as

values are not repeated per the rules in this paragraph. SW-PCs are not required to check for duplicate Request IDs.

5.7. SW Request

A SW-PV sends this attribute to a SW-PC to request that the SW-PC send software inventory information to the SW-PV. A SW-PC MUST NOT send this attribute.

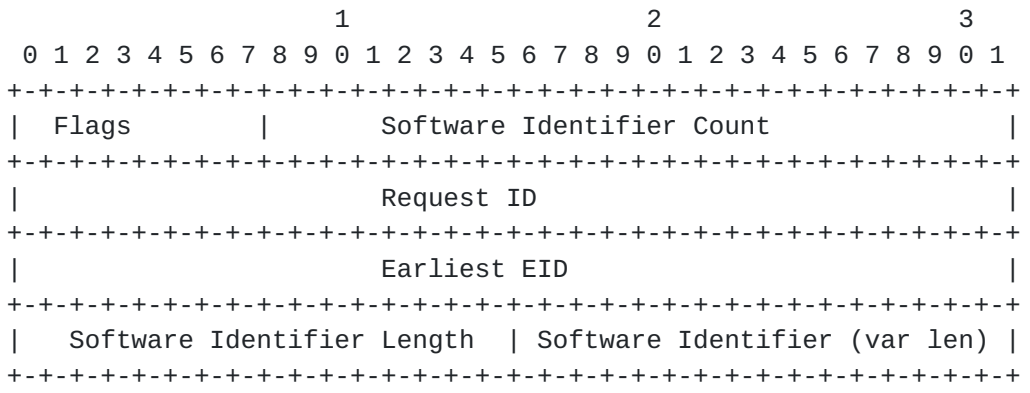


Figure 6: SW Request Attribute

Field	Description
Flags: Bit 0 - Clear Subscriptions	If set (1), the SW-PC MUST delete all subscriptions established by the requesting SW-PV (barring any errors).
Flags: Bit 1 - Subscribe	If set (1), in addition to responding to the request as described, the SW-PC MUST establish a subscription with parameters matching those in the request attribute (barring any errors).
Flags: Bit 2 - Result Type	If unset (0), the SW-PC's response MUST include software inventory evidence records and thus the response MUST be a Software Inventory, a Software Events, or a PA-TNC Error attribute. If set (1), the response MUST NOT include software inventory evidence records and thus the response MUST be a Software Identifier Inventory, a Software Identifier Events, or a PA-TNC Error attribute.
Flags: Bit	Reserved for future use. This field MUST be set

3-7 - Reserved	to zero on transmission and ignored upon reception.
Software Identifier Count	A 3-byte unsigned integer indicating the number of Software Identifiers that follow. If this value is non-zero, this is a targeted request, as described in Section 3.5. The Software Identifier Length and Software Identifier fields are repeated, in order, the number of times indicated in this field. In the case where Software Identifiers are present, the SW-PC MUST only report software that corresponds to the identifiers the SW-PV provided in this attribute (or with a PA-TNC Error attribute). This field value MAY be 0, in which case there are no instances of the Software Identifier Length and Software Identifier fields. In this case, the SW-PV is indicating an interest in all software inventory evidence records on the endpoint (i.e., this is not a targeted request).
Request ID	A value that uniquely identifies this SW Request from a particular SW-PV.
Earliest EID	In the case where the SW-PV is requesting software events, this field contains the EID value of the earliest event the SW-PV wishes to have reported. (Note - the report will be inclusive of the event with this EID value.) In the case where the SW-PV is requesting an inventory, then this field MUST be 0. (0x00000000) In the case where this field is non-zero, the SW-PV is requesting events and the SW-PC MUST respond using a Software Events, Software Identifier Events, or a PA-TNC Error attribute. In the case where this field is zero, the SW-PV is requesting an inventory and the SW-PC MUST respond using a Software Inventory, a Software Identifier Inventory, or a PA-TNC Error attribute.
Software Identifier Length	A 2-byte unsigned integer indicating the length in bytes of the Software Identifier field.
Software Identifier	A string containing the Software Identifier value from a software inventory evidence record. This field value MUST be normalized to Network Unicode

| | | format, as described in Section 5.5. This string | | | MUST NOT be NULL terminated. |

Table 3: SW Request Attribute Fields

The SW-PV sends the SW Request attribute to a SW-PC to request the indicated information. Note that between the Result Type flag and the Earliest EID field, the SW-PC is constrained to a single possible SW Response attribute type (or a PA-TNC Error attribute) in its response to the request.

The Subscribe and Clear Subscription flags are used to manage subscriptions for the requesting SW-PV on the receiving SW-PC. Specifically, an attribute with the Subscribe flag set seeks to establish a new subscription by the requesting SW-PV to the given SW-PC, while an attribute with the Clear Subscription flag seeks to delete all existing subscriptions by the requesting SW-PV on the given SW-PC. Note that, in the latter case, only the subscriptions associated with the Connection ID and the Posture Validator ID of the requester are deleted as described in Section 3.8.3. A newly established subscription has the parameters outlined in the Request attribute. Specifically, the Result Type flag indicates the type of result to send in fulfillment of the subscription, the value of the Earliest EID field indicates whether the fulfillment attributes list inventories or events, and the fields describing Software Identifiers (if present) indicate if and how a subscription is targeted. In the case that the SW-PC is unable or unwilling to comply with the SW-PV's request to establish or clear subscriptions, the SW-PC MUST respond with a PA-TNC Error attribute with the SW_SUBSCRIPTION_DENIED_ERROR error code. If the SW-PV requests that subscriptions be cleared but has no existing subscriptions, this is not an error.

An attribute requesting the establishment of a subscription is effectively doing double-duty, as it is a request for an immediate response from the SW-PC in addition to setting up the subscription. Assuming the SW-PC is willing to comply with the subscription it MUST send an appropriate response attribute to a request with the Subscribe flag set containing all requested information. The same is true of the Clear Subscription flag - assuming there is no error the SW-PC MUST generate a response attribute without regard to the presence of this flag in addition to clearing its subscription list.

Both the Subscribe and Clear Subscription flags MAY be set in a single SW Request attribute. In the case where this request is successful, the end result MUST be equivalent to the SW-PC clearing its subscription list for the given SW-PV first and then creating a new subscription in accordance with the request parameters. In other

words, do not first create the new subscription and then clear all the subscriptions including the one that was just created. In the case that the requested actions are successfully completed, the SW-PC MUST respond with a SW Response attribute. The specific type of SW Response attribute depends on the Result Type and Earliest EID fields, as described above. In the case where there is a failure that prevents some part this request from completing, the SW-PC MUST NOT add a new subscription, MUST NOT clear the old subscriptions, and the SW-PC MUST respond with a PA-TNC Error attribute. In other words, the SW-PC MUST NOT partially succeed at implementing such a request; either all actions succeed, or none succeed.

The Earliest EID field is used to indicate if the SW-PV is requesting an inventory or event list from the SW-PC. A value of 0 (0x00000000) represents a request for inventory information. Otherwise, the SW-PV is requesting event information. For Earliest EID values other than 0, the SW-PC's response MUST respond with event records, as described in Section 3.7. Note that the request does not identify a particular EID Epoch, since responses can only include events in the SW-PC's current EID Epoch.

The Software Identifier Count indicates the number of Software Identifiers in the attribute. This number might be any value between 0 and 16,777,216, inclusive. A single Software Identifier is represented by the following fields: Software Identifier Length and Software Identifier. These fields are repeated a number of times equal to the Software Identifier Count, which may be 0. The Software Identifier Length field indicates the number of bytes allocated to the Software Identifier field. The Software Identifier field contains a Software Identifier as describe in Section 3.4.1. The presence of one or more Software Identifiers is used by the SW-PV to indicate a targeted request, which seeks only inventories of or events affecting software corresponding to the given identifiers. The SW-PC MUST only report software that matched the Software Identifiers provided in the SW-PVs SW Request attribute.

5.8. Software Identifier Inventory

A SW-PC sends this attribute to a SW-PV to convey the inventory of the endpoint's Software Inventory Evidence Collection without the inclusion of software inventory evidence records. This list might represent a complete inventory or a targeted list of records, depending on the parameters in the SW-PV's request. A SW-PV MUST NOT send this attribute. The SW-PC either sends this attribute in fulfillment of an existing subscription where the establishing request has a Result Type of 1 and the Earliest EID is zero, or in direct response to a SW Request attribute where the Result Type is 1 and the Earliest EID is zero.

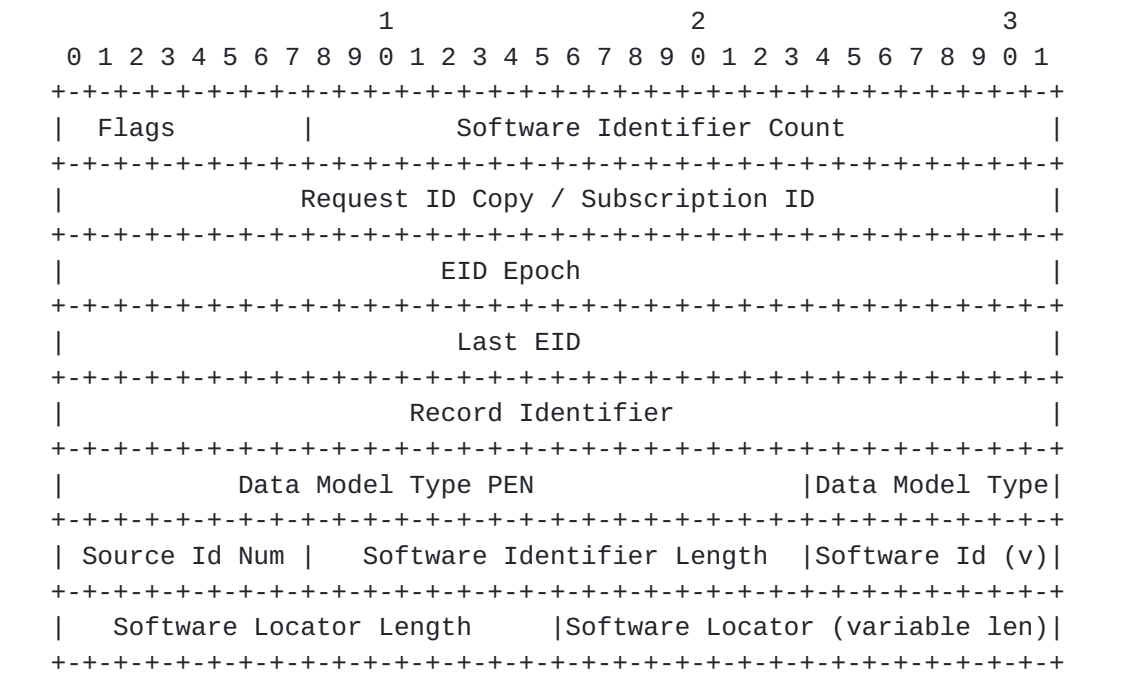


Figure 7: Software Identifier Inventory Attribute

Field	Description
Flags: Bit 0 - Subscription Fulfillment	In the case that this attribute is sent in fulfillment of a subscription this bit MUST be set (1). In the case that this attribute is a direct response to a SW Request, this bit MUST be unset (0).
Flags: Bit 1-7 - Reserved	Reserved for future use. This field MUST be set to zero on transmission and ignored upon reception.
Software Identifier Count	The number of Software Identifiers that follow. This field is an unsigned integer. The Record Identifier, Data Model Type PEN, Data Model Type, Source Identification Number, Software Identifier Length, Software Identifier, Software Locator Length, and Software Locator fields are repeated, in order, the number of times indicated in this field. This field value MAY be 0, in which case there are no instances of these fields.

Request ID Copy / Subscription ID	In the case where this attribute is in direct response to a SW Request attribute from a SW-PV, this field MUST contain an exact copy of the Request ID field from that SW Request. In the case where this attribute is sent in fulfillment of an active subscription, this field MUST contain the Subscription ID of the subscription being fulfilled by this attribute.
EID Epoch	The EID Epoch of the Last EID value. This field is an unsigned 4-byte integer.
Last EID	The EID of the last event recorded by the SW-PC, or 0 if the SW-PC has no recorded events. This field is an unsigned 4-byte integer.
Record Identifier	A 4-byte, unsigned integer containing the Record Identifier value from a software inventory evidence record.
Data Model Type PEN	A 3-byte unsigned integer containing the Private Enterprise Number (PEN) of the organization that assigned the meaning of the Data Model Type value.
Data Model Type	A 1-byte unsigned integer containing an identifier number that identifies the data model of the reported record.
Source Identification Number	The Source Identification Number associated with the source from which this software installation inventory instance was reported.
Software Identifier Length	A 2-byte unsigned integer indicating the length in bytes of the SW ID field.
Software Identifier	A string containing the Software Identifier value from a software inventory evidence record. This field value MUST be normalized to Network Unicode format, as described in Section 5.5. This string MUST NOT be NULL terminated.
Software Locator Length	A 2-byte unsigned integer indicating the length in bytes of the Software Locator field.
Software Locator	A string containing the Software Locator value. This is expressed as a URI. This field value

	MUST be normalized to Network Unicode format as described in Section 3.4.4. This string MUST NOT be NULL terminated.	

Table 4: Software Identifier Inventory Attribute Fields

In the case that this attribute is sent in fulfillment of a subscription, the Subscription Fulfillment bit MUST be set (1). In the case that this attribute is sent in direct response to a SW Request, the Subscription Fulfillment bit MUST be unset (0). Note that the SW Response attribute sent in direct response to a SW Request that establishes a subscription (i.e., a subscription's establishing request) MUST be treated as a direct response to that SW Request (and thus the Subscription Fulfillment bit is unset). SW Response attributes are only treated as being in fulfillment of a subscription (i.e., Subscription Fulfillment bit set) if they are sent following a change event, as shown in Figure 3.

The Software Identifier Count field indicates the number of Software Identifiers present in this inventory. Each Software Identifier is represented by the following set of fields: Record Identifier, Data Model Type, Software Identifier Length, Software Identifier, Software Locator Length, and Software Locator. These fields will appear once for each reported record.

When responding directly to a SW Request attribute, the Request ID Copy / Subscription ID field MUST contain an exact copy of the Request ID field from that SW Request. When this attribute is sent in fulfillment of an existing subscription on this Posture Collector, then this field MUST contain the Subscription ID of the fulfilled subscription.

The EID Epoch field indicates the EID Epoch of the Last EID value. The Last EID field MUST contain the EID of the last recorded change event (see Section 3.7 for more about EIDs and recorded events) at the time this inventory was collected. In the case where there are no recorded change events at the time that this inventory was collected, the Last EID field MUST contain 0. These fields can be interpreted to indicate that the provided inventory reflects the state of the endpoint after all changes up to and including this last event have been accounted for.

The Data Model Type PEN and Data Model Type fields are used to identify the data model associated with the given software record. These fields are discussed more in Section 3.4.2.

The Source Identification Number field is used to identify the source that provided the given record, as described in Section 3.1.

5.9. Software Identifier Events

A SW-PC sends this attribute to a SW-PV to convey events where the affected records are reported without software inventory evidence records. A SW-PV MUST NOT send this attribute. The SW-PC either sends this attribute in fulfillment of an existing subscription where the establishing request has a Result Type is 1 and the Earliest EID is non-zero, or in direct response to a SW Request attribute where the Result Type is 1 and the Earliest EID is non-zero.

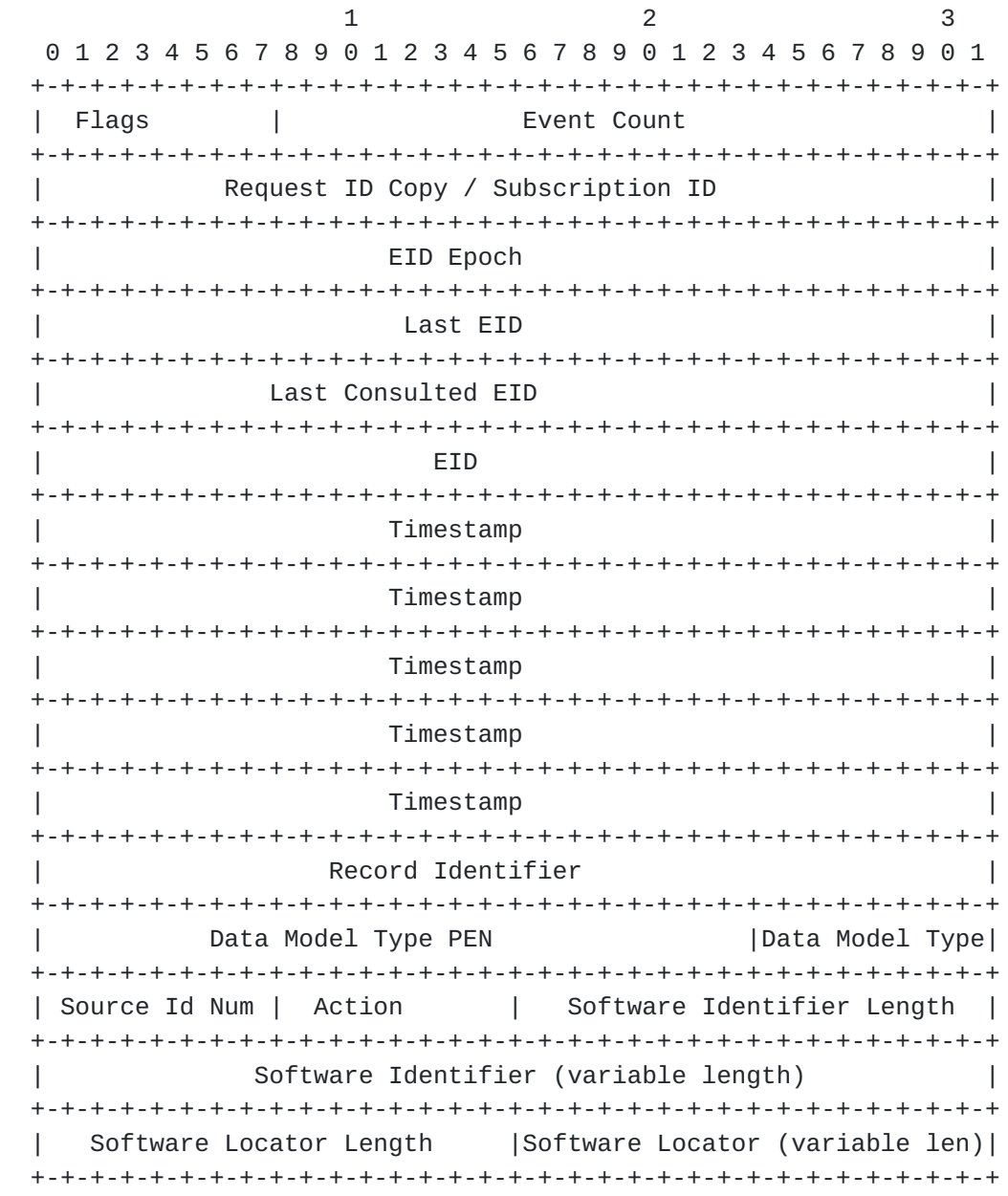


Figure 8: Software Identifier Events Attribute

Field	Description
Flags: Bit 0 - Subscription Fulfillment	In the case that this attribute is sent in fulfillment of a subscription this bit MUST be set (1). In the case that this attribute is a direct response to a SW Request, this bit MUST be unset (0).

Flags: Bit 1-7 - Reserved	Reserved for future use. This field MUST be set to zero on transmission and ignored upon reception.
Event Count	The number of events that are reported in this attribute. This field is a 3-byte unsigned integer. The EID, Timestamp, Record Identifier, Data Model Type PEN, Data Model Type, Source Identification Number, Action, Software Identifier Length, Software Identifier, Software Locator Length, and Software Locator fields are repeated, in order, the number of times indicated in this field. This field value MAY be 0, in which case there are no instances of these fields.
Request ID Copy / Subscription ID	In the case where this attribute is in direct response to a SW Request attribute from a SW-PV, this field MUST contain an exact copy of the Request ID field from that SW Request. In the case where this attribute is sent in fulfillment of an active subscription, this field MUST contain the Subscription ID of the subscription being fulfilled by this attribute.
EID Epoch	The EID Epoch of the Last EID value. This field is an unsigned 4-byte integer.
Last EID	The EID of the last event recorded by the SW-PC, or 0 if the SW-PC has no recorded events. This field contains the EID of the SW-PC's last recorded change event (which might or might not be included as an event record in this attribute).
Last Consulted EID	The EID of the last event record that was consulted when generating the event record list included in this attribute. This is different from the Last EID field value if and only if this attribute is conveying a partial list of event records. See Section 3.7.5 for more on partial list of event records.
EID	The EID of the event in this event record.
Timestamp	The timestamp associated with the event in this event record. This timestamps is the SW-PC's best understanding of when the given event

		occurred. Note that this timestamp might be an estimate. The Timestamp date and time MUST be represented as an RFC 3339 [5] compliant ASCII string in Coordinated Universal Time (UTC) time with the additional restrictions that the 'T' delimiter and the 'Z' suffix MUST be capitalized and fractional seconds (time-secfrac) MUST NOT be included. This field conforms to the date-time ABNF production from section 5.6 of RFC 3339 [RFC3339] with the above restrictions. Leap seconds are permitted and SW-PVs MUST support them. The Timestamp string MUST NOT be NULL terminated or padded in any way. The length of this field is always 20 octets.
Record Identifier	A 4-byte, unsigned integer containing the Record Identifier value from a software inventory evidence record.	
Data Model Type PEN	A 3-byte unsigned integer containing the Private Enterprise Number (PEN) of the organization that assigned the meaning of the Data Model Type value.	
Data Model Type	A 1-byte unsigned integer containing an identifier number that identifies the data model of the reported record.	
Source Identification Number	The Source Identification Number associated with the source from which this software installation inventory instance was reported.	
Action	The type of event that is recorded in this event record. Possible values are: 1 = CREATION - the addition of a record to the endpoint's Software Inventory Evidence Collection; 2 = DELETION - the removal of a record from the endpoint's Software Inventory Evidence Collection; 3 = ALTERATION - There was an alteration to a record within the endpoint's Software Inventory Evidence Collection. All other values are reserved for future use and MUST NOT be used when sending attributes. In the case where a SW-PV receives an event record that uses an action value other than the ones defined here, it MUST ignore that event record but SHOULD process other event records in this attribute as normal.	

Software Identifier Length	A 2-byte unsigned integer indicating the length in bytes of the SW ID field.
Software Identifier	A string containing the Software Identifier value from a software inventory evidence record. This field value MUST be normalized to Network Unicode format, as described in Section 5.5. This string MUST NOT be NULL terminated.
Software Locator Length	A 2-byte unsigned integer indicating the length in bytes of the Software Locator field.
Software Locator	A string containing the Software Locator value. This is expressed as a URI. This field value MUST be normalized to Network Unicode format as described in Section 3.4.4. This string MUST NOT be NULL terminated.

Table 5: Software Identifier Events Attribute Fields

The first few fields in the Software Identifier Events attribute mirror those in the Software Identifier Inventory attribute. The primary difference is that, instead of conveying an inventory, the attribute conveys zero or more event records, consisting of the EID, timestamp, Record Identifier, action type, data model type, Software Identifier, and Software Locator of the affected software inventory evidence record.

With regard to the Timestamp field, it is important to note that clock skew between the SW-PC and SW-PV as well as between different SW-PCs within an enterprise might make correlation of timestamp values difficult. This specification does not attempt to resolve clock skew issues, although other mechanisms outside of this specification do exist to reduce the impact of clock skew and make the timestamp more useful for such correlation. Instead, Software Inventory Message and Attributes for PA-TNC uses Timestamp value primarily as a means to indicate the amount of time between two events on a single endpoint. For example, by taking the difference of the times for when a record was removed and then subsequently re-added, one can get an indication as to how long the system was without the given record (and, thus without the associated software). Since this will involve comparison of timestamp values all originating on the same system, clock skew between the SW-PC and SW-PV is not an issue. However, if the SW-PC's clock was adjusted between two recorded events, it is possible for such a calculation to lead to incorrect understandings of the temporal distance between

events. Users of this field need to be aware of the possibility for such occurrences. In the case where the Timestamp values of two events appear to contradict the EID ordering of those events (i.e., the later EID has an earlier timestamp) the recipient MUST treat the EID ordering as correct.

All events recorded in a Software Identifier Events Attribute are required to be part of the same EID Epoch. Specifically, all reported events MUST have an EID from the same EID Epoch as each other, and which is the same as the EID Epoch of the Last EID and Last Consulted EID values. The SW-PC MUST NOT report events with EIDs from different EID Epochs.

The Last Consulted EID field contains the EID of the last event record considered for inclusion in this attribute. If this attribute contains a partial event set (as described in Section 3.7.5) this field value will be less than the Last EID value; if this attribute contains a complete event set, the Last EID and Last Consulted EID values are identical.

If multiple events are sent in a Software Identifier Events attribute, the order in which they appear within the attribute is not significant. The EIDs associated with them are used for ordering the indicated events appropriately. Also note that a single software record might be reported multiple times in an attribute, such as if multiple events involving the associated record were being reported.

5.10. Software Inventory

A SW-PC sends this attribute to a SW-PV to convey a list of inventory records. A SW-PV MUST NOT send this attribute. The SW-PC either sends this attribute in fulfillment of an existing subscription where the establishing request had a Result Type of 0 and the Earliest EID is zero, or in direct response to a SW Request attribute where the Result Type is 0 and the Earliest EID is zero.

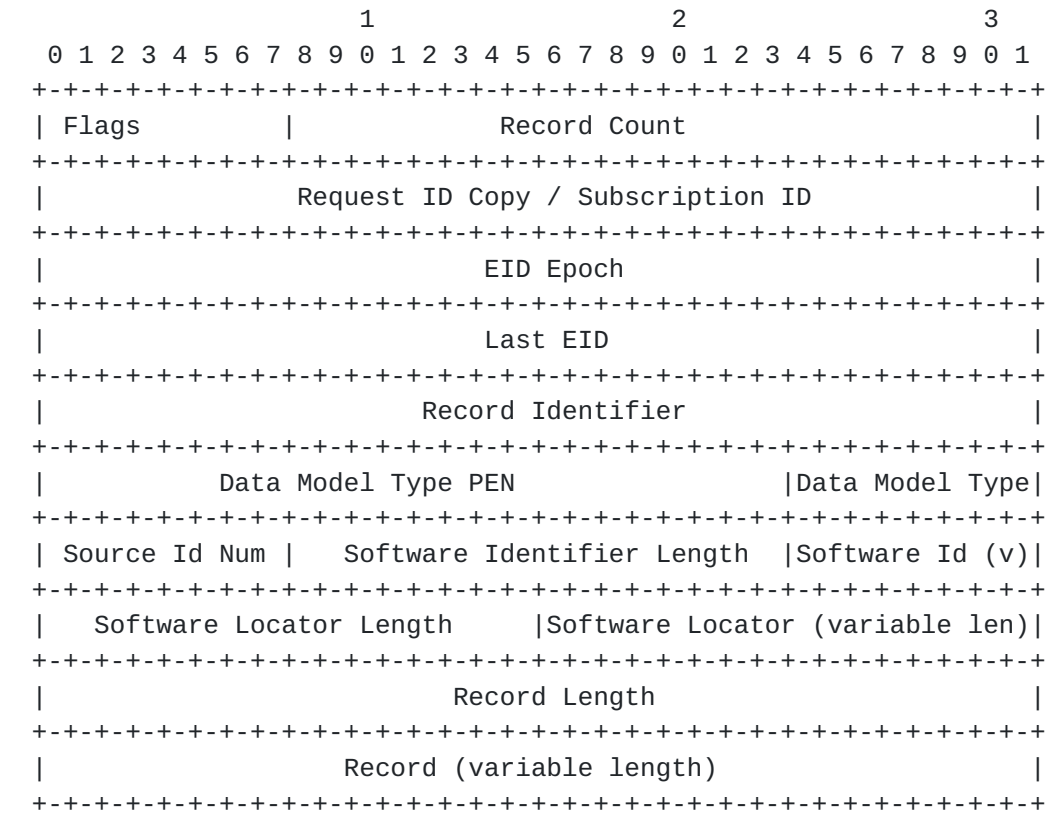


Figure 9: Software Inventory Attribute

Field	Description
Flags: Bit 0 - Subscription Fulfillment	In the case that this attribute is sent in fulfillment of a subscription this bit MUST be set (1). In the case that this attribute is a direct response to a SW Request, this bit MUST be unset (0).
Flags: Bit 1-7 - Reserved	Reserved for future use. This field MUST be set to zero on transmission and ignored upon reception.
Record Count	The number of records that follow. This field is a 3-byte unsigned integer. The Record Identifier, Data Model Type PEN, Data Model Type, Source Identification Number, Software Identifier Length, Software Identifier, Software Locator Length, Software Locator, Record Length, and Record fields are repeated, in order, the number of times indicated in this field. This

	field value MAY be 0 in which case there are no instances of these fields.
Request ID Copy / Subscription ID	In the case where this attribute is in direct response to a SW Request attribute from a SW-PV, this field MUST contain an exact copy of the Request ID field from that SW Request. In the case where this attribute is sent in fulfillment of an active subscription, this field MUST contain the Subscription ID of the subscription being fulfilled by this attribute.
EID Epoch	The EID Epoch of the Last EID value. This field is an unsigned 4-byte integer.
Last EID	The EID of the last event recorded by the SW-PC, or 0 if the SW-PC has no recorded events. This field is an unsigned 4-byte integer.
Record Identifier	A 4-byte, unsigned integer containing the Record Identifier value from a software inventory evidence record.
Data Model Type PEN	A 3-byte unsigned integer containing the Private Enterprise Number (PEN) of the organization that assigned the meaning of the Data Model Type value.
Data Model Type	A 1-byte unsigned integer containing an identifier number that identifies the data model of the reported record.
Source Identification Number	The Source Identification Number associated with the source from which this software installation inventory instance was reported.
Software Identifier Length	A 2-byte unsigned integer indicating the length in bytes of the SW ID field.
Software Identifier	A string containing the Software Identifier value from a software inventory evidence record. This field value MUST be normalized to Network Unicode format, as described in Section 5.5. This string MUST NOT be NULL terminated.
Software Locator Length	A 2-byte unsigned integer indicating the length in bytes of the Software Locator field.

Software Locator	A string containing the Software Locator value. This is expressed as a URI. This field value MUST be normalized to Network Unicode format as described in Section 3.4.4. This string MUST NOT be NULL terminated.
Record Length	This is a 4-byte unsigned integer indicating the length of the Record field in bytes.
Record	A software inventory evidence record as a string. The record MUST be converted and normalized to Network Unicode format, as described in Section 5.5. This string MUST NOT be NULL terminated.

Table 6: Software Inventory Attribute Fields

The Software Inventory attribute contains some number of software inventory evidence records along with the core response attribute fields. Given that the size of records can vary considerably, the length of this attribute is highly variable and, if transmitting a complete inventory, can be extremely large. Enterprises might wish to constrain the use of Software Inventory attributes to targeted requests to avoid over-burdening the network unnecessarily.

When copying a software inventory evidence record into the Record field, the record MUST be converted and normalized to use Network Unicode format prior to its inclusion in the record field.

5.11. Software Events

A SW-PC sends this attribute to a SW-PV to convey a list of events that include software inventory evidence records. A SW-PV MUST NOT send this attribute. The SW-PC either sends this attribute in fulfillment of an existing subscription where the establishing request has a Result Type of 0 and the Earliest EID is non-zero, or in direct response to a SW Request attribute where the Result Type is 0 and the Earliest EID is non-zero.

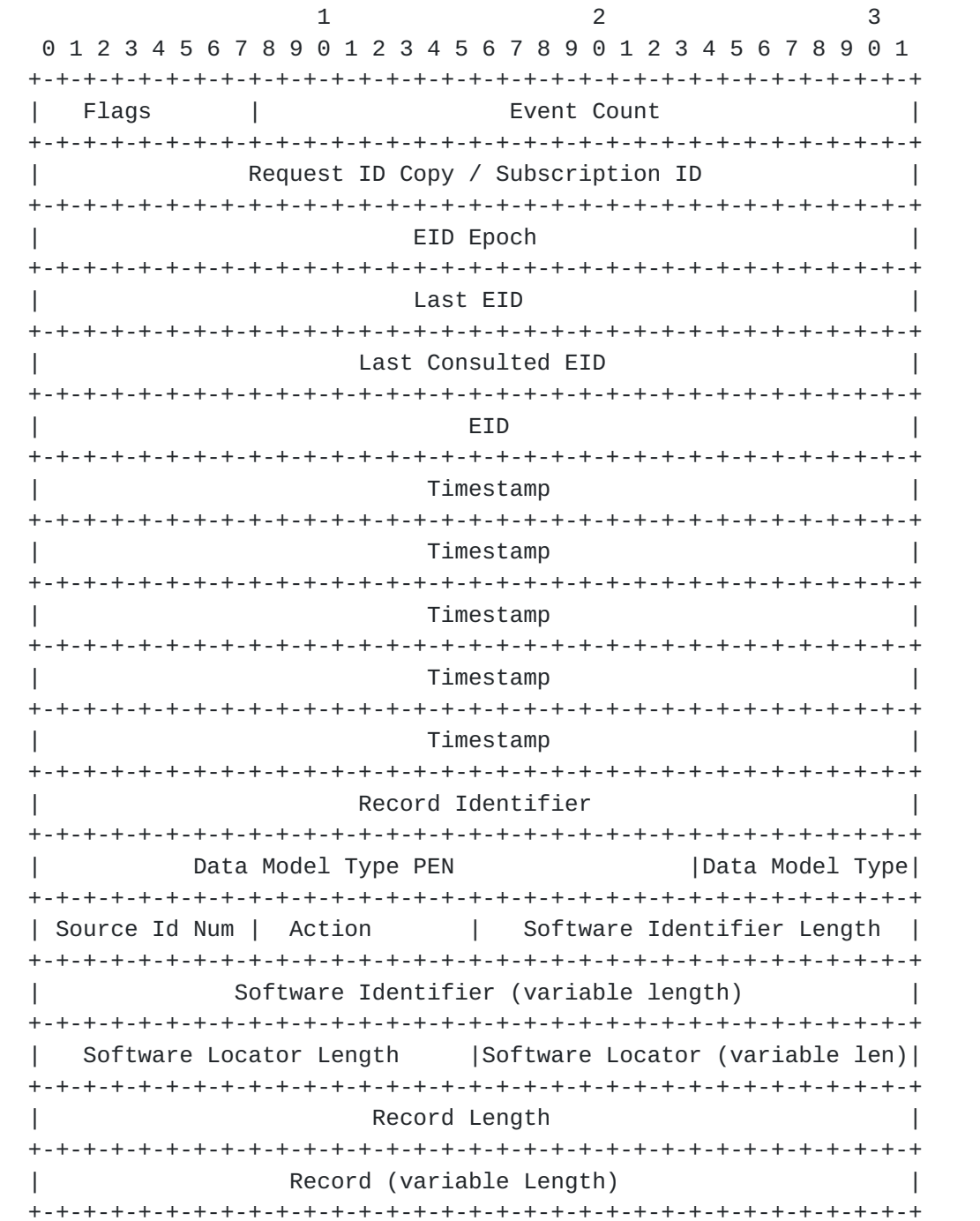


Figure 10: Software Events Attribute

Field	Description
Flags: Bit 0 - Subscription	In the case that this attribute is sent in fulfillment of a subscription this bit MUST be

Fulfillment	set (1). In the case that this attribute is a direct response to a SW Request, this bit MUST be unset (0).
Flags: Bit 1-7 - Reserved	Reserved for future use. This field MUST be set to zero on transmission and ignored upon reception.
Event Count	The number of events being reported in this attribute. This field is a 3-byte unsigned integer. The EID, Timestamp, Record Identifier, Data Model Type PEN, Data Model Type, Source Identification Number, Action, Software Identifier Length, Software Identifier, Software Locator Length, Software Locator, Record Length, and Record fields are repeated, in order, the number of times indicated in this field. This field value MAY be 0, in which case there are no instances of these fields.
Request ID Copy / Subscription ID	In the case where this attribute is in direct response to a SW Request attribute from a SW-PV, this field MUST contain an exact copy of the Request ID field from that SW Request. In the case where this attribute is sent in fulfillment of an active subscription, this field MUST contain the Subscription ID of the subscription being fulfilled by this attribute.
EID Epoch	The EID Epoch of the Last EID value. This field is an unsigned 4-byte integer.
Last EID	The EID of the last event recorded by the SW-PC, or 0 if the SW-PC has no recorded events. This field contains the EID of the SW-PC's last recorded change event (which might or might not be included as an event record in this attribute).
Last Consulted EID	The EID of the last event record that was consulted when generating the event record list included in this attribute. This is different from the Last EID field value if and only if this attribute is conveying a partial list of event records. See Section 3.7.5 for more on partial list of event records.
EID	The EID of the event in this event record.

Timestamp	The timestamp associated with the event in this event record. This timestamp is the SW-PC's best understanding of when the given event occurred. Note that this timestamp might be an estimate. The Timestamp date and time MUST be represented as an RFC 3339 [5] compliant ASCII string in Coordinated Universal Time (UTC) time with the additional restrictions that the 'T' delimiter and the 'Z' suffix MUST be capitalized and fractional seconds (time-secfrac) MUST NOT be included. This field conforms to the date-time ABNF production from section 5.6 of RFC 3339 [RFC3339] with the above restrictions. Leap seconds are permitted and SW-PVs MUST support them. The Timestamp string MUST NOT be NULL terminated or padded in any way. The length of this field is always 20 octets.
Record Identifier	A 4-byte, unsigned integer containing the Record Identifier value from a software inventory evidence record.
Data Model Type PEN	A 3-byte unsigned integer containing the Private Enterprise Number (PEN) of the organization that assigned the meaning of the Data Model Type value.
Data Model Type	A 1-byte unsigned integer containing an identifier number that identifies the data model of the reported record.
Source Identification Number	The Source Identification Number associated with the source from which this software installation inventory instance was reported.
Action	The type of event that is recorded in this event record. Possible values are: 1 = CREATION - the addition of a record to the endpoint's Software Inventory Evidence Collection; 2 = DELETION - the removal of a record from the endpoint's Software Inventory Evidence Collection; 3 = ALTERATION - There was an alteration to a record within the endpoint's Software Inventory Evidence Collection. All other values are reserved for future use and MUST NOT be used when sending attributes. In the case where a SW-PV receives an event record that uses an action

		value other than the ones defined here, it MUST ignore that event record but SHOULD process other event records in this attribute as normal.
Software Identifier Length		A 2-byte unsigned integer indicating the length in bytes of the Software Identifier field.
Software Identifier		A string containing the Software Identifier value from a software inventory evidence record. This field value MUST be normalized to Network Unicode format, as described in Section 5.5. This string MUST NOT be NULL terminated.
Software Locator Length		A 2-byte unsigned integer indicating the length in bytes of the Software Locator field.
Software Locator		A string containing the Software Locator value. This is expressed as a URI. This field value MUST be normalized to Network Unicode format as described in Section 3.4.4. This string MUST NOT be NULL terminated.
Record Length		This is a 4-byte unsigned integer indicating the length of the Record field in bytes.
Record		A software inventory evidence record as a string. The record MUST be converted and normalized to Network Unicode format, as described in Section 5.5. This string MUST NOT be NULL terminated.

Table 7: Software Events Attribute Fields

The fields of this attribute are used in the same way as the corresponding fields of the previous attributes. As with the Software Inventory attribute, a Software Events attribute can be quite large if many events have occurred following the event indicated by a request's Earliest EID. As such, it is recommended that the SW Request attributes only request full records be sent (Result Type set to 0) in a targeted request, thus constraining the response just to records that match a given set of Software Identifiers.

As with the Software Identifier Events Attribute, this attribute MUST only contain event records with EIDs coming from the current EID Epoch of the SW-PC.

As with the Software Inventory Attribute, the SW-PC MUST perform conversion and normalization of the record.

5.12. Subscription Status Request

A SW-PV sends this attribute to a SW-PC to request a list of active subscriptions for which the requesting SW-PV is the subscriber. A SW-PC MUST NOT send this attribute.

This attribute has no fields.

A SW-PC MUST respond to this attribute by sending a Subscription Status Response attribute (or a PA-TNC Error attribute if it is unable to correctly provide a response).

5.13. Subscription Status Response

A SW-PC sends this attribute to a SW-PV to report the list of active subscriptions for which the receiving SW-PV is the subscriber. A SW-PV MUST NOT send this attribute.

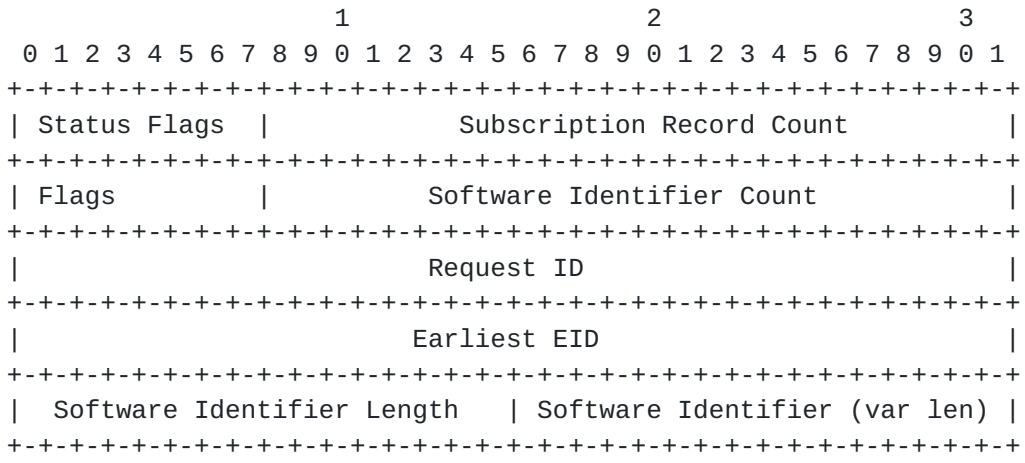


Figure 11: Subscription Status Response Attribute

Field	Description
Flags: Bit 0-7 - Reserved	Reserved for future use. This field MUST be set to zero on transmission and ignored upon reception.
Subscription Record Count	The number of subscription records that follow. This field is a 3-byte unsigned integer. The Flags, Software Identifier Count, Request ID, Earliest EID, Software Identifier Length, and Software Identifier fields are repeated, in order, the number of times indicated in this field. This field value MAY be 0 in which case there are no instances of these fields.
Flags, Software Identifier Count, Request ID, Earliest EID, Software Identifier Length, and Software Identifier	For each active subscription, these fields contain an exact copy of the fields with the same name as provided in the subscription's establishing request.

Table 8: Subscription Status Response Fields

A Subscription Status Response contains zero or more subscription records. Specifically, it MUST contain one subscription record for each active subscription associated with the party that sent the Subscription Status Request to which this attribute is a response. As described in Section 3.8.2, the SW-PC MUST use the requester's Connection ID and its Posture Validator ID to determine which subscriptions are associated with the requester.

A SW-PC MUST send a Subscription Status Response attribute in response to a Subscription Status Request attribute. The only exception to this is if the SW-PC experiences an error condition that prevents it from correctly populating the Subscription Status Response attribute, in which case it MUST respond with a PA-TNC Error attribute appropriate to the type of error experienced. If there are no active subscriptions associated with the requesting party, the Subscription Status Response attribute will consist of its Status Flags field, a Subscription Record Count field with a value of 0, and no additional fields.

Each subscription record included in a Subscription Status Response attribute duplicates the fields of a SW Request attribute that was the establishing request of a subscription. Note that the Request ID field in the record captures the Subscription ID associated with the given subscription record (since the Subscription ID is the same as the Request ID of the establishing request). Note also that if the establishing request is targeted, then its Record Count field will be non-zero and, within that subscription record, the Software Identifier Length and Software Identifier fields are repeated, in order, the number of times indicated in the Record Count field. As such, each subscription record can be different sizes. If the establishing request is not targeted (Record Count field is 0), the subscription record has no Software Identifier Length or Software Identifier fields.

When a SW-PV compares the information received in a Subscription Status Response to its own records of active subscriptions it should be aware that the SW-PC might be unable to distinguish this SW-PV from other SW-PVs on the same NEA Server. As a result, it is possible that the SW-PC will report more subscription records than the SW-PV recognizes. For this reason, SW-PVs SHOULD NOT automatically assume that extra subscriptions reported in a Subscription Status Response indicate a problem.

5.14. Source Metadata Request

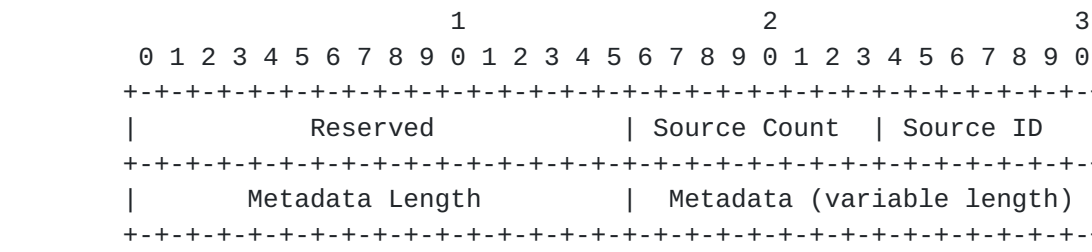
A SW-PV sends this attribute to a SW-PC to request metadata about sources that the SW-PC is using to collect software inventory information. A SW-PC MUST NOT send this attribute.

This attribute has no fields.

A SW-PC MUST respond to this attribute by sending a Sources Metadata Response attribute (or a PA-TNC Error attribute if it is unable to correctly provide a response).

5.15. Source Metadata Response

A SW-PC sends this attribute to a SW-PV to provide descriptive metadata about the sources of software inventory information used by the SW-PC. A SW-PV MUST NOT send this attribute.



Source Metadata Response Attribute

Field	Description
Reserved	Reserved for future use. This field MUST be set to zero on transmission and ignored upon reception.
Source Count	The number of source records that follow. The Source ID, Metadata Size, and Metadata fields are repeated, in order, the number of times indicated by this field. This field MAY be 0, in which case no fields follow (but this would only be done to indicate that the SW-PC has no active sources, which would not be a usual situation).
Source ID	The Source ID number associated with the described source for any communications with the recipient SW-PV.
Metadata Length	A 2-byte unsigned integer indicating the length in bytes of the Metadata field.
Metadata	A string containing descriptive metadata about the indicated data source. This string MUST NOT be NULL terminated.

Source Metadata Response Fields

A Source Metadata Response attribute contains 0 or more records, each describing one of the data sources the SW-PC uses to collect software inventory information. It SHOULD contain one metadata record for each source that the SW-PC uses. (There might be reasons not to inform certain SW-PVs of the presence of certain data sources.) The attribute MUST contain a metadata record for each source that has been identified in inventory or event messages to the given SW-PV.

A SW-PC MUST send a Source Metadata Response attribute in response to a Source Metadata Request attribute. The only exception to this is if the SW-PC experiences an error condition that prevents it from correctly populating the Source Metadata Response attribute, in which case it MUST respond with a PA-TNC Error attribute appropriate to the type of error experienced.

The Source Count field indicates how many source metadata records are included in the attribute. Each included record consists of a Source Identifier, Metadata Length, and Metadata field.

The Source Identifier corresponds to the Source Identifier field in inventory and event messages. In the case where the Source Identifier value in a Source Metadata Response attribute matches a Source Identifier associated with an event or inventory record and both the Source Metadata Response and the inventory/event record were sent to the same SW-PV, the source described in the Metadata field MUST be the same source that provided the event or inventory record associated with the Source Identifier. Recall that a SW-PC MAY use different Source Identifier associations with different SW-PVs. When sending to a giving SW-PV, the SW-PC MUST use the recipient SW-PVs Source Identifier associations.

The Metadata Length indicates the length, in bytes, if the Metadata field. The Metadata field contains information about the indicated data source. This specification does not dictate a format for the contents of the Metadata field. This field MAY include machine-readable information. For broadest utility, the Metadata field SHOULD include human-readable, descriptive information about the data source.

5.16. PA-TNC Error as Used by Software Inventory Message and Attributes for PA-TNC

The PA-TNC Error attribute is defined in the PA-TNC specification [RFC5792], and its use here conforms to that specification. A PA-TNC Error can be sent due to any error in the PA-TNC exchange and might also be sent in response to error conditions specific to the Software Inventory Message and Attributes for PA-TNC exchange. The latter case utilizes error codes defined below.

A PA-TNC Error MUST be sent by a SW-PC in response to a SW Request in the case where the SW-PC encounters a fatal error (i.e., an error that prevents further processing of an exchange) relating to the attribute exchange. A SW-PV MUST NOT send this attribute. In the case where the SW-PV experiences a fatal error, it MUST handle the error without sending a PA-TNC Error attribute. The SW-PV MAY take

other actions in response to the error, such as logging the cause of the error, or even taking actions to isolate the endpoint.

A PA-TNC Error attribute is sent instead of a SW Response attribute due to factors that prevent the reliable creation of a SW Response. As such, a SW-PC MUST NOT send both a PA-TNC Error attribute and a SW Response attribute in response to a single SW Request attribute.

Table 9 lists the Error Code values for the PA-TNC Error attribute specific to the Software Inventory Message and Attributes for PA-TNC exchange. Error codes are shown in both hexadecimal and decimal format. In all of these cases, the Error Code Vendor ID field MUST be set to 0x000000, corresponding to the IETF SMI Private Enterprise Number. The Error Information structures for each error type are described in the following subsections.

Note that a message with a Software Inventory Message and Attributes for PA-TNC attribute might also result in an error condition covered by the Standard PA-TNC Error Codes defined in PA-TNC. For example, a SW Attribute might have an invalid parameter, leading to an error code of "Invalid Parameter". In this case, the SW-PC MUST use the appropriate PA-TNC Error Code value as defined in Section 4.2.8 of PA-TNC specification.

Error Code Value	Description
0x00000020 (32)	SW_ERROR. This indicates a fatal error (i.e., an error that precludes the creation of a suitable response attribute) other than the errors described below but still specific to the processing of SW Attributes. The Description field SHOULD contain additional diagnostic information.
0x00000021 (33)	SW_SUBSCRIPTION_DENIED_ERROR. This indicates that the SW-PC denied the SW-PV's request to establish a subscription. The Description field SHOULD contain additional diagnostic information.
0x00000022 (34)	SW_RESPONSE_TOO_LARGE_ERROR. This indicates that the SW-PC's response to the SW-PV's request was too large to be serviced. The error information structure indicates the largest possible size of a response supported by the SW-PC (see

		Section 5.16.2). The Description field SHOULD contain additional diagnostic information.
0x00000023 (35)	SW_SUBSCRIPTION_FULFILLMENT_ERROR. This	indicates that the SW-PC experienced an error fulfilling a given subscription. The error information includes the Subscription ID of the relevant subscription, as well as a sub-error that describes the nature of the error the SW-PC experienced. The SW-PC and SW-PV MUST treat the identified subscription as cancelled.
0x00000024 (36)	SW_SUBSCRIPTION_ID_REUSE_ERROR. This	indicates that the SW-PC received a SW Request from a given SW-PV where the Request ID of that SW Request is currently used as the Subscription ID of an active subscription with that SW-PV. This error does not cancel the identified subscription.
0x00000025-0x0000002F (37-47)	RESERVED. These Error Codes are reserved	for use by future revisions of the Software Inventory Message and Attributes for PA-TNC specification. Any PA-TNC Error attribute using one of these Error Codes MUST be treated as indicating a fatal error on the sender without further interpretation.

Table 9: PA-TNC Error Codes for Software Inventory Message and Attributes for PA-TNC

The following subsections describe the structures present in the Error Information fields.

5.16.1. SW_ERROR, SW_SUBSCRIPTION_DENIED_ERROR and SW_SUBSCRIPTION_ID_REUSE_ERROR Information

The SW_ERROR error code indicates that the sender (the SW-PC) has encountered an error related to the processing of a SW Request attribute but which is not covered by more specific SW error codes. The SW_SUBSCRIPTION_DENIED_ERROR is used when the SW-PV requests to establish a subscription or clear all subscriptions from the given

SW-PV, but the SW-PC is unable or unwilling to comply with this request. The SW_SUBSCRIPTION_ID_REUSE_ERROR is used when the SW-PC receives a SW Request whose Request ID duplicates a Subscription ID of an active subscription with the request's sender. All of these error codes use the following error information structure.

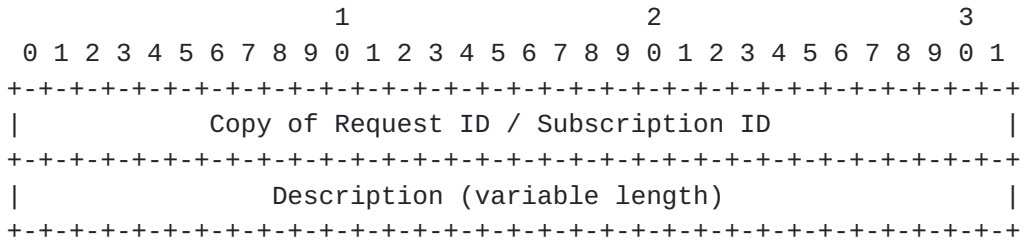


Figure 12: SW Error, Subscription Error, and Subscription ID Reuse Information

Field	Description
Copy of Request ID / Subscription ID	In the case that this error condition is generated in direct response to a SW Request attribute, this field MUST contain an exact copy of the Request ID field in the SW Request attribute that caused this error. In the case that the attribute in question is generated in fulfillment of an active subscription, this field MUST contain the Subscription ID of the subscription for which the attribute was generated. (This is only possible if the error code is SW_ERROR as the other errors are not generated by subscription fulfillment.) Note that, in this case, the indicated error appears as a sub-error for a SW_SUBSCRIPTION_FULFILLMENT_ERROR, as described in Section 5.16.3.
Description	A UTF-8 string describing the condition that caused this error. This field MAY be 0-length. However, senders SHOULD include some description in all PA-TNC Error attributes of these types. This field MUST NOT be NULL terminated.

Table 10: SW Error, Subscription Error, and Subscription ID Reuse Information Fields

This error information structure is used with SW_ERROR, SW_SUBSCRIPTION_DENIED_ERROR, and SW_SUBSCRIPTION_ID_REUSE_ERROR status codes to identify the SW Request attribute that precipitated the error condition and to describe the error. The Description field contains text describing the error. The SW-PC MAY encode machine-interpretable information in this field, but SHOULD also include a human-readable description of the error, since the receiving SW-PV might not recognize the SW-PC's encoded information.

5.16.2. SW_RESPONSE_TOO_LARGE_ERROR Information

The SW_RESPONSE_TOO_LARGE_ERROR error code indicates that a response generated by a SW-PC in response to a SW-PV's SW Request attribute was too large to send.

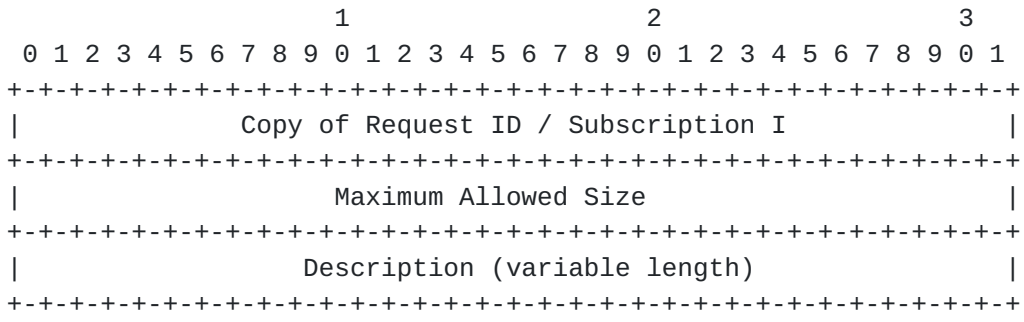


Figure 13: SW Response Too Large Error Information

Field	Description
Copy of Request ID / Subscription ID	In the case that the attribute in question is generated in direct response to a SW Request, this field MUST contain an exact copy of the Request ID field in the SW Request attribute that caused this error. In the case that the attribute in question is generated in fulfillment of an active subscription, this field MUST contain the Subscription ID of the subscription for which the attribute was generated. Note that, in the latter case, the SW_RESPONSE_TOO_LARGE_ERROR appears as a sub-error for a SW_SUBSCRIPTION_FULFILLMENT_ERROR, as described in Section 5.16.3.
Maximum Allowed Size	This field MUST contain an unsigned integer indicating the largest permissible size, in bytes, of SW Attribute that the SW-PC is currently willing to send in response to a SW Request attribute.
Description	A UTF-8 string describing the condition that caused this error. This field MAY be 0-length. However, senders SHOULD include some description in all PA-TNC Error attributes of these types. This field MUST NOT be NULL terminated.

Table 11: SW Response Too Large Error Information Fields

This error structure is used with the SW_RESPONSE_TOO_LARGE_ERROR status code to identify the SW Request attribute that precipitated the error condition and to describe the error. The Maximum Allowed Size field indicates the largest attribute the SW-PC is willing to send in response to a SW Request under the current circumstances. Note that under other circumstances, the SW-PC might be willing to return larger or smaller responses than indicated (such as if the endpoint connects to the NEA Server using a different network protocol). The other fields in this error information structure have the same meanings as corresponding fields in the SW_ERROR and SW_SUBSCRIPTION_DENIED_ERROR information structure.

5.16.3. SW_SUBSCRIPTION_FULFILLMENT_ERROR Information

The SW_SUBSCRIPTION_FULFILLMENT_ERROR error code indicates that the SW-PC encountered an error while fulfilling a subscription. The bytes after the first 4 octets duplicate a PA-TNC Error attribute (as

described in Section 4.2.8 of PA-TNC) that is used to identify the nature of the encountered error.

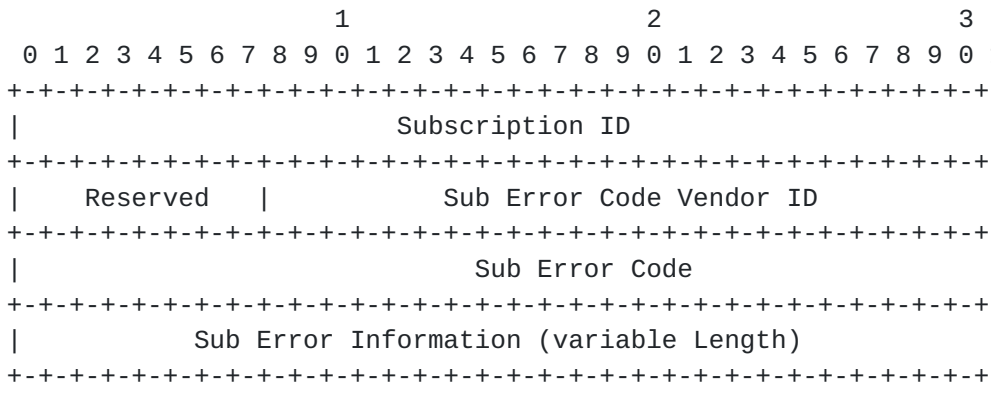


Figure 14: SW Subscription Fulfillment Error Information

Field	Description
Subscription ID	This field MUST contain the Subscription ID of the subscription whose fulfillment caused this error.
Reserved	This field MUST contain the value of the Reserved field of a PA-TNC Error attribute that describes the error condition encountered during subscription processing.
Sub Error Code Vendor ID	This field MUST contain the value of the Error Code Vendor ID field of a PA-TNC Error attribute that describes the error condition encountered during subscription processing.
Sub Error Code	This field MUST contain the value of the Error Code field of a PA-TNC Error attribute that describes the error condition encountered during subscription processing.
Sub Error Information	This field MUST contain the value of the Error Information field of a PA-TNC Error attribute that describes the error condition encountered during subscription processing.

Table 12: SW Subscription Fulfillment Error Information Fields

This error structure is used with the SW_SUBSCRIPTION_FULFILLMENT_ERROR status code. The first 4 octets of this error structure contain the Subscription ID of the subscription that was being fulfilled when the error occurred. The remaining fields of this error structure duplicate the fields of a PA-TNC Error attribute, referred to as the "sub-error". The error code of the sub-error corresponds to the type of error that the SW-PC encountered while fulfilling the given subscription. The sub-error MUST NOT have an error code of SW_SUBSCRIPTION_FULFILLMENT_ERROR.

The SW-PC sending a PA-TNC Error attribute with this error code, and the SW-PV receiving it, MUST treat the subscription identified by the Subscription ID field as cancelled. All other subscriptions are unaffected.

6. Supported Data Models

Software Inventory Message and Attributes for PA-TNC supports an extensible list of data models for representing and transmitting software inventory information. This list of data models appears in the Software Data Model IANA registry (see Section 10.4). This document provides guidance for an initial set of data models. Other documents might provide guidance on the use of new data models by Software Inventory Message and Attributes for PA-TNC, and will be referenced by extensions to the Software Data Model IANA registry.

6.1. ISO 2015 SWID Tags using XML

The International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC) published the specification governing SWID tag construction and use in 2009 with a revised version published in 2015. [SWID15] Since that time, a growing number of vendors have integrated SWID tags into their software products. Doing so significantly simplifies the task of identifying these pieces of software: instead of relying on discovery processes that look for clues as to software presence, such as the presence of particular files or registry keys. A readily available list of SWID tags provides simple and immediate evidence as to the presence of the given piece of software.

SWID Message and Attributes for PA-TNC has no reliance on the presence or management of SWID tags on an endpoint as described in the ISO specification. However, the data model for describing software that is presented in the ISO specification provides a robust and comprehensive way of describing software and is adopted here as a means of representing and transmitting software information. It should be emphasized, the use of the ISO SWID tag data model makes no assumption as to whether the source of the recorded information was,

in fact, an ISO SWID tag harvested from the endpoint or whether the information was created using some other source and normalized to the SWID format.

6.1.1. Guidance on Normalizing Source Data to ISO 2015 SWID Tags using XML

Any record associated with this Software Data Model Type MUST conform to the ISO/IEC 19770-2-2015 [SWID15] specification. Any such tag SHOULD use a UTF-8 encoding, but MUST NOT alter the existing encoding if doing so would invalidate digital signatures included in the tag.

If generating a new ISO 2015 SWID tag, the software generating the tag MUST use a Tag Creator RegID that is associated with the generating software, unless this is impossible, in which case it MUST use the "http://invalid.unavailable" Tag Creator RegID value. Do not use a RegID associated with any other party. In particular, it is incorrect to use a Tag Creator RegID associated with the software being described by the tag, the enterprise that is using the software, or any other entity except that of the party that built the tool that is generating the SWID tag. This reflects the requirement that the Tag Creator RegID identify the party that created the tag.

6.1.2. Guidance on Creation of Software Identifiers from ISO 2015 SWID Tags

A Software Identifier generated from an ISO 2015 SWID tag is expressed as a concatenation of the value of the Tag Creator RegID field and the Unique ID field. Specifically, it MUST be of the form: TAG_CREATOR_REGID "_" "_" UNIQUE_ID. Specifically, the Software Identifier consists of, the Tag Creator RegID and the Unique ID from the tag connected with a double underscore (_), without any other connecting character or whitespace.

6.2. ISO 2009 SWID Tags using XML

As noted above, ISO's SWID tag specification provides a useful data model for representation of software information. As of the writing of this specification, while the 2015 specification is considered more comprehensive and addresses some issues with the 2009 specification, 2009-format SWID tags remain far more common in deployments. For this reason, ISO 2009 SWID tags are included in the Software Data Model IANA table.

6.2.1. Guidance on Normalizing Source Data to ISO 2009 SWID Tags using XML

Any record associated with this Software Data Model Type MUST conform to the ISO/IEC 19770-2-2009 [SWID09] specification. Any such tag SHOULD use a UTF-8 encoding, but MUST NOT alter the existing encoding if doing so would invalidate digital signatures included in the tag.

If generating a new ISO 2009 SWID tag, the software generating the tag MUST use a Tag Creator RegID that is associated with the generating software unless this is impossible, in which case it MUST use "http://invalid.unavailable", which indicates the Tag Creator is unknown. Do not use a RegID associated with any other party. In particular, it is incorrect to use a Tag Creator RegID associated with the software being described by the tag, the enterprise that is using the software, or any other entity except that of the party that built the tool that is generating the SWID tag. This reflects the requirement that the Tag Creator RegID identify the party that created the tag.

6.2.2. Guidance on Creation of Software Identifiers from ISO 2009 SWID Tags

A Software Identifier generated from an ISO 2009 SWID tag is expressed as a concatenation of the value of the Tag Creator RegID field and the Unique ID field. Specifically, it MUST be of the form: TAG_CREATOR_REGID "_" "_" UNIQUE_ID. Specifically, the Software Identifier consists of, the Tag Creator RegID and the Unique ID from the tag connected with a double underscore (_), without any other connecting character or whitespace.

7. Security Considerations

This section discusses some of the security threats facing Posture Collectors and Posture Validators that implement the Software Inventory Message and Attributes for PA-TNC protocol. This section primarily notes potential issues for implementers to consider, although it does contain a handful of normative requirements to address certain security issues. The issues identified below focus on capabilities specific to this document. Implementers are advised to consult other relevant NEA specifications, particularly [RFC5209] (the NEA Architecture) and [RFC5792] (PA-TNC), for security issues that are applicable to such components in general.

7.1. Evidentiary Value of Software Inventory Evidence Records

The degree to which an endpoint's Software Inventory Evidence Collection accurately reflects the endpoint's actual software load and any changes made to this software load is dependent on the accuracy of the tools used to populate and manage the software inventory evidence records in this collection. Some tools might not be designed to update records in the Software Inventory Evidence Collection in real time, resulting in a collection that is out-of-sync with actual system state. Moreover, tools might inaccurately characterize software, or fail to properly record its removal. Finally, it is likely that there will be software on the endpoint that is not tracked by any source and thus is not reflected in the Software Inventory Evidence Collection. Users of collected software inventory evidence records need to understand that the information provided by the Software Inventory Message and Attributes for PA-TNC capability cannot be treated as completely accurate. Nonetheless, having endpoints report this information can still provide useful insights into the state of the endpoint's software load, and can alert administrators and policy tools of situations that require remediation.

7.2. Sensitivity of Collected Records

Software records on an endpoint are generally not considered to be sensitive, although there can be exceptions to this generalization as noted in the section on Privacy Considerations. In general, an endpoint's Software Inventory Evidence Collection can be browsed and individual records read by any party with access to the endpoint. This is generally not considered to be problematic, as those with access to the endpoint can usually learn of everything disclosed by that endpoint's records simply by inspecting other parts of the endpoint.

The situation changes when an endpoint's inventory records are collected and stored off of the endpoint itself, such as on a NEA Server or CMDB. Inventory records represent a wealth of information about the endpoint in question and, for an adversary who does not already have access to the endpoint, a collection of the endpoint's inventory records might provide many details that are useful for mounting an attack. A list of the inventory records associated with an endpoint reveals a list of software installed on the endpoint. This list can be very detailed, noting specific versions and even patch levels, which an adversary can use to identify vulnerable software and design efficacious attacks.

In addition, other information might also be gleaned from a collection of software inventory records:

- An inventory record might include information about where the product was installed on a given endpoint. This can reveal details about the file organization of that endpoint that an attacker can utilize.

- An inventory record might include information about how the software was provided to the endpoint, who in an organization signs off on the package release, and who packaged the product for installation. This information might be used as a starting point for the development of supply chain attacks.

- Events affecting inventory records are reported with timestamps indicating when each given event occurred. This can give the attacker an indication of how quickly an organization distributes patches and updates, helping the attacker determine how long an attack window might remain open.

Any consolidated software inventory is a potential risk, because such an inventory can provide an adversary an insight into the enterprise's configuration and management process. It is recommended that a centralized software inventory record collection be protected against unauthorized access. Mechanisms to accomplish this can include encrypting the data at rest, ensuring that access to the data is limited only to authorized individuals and processes, and other basic security precautions.

7.3. Integrity of Endpoint Records

SW-PCs maintain records of detected changes to the endpoint's Software Inventory Evidence Collection. These records are used to respond to a SW-PV's request for change events. The SW-PV might use a list of reported events to update its understanding of the endpoint's Software Inventory Evidence Collection without needing to receive a full inventory report from the SW-PC. For this reason, preserving the integrity of the SW-PC's record of events is extremely important. If an attacker modifies the SW-PC's record of changes to the endpoint's Software Inventory Evidence Collection, this might cause the SW-PV's understanding of the endpoint's Software Inventory Evidence Collection to differ from its actual state. Results might include leading the SW-PV to believe that absent software was present, that present software was absent, that patches have been installed even if this is not the case, or to be unaware of other changes to Software Inventory Evidence Records. As such, the SW-PC MUST take steps to protect the integrity of its event records.

In addition, records of established SW-PV subscriptions also require protection against manipulation or corruption. If an attacker is able to modify or delete records of an established subscription by a

SW-PV, the SW-PC might fail to correctly fulfill this subscription. The SW-PV would not be aware that its subscription was not being correctly fulfilled unless it received additional information that indicated a discrepancy. For example, the SW-PV might collect a full inventory and realize from this that certain events had not been correctly reported in accordance with an established subscription. For this reason, the SW-PC MUST protect the integrity of subscription records.

7.4. SW-PC Access Permissions

A SW-PC requires sufficient permissions to collect Software Inventory Evidence Records from all of its supported sources, as well as sufficient permissions to interact with the endpoint's Posture Broker Client. With regard to the former, this might require permissions to read the contents of directories throughout the file system. Depending on the operating environment and other activities undertaken by a SW-PC (or software that incorporates a SW-PC as one of its capabilities) additional permissions might be required by the SW-PC software. The SW-PC SHOULD NOT be granted permissions beyond what it needs in order to fulfill its duties.

7.5. Sanitization of Record Fields

Not all sources of software inventory evidence are necessarily tightly controlled. For example, consider a source that gathers .swid files from the endpoint's file system. Any party could create a new .swid file that could be collected and turned into a Software Inventory Evidence Record. As a result, it is important that the contents of source information not be automatically trusted. In particular, tools that read source information and the Software Inventory Evidence Records derived therefrom, including SW-PCs, need to be careful to sanitize input to prevent buffer overflow attacks, encoding attacks, and other weaknesses that might be exploited by an adversary who can control the contents of a record.

7.6. PA-TNC Security Threats

In addition to the aforementioned considerations the Software Inventory Message and Attributes for PA-TNC protocol is subject to the same security threats as other PA-TNC transactions, as noted in Section 5.2 of PA-TNC [RFC5792]. These include, but are not limited to, attribute theft, message fabrication, attribute modification, attribute replay, attribute insertion, and denial of service. Implementers are advised to consult the PA-TNC specification to better understand these security issues.

8. Privacy Considerations

As noted in Section 7.2, if an adversary can gain an understanding of the software installed on an endpoint, they can utilize this to launch attacks and maintain footholds on this endpoint. For this reason, the NEA Server needs to ensure adequate safeguards are in place to prevent exposure of collected inventory records. For similar reasons, it is advisable that an endpoint only send records to a NEA Server that is authorized to receive this information and that can be trusted to safeguard this information after collection.

9. Relationship to Other Specifications

This specification makes frequent reference to and use of other specifications. This section describes these relationships.

This specification is expected to participate in a standard NEA architecture. As such, it is expected to be used in conjunction with the other protocols used in a NEA exchange. In particular, SW Attributes are conveyed over PB-TNC [RFC5793], which is in turn conveyed over some variant of PT (either PT-TLS [RFC6876] or PT-EAP [RFC7171]). These protocols have an especially important role, as they are responsible for ensuring that attributes defined under this specification are delivered reliably, securely, and to the appropriate party.

It is important to note that the Product Information, Numeric Version, and String Version attributes defined in the PA-TNC specification [RFC5792] are also meant to convey information about installed applications and the versions thereof. As such, there is some conceptual overlap between those attributes and the intent of this specification. However, PA-TNC was designed to respond to very specific queries about specific classes of products, while the Software Inventory Message and Attributes for PA-TNC specification is able to convey a broader query, resulting in a more comprehensive set of evidence regarding an endpoint's installed software. As such, this specification provides important capabilities not present in the PA-TNC specification.

10. IANA Considerations

This section extends multiple existing IANA registries. Specifically, it extends the PA-TNC Attribute Types and PA-TNC Error Codes defined in the PA-TNC specification [RFC5792] and the PA-Subtypes registry defined in the PB-TNC specification [RFC5793] and extended in PA-TNC. This specification only adds values to these registries and does not alter how these registries work or are

maintained. Consult the appropriate specifications for details on the operations and maintenance of these registries.

10.1. PA Subtypes

The following is an extension to the PA Subtype registry [1] defined in section 7.2 of the PA-TNC specification [RFC5792].

PEN	Integer	Name	Defining Specification
0	9	SW Attributes	Software Inventory Message and Attributes for PA-TNC

10.2. Registry for PA-TNC Attribute Types

Section 5.4 of this specification defines several new PA-TNC attributes. The following values are added to the registry for PA-TNC Attribute Types defined in the PA-TNC specification. Note that Table 2 in Section 5.4 lists these attributes but uses a hexadecimal value to identify their associated integer. The integer values given in that table are identical to those provided here. Note also that Table 2 includes an entry for PA-TNC Error attributes, but the IANA information associated with that attribute is already defined in the PA-TNC specification and is not reproduced here.

PEN	Integer	Name	Defining Specification
0	17	SW Request	Software Inventory Message and Attributes for PA-TNC
0	18	Software Identifier Inventory	Software Inventory Message and Attributes for PA-TNC
0	19	Software Identifier Events	Software Inventory Message and Attributes for PA-TNC
0	20	Software Inventory	Software Inventory Message and Attributes for PA-TNC
0	21	Software Events	Software Inventory Message and Attributes for PA-TNC
0	22	Subscription Status Request	Software Inventory Message and Attributes for PA-TNC
0	23	Subscription Status Response	Software Inventory Message and Attributes for PA-TNC
0	24	Source Metadata Request	Software Inventory Message and Attributes for PA-TNC
0	25	Source Metadata Response	Software Inventory Message and Attributes for PA-TNC
0	26 - 31	Reserved for future use	Software Inventory Message and Attributes for PA-TNC

10.3. Registry for PA-TNC Error Codes

Section 5.16 of this specification defines several new PA-TNC Error Codes. The following values are added to the registry for PA-TNC Error Codes defined in the PA-TNC specification. Note that Table 9 in Section 5.16 lists these codes but uses a hexadecimal value to identify their associated integer. The integer values given in that table are identical to those provided here.

PEN	Integer	Name	Defining Specification
0	32	SW_ERROR	Software Inventory Message and Attributes for PA-TNC
0	33	SW_SUBSCRIPTION_DENIED_ERROR	Software Inventory Message and Attributes for PA-TNC
0	34	SW_RESPONSE_TOO_LARGE_ERROR	Software Inventory Message and Attributes for PA-TNC
0	35	SW_SUBSCRIPTION_FULFILLMENT_ERROR	Software Inventory Message and Attributes for PA-TNC
0	36	SW_SUBSCRIPTION_ID_REUSE_ERROR	Software Inventory Message and Attributes for PA-TNC
0	37-47	Reserved for future use	Software Inventory Message and Attributes for PA-TNC

10.4. Registry for Software Data Models

The name for this registry is "Software Data Model Types". Each entry in this registry should include a human readable name, an SMI Private Enterprise Number, a decimal integer value between 1 and 2^8-1 (inclusive), and a reference to the specification where the use of this data model is defined. This specification needs to provide

both a description of the format used by the data model and the procedures by which Software Identifiers are derived from a record expressed using this data model. Note that a specification that just defines the data model structure and its use is generally not sufficient as it would likely lack the procedures for constructing a Software Identifier. This is why the table below references the current specification for ISO SWID tags, rather than using the actual ISO SWID tag specification.

The following entries for this registry are defined in this document. They are the initial entries in the registry for Software Data Model Types. Additional entries to this registry are added by Expert Review with Specification Required, following the guidelines in [RFC5226].

PEN	Integer	Name	Defining Specification
0	1	ISO 2015 SWID Tags using XML	**CURRENT SPECIFICATION**
0	2	ISO 2009 SWID Tags using XML	**CURRENT SPECIFICATION**
0	192-255	Reserved for local enterprise use	N/A

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <http://www.rfc-editor.org/info/rfc3339>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <http://www.rfc-editor.org/info/rfc3986>.

[RFC5198] Klensin, J. and M. Padlipsky, "Unicode Format for Network Interchange", RFC 5198, DOI 10.17487/RFC5198, March 2008, <http://www.rfc-editor.org/info/rfc5198>.

[RFC5209] Sangster, P., Khosravi, H., Mani, M., Narayan, K., and J. Tardo, "Network Endpoint Assessment (NEA): Overview and Requirements", RFC 5209, DOI 10.17487/RFC5209, June 2008, <http://www.rfc-editor.org/info/rfc5209>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

[RFC5792] Sangster, P. and K. Narayan, "PA-TNC: A Posture Attribute (PA) Protocol Compatible with Trusted Network Connect (TNC)", RFC 5792, DOI 10.17487/RFC5792, March 2010, <http://www.rfc-editor.org/info/rfc5792>.

[SWID09] The International Organization for Standardization/ International Electrotechnical Commission, "Information Technology - Software Asset Management - Part 2: Software Identification Tag, ISO/IEC 19770-2", November 2009.

[SWID15] The International Organization for Standardization/ International Electrotechnical Commission, "Information Technology - Software Asset Management - Part 2: Software Identification Tag, ISO/IEC 19770-2", October 2015.

11.2. Informative References

[RFC5793] Sahita, R., Hanna, S., Hurst, R., and K. Narayan, "PB-TNC: A Posture Broker (PB) Protocol Compatible with Trusted Network Connect (TNC)", RFC 5793, DOI 10.17487/RFC5793, March 2010, <http://www.rfc-editor.org/info/rfc5793>.

[RFC6876] Sangster, P., Cam-Winget, N., and J. Salowey, "A Posture Transport Protocol over TLS (PT-TLS)", RFC 6876, DOI 10.17487/RFC6876, February 2013, <http://www.rfc-editor.org/info/rfc6876>.

[RFC7171] Cam-Winget, N. and P. Sangster, "PT-EAP: Posture Transport (PT) Protocol for Extensible Authentication Protocol (EAP) Tunnel Methods", RFC 7171, DOI 10.17487/RFC7171, May 2014, <http://www.rfc-editor.org/info/rfc7171>.

11.3. URIs

[1] https://www.iana.org/assignments/pb-tnc-parameters/pb-tnc-parameters.xhtml#pb-tnc-parameters-2

Authors' Addresses

Charles Schmidt
The MITRE Corporation
202 Burlington Road
Bedford, MA 01730
USA

Email: cmschmidt@mitre.org

Daniel Haynes
The MITRE Corporation
202 Burlington Road
Bedford, MA 01730
USA

Email: dhaynes@mitre.org

Chris Coffin
The MITRE Corporation
202 Burlington Road
Bedford, MA 01730
USA

Email: ccoffin@mitre.org

David Waltermire
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, Maryland
USA

Email: david.waltermire@nist.gov

Jessica Fitzgerald-McKay
Department of Defense
9800 Savage Road
Ft. Meade, Maryland
USA

Email: jmfitz2@nsa.gov