TCG Trusted Network Connect

TNC IF-TNCCS: Protocol Bindings for SoH

Specification Version 1.0
Revision 0.08
21 May 2007
Published

Contact: admin@trustedcomputinggroup.org.

TCG PUBLISHED

TCG

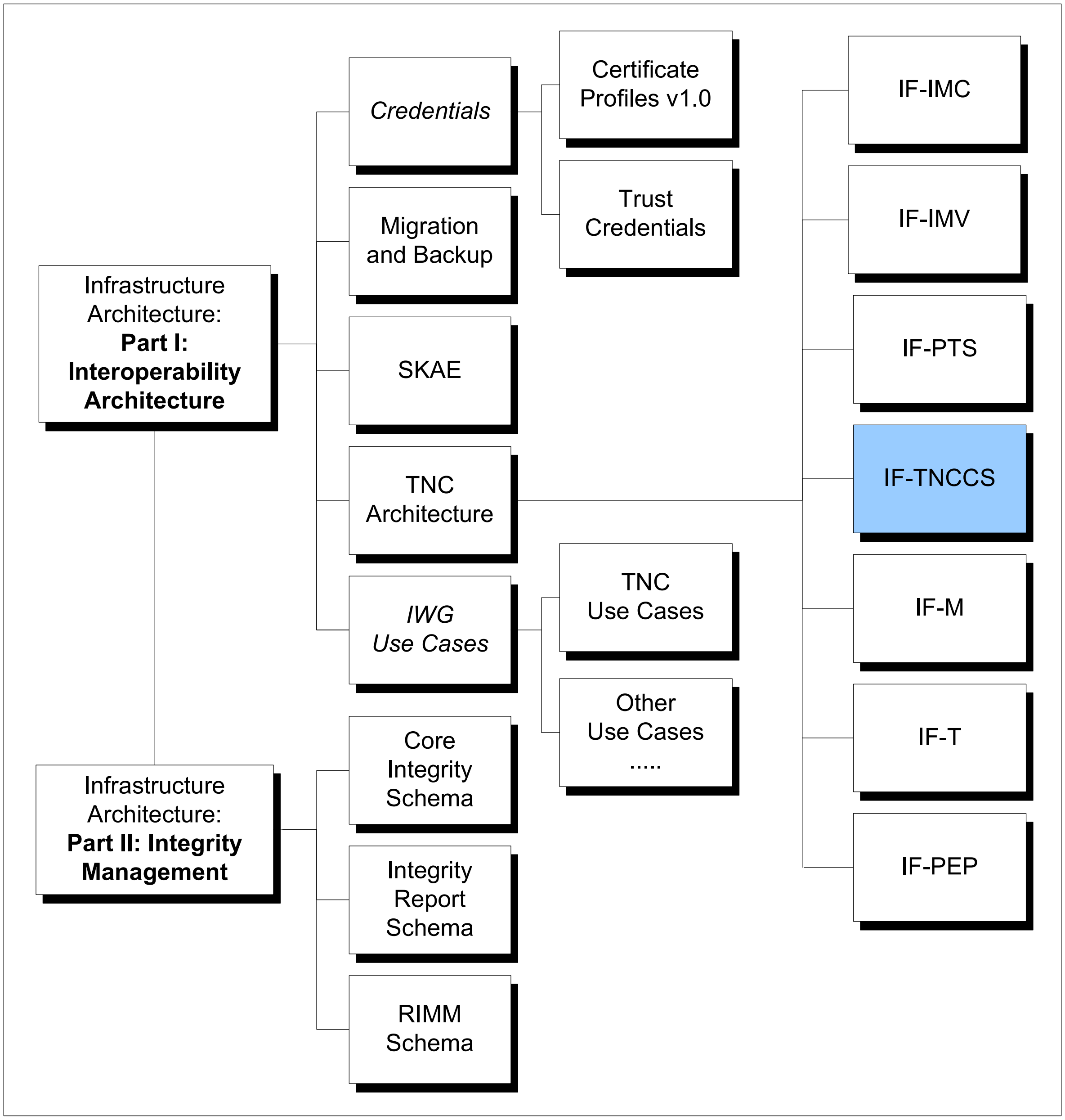
Disclaimer

THIS SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Without limitation, TCG disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification and to the implementation of this specification, and TCG disclaims all liability for cost of procurement of substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, direct, indirect, or special damages, whether under contract, tort, warranty or otherwise, arising in any way out of use or reliance upon this specification or any information herein.

Contact the Trusted Computing Group at www.trustedcomputinggroup.org for information on specification licensing through membership agreements.

Any marks and brands contained herein are the property of their respective owners.

TNC Document Roadmap



Acknowledgements

The TCG wishes to thank all those who contributed to this specification.

Scott Kelly	**Aruba Networks**
Amit Agarwal	**Avaya**
Mahalingam Mani	**Avaya**
Sung Lee	**Fujitsu Limited**
Mauricio Sanchez	**Hewlett-Packard**
Diana Arroyo	**IBM**
Chris Hessing	**Identity Engines**
Morteza Ansari	**Infoblox**
Stuart Bailey	**Infoblox**
Ivan Pulleyn	**Infoblox**
Ravi Sahita	**Intel Corporation**
Steve Hanna (TNC co-chair)	**Juniper Networks**
PJ Kirner	**Juniper Networks**
Lisa Lorenzin	**Juniper Networks**
Dean Sheffield	**Juniper Networks**
John Jerrim	**Lancope**
Ryan Hurst (Editor)	**Microsoft Corporation**
Chris Salter	**National Security Agency**
Meenakshi Kaushik	**Nortel**
Ron Pon	**Nortel**
Paul Sangster (TNC co-chair)	**Symantec Corporation**
Curtis Simonson	**University of New Hampshire InterOperability Labs**
Rod Murchison	**Vernier Networks**
Michelle Sommerstad	**Vernier Networks**
Thomas Hardjono	**Wave Systems**
Greg Kazmierczak	**Wave Systems**

Special thanks to the following individuals who contributed to the discussions and/or wordings used in this document:

Ram Vadali	Microsoft
Mudit Goel	Microsoft
Paul Mayfield	Microsoft
Tom Kelnar	Microsoft
Priyank Patel	Microsoft
Khaja Ahmed	Microsoft
Calvin Choe	Microsoft
Bernard Aboba	Microsoft
Jun Wang	Microsoft

Table of Contents

1 Introduction

1.1 Scope and Audience

The Trusted Network Connect Work Group (TNCWG) is defining an open solution architecture that enables network operators to enforce policies regarding endpoint integrity when granting access to a network infrastructure. Integrity measurements are carried between the TNC Client and TNC Server on a protocol called IF-TNCCS (Trusted Network Connect Client-Server), as shown in figure 1 below.

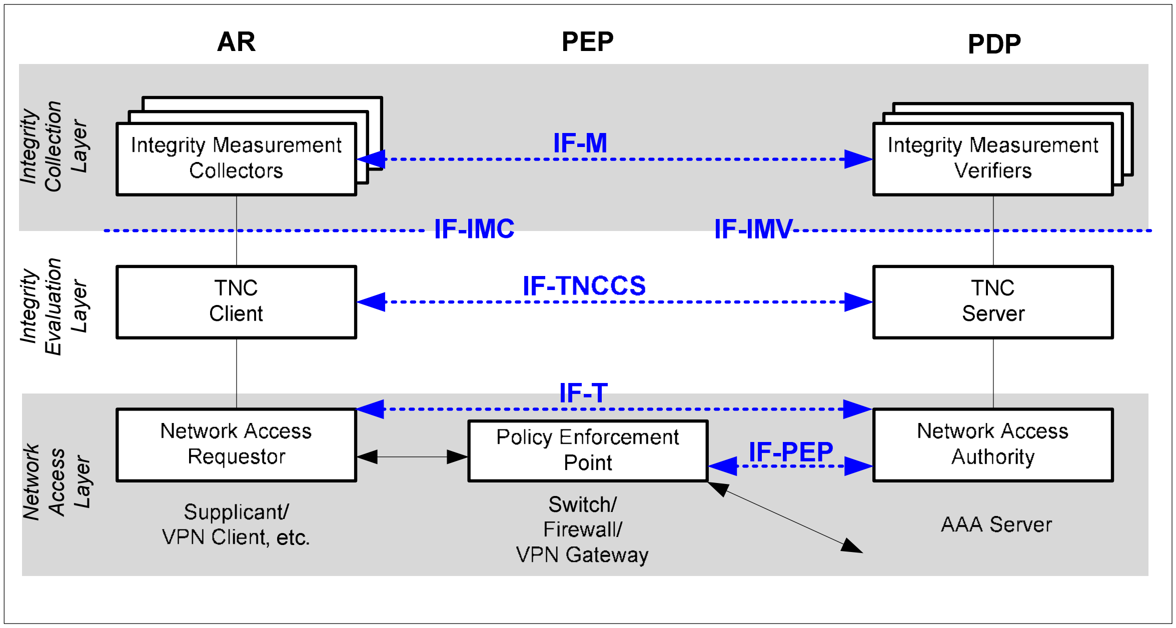

AR
PEP
PDP
Integrity Collection Layer
Integrity Evaluation Layer
Network Access Layer
Integrity Measurement Collectors
IF-M
Integrity Measurement Verifiers
IF-IMC
IF-IMV
TNC Client
IF-TNCCS
TNC Server
IF-T
Network Access Requestor
Policy Enforcement Point
IF-PEP
Network Access Authority
Supplicant/ VPN Client, etc.
Switch/ Firewall/ VPN Gateway
AAA Server

Figure 1 - TNC Architecture

This specification defines an alternate binding for IF-TNCCS called IF-TNCCS-SOH. This protocol is identical to the Statement of Health (SOH) protocol used by Microsoft Network Access Protection (NAP) [10] systems and therefore enables interoperability with such systems. This protocol was not invented by the TNC, but is considered a valid IF-TNCCS protocol within the TNC architecture. Future versions of the SOH binding for IF-TNCCS will be specified by the TNC group. Architects, designers, developers, and technologists interested in the development, deployment, and interoperation of trusted systems will find this document necessary in providing a specific mechanism for communicating integrity information.

Before reading this document any further, the reader should review and understand the TNC architecture as described in [7]. If the reader is building a TNC Client that supports IF-IMC, the reader is encouraged to read [3] prior to reading this document. If the reader is building a TNC Server that supports IF-IMV, the reader is encouraged to read [4] prior to reading this document.

1.2 Keywords

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC 2119 [1] 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.

2 Background

2.1 Role of IF-TNCCS-SOH

The SoH binding for IF-TNCCS (herein referred to as IF-TNCCS-SOH) plays the same role in the TNC architecture as the XML binding of IF-TNCCS. It describes a standard way for the TNC Client and the TNC Server to exchange messages. However, IF-TNCCS-SOH does this in a manner that is compatible with the Microsoft Network Access Protection (NAP) system. More specifically, this interface defines a protocol and format for carrying:

(a) Messages from IMCs to IMVs (such as integrity measurements)

(b) Messages from IMVs to IMCs (such as requests for additional integrity measurements, or remediation instructions)

(c) Messages from TNC Clients to TNC Servers (such as control messages)

(d) Messages from TNC Servers to TNC Clients (such as the TNCCS-Recommendation message)

Note that the contents of the messages being passed between the IMCs and IMVs ((a) and (b) above) are opaque to the IF-TNCCS-SOH layer. IF-TNCCS-SOH relies on the underlying transport protocol (IF-T) to provide a secure authenticated channel to protect the messages in transit between the TNC Client and the TNC Server and ensure they are delivered to the correct TNCC or TNCS.

2.2 Differences between IF-TNCCS and IF-TNCCS-SOH

The key differences between IF-TNCCS and IF-TNCCS-SOH can be summarized as:

- IF-TNCCS-SOH is Type-Length-Value (TLV) based while IF-TNCCS is XML based.
- IF-TNCCS-SOH supports only a single exchange of one 4KB packet while IF-TNCCS supports multiple exchanges and no packet size restriction
- IF-TNCCS-SOH exchanges only reason code identifiers between TNC client and TNC server while IF-TNCCS exchanges user locality and strings.
- IF-TNCCS-SOH does not yet support integration with Trusted Platform Modules for boot measurement information.

2.3 NAP and TNC Architectures

Figure 2 demonstrates the mapping from TNC components to components in the NAP architecture.

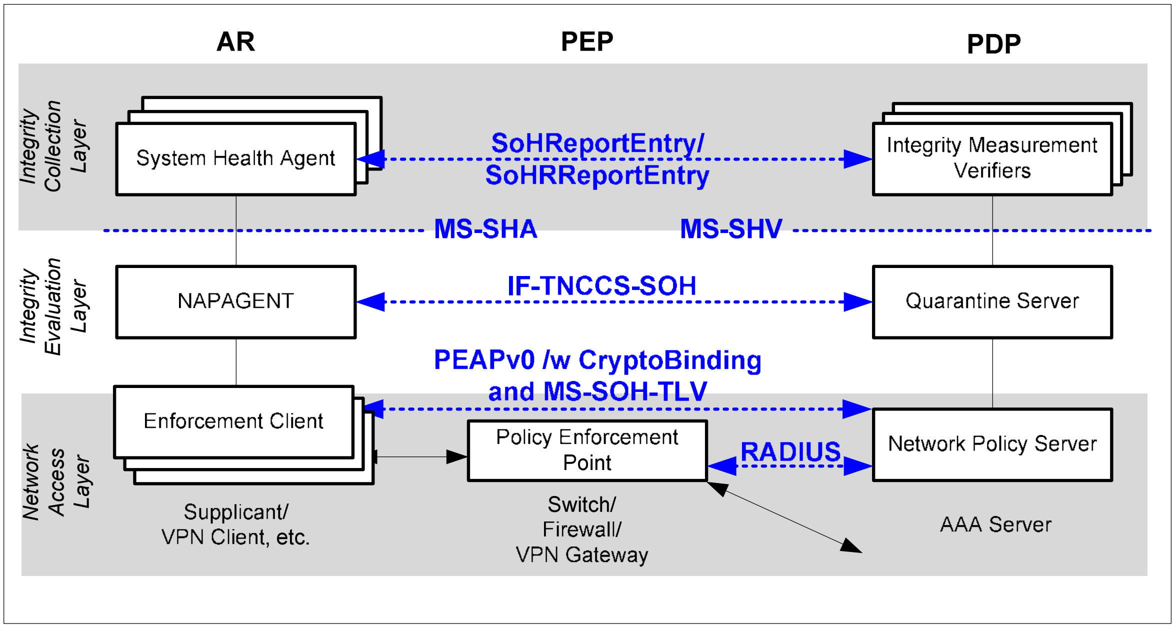


Figure 2: Mapping of NAP architecture and protocols to equivalent TNC architecture and protocols

2.4 Support for IF-TNCCS and IF-TNCCS-SOH

IF-TNCCS-SOH is not compatible with the XML binding of IF-TNCCS and has a slightly different feature set. To be compliant with IF-TNCCS, TNC clients and servers MUST implement either protocol or both, depending on their requirements. If compatibility with Microsoft NAP architectures is important, then IF-TNCCS-SOH should be implemented. If support for multiple round trips between TNC client and server or other features provided only by IF-TNCCS are needed, then IF-TNCCS should be implemented.

2.5 Supported Use Cases

Use cases that IF-TNCCS-SOH supports are as follows.

- During the Integrity Check Handshake, the IMCs send a batch of messages (typically integrity measurements) to the IMVs. The messages from the IMCs to the IMVs are carried through IF-TNCCS-SOH.
- Upon receiving a batch of messages from the IMCs during the Integrity Check Handshake, the IMVs optionally respond with a batch of messages (remediation instructions, etc). The messages from the IMVs to the IMCs are carried through IF-TNCCS-SOH.
- The TNCC may send a batch of messages (standardized or vendor-specific) to the TNCS. The messages from the TNCC to the TNCS are carried through IF-TNCCS-SOH.
- The TNC Server sends reason codes to the TNC Client which can use these with a local string resource to present a localized string instead of sending language preferences from the TNC Client to the TNC Server.

2.6 Non-supported Use Cases

Several use cases, including but not limited to this one, are not provided by (but not prevented by) IF—TNCCS-SOH:

- A TNC Client or TNC Server triggers multiple round trips within a single TNC handshake. This version of IF-TNCCS-SOH only supports a single round trip per session.

2.7 Requirements

Here are the requirements that the IF-TNCCS-SOH protocol must meet in order to successfully play its role in the TNC architecture.

- Meets the needs of the TNC architecture

  Use cases described in the TNC architecture are supported with the exception of those that require multiple round trips are not supported by the IF-TNCCS-SOH protocol.

- Efficient

  The TNC architecture delays 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, it is essential to minimize delays and make IMC-IMV communications as rapid and efficient as possible. Efficiency is also important when you consider that some network endpoints are small and low-powered and that some networks have high latency, high cost, or low bandwidth.

- Easy to use and implement

  The protocol should be easy for TNC Client and TNC Server vendors to use and implement. It should allow them to enhance existing products to support the TNC architecture and integrate legacy code without requiring substantial changes. The protocol should also make things easy for system administrators and end-users. Components of the TNC architecture should plug together automatically without requiring manual configuration.

- Compatible with existing NAP clients and servers

  The IF-TNCCS-SOH protocol must be compatible with existing NAP clients and servers that implement version 1 and 2 of this protocol as described in this document, working with them without requiring modifications.

- Internationalized

  IF-TNCCS-SOH must be able to support being used in multiple localities, in particular the TNC Server should pass reason codes so that the TNC Client can display the use the local specific value.

2.8 Assumptions

Here are the assumptions that the IF-TNCCS-SOH protocol makes about other components in the TNC architecture.

- Format of integrity measurements

The format of the IMC-IMV messages that IF-TNCCS-SOH conveys between the TNC Client and the TNC Server is opaque to IF-TNCCS-SOH. Each IMC-IMV message is simply represented as a binary piece of data within the IF-TNCCS-SOH protocol.

- Limited Message Size

  An IF-TNCCS-SOH packet cannot be larger than 4K bytes in size to maintain compatibility with existing NAP clients and servers. IF-TNCCS-SOH assumes that the underlying IF-T can handle messages of this size as no fragmentation is supported by the TNCC and TNCS.

- Transport

  IF-T is the underlying transport protocol for ALL IF-TNCCS-SOH communication. It is assumed that IF-T will provide a reliable transport mechanism, ensuring the timely delivery of IF-TNCCS-SOH messages.

- Security

  IF-TNCCS-SOH relies on IF-T for secure transport of messages being exchanged between the TNC Client and the TNC Server. IF-T should indicate what security services it is providing (e.g. encrypted data tunnel). IF-TNCCS-SOH does not provide a means for mutual authentication of the TNC Client and the TNC Server.

2.9 Standards Assignments

The following table represents the parameters that have been assigned by standard bodies that are relevant to this specification:

Parameter	Value	Reference
IANA SMI Vendor ID for Microsoft	0x137 (311)	[9]

3 Messages

3.1 Protocol Overview

The IF-TNCCS-SOH protocol consists of two messages: an SoH (Statement of Health) message from the TNC Client to the TNC Server and a responding SoHR (Statement of Health Response) message from the TNC Server to the TNC Client. There is at most one SoH and one SoHR message exchanged for each TNC handshake in an implementation supporting this protocol. The SoH message consists of three major elements: a header, a SSoH (System Statement of Health - information from the TNC Client), and one or more SoHReportEntries (messages from the IMCs). The SoHR message has a similar structure to the SoH message: a header, a SSoHR (System Statement of Health Response - information from the TNC Server), and one or more SoHReportEntries (messages from the IMVs).

The SoH and SoHR use identical formats that rely on use of nested Type-Length-Values (TLV) and multiple Type-Values (TVs). This allows for compactness and extensibility, but can be confusing.

The meaning of a TLV can change depending on the context. In some parts of the message, only one particular Type is allowed. Read the following sections carefully to understand how the protocol works, which types are permitted where, and what they mean. Refer to the Table of Contents to quickly find documentation on particular types and structures.

Type-Value (TV) structures are used instead of TLVs in various points to save space. The meaning of Type 1 in a TV structure is different from Type 1 in a TLV structure.

The first TLV in every SoH is the System Statement of Health (SSoH) after the header. The value part of the SSoH is a sequence of Type-Value (TV) structures. Allowable values for SSOH TVs are described later in this section.

The TLVs that follow the SSoH are grouped in SoHReportEntry sets. The System-Health-ID TLV marks the beginning of each set of TLVs that constitute an SoHReportEntry and MUST be present. Additinally, there MUST only be one SoHReportEntry per System-Health-ID. Each subsequent TLV in an SoHReportEntry is called an SoHAttribute and there may be 0 or more of these per SoHReportEntry.

Implementations MUST use System-Health-Id to determine which IMC messages should be delivered to which IMV.

A graphical representation of the top level structure for a SoH follows.

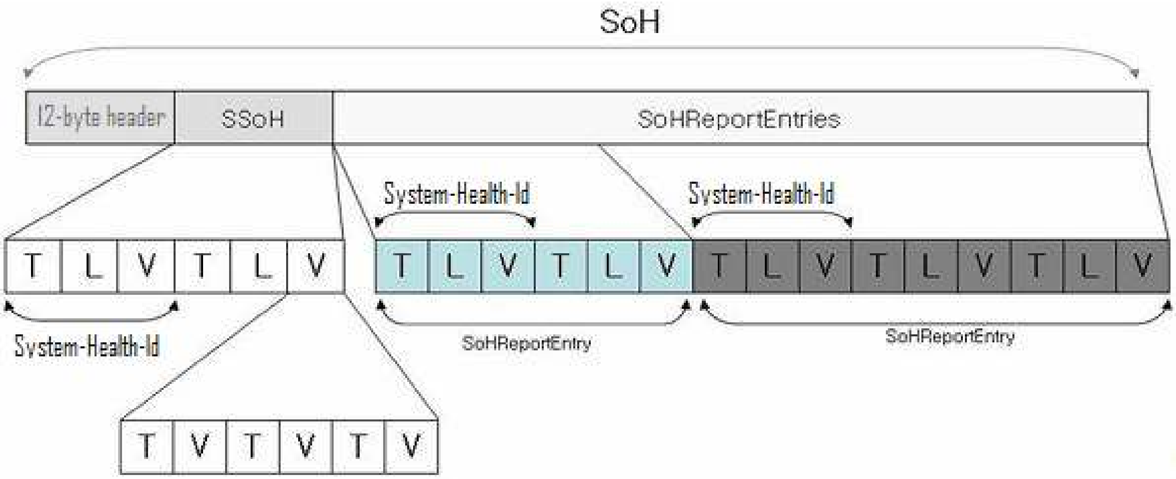


Figure 3: SoH without a Sub-Mode header

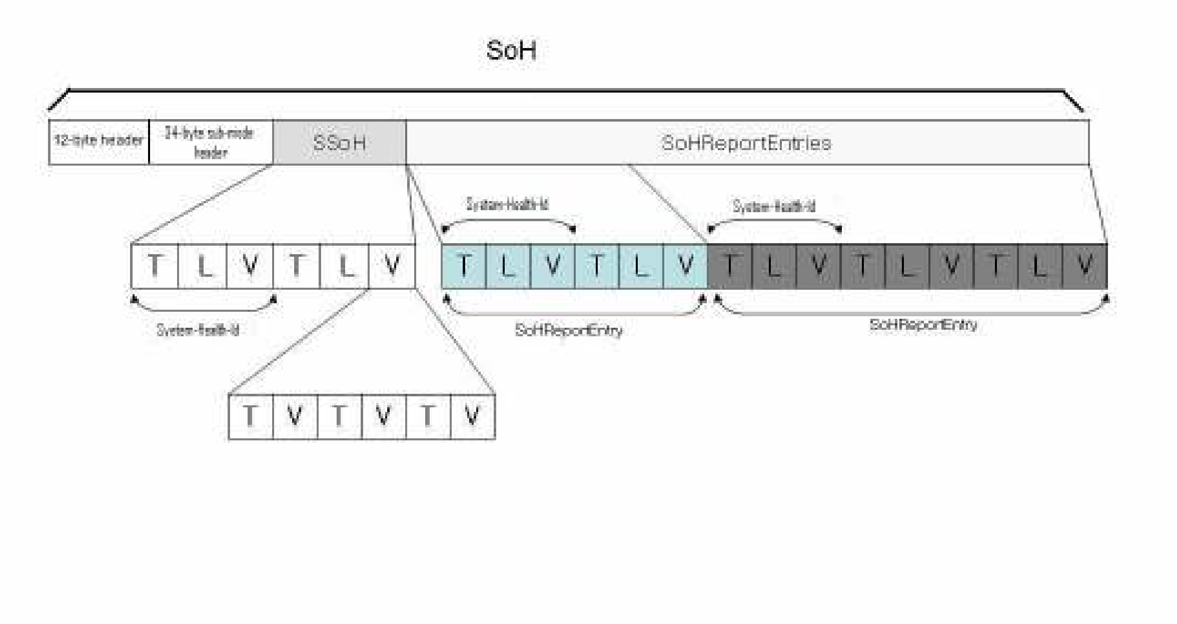


Figure 4: SoH with a Sub-Mode header, version 2 only.

The SoH and SoHR messages use a nested Type-Length-Value (TLV) structure. This is compact and extensible but can be confusing. The meaning of a TLV can change depending on the context. In some parts of the message, only one particular Type is allowed. Read the following specification carefully to understand how the protocol works, which types are permitted where, and what they mean. Refer to the Table of Contents to quickly find documentation on particular types and structures.

Type-Value (TV) structures are used instead of TLVs in the SSoH and SSoHR structures to save space. The meaning of type 1 in a TV structure is different from type 1 in a TLV structure. The table of TLV types is contained in section 3.7 while the table of TV types is in section 3.8.

3.2 Version Handling

It is expected that a TNC Client and Server may implement different versions of this specification.

The IF-TNCCS-SOH protocol at this time has two versions; the version number is specified in the inner type field in the SoH/SoHR headers. One key difference between a version 2 message and a version 1 message is that version 2 messages contain a SoH Mode Sub-Header as specified in section 3.5.1.2. This SoH Mode Sub-Header is intended to allow a later incarnation of the protocol to have newer modes of operation. There is no other functional difference between the two.

When a TNC server receives a TNCCS-SOH request message from a TNC client, it MUST create a TNCCS-SOH response with the same version as the request, if it understands that version. If a client or server does not understand the version included in the message it MUST drop that request as invalid.

As seen in sections 3.5.1.1 and 3.6.1.1, the version of the message affects only the headers.

The intention of the above is to allow a TNC Client that implements one version of IF-TNCCS-SOH to interoperate with a TNC server that implements multiple versions of the protocol.

3.3 Type-Length-Value (TLV) Overview

The following diagram specifies the TLV (M, R, TLV Type, Length, and Value) structure that forms the basis for the SoH and SoHR message messages attributes.

The fields of the structure are transmitted in network-byte order from left to right. Bit 0 is the most significant bit.

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>20</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>30</th><th>1</th></tr>
<tr><td><i>M</i></td><td><i>R</i></td><td colspan="14"><i>TLV Type</i></td><td colspan="16"><i>Length</i></td></tr>
<tr><td colspan="32"><i>Value (variable-length)</i></td></tr>
<tr><td colspan="32"><i>...</i></td></tr>
</table>

Figure 5: Type-Length-Value (TLV) packet.

M: The M bit has the following possible values:

Value	Meaning
0	This TLV MAY be supported (non-mandatory).
1	This is a **mandatory TLV**.

Windows Behavior: *Windows VISTA, XP NAP and Longhorn Server ignore this bit.*

R: The R bit is reserved and MUST be set to zero and ignored on receipt.

TLV Type: 14-bit unsigned integer in network-byte order that indicates the type of data in the Value field. Valid Type values are specified in Section 3.7.

Length: 16-bit unsigned integer in network-byte order that indicates the length, in bytes, of the Value field.

Value (variable): The value MUST be formatted in accordance with the type specified in the TLV Type field.

3.4 Type-Value (TV) Overview

The following diagram shows the standard Type-Value (TV) structure that is used by all the System Statement of Health (SSoH) Attributes (see Section 3.8).

The fields of the structure are transmitted in network-byte order from left to right.

<table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>1
0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>2
0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>3
0</td><td>1</td></tr>
<tr><td colspan="8"><i>Type</i></td><td colspan="24"><i>Value (variable)</i></td></tr>
<tr><td colspan="32">...</td></tr>
</table>

Figure 6: TNCC-SOH The Type-Value (TV) packet.

Type: 8-bit unsigned integer that indicates the type of data in the Attribute Value field.

Value (variable): The length of the Value is defined by the Type. Each type is of a fixed length although different Types have different values.

3.5 Statement of Health (SoH) Message

The SoH message is used to represent an endpoint's claims about its integrity. It contains a header and a value field which forms the rest of the message content.

3.5.1.1 SoH Header

The fields of the header are transmitted in network-byte order from left to right.

<table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>1
0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>2
0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>3
0</td><td>1</td></tr>
<tr><td colspan="2"><i>Rvd</i></td><td colspan="14"><i>Outer Type</i></td><td colspan="16"><i>Length</i></td></tr>
<tr><td colspan="32"><i>IANA SMI Code for Microsoft</i></td></tr>
<tr><td colspan="16"><i>Inner Type</i></td><td colspan="16"><i>Inner Length</i></td></tr>
<tr><td colspan="32"><i>Value (variable)</i></td></tr>
<tr><td colspan="32">...</td></tr>
</table>

Figure 7: SoH Header structure

Rvd: The Rvd field is reserved and MUST be set to zero and ignored on receipt.

Outer Type: 14-bit unsigned integer in network-byte order that MUST be set to 7 (0x7).

Length: 16-bit unsigned integer in network-byte order that indicates the length, in bytes, of the IANA SMI Code for Microsoft, Inner Type, Inner Length and Value fields.

IANA SMI Code for Microsoft: 32-bit unsigned integer in network-byte order that MUST be set to 0x137 (311) (See section 2.9). This value, in combination with the 'Inner Type' value described below, allows implementations to identify that these messages belong to the TNCCS-SoH protocol. This is useful when implementations at either the client or the server side get other messages formatted similarly, such as EAP TLVs.

Inner Type: 16-bit unsigned integer in network-byte order that can have the value 0x0001 or 0x0002. This determines the version of the message content, and dictates the format of the data in the Value field. If 0x0002 is specified the sub-mode header MUST be present (see Section 3.5.1.2)

Inner Length: 16-bit unsigned integer in network-byte order that indicates the length, in bytes, of the Value field.

Value (variable): Variable length field in network-byte order that contains data as follows:

If the Inner Type field is 0x0001:

- SSoH TLV (one only)
- SoHReportEntry TLV sets (zero or more)

If the Inner Type field is 0x0002:

- SoH Mode Sub-Header (one only)
- SSoH TLV (one only)
- SoHReportEntry TLV (zero or more)

3.5.1.2 SoH Mode Sub-Header

The SoH Mode Sub-Header is used to facilitate future extensibility of the TNCC-SOH protocol, it allows for future versions of the protocol to incorporate encryption while retaining compatibility with this version of the protocol.

The fields of the header are transmitted in network-byte order from left to right.

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>1
0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>2
0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>3
0</th><th>1</th></tr>
<tr><td colspan="2"><i>Rvd</i></td><td colspan="14"><i>Outer Type</i></td><td colspan="16"><i>Length</i></td></tr>
<tr><td colspan="32"><i>IANA SMI Code for Microsoft</i></td></tr>
<tr><td colspan="32"><i>Value</i></td></tr>
<tr><td colspan="32"><i>(Value cont'd for 5 rows)</i></td></tr>
<tr><td colspan="16"></td></tr>
</table>

Figure 8: SoH Mode Sub-Header Structure.

Rvd: The Rvd field is reserved, MUST be set to zeros and ignored on receipt.

Outer Type: 14-bit unsigned integer in network-byte order that MUST be set to 7 (0x7).

Length: 16-bit unsigned integer in network-byte order that indicates the length, in bytes, of the Value field and IANA SMI Code for Microsoft.

IANA SMI Code for Microsoft: 32-bit unsigned integer in network-byte order that MUST be set to 0x137 (311) (See section 2.9).

Value: 0x1A (26) bytes formatted as follows:

Correlation ID: 24 bytes that have the same value as the value field of MS-CorrelationId (see section 3.8.6)

Intent Flag: 1 byte that MUST be 0x01 for request (SoH message) and 0x00 for response (SoHR)

Content-Type Flag: 1 byte that MUST be 0x0. This field is intended to help in SoH protocol enhancements in the future, recipients should drop packets containing unknown types.

3.5.1.3 System SoH (SSoH)

SSoH is used to represent generic information about the host, the message containing the SSoH and the current health state of the host.

SSoH has no header of its own and is simply constructed as the following ordered sequence of SoH Attributes and SSoH Attributes:

- System-Health-Id Attribute (see Section 3.7.1).

  The value of System-Health-Id MUST be decimal 79616 (0x00013700), created in accordance to the rules in Section 3.7.1.

- Vendor-Specific Attribute (see Section 3.7.3), the Vendor-Specific Attribute MUST have the following fields set to the specified values:

  Vendor ID: Microsoft SMI code, as specified in [9].

  Value: MUST contain the following ordered sequence of SSoH Attributes, other SSoH Attributes MAY follow the sequence:

- MS-Machine-Inventory (see Section 3.8.1)
- MS-Quarantine-State (see Section 3.8.2)
- MS-Packet-Info (see Section 3.8.3)
- MS-SystemGenerated-Ids (optional) (see Section 3.8.4)
- MS-MachineName (see Section 3.8.5)
- MS-CorrelationId (see Section MS-CorrelationId (section 3.2.4.7))

- MS-Machine-Inventory-Ex (optional) (see Section 3.8.8)

3.5.1.4 SoHReportEntry

The SoHReportEntry message is used to represent a set of SoH Attributes; it has no header of its own and is simply constructed as a set of SoH attributes (see Section 4).

System-Health-Id MUST be the first attribute. The SoHReportEntry message MAY contain zero or more additional SoH Attributes after System-Health-Id.

3.6 Statement of Health Response (SoHR) Message

The SoHR message is used to transport information about the result of the evaluation of a SoH message by the health policy server. It contains a header and a value field which forms the rest of the message content.

3.6.1.1 SoHR Header

The fields of the header are transmitted in network-byte order from left to right.

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>1
0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>2
0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>3
0</th><th>1</th></tr>
<tr><td colspan="2"><i>Rvd</i></td><td colspan="14"><i>Outer Type</i></td><td colspan="16"><i>Length</i></td></tr>
<tr><td colspan="32"><i>IANA SMI Code for Microsoft</i></td></tr>
<tr><td colspan="16"><i>Inner Type</i></td><td colspan="16"><i>Inner Length</i></td></tr>
<tr><td colspan="32"><i>Value (variable)</i></td></tr>
<tr><td colspan="32"><i>...</i></td></tr>
</table>

Figure 9: SoHRHeader Structure.

Rvd: The Rvd field is reserved and MUST be set to zeros and ignored on receipt.

Outer Type: 14-bit unsigned integer in network-byte order that MUST be set to 7 (0x7).

Length: 16-bit unsigned integer in network-byte order that indicates the length, in bytes, of the IANA SMI Code for Microsoft, Inner Type, Inner Length and Value field.

IANA SMI Code for Microsoft: 32-bit unsigned integer in network-byte order that MUST be set to 0x137 (311) (See section 2.9). This value, in combination with the 'Inner Type' value described below, allows implementations to identify that these messages belong to the SoH protocol. This is useful when implementations at either the client or the server side get other messages formatted similarly, as EAP TLVs.

Inner Type: 16-bit unsigned integer in network-byte order that MUST be set to 0x0001 or 0x0002. This value MUST be same as the Inner Type value in the corresponding SoH message that the server received. This determines the version of the message content, and dictates the format of the data in the Value field.

Inner Length: 16-bit unsigned integer in network-byte order that indicates the length, in bytes, of the Value field.

Value (variable): Variable length field in network-byte order that contains data as follows:

If the value of the Inner Type field is 0x0001:

- SSoHR (one only)
- SoHRReportEntry (zero or more)

If the value of the Inner Type field is 0x0002:

- SoH Mode Sub-Header (one only)
- SSoHR (one only)
- SoHRReportEntry (zero or more)

3.6.1.2 System SoH Response (SSoHR)

SSoHR is used to represent generic information about the health policy server. SSoHR has no header of its own and is simply constructed as the following ordered sequence of SoH Attributes and SSoH Attributes:

- System-Health-Id Attribute (see Section 3.7.1).

  The value of this attribute MUST be decimal 79616 (0x00013700), created in accordance to the rules in section 3.7.1.

- Vendor-Specific Attribute (see Section 3.7.3), the Vendor-Specific Attribute MUST have the following fields set to the specified values:

  Vendor ID: Microsoft SMI code, as specified in [9].

  Value: MUST contain the following ordered sequence of SSoHR Attributes:

- MS-Packet-Info (see Section 3.8.3)
- MS-MachineName (see Section 3.8.5)
- MS-CorrelationId (see Section 3.8.6)
- MS-Quarantine-State (see Section 3.8.2)
- MS-Installed-SHVs (optional) (see Section 3.8.7)

3.6.1.3 SoHRReportEntry

The SoHRReportEntry is used to represent a set of SoH Attributes. It has no header of its own and is simply constructed as a set of SoH Attributes.

System-Health-Id MUST be the first attribute. System-Health-Id MUST be followed by either a Compliance-Result-Codes, a Failure Category, or both attributes.

Finally these messages MAY be followed by any set of SoH Attributes (see Section 3.3).

- System-Health-Id
- SoHRAttribute (one or more) (MUST include a Compliance-Result-Code, a Failure Category, or both)

3.7 SoH Attributes

SoH Attributes are used to construct valid SoHReportEntry and SoHRReportEntry elements. A collection of SoHAttributes in which the first attribute is the System-Health-Id constitutes a SoHReportEntry (See Section 3.3).

When using a TLV to contain an SoH Attribute, the TLV Types with values between and including 0–256 are reserved for use in the SoH protocol and MUST NOT be used for other attributes. Only one TLV of a given type should be present for a given type within a message.

The following TLV Types have been reserved and have special meanings when used within this protocol. This table lists TLV types that are interpreted by a TNC Client or TNC Server that implement this protocol.

Type	Length	Name	Meaning	Format
2	4 bytes	System-Health-Id	ID of the IMC or IMV that generated the SoHReportEntry **or** SoHRReportEntry	See Section 3.7.1
3	(Variable) MUST be a multiple of 4 bytes.	IPv4-Fixup-Servers	IPv4 addresses of the servers to be used by the client for remediation.	TLV whose value is a array of 4-byte IPv4 addresses
4	(Variable)	Compliance-Result-Codes	Result codes specifying whether or not the client machine is compliant	See Section 3.7.2
7	(Variable)	Vendor-Specific	The value field contains a vendor-specific TLV	See Section 3.7.3

Type	Length	Name	Meaning	Format
14	1 byte	Failure Category	A code that indicates the failure category	See Section 3.7.4
15	(Variable) MUST be a multiple of 16 bytes.	IPv6-Fixup Servers	IPv6 addresses of the servers to be used by the client for remediation.	TLV whose value is a array of 16-byte Ipv6 addresses

Implementers can define their own TLV types (with types equal or greater than 256) for use within their own vendor-specific SoHReportEntry and SoHRReportEntry messages.

Since there is no mechanism to avoid two vendors defining the same TLV type, vendors should be prepared for such collisions. The collisions can be handled by including such types only within an SOHReportEntry or SOHRReportEntry whose System-Health-ID value identifies a vendor who defined (or recognizes) that interpretation of the TLV type in question.

The following optional TLV attributes, provided as part of this specification for the purpose of completeness only, are not processed by a TNC Client or Server. These attributes MAY be used to construct SoHReportEntries or SoHRReportEntries (IF-M messages). When used, the lengths specified in the table MUST be honored.

Type	Length	Name	Value	Format
5	8 bytes	Time-Of-Last-Update	UTC time when client machine was last successfully updated by the fix up server (measured as a number of seconds since midnight, January 1, 1970)	64-bit unsigned integer
6	(Variable)	Client-Id	An identifier that is used by a health entity to identify itself to a server.	Byte sequence

Type	Length	Name	Value	Format
8	1 byte	Health-Class	Type of health check/verification that the System Health entity is performing Possible values include: Firewall = 0, Patch Level = 1 Antivirus = 2 Critical Update = 3	8-bit unsigned integer
9	(Variable)	Software-Version	Version of the software installed on the client machine	Byte sequence
10	(Variable)	Product-Name	Name of the product installed on the client machine	Byte sequence
11	(Variable)	Health Class Status	Status code for the health-class type given by the Health-Class TLV The values of the status code are implementation specific	Byte sequence
12	8 bytes	SoHGenerationTime	UTC time when the SoHReportEntry was generated (a 64-bit unsigned integer of seconds since 1/1/70)	8-byte value
13	(Variable) MUST be a multiple of 4 bytes.	Error Codes	Error codes for specific operations that can be contained in the SoHReportEntry or the SoHRReportEntry	List of HRESULT [12] error codes

3.7.1 System-Health-ID Attribute

The ID of the component that generated the SoHReportEntry (or the SoHRReportEntry) MUST be the first TLV present in the SoHReportEntry or SoHRReportEntry.

The TLV containing the System-Health-Id attribute has the following values set:

M: The M bit MUST be set to 0.

R: The R bit is reserved and MUST be set to zero and ignored on receipt.

TLV Type: The **TLV** Type MUST be set to 2.

Value: 32-bit unsigned integer used to represent the Health ID of the SoHReportEntry. The value MUST be globally unique and be formatted as follows:

										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
IANA SMI Code for Vendor																								*Id*							

Figure 10: System-Health-ID Attribute.

IANA SMI Code for Vendor: 24-bit unsigned integer in network-byte order that MUST contain the IANA SMI code for the vendor whose component produced the message.

Id: 8-bit unsigned integer used to identify different components from the same vendor. Any value can be specified by the vendor for use by its components.

3.7.2 Compliance-Result-Codes Attribute

The result of the evaluation of the SoHReportEntry is used to specify whether or not the client machine is compliant with policy.

An SoHRReportEntry MUST contain a Compliance-Result-Code attribute, a Failure Category Code attribute (section 3.7.4), or both. An SoHReportEntry MAY contain this attribute.

The Compliance-Result-Codes attribute MUST NOT be present in a failure case.

The TLV values of a Compliance-Result-Codes attribute are as follows:

M: The M bit MUST be set to 0.

R: The R bit is reserved and MUST be set to zero and ignored on receipt.

TLV Type: The TLV Type MUST be set to 4.

Value: Array of HRESULTS [12] in network-byte order, these HRESULTS are vendor specific and MAY be used by the implementation for diagnostic purposes. Implementations SHOULD use Zero to indicate success.

3.7.3 Vendor-Specific Attribute

Represents vendor-specific data, in which the format of the data is known only to the specified vendor.

This attribute is used to carry implementation specific data from the client, to the server and back; for example an anti-virus vendor may include data about the signature database version and the time in which the last complete scan was performed.

This attribute MAY be present in a SoHReportEntry and SoHRReportEntry.

The **TLV** values of a Vendor-Specific attribute are as follows:

M: The M bit MUST be set to 0.

R: The R bit is reserved and MUST be set to zero and ignored on receipt.

TLV Type: The TLV Type MUST be set to 7.

Value: The value field MUST be formatted as follows:

<table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>1
0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>2
0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>3
0</td><td>1</td></tr>
<tr><td colspan="32"><i>Vendor ID</i></td></tr>
<tr><td colspan="32"><i>Data (variable)</i></td></tr>
<tr><td colspan="32"><i>...</i></td></tr>
</table>

Figure 11: Vendor Specific Attribute.

Vendor ID: 32-bit unsigned integer in network-byte order that contains the IANA assigned SMI for the vendor whose data is be specified in the Data field.

Data (variable): The format of the data field is vendor specific.

3.7.4 Failure Category Attribute

Used to classify type of failure that occurred. An SoHRReportEntry MUST contain a Failure Category attribute, a Compliance-Result-Code attribute, or both. This attribute MAY be present in an SoHReportEntry.

The TLV values of a Failure Category attribute are as follows:

M: The M bit MUST be set to 0.

R: The R bit is reserved and MUST be set to zero and ignored on receipt.

TLV Type: The TLV Type MUST be set to 14.

Value: 8 bit field containing one of the values from the following table, all other values MUST be treated as a failure:

Value	Meaning
0	No failure occurred.
1	A failure which is not due to client or server components or communications.
2	A failure due to client component.
3	A failure due to client communication.
4	A failure due to server component.
5	A failure due to server communication.

3.8 SSoH and SSoHR Attributes

SSoHAttribute attributes are used to construct valid SSoHReportEntry and SSoHRReportEntry TLVs, respectively. SSoHAttribute MUST be contained in the Value field of a TV.

When using a TV to contain an SSoH or SSoHR attribute, the SSoH and SSoHR headers are not extensible. The following TV Types have been defined for use within this protocol:

Value	Meaning
1	MS-Machine-Inventory
2	MS-Quarantine-State
3	MS-Packet-Info
4	MS-SystemGenerated-Ids

Value	Meaning
5	MS-MachineName
6	MS-CorrelationId
7	MS-Installed-Shvs
8	MS-Machine-Inventory-Ex

When an implementation encounters a unknown type it MUST ignore the remaining TVs. All TV Types (0 through 255) are reserved for future use in the TNCC-SoH protocol and MUST NOT be used.

3.8.1 MS-Machine-Inventory Attribute

The MS-Machine-Inventory attribute is used to communicate information about the host operating system and its processor architecture, these values MAY be used by a server to make policy decisions.

The values of returned by the Windows TNC Client are based on the returns of the GetVersionEx API [11].

Operating systems other than Windows MAY specify platform specific values, fields that are not relevant for a given operating system MUST be set to zero.

The MS-Machine-Inventory attribute MUST be present in an SSoHReportEntry and MAY be present in an SSoHRReportEntry.

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>1
0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>2
0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>3
0</th><th>1</th></tr>
<tr><td colspan="32"><i>osVersionMajor</i></td></tr>
<tr><td colspan="32"><i>osVersionMinor</i></td></tr>
<tr><td colspan="32"><i>osVersionBuild</i></td></tr>
<tr><td colspan="16"><i>spVersionMajor</i></td><td colspan="16"><i>spVersionMinor</i></td></tr>
</table>

										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
procArch																															

Figure 12: MS-Machine-Inventory Attribute.

osVersionMajor: 32-bit unsigned integer in network-byte order that is used to represent the major version of the host operating system.

osVersionMinor: 32-bit unsigned integer in network-byte order that is used to represent the minor version of the host operating system.

osVersionBuild: 32-bit unsigned integer in network-byte order that is used to represent the build number of the host operating system.

spVersionMajor: 16-bit unsigned integer in network-byte order that is used to represent the major version of the service pack installed on host operating system.

spVersionMinor: 16-bit unsigned integer in network-byte order that is used to represent the minor version of the service pack installed on host operating system.

procArch: 16-bits in network-byte order that is used to represent the processor architecture of the host.

3.8.2 MS-Quarantine-State Attribute

The MS-Quarantine-State attribute is used to communicate information about the desired or resulting permission to a requested network resource for an endpoint. This attribute MUST be present in both the SSoHReportEntry and the SSoHRReportEntry.

The first 16 bits is a field is called Flags. This field contains the first four fields Reserved, qState, f and Reserved fields.

										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved1								*Reserved2*				*F*	*qState*			*ProbTime*															
(ProbTime cont'd for 1 rows)																															
																urlLenInBytes															
url (variable)																															
...																															

Figure 13: MS-Quarantine-State Attribute.

Reserved1: 8-bit reserved field that MUST be set to zero and ignored on receipt.

Reserved2: 4-bit reserved field that MUST be set to zero and ignored on receipt.

f: 1-bit field that indicates the health policy server requests the host fixes up before attempting to access its resource again.

Value	Meaning
0	Remediation not required by policy.
1	Remediation required by policy.

qState: 3-bit field that has the following potential values:

Value	Meaning
1	Network connectivity is not being restricted.
2	Network connectivity is not being restricted but may be at a later time.
3	Network connectivity is being restricted.

ProbTime: 64-bits used to represent the time in which the host will be on probation. Probation allows an implementation to grant a client temporary authorization for a period even when the health check fails. At the end of the probation period the client SHOULD revalidate its health. The behavior is implementation dependent and SHOULD be policy driven. The value is formatted as a 64-bit value representing the number of 100-nanosecond intervals since January 1, 1601 (UTC).

urlLenInBytes: 16-bits in network-byte order used to represent the length of the Uniform Resource Locator (URL) and includes NULL string termination character**.**

url (variable): UTF-8, as specified in [2], representation of a URL that can be presented to users for more information on why they were assigned this state.

3.8.3 MS-Packet-Info Attribute

The MS-Packet-Info attribute is used to communicate information version and intent (request or response) of the SSoH and SSoHR. This attribute MUST be present in both the SSoH and the SSoHR.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Reserved				*r*	*Vers*																										

Figure 14: MS-Packet-Info Attribute.

Reserved: The 4-bit Reserved Bits are reserved and MUST be set to zero and ignored on receipt.

r: 1-bit Response/request flag. The value indicates whether the attribute contains a request or a response, the field has the following values.

Value	Meaning
0	Packet is response (SSoHR)
1	Packet is a request (SSoH)

Vers: The 3-bit protocol version. MUST be 1 and ignored on receipt. This is not to be confused with the version number set in the header.

3.8.4 MS-SystemGenerated-Ids Attribute

The MS-SystemGenerated-Ids attribute is used to represent which of the installed IMCs were unable to generate their SoHReportEntry and what error information is available for the failure. This attribute MAY be present in an SSoH and SHOULD NOT be present in an SSoHR.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Length																*idList (variable)*															
...																															

Figure 15: MS-SystemGenerated-Ids Attribute.

Length: 16-bit unsigned integer in network-byte order that indicates the length, in bytes, of the idList field.

idList (variable): Array of identifiers for the components that generated the SoHReportEntry/SoHRReportEntrys in the message that contains this attribute. Identifiers MUST use MS-SystemGenerated-Ids sub-attribute format.

3.8.4.1 MS-SystemGenerated-Ids Sub-Attribute

										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
IANA SMI Code for Vendor																								*Id*							

Figure 16: The MS-SystemGenerated-Ids Sub packet.

IANA SMI Code for Vendor: 24-bit unsigned integer in network-byte order that MUST contain the IANA SMI code for the vendor whose component produced the message.

Id: 8-bit unsigned integer used to identify different components from the same vendor, any value can be specified by the vendor for use by its components. This value, combined with the 'IANA SMI Code for Vendor', described above, allows the routing of a SoHReportEntry from a client component to the corresponding server side component that can deal with it. Similarly, the SoHRReportEntry is routed to the originator based on this ID. The 'IANA SMI Code for Vendor' by itself is not sufficient because a given vendor’s products may have multiple components. The 8 bit component identifier fully identifies the source and destination of each SoHReportEntry and SoHRReportEntry.

3.8.5 MS-MachineName Attribute

The MS-MachineName attribute is used to communicate the name of the machine that generated the message. This attribute MUST be present in both the SSoH and the SSoHR.

										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Length																*Fully Qualified Domain Name (variable)* *e.g.* hostname@domain.com															
...																															

Figure 17: The MS-MachineName packet.

Length: 16-bit field in network-byte order that represents the length of the machine name field.

machineName (variable): Null terminated UTF-8 encoded string field used to represent the computer name of the machine that generated the message.

3.8.6 MS-CorrelationId Attribute

The MS-CorrelationId attribute is used for diagnostic purposes, specifically it facilitates correlation messages related to a single transaction together across multiple machines. This attribute MUST be present in both the SSoH and the SSoHR.

This value MUST be globally unique across all clients to prevent mis-association of messages.

										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
correlationId																															
(correlationId cont'd for 5 rows)																															

Figure 18: The MS-CorrelationId packet.

correlationId: 192-bit field in network-byte order that represents a unique transaction identifier shared across SSoH and SSoHR messages. The format of the correlationId is implementation specific.

3.8.7 MS-Installed-Shvs Attribte

The MS-Installed-Shvs is a list of identifiers of IMV services that can evaluate SoHs on the TNC Server.

These values MAY be used as hints to determine what **health messages** to send.

This attribute SHOULD NOT be present in the SSoH .

										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Length																*idList (variable)*															
...																															

Figure 19: The MS-Installed-Shvs packet.

Length: 16-bit unsigned integer in network-byte order that indicates the length, in bytes, of the idList field.

idList (variable): Array of identifiers for the components that generated the SoHRs messages on the server. The identifiers MUST use MS-Install-Shvs sub-attribute format.

3.8.7.1 MS-Installed-Shvs Sub-Attribute

										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
IANA SMI Code for Vendor																								*Id*							

Figure 20: The MS-Installed-Shvs-Sub packet.

IANA SMI Code for Vendor: 24-bit unsigned integer in network-byte order that MUST contain the IANA SMI code for the vendor whose component produced the message.

Id: 8-bit unsigned integer used to identify different components from the same vendor, any value can be specified by the vendor for use by its components.

3.8.8 MS-Machine-Inventory-Ex Attribute

The MS-Machine-Inventory-Ex is used to communicate additional information about the host operating system sending the attribute. This attribute MUST be present in the SSoH and MAY be present in the SSoHR.

The values of returned by the Windows TNC Client are based on the returns of the GetVersionEx API [11].

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Reserved																															
ProductType																															

Figure 21: The MS-Machine-Inventory packet.

Reserved: 32-bit reserved value, MUST be ignored on receipt.

ProductType: 8-bit field used to represent the type of the operating system.

4 Protocol Details

4.1 Common Details

4.1.1 Abstract Data Model

These are specified in Sections 4.2.1 and 4.3.1.

4.1.2 Timers

The SoH protocol includes no timers. The transports over which an SoH is transported may have timers associated with them to achieve guaranteed and in-order delivery.

4.1.3 Initialization

The SoH protocol does not require explicit initialization. The transports that carry it may.

4.1.4 Higher-Layer Triggered Events

There are no common higher-layer triggered events.

4.1.5 Message Processing Events and Sequencing Rules

There are no common message processing events or sequencing rules.

4.1.6 Timer Events

There are no common timer events.

4.1.7 Other Local Events

There are no common local events.

4.2 Client Specific Details

4.2.1 Abstract Data Model

The SoH protocol requires a single piece of state to be tracked on the SoH Client:

MS-CorrelationId Cache: The SoH client must maintain a cache of the MS-CorrelationId values it sends. The cache is used to ensure that a received SoHR corresponds to an SoH that was sent.

Note that the cache can be implemented using a variety of techniques. An implementation is at liberty to implement such data in any way it pleases.

4.2.2 Timers

See section 4.1.2.

4.2.3 Initialization

See section 4.1.3.

4.2.4 Higher-Layer Triggered Events

The following events can result in SoHs being sent by the TNC client:

1. A user reboots a machine.

2. The status of the client changes. For example, the firewall on the client is turned off.

In addition, events specific to the transport that carries the SoH messages can result in SoHs being sent by the SoH client. For example, when an IF-T is used to carry SoH messages, the re-authentication to the NAA can result in an SoH message being sent to the server.

4.2.5 Message Processing Events and Sequencing Rules

The processing of SoH and SoHR messages on the client is implementation specific and often involves the use of third-party components. An SoH contains data (contained in the SoHReportEntry values) that reports the client's current status to the health policy server. This value of this data is typically provided by software on the client that provides security services such as an anti-virus client or a patch client. Likewise, the validation of this data is typically provided by an anti-virus server or patch server. The SoH protocol itself simply provides mechanism for this data to be supplied and evaluated.

The TNCC and TNCS do not interpret the SohReportEntry and SoHRReportEntry directly. They instead rely on the IMC and IMV to interpret these messages. To route the report entries to the appropriate component, the TNCC and TNCS use the System-Health-Id to delineate each report entry.

An SoH client does the following:

1. Creates valid SoH messages containing host status information per locally configured policy. The message version SHOULD be version 2.

2. Sends the SoH messages over IF-T.

3. Receives an SoHR from the TNC server over IF-T.

4. Process the SoHRs received.

4.2.5.1 Sending SoHs

An SoH client MUST ensure that all SoHs sent contain unique values in the MS-CorrelationId value of the SSoH included in the SoH.

4.2.5.2 Receiving SoHs

An SoH client MUST silently discard any message received that is not a valid SoHR.

4.2.5.3 Sending SoHRs

An SoH client MUST NOT send an SoHR message.

4.2.5.4 Receiving SoHRs

The SoH client MUST ensure that every received SoHR is properly formed, including validating the length of each attribute. If the lengths are invalid, the SoH client MUST discard the SoHR message.

The SoH client MUST ensure that the SoHRAttributeSet value in the SoHR contains at least a Compliance-Result-Codes (section 3.7.2) or Failure Category (section 3.7.4) attribute. If that is not the case, the SoH client MUST discard the SoHR message.

The SoH client MUST discard any received SoHR message that contains an MS-CorrelationId (section 3.8.6) value in the SSoHR attribute that does not correspond to an MS-CorrelationId (section 3.8.6) value previously sent in an SoH message.

4.2.6 Timer Events

Probation expiry timer event: Probation allows an implementation to grant a client temporary authorization for a period even when the health check fails. At the end of the probation period the TNCC SHOULD revalidate its health, enforcement of probation periods is the responsibility of the PEP.

4.2.7 Other Local Events

The following events can result in SoHs being sent by the TNCC:

1. A new IP address is configured or assigned to the TNCC.

2. A new SoH transport protocol completes initialization.

3. 802.1x session authentication is started on the client.

4.3 Server Specific Details

4.3.1 Abstract Data Model

No ADM required.

4.3.2 Timers

See Section 4.1.2

4.3.3 Initialization

See Section 4.1.3.

4.3.4 Higher-Layer Triggered Events

No higher-layer triggered events.

4.3.5 Message Processing Events and Sequencing Rules

The processing of SoH and SoHR messages on the server is implementation specific and often involves the use of third-party components. An SoHR contains data (contained in the SoHRReportEntry values) that reports the results of an evaluation of the client's current status. This value of this data is typically provided by other servers that provide security services such as an anti-virus server or a patch server. The SoH protocol itself simply provides mechanism for the client status to be supplied so that it can be evaluated.

A health policy server does the following:

1. Receives and processes SoHs from TNC client over IF-T.

2. Creates SoHRs.

3. Sends SoHRs to TNC Client over IF-T.

4.3.5.1 Sending SoHs

The Health Policy Server MUST NOT send an SoH.

4.3.5.2 Receiving SoHs

The SoH server MUST ensure that every received SoH is properly formed, including validating the length of each attribute. The SoH server MUST then use the received SoH to evaluate the compliance of the SoH against policy.

4.3.5.3 Sending SoHRs

The SoH server MUST ensure that every SoHR it sends is properly formed, including validating the length of each attribute. The SoH server MUST include at least a valid Compliance-Result-Codes or Failure Category attribute in the SoHRAttributeSet of the SoHR.

The SoH server MUST NOT send an SoHR that is not in response to an SoH previously received. The SoH server MUST populate the value of the MS-CorrelationId (section 3.8.6) attribute in the SSoHR with the value of the MS-CorrelationId (section 3.8.6) attribute in the SoH to which this SoHR is a response.

4.3.5.4 Receiving SoHRs

The SoH Server MUST discard any message that is not a valid SoH.

4.3.6 Timer Events

There are no timer events on the SoH server.

4.3.7 Other Local Events

The following events can result in an SoHR being sent by the SoH Server:

- A RADIUS/EAP packet containing an MS SoH attribute is received.

5 Security Considerations

5.1 Security Considerations for Implementers

Security for health messages SHOULD be provided by IF-T. IF-T SHOULD guard against replay, tampering and provide confidentiality and authentication of Health Messages. Health messages SHOULD NOT be transmitted in the clear if the transport protocol itself does not encrypt the communication.

The following risks will be mitigated when IF-T provides security for Health Messages:

1. Confidentiality from Passive Observation – Health messages contain information that may not only disclose personally identifying information of a user but also disclose a current vulnerability in the system. That is the nature of the message and the service it provides. For this reason, it is important to preserve the confidentiality of these messages. It should not be possible for an attacker (man in the middle) to view the contents of health messages as they are transmitted.

2. Spoofing – Health Messages (an SoH) are token's that potentially enable a client's access a protected resource. It should not be possible for anyone other than the system which created the SoH to use it. This requires that the authenticity of the source be verified. Similarly, an SoHR potentially causes a client to execute code that may be unsafe. For this reason, it is important to prevent an attacker from being able to spoof such messages. Thus, it should not be possible for an attacker to impersonate a NAS or EAP Server or a client.

3. Active Tampering – For the risks discussed above, it should not be possible for an attacker to modify the SoHR, undetected. Similarly, tampering of the SoH may cause a client to be given access when it must not and vice versa. The security provided by the transport mechanism should prevent tampering of these messages.

4. Replaying – A message that causes a client to be granted access can potentially be retransmitted by another client to incorrectly give it access. This should be prevented by ensuring idempotence of SoH and SoHR messages as observed by a man in the middle who is able to view the transport level communication.

There are a set of attacks for which no good measures currently exist. These are documented here to make developers aware of these.

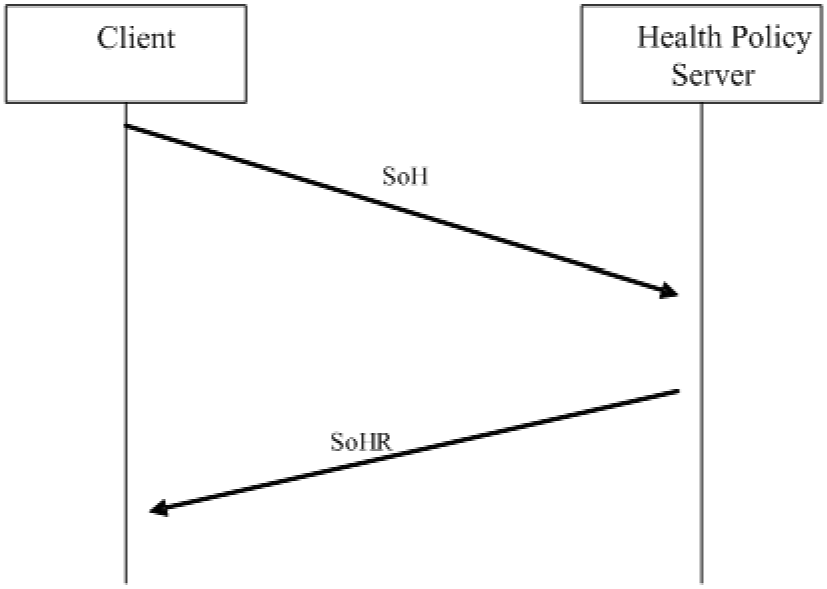
There are no reliable measures that can prevent a denial of service attack on either a client or a NAS. Such attacks may include network flooding or tampering of communication by an attacker who is on the path between a client and a NAS (in the case of WiFi for example).

There is always the potential that the host itself is compromised by some kernel mode malware. In such cases, the SoHs, SoHAttributes produced by the client can not be trusted. The solution to this problem is to leverage trusted hardware such as a TPM.

6 Example Message Flows

This is a simple protocol with a single exchange. The party seeking access to a network resource sends the SoH and receives an SoHR. It is represented graphically below.

Figure 22: Basic SoH and SoHR exchange



In all cases, a transport protocol is involved in sending the messages in both directions. The transport protocol is typically the authentication protocol that mediates access to the network resource. This simple flow applies to all use cases. See Section Protocol Details (section 4) for specifics about the flow.

When the SoH is being sent, it is likely that the client is requesting access to some service and is being required to proved 'good health' as a precondition. When the SoH is received it is likely that the receiver will forward it to some infrastructure server that will evaluate the SoH and return the response (SoHR) to the client, via the original receiver of the SoH.

The receipt of an SoHR by the client, generally, will allow access to the service being requested, if a such was being requested. In cases when the health of the client is not 'good' the SoHR is likely to contain sufficient instructions to permit the client to seek and get remedied. Once the client is remedied it can initiate the protocol again, this time in good health.

7 References

7.1 Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.ietf.org/rfc/rfc2119.txt

[2] F. Yergeau, "UTF-8, a transformation format of ISO 10646", RFC 3629, February 2000, http://www.ietf.org/rfc/rfc3629.txt

7.2 Informative References

[3] Trusted Computing Group, *TNC IF-IMC*, Specification Version 1.2, May 2006.

[4] Trusted Computing Group, *TNC IF-IMV*, Specification Version 1.2, May 2006.

[5] Trusted Computing Group, *TNC IF-T*, Specification Version 1.0, May 2006.

[6] Trusted Computing Group, *TNC IF-PEP*, Specification Version 1.0, May 2006.

[7] Trusted Computing Group, *TNC Architecture for Interoperability*, Specification Version 1.1, May 2006.

[8] Trusted Computing Group, *TNC IF-TNCCS, Specification Version 1.1,* Febuary 2007

[9] Internet Assigned Numbers Authority, "IANA-registered Private Enterprises", http://www.iana.org/assignments/enterprise-numbers

[10] Microsoft, Introduction to Network Access Protection, http://www.microsoft.com/downloads/details.aspx?FamilyID=5d5e243a-23a8-479c-9f2d-37d6d79153e7&DisplayLang=en

[11] Microsoft Developer Network (MSDN), GetVersionEX, http://msdn2.microsoft.com/en-us/library/ms724451.aspx

[12] Microsoft Developer Network (MSDN), HRESULT, http://msdn2.microsoft.com/en-us/library/ms526450.aspx