[MS-SOH]:
Statement of Health for Network Access Protection (NAP) Protocol Specification

Revision Summary

Date	Revision History	Revision Class	Comments
03/14/2007	1.0	Major	Updated and revised the technical content.
04/10/2007	1.1	Minor	Updated the technical content.
05/18/2007	1.2	Minor	Addressed EU feedback
06/08/2007	1.2.1	Editorial	Revised and edited the technical content.
07/10/2007	1.2.2	Editorial	Revised and edited the technical content.
08/17/2007	1.2.3	Editorial	Revised and edited the technical content.
09/21/2007	2.0	Major	Updated and revised the technical content.
10/26/2007	2.0.1	Editorial	Revised and edited the technical content.
01/25/2008	3.0	Major	Updated and revised the technical content.
03/14/2008	3.0.1	Editorial	Revised and edited the technical content.
06/20/2008	4.0	Major	Updated and revised the technical content.
07/25/2008	4.0.1	Editorial	Revised and edited the technical content.
08/29/2008	4.0.2	Editorial	Revised and edited the technical content.
10/24/2008	4.0.3	Editorial	Revised and edited the technical content.
12/05/2008	5.0	Major	Updated and revised the technical content.
01/16/2009	5.0.1	Editorial	Revised and edited the technical content.
02/27/2009	5.0.2	Editorial	Revised and edited the technical content.
04/10/2009	5.0.3	Editorial	Revised and edited the technical content.
05/22/2009	5.0.4	Editorial	Revised and edited the technical content.
07/02/2009	5.0.5	Editorial	Revised and edited the technical content.
08/14/2009	5.0.6	Editorial	Revised and edited the technical content.
09/25/2009	5.1	Minor	Updated the technical content.
11/06/2009	5.1.1	Editorial	Revised and edited the technical content.
12/18/2009	6.0	Major	Updated and revised the technical content.
01/29/2010	6.0.1	Editorial	Revised and edited the technical content.
03/12/2010	7.0	Major	Updated and revised the technical content.

Date	Revision History	Revision Class	Comments
04/23/2010	8.0	Major	Updated and revised the technical content.
06/04/2010	9.0	Major	Updated and revised the technical content.
07/16/2010	9.0	No change	No changes to the meaning, language, or formatting of the technical content.
08/27/2010	9.0	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2010	10.0	Major	Significantly changed the technical content.
11/19/2010	11.0	Major	Significantly changed the technical content.
01/07/2011	11.0	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	12.0	Major	Significantly changed the technical content.
03/25/2011	13.0	Major	Significantly changed the technical content.
05/06/2011	14.0	Major	Significantly changed the technical content.
06/17/2011	15.0	Major	This document has been updated by The Technical Committee.

Contents

1 Introduction

This document specifies the Statement of Health for Network Access Protection (NAP) Protocol in which a client and a server exchange **Statement of Health (SoH)** and **Statement of Health Response (SoHR)** messages. This protocol, along with appropriate **authentication** protocols, helps enterprises to ensure that users of their network resources are not only authenticated but also using systems that conform with corporate **policies**. Typically the policies relevant to this protocol relate to security update management, configuration for antivirus products, firewall settings, and measures for security health and system health.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

authentication
authorization
computer name
Dynamic Host Configuration Protocol (DHCP)
EAP server
enforcement client
FILETIME
fix-up servers
health ID
health messages
health policy server
health registration authority (HRA)
health state
HRESULT
mandatory type-length-value
Network Access Protection (NAP)
Network Access Protection (NAP) client
network access server (NAS)
policy
remediation server
Remote Authentication Dial-In User Service (RADIUS)
session
statement of health (SoH)
statement of health (SoH) client
statement of health response (SoHR)
system health entity

The following terms are specific to this document:

Health Certificate Enrollment Protocol (HCEP): A protocol designed to accomplish health certificate enrollment. Health certificates encapsulate the client's compliance to **policy** in a way that can be presented to interested parties without requiring those parties to perform the validation themselves.

IANA SMI: Structure and Identification of Management Information for TCP/IP-based Internets (SMI), a data structure defined by the Internet Assigned Numbers Authority (IANA) to manage hosts and gateways on the Internet. As specified in [IANA-ENT], [IANA-NMP], and [RFC1155].

idempotence: An operation where if the operation is applied one or more times, then no differences, no errors, and no inconsistencies will result. Example: abs(x) == abs(abs(x)) == abs(abs(abs(x))) == ... for all x.

man in the middle attack (MITM): A security attack in which an attacker intercepts and possibly modifies data that is transmitted between two users. The attacker pretends to be the other person to each user. In a successful **MITM** attack, the users are unaware that there is an attacker, which is intercepting and modifying their data, between them. Also referred to as a bucket brigade attack.

NAP EC API: Provides a set of function calls that allow NAP **enforcement clients** to register with the **NAP** agent, to request system health status, and pass system health remediation information to the **NAP** agent. The **NAP EC API** allows vendors to create and install additional **NAP EC**. For more information about this API, see [MS-NAPSO] and [MSDN-NAPAPI].

NAP enforcement client (NAP EC): The NAP **enforcement client** components are part of the **NAP client**. A **NAP EC** can be defined for different type of network access or communication.

PEP channel: An abstract interface that is used by the **NAP client** to transport **SoH** messages to and from the PEP. Examples of **PEP channels** include **DHCP** and HTTP/S used to transport **SoH** messages.

SHA API: Provides a set of function calls that allow **SHAs** to interact with the **NAP** agent to register **SHAs**, indicate system health status, respond to queries for system health status from the **NAP** agent, and for the **NAP** agent to pass system health remediation information to a **SHA**. The **SHA API** enables vendors to create and install additional **SHAs**. For more information about this API, see [MS-NAPSO] and [MSDN-NAPAPI].

SHV API: Provides a set of function calls that enable **SHVs** to interact with the **NAP** administration server component to register **SHVs**, receive **SoHs**, and send **SoHRs**. The **SHV API** is provided with the **NAP** platform. For more information about this API, see [MS-NAPSO] and [MSDN-NAPAPI].

statement of health ReportEntry (SoH ReportEntry): A collection of data that represents a specific aspect of the **health state** of a client.

statement of health response ReportEntry (SoHR ReportEntry): A collection of data that represents the evaluation of a specific aspect of the **health state** of a client, according to network **policies**.

system health agent (SHA): The client components that make declarations on a specific aspect of the client **health state** and generate an **SoH ReportEntry**.

system health validator (SHV): The server counterpart to the **System Health Agent (SHA)**, which is responsible for verifying the declarations of client **health state** made by the respective **SHA**. The **SHV** generates an **SoHR ReportEntry**.

type-length-value (TLV): An information element that is encoded within a protocol. Type and Length fields are a fixed size (1 to 4 bytes), and the Value field is variable length. Type indicates what kind of field is encoded; Length indicates the size of Value; and Value defines the data portion of this **type-length-value (TLV)** element.

type-value (TV): An information element that is encoded within a protocol. The Type field is of a fixed size. Type indicates both what kind of value is encoded and the length of the Value field (by implication). This is because each type in a **type-value (TV)** is of a fixed and known length.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

References to Microsoft Open Specification documents do not include a publishing year because links are to the latest version of the documents, which are updated frequently. References to other documents include a publishing year when one is available.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-DHCPE] Microsoft Corporation, "Dynamic Host Configuration Protocol (DHCP) Extensions".

[IANA-ENT] Internet Assigned Numbers Authority, "Private Enterprise Numbers", January 2007, http://www.iana.org/assignments/enterprise-numbers

[IANA-NMP] Internet Assigned Numbers Authority, "Network Management Parameters", http://www.iana.org/assignments/smi-numbers

[MS-DTYP] Microsoft Corporation, "Windows Data Types".

[MS-HCEP] Microsoft Corporation, "Health Certificate Enrollment Protocol Specification".

[MS-RNAP] Microsoft Corporation, "Vendor-Specific RADIUS Attributes for Network Access Protection (NAP) Data Structure".

[MS-WSH] Microsoft Corporation, "Windows Security Health Agent (WSHA) and Windows Security Health Validator (WSHV) Protocol Specification".

[RFC1155] Rose, M., and McCloghrie, K., "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, May 1990, http://www.ietf.org/rfc/rfc1155.txt

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

[RFC2781] Hoffman, P., and Yergeau, F., "UTF-16, an encoding of ISO 10646", RFC 2781, February 2000, http://www.ietf.org/rfc/rfc2781.txt

[RFC2865] Rigney, C., Willens, S., Rubens, A., and Simpson, W., "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000, http://www.ietf.org/rfc/rfc2865.txt

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary".

[MS-NAPSO] Microsoft Corporation, "Network Policy and Access Services System Overview".

[MS-PEAP] Microsoft Corporation, "Protected Extensible Authentication Protocol (PEAP) Specification".

[MSDN-INapSysHA] Microsoft Corporation, "INapSystemHealthAgentCallback Interface", http://msdn.microsoft.com/en-us/library/aa369655(v=VS.85).aspx

[MSDN-INapSysHV] Microsoft Corporation, "INapSystemHealthValidator Interface", http://msdn.microsoft.com/en-us/library/aa369692(VS.85).aspx

[MSDN-NAPAPI] Microsoft Corporation, "NAP Interfaces", http://msdn.microsoft.com/en-us/library/aa369705(v=VS.85).aspx

[MSDN-OSVERSIONINFOEX] Microsoft Corporation, "OSVERSIONINFOEX" Structure, http://msdn.microsoft.com/en-us/library/ms724833.aspx

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997, http://www.ietf.org/rfc/rfc2131.txt

[RFC2409] Harkins, D., and Carrel, D., "The Internet Key Exchange (IKE)", RFC 2409, November 1998, http://www.ietf.org/rfc/rfc2409.txt

1.3 Overview

It is common for network administrators to require authentication and **authorization** for users or devices attaching to their networks. Likewise, administrators require that the devices conform with the security policies of the organization if they are to access the network. For example, an administrator might require that every client accessing the network have a host firewall configured in a particular manner to protect both the network and the client computer.

There are a number of ways in which an administrator can approach this particular problem. Historically, the most common way has been through the creation of written policies that administrators hope users will follow.

This approach only works well with small groups of trustworthy and technically adept personnel. It fails in scenarios in which large groups of users are involved. Many users are either unable or unwilling to follow the prescribed policies.

Network Access Protection (NAP) is a system developed by Microsoft to provide a more reliable alternative to this problem. NAP uses the Statement of Health for NAP Protocol to manage a computer's conformance with corporate security policies. The Statement of Health for NAP Protocol uses Statement of Health (SoH) and Statement of Health Response (SoHR) messages exchanged between a client and a server to validate client conformance with corporate security policies. This protocol can be used in any other mechanism intended to manage the health of connected resources.

Note The term Statement of Health (SoH) occurs both in the name of the protocol and the name of the message from the client to the server. In the interest of clarity, whenever "Statement of Health" is used to refer to the protocol, the phrase "Statement of Health for NAP Protocol" will be used.

The notion of health in this context has to do with conformance to corporate policies for how a system should be configured. A system is healthy if it conforms to corporate policy. It is not healthy if it does not conform to policy.

The purpose of the SoH message is to report the **health state** of the client that is in the process of requesting access to a network resource. SoH messages are typically exchanged as part of the process to authenticate and authorize the client or user. The client uses the SoH to report its state so that the health state of the client can be evaluated against the corporate policy.

The purpose of the Statement of Health Response (SoHR) is two-fold. It indicates whether the client meets policy requirements based on evaluation of the SoH. This allows the server/service to

allow/disallow the connection request. Additionally, the SoHR communicates to the client what, if any, measures it must take in order to conform with policy, in the event that it is not conformant.

These messages may be carried within other authentication and authorization protocols, such as the **Health Certificate Enrollment Protocol (HCEP)**, as specified in [MS-HCEP] section 1.2. Carrying the Statement of Health for NAP Protocol in a higher-layer transport protocol that has built-in security measures has the advantage of securing the Statement of Health for NAP Protocol.

The Statement of Health for NAP Protocol is a simple protocol in which there is a single exchange between the client and the server. The flow is as follows in the case of its successful use in HCEP:

1. The client sends an SoH inside an HCEP message that is posted to a **health registration authority (HRA)**.
2. The SoH is then checked for conformance with network policies.
3. The result is encoded in an SoHR.
4. The HRA replies to the client by sending an HCEP message that includes the SoHR inside.

This process is as specified in [MS-HCEP] section 1.3.

Note The processing done in steps 2 and 3 are specific to the protocol that carries the SoH/SoHR messages.

1.4 Relationship to Other Protocols

The Statement of Health for NAP Protocol can run on any transport protocol. The **SoH client** and server MUST support one or more of the following transport protocols:

- Protected Extensible Authentication Protocol, as specified in [MS-PEAP]
- Health Certificate Enrollment Protocol, as specified in [MS-HCEP]
- Dynamic Host Configuration Protocol (DHCP) Extensions, as specified in [MS-DHCPE]
- Vendor-Specific RADIUS Attributes for Network Access Protection (NAP) Data Structure, as specified in [MS-RNAP] and [RFC2865]

It MAY support additional implementation-defined protocols.

The protocol also uses services and applications (**system health agents (SHAs)**) on the client side to produce the information that must be included in the SoH. Similarly, on the server side, the protocol uses services (**system health validators (SHVs)**) that process and evaluate the SoH and produce an SoHR in response. The interface between the component implementing the protocol and the component providing these services is an implementation choice. Microsoft Windows® includes one specific SHA to report the health state of the systerm and one specific SHV to evaluate the health state. These are the Windows Security Health Agent (WSHA) and Windows Security Health Validator (WSHV), as specified in [MS-WSH]. Notice that SHAs and SHVs can be implemented by third-parties.

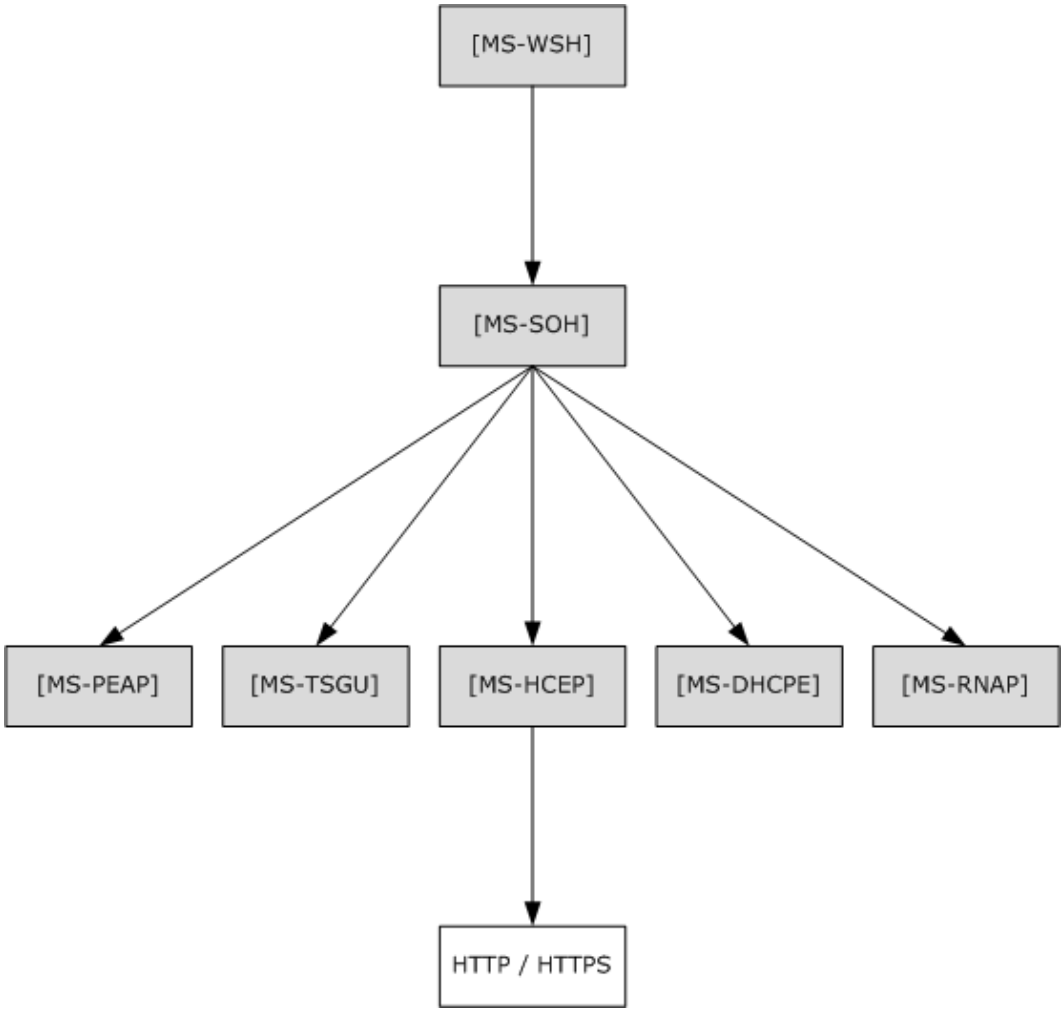


Figure 1: Relationship to other protocols

1.5 Prerequisites/Preconditions

For a Statement of Health for NAP Protocol exchange to occur, the SoH Client must have a **session** with a suitable transport protocol established that uses a **Health Policy Server** or to an intermediary server that can relay the Statement of Health for NAP Protocol to the Health Policy Server.

As a precondition, the SoH Client is required to be able to construct valid SoH messages in the format defined in section 2. It is also a requirement that the client be able to process well-formed SoHR messages. Similarly, as a precondition, the SoH server must be able to process SoH messages and construct SoHR messages.

The actual content of the SoH messages is implementation-specific.

As explained in section 1.7, there are two protocol versions. The SoH Client should send version 2 messages. The SoH server must accept version 2 messages and may accept version 1 messages. The SoH server must create a response that matches the version of the received request.<1>

The most common use for the Statement of Health for NAP Protocol is one in which a client connects to a **network access server (NAS)** and the NAS connects as a **Remote Authentication Dial-In User Service (RADIUS)** client to a RADIUS server. This scenario is specified in [MS-RNAP] section 1.3. Each vendor specifies how the transport handles the SoH message.<2>

1.6 Applicability Statement

The Statement of Health for NAP Protocol is designed to provide to enterprises a mechanism that helps ensure the systems that they manage are healthy.

The protocol may be used in conjunction with an authentication protocol for network access. It is also possible to use the protocol independently of any authentication process. Thus, the protocol does not need to be tied to authentication and authorization, but may only be used to manage the health of the enterprise computing resources. An example of such usage is when DHCP is used as the carrier of SoH or SoHR messages.

Another way of using the Statement of Health for NAP Protocol is to obtain a credential (for example, a X.509 certificate) by using an enrollment process that includes this protocol. Possessing such a credential is the equivalent of having had the client evaluated to be in good health. The Health Certificate Enrollment Protocol (HCEP), as specified in [MS-HCEP], is an example of this usage.

The result of the Health Certificate Enrollment Protocol, in the successful case, is to provide the client with a X.509 certificate that can be used in an authentication protocol such as the Internet Key Exchange (IKE) Protocol (for more information, see [RFC2409]).

1.7 Versioning and Capability Negotiation

The Statement of Health for NAP Protocol has two versions. A version 2 message differs from a version 1 message in that it has an SoH mode subheader, as specified in section 2.2.7. This SoH mode subheader is intended to allow a later version of the protocol to have newer modes of operation. There is no other functional difference between the two versions.

When a server receives a request message from a client, the server creates a response with the same version as the request. The version of the message affects only the headers, as defined in sections 2.2.5 and 2.2.6.

1.8 Vendor-Extensible Fields

The Statement of Health for NAP Protocol provides a single vendor-extensible field in its SoH and SoHR messages (see section 2.2.3.3).

1.9 Standards Assignments

Parameter	Value	Reference
IANA SMI vendor ID for Microsoft	0x137 (decimal 311)	[IANA-ENT]

2 Messages

The following sections specify transport and the syntax of attributes for the Statement of Health for NAP Protocol.

This protocol references commonly used data types as defined in [MS-DTYP].

2.1 Transport

The Statement of Health for NAP Protocol does not provide its own transport. It MUST be carried in some other protocol that provides transport for it. It SHOULD be carried in one of the following protocols:

- Health Certificate Enrollment Protocol (HCEP), as specified in [MS-HCEP].
- Remote Authentication Dial-In User Service (RADIUS), as specified in [MS-RNAP] sections 2.2.1.8 or section 2.2.1.19.
- Protected Extensible Authentication Protocol (PEAP), as specified in [MS-PEAP] section 2.2.4.
- Dynamic Host Configuration Protocol (DHCP), as specified in [MS-DHCPE] section 2.2.2.

2.2 Message Syntax

The SoH and the SoHR messages are identical structures. They are composed of a header followed by a set of **type-length-values (TLVs)**.

The first TLV in every SoH/SoHR message is the System-Health-IDTLV of the System Statement of Health (SSoH)/System Statement of Health Response (SSoHR). The value part of the SSoH/SSoHR is a sequence of **type-value (TV)** structures. Allowable values of the TV structures are defined later in this section.

The TLVs that follow the SSoH/SSoHR are grouped in SoHReportEntry/SoHRReportEntry sets. The System-Health-IDTLV marks the beginning of each set of TLVs that constitute an SoHReportEntry/SoHRReportEntry set, and MUST be present. Each TLV in an SoHReportEntry/SoHRReportEntry set is called an SoHAttribute/SoHRAttribute. There can be zero or more SoHAttributes per SoHReportEntry set (see section 2.2.5.3) besides the System-Health-IDTLV. In addition to the System-Health-IDTLV (see section 2.2.6.3), there MUST be one or more SoHRAttributes per SoHRReportEntry set.

These structures and their allowable values are defined in the following figures. The fields of these structures are transmitted in network-byte order from left to right as shown in the figures and tables throughout this section.

A graphic representation of the top-level structure follows.

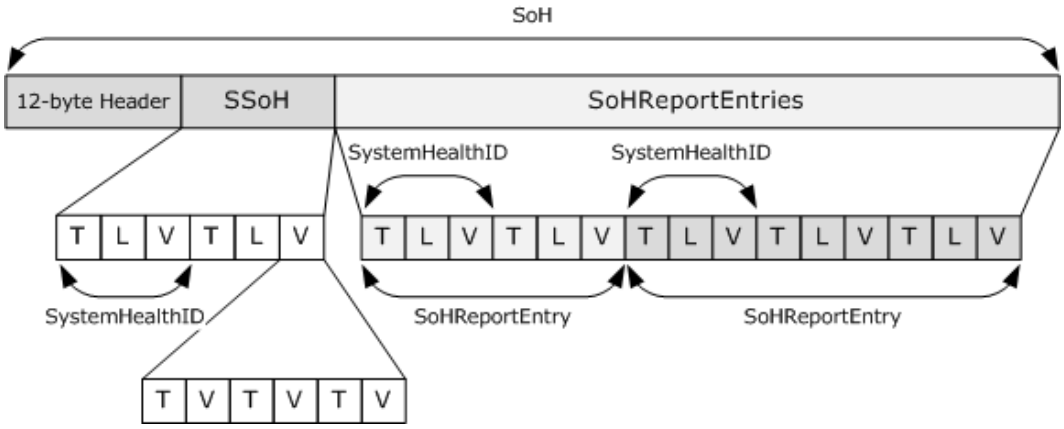


Figure 2: SoH shown without sub-mode header

See section 2.2.7.

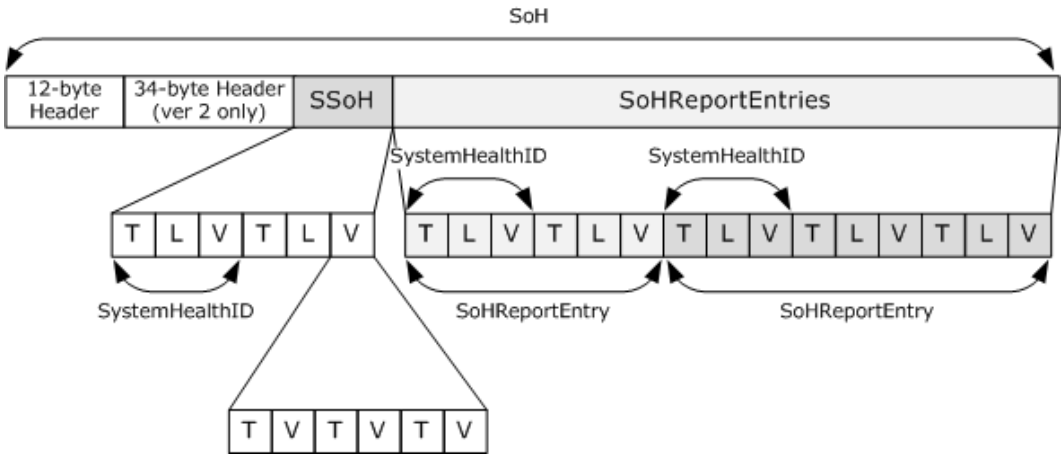


Figure 3: SoH shown with sub-mode header

See section 2.2.7.

2.2.1 Type-Length-Value (TLV) Packet

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

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
M	R	TLV Type														Length															
Value (variable)																															
...																															

M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a **mandatory TLV**.

R (1 bit): The **R** bit is reserved and MUST be set to zero and ignored on receipt.

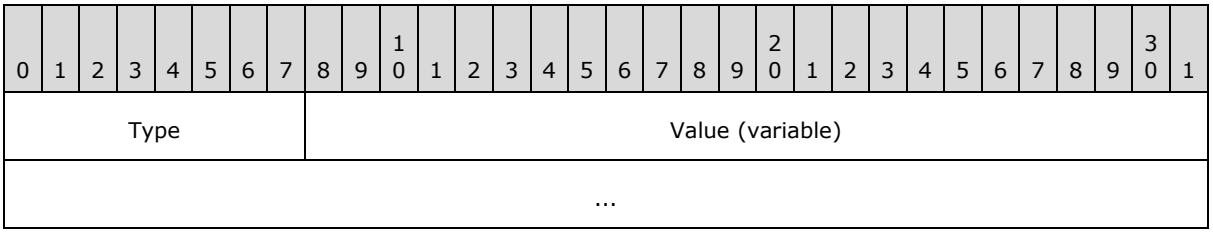
TLV Type (14 bits): A 14-bit unsigned integer that MUST indicate the type of data in the **Value** field. Valid **TLV Type** values MUST be one of those specified in section 2.2.3.

Length (2 bytes): A 16-bit unsigned integer that MUST specify 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.

2.2.2 Type-Value (TV) Packet

The following packet diagram shows the standard type-value (TV) structure that is used by all System Statement of Health (SSoH)/System Statement of Health Response (SSoHR) attributes, as specified in section 2.2.4.



Type (1 byte): An 8-bit unsigned integer that indicates the type of data in the attribute **Value** field.

Value (variable): The length of the **Value** field MUST correspond to the type as defined by the **Type** field. Each type is of a fixed length, although different types have different values.

2.2.3 SoHAttributes / SoHRAttributes

The SoHAttribute/SoHRAttribute elements are messages that are used to construct valid SoH/SoHR messages. A collection of SoHAttributes/SoHRAttributes in which the first attribute is the System-Health-ID constitutes an SoHReportEntry/SoHRReportEntry.

When using a TLV to contain an SoH/SoHR attribute, the TLV types, with values between and including 0 through 256, are reserved for use in the Statement of Health for NAP Protocol and MUST NOT be used for other attributes. The following TLV types MUST be supported for use within this protocol.

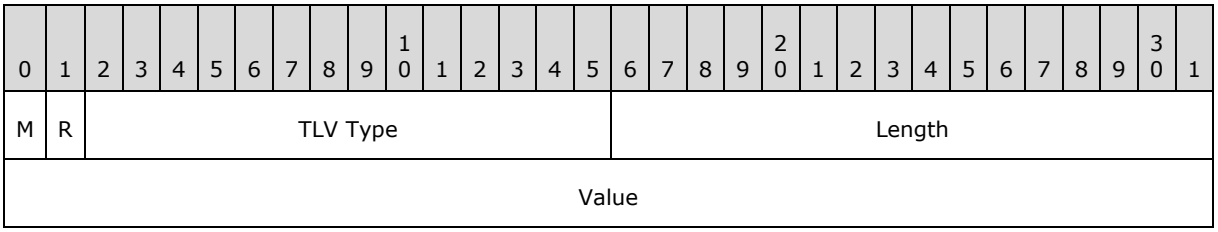
Type	Name	Meaning
2	System-Health-ID	ID of the system health agent (SHA) or system health validator (SHV) that generated the SoHReportEntry or SoHRReportEntry set.

Type	Name	Meaning
4	Compliance-Result-Codes	Result codes specifying if the client computer is compliant.
7	Vendor-Specific	The **Value** field contains a vendor-specific TLV.
14	Failure Category	A code that indicates the Failure Category.

A list of optional TLVs are provided in section 2.2.3.5.

2.2.3.1 System-Health-ID Packet

The ID of the component that generated the SoHReportEntry/SoHRReportEntry set (for the SoH or SoHR message) MUST be the first TLV present in the SoHReportEntry/SoHRReportEntry.



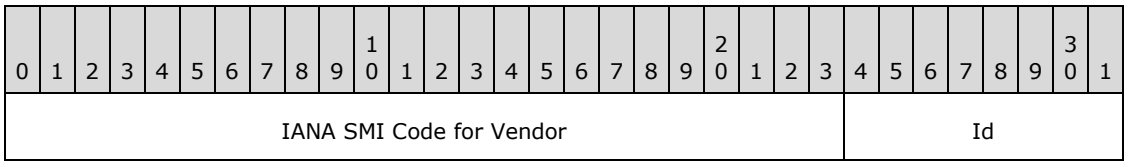
M (1 bit): The **M** bit MUST be set to zero.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

TLV Type (14 bits): The **TLV Type** for this packet type MUST be set to 2.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field. For this packet type, MUST be set to 4.

Value (4 bytes): A 32-bit unsigned integer used to represent the **Health ID** of the SoHReportEntry/SoHRReportEntry. The value MUST be formatted as follows.

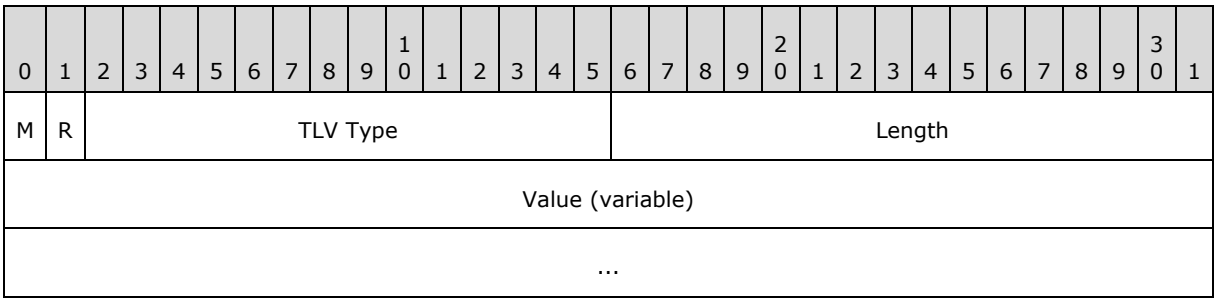


IANA SMI Code for Vendor (3 bytes): A 24-bit unsigned integer that MUST contain the IANA SMI code for the vendor whose component produced the message.

Id (1 byte): An 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>

2.2.3.2 Compliance-Result-Codes

The result of the evaluation of the SoH message is used to specify whether the client computer is compliant with policy. An SoHR message MUST contain this attribute, a Failure Category attribute, or both. An SoH MAY contain this attribute.



M (1 bit): The **M** bit MUST be set to zero.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

TLV Type (14 bits): The **TLV Type** for this packet type MUST be set to 4.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field.

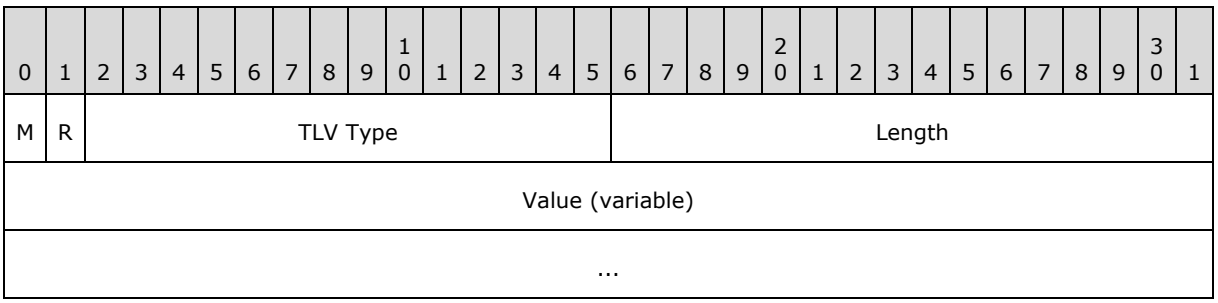
Value (variable): An array of **HRESULT** values that indicate the results of evaluation by the server.

2.2.3.3 Vendor-Specific Packet

The Vendor-Specific packet represents vendor-specific data, in which the format of the data is known only to the vendor.

This attribute is used to carry implementation-specific data from the client to the server and back. For example, an antivirus vendor can include data about the signature database version and the time at which the last complete scan was performed.

This attribute can be present in an SoH and SoHR message. It is also present in SSoH and SSoHR.



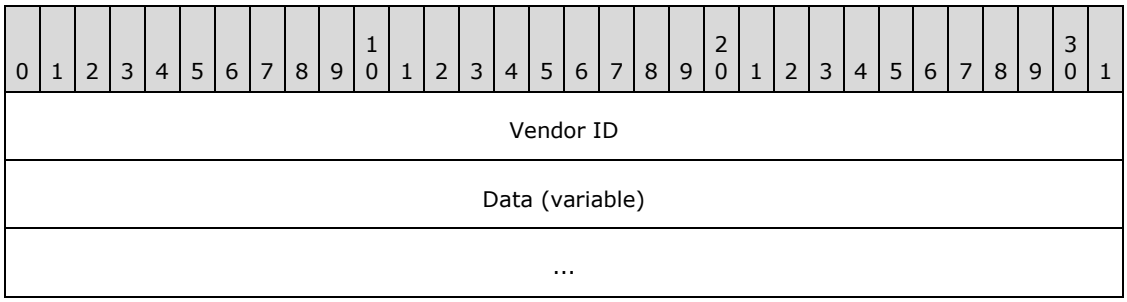
M (1 bit): The **M** bit MUST be set to zero.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

TLV Type (14 bits): The **TLV Type** for this packet type MUST always be 7.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field.

Value (variable): The **Value** field MUST be formatted as follows.

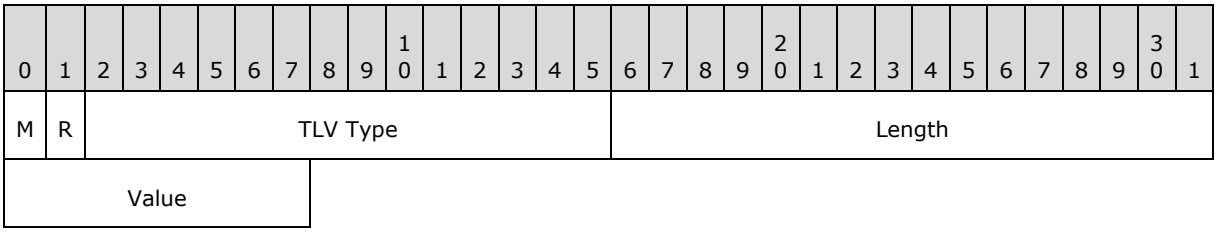


Vendor ID (4 bytes): A 32-bit unsigned integer that SHOULD specify the assigned IANA-assigned SMI for the vendor whose data is to be specified in the **Data** field.<4> NAP does not interpret this field in SoH or SoHR messages. The vendor can use it for any purpose. However, NAP does interpret this field in SSoH and SSoHR.

Data (variable): The format of the **Data** field is vendor specific. An example can be found in the MSSHA implementation in WSHA SoH and WSHV SoHR.

2.2.3.4 Failure Category

The Failure Category attribute is used to classify the type of failure that occurred. An SoHR message MUST contain this TLV, a Compliance-Result-CodeTLV, or both. This attribute MAY be present in an SoH message.



M (1 bit): The **M** bit MUST be set to zero.

R (1 bit): The **R** bit is reserved and MUST be set to zero and ignored on receipt.

TLV Type (14 bits): The **TLV Type** MUST be set to 14.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field. For this packet type, MUST be set to 1.

Value (1 byte): An 8-bit field that MUST contain one of the following values:

Value	Meaning
0	No failure occurred.
1	Failure that is not due to components or communications of the client or server
2	Failure due to client component.
3	Failure due to client communication.
4	Failure due to server component.

Value	Meaning
5	Failure due to server communication.

2.2.3.5 Optional TLVs

The following table contains a list of optional TLVs that can be used by an implementation. The TLVs that are not optional are excluded from the table that follows and can be found in section 2.2.3. These TLVs can be present in an SoH or SoHR message. If these types are used to construct SoH or SoHR messages, the lengths specified in the following sections for each optional TLV MUST be honored.<5>

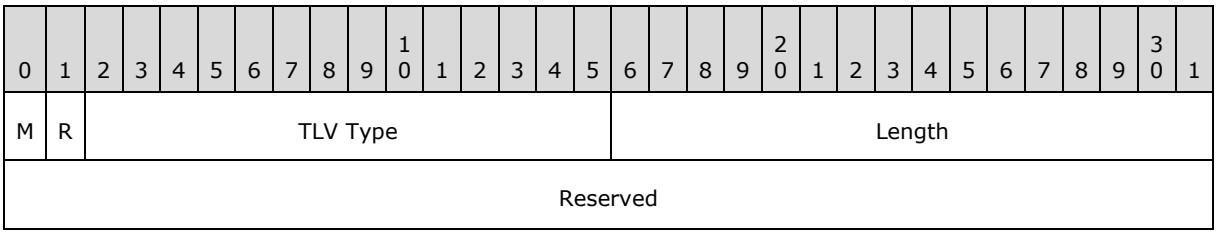
Type	Name	Value	Length in bytes
0	Reserved, specified in section 2.2.3.5.1.	Reserved	4
1	Reserved, specified in section 2.2.3.5.2.	Reserved	4
3	IPv4 Fix-up Servers, specified in section 2.2.3.5.3.	IPv4 addresses of the **fix-up servers**	Variable
5	Time-of-Last-Update, specified in section 2.2.3.5.4.	UTC time when client computer was last updated which is measured as the number of 100-nanosecond intervals since January 1, 1601 (UTC)	8
6	Client-Id, specified in section 2.2.3.5.5.	Identifier for the client	Variable
8	Health-Class, specified in section 2.2.3.5.6.	Type of health check that the SHA is performing (firewall, antivirus, critical update, and so on)	1
9	Software-Version, specified in section 2.2.3.5.7.	Version of the software installed on the client computer	1
10	Product-Name, specified in section 2.2.3.5.8.	Name of the product installed on the client computer	Variable
11	Health Class Status, specified in section 2.2.3.5.9.	Status code for the health-class type given by the Health-Class TLV	Variable
12	SoHGenerationTime, specified in section 2.2.3.5.10.	UTC time when the SoH message was generated	8
13	Error Codes, specified in section 2.2.3.5.11.	Error codes for specific operations that can be contained in the SoHReportEntry or the SoHRReportEntry set	Variable
15	IPv6 Fix-up Servers, specified in section 2.2.3.5.12.	IPv6 addresses of the fix-up servers	Variable

Type	Name	Value	Length in bytes
16-255	Reserved	Reserved	N/A

The following sections detail the defined optional TLVs 0 through 15.

2.2.3.5.1 Optional TLV 0: Reserved

The Optional TLV 0: Reserved packet is reserved for future use, and MUST be ignored if received in error.



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

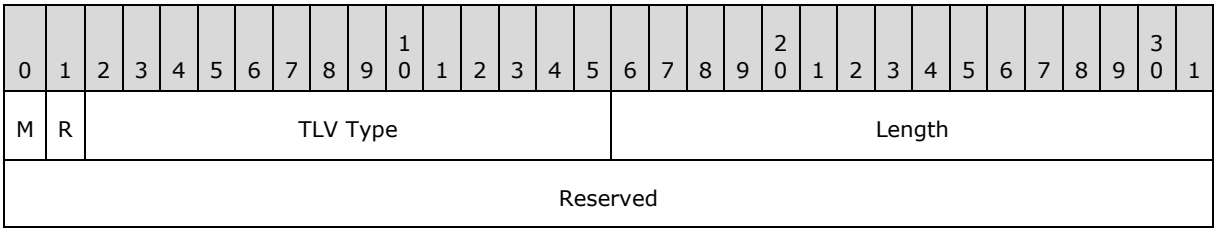
TLV Type (14 bits): Reserved. MUST always be 0.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the Reserved field. For this packet type, MUST be set to 4.

Reserved (4 bytes): Reserved for future use. MUST be 0 and MUST be ignored if received in error.

2.2.3.5.2 Optional TLV 1: Reserved

The Optional TLV 1: Reserved packet is reserved for future use and MUST be ignored if received in error.



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved and MUST be set to zero and ignored on receipt.

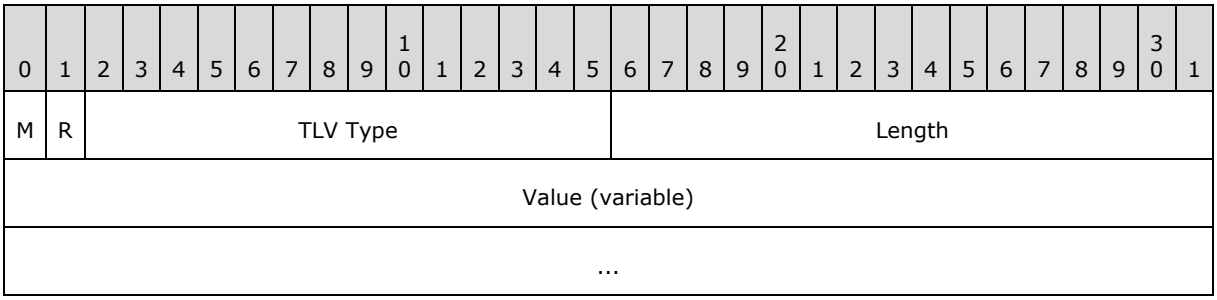
TLV Type (14 bits): Reserved. For this packet type, MUST always be 1.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the Reserved field. For this packet type, MUST be set to 4.

Reserved (4 bytes): Reserved for future use. MUST be 0 and MUST be ignored if received in error.

2.2.3.5.3 Optional TLV 3: IPv4 Fix-up Servers

The Optional TLV 3: IPv4 Fix-up Servers packet provides the addresses of the fix-up servers. An SoH message SHOULD NOT contain this attribute. An SoHR message MAY contain this attribute. When this attribute appears in an SoH or SoHR message, it MAY appear multiple times. An SHA can use this information to perform remediation.



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved and MUST be set to zero and ignored on receipt.

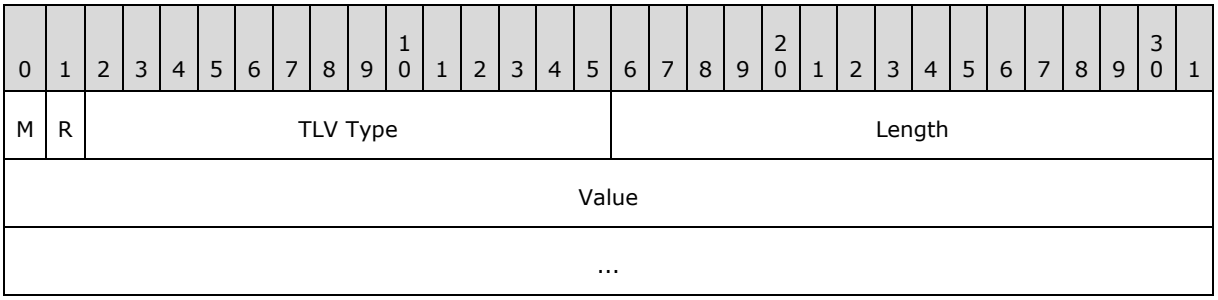
TLV Type (14 bits): The **TLV Type** for this packet type MUST always be 3.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field.

Value (variable): An array containing the 4-byte IPv4 addresses of the fix-up servers.

2.2.3.5.4 Optional TLV 5: Time-of-Last-Update

The Optional TLV 5: Time-of-Last-Update packet specifies the UTC time when the client computer was last updated which is measured as the number of 100-nanosecond intervals since January 1, 1601 (UTC).<6>



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

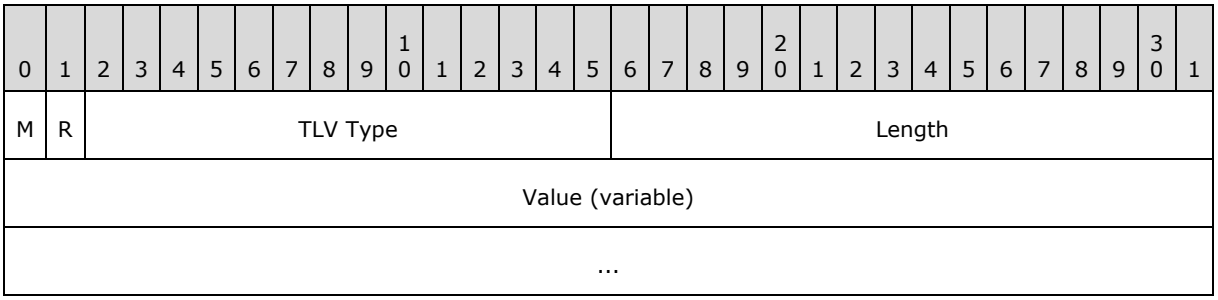
TLV Type (14 bits): The **TLV Type** for this packet type MUST always be 5.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field. For this packet type, MUST be set to 8.

Value (8 bytes): MUST be the UTC time when the client computer was last updated which is measured as the number of 100-nanosecond intervals since January 1, 1601 (UTC).

2.2.3.5.5 Optional TLV 6: Client-ID

The Optional TLV 6: Client-ID packet specifies the client identifier.<7>



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.

Value	Meaning
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

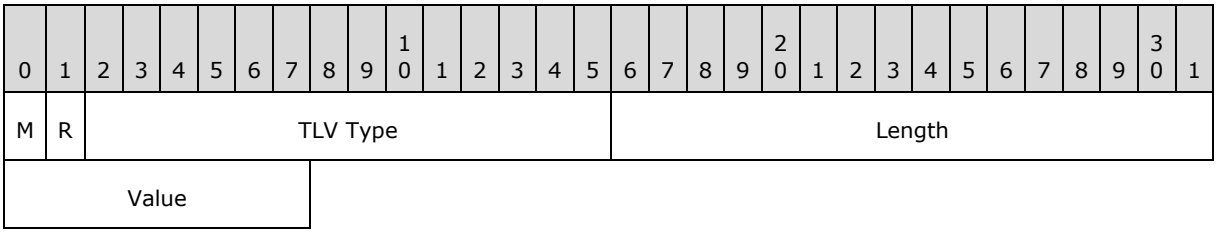
TLV Type (14 bits): The **TLV Type** for this packet type MUST always be 6.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field.

Value (variable): MUST specify the client identifier as a null-terminated string.

2.2.3.5.6 Optional TLV 8: Health-Class

The Optional TLV 8: Health-Class packet specifies the type of health check that the SHA is performing (firewall, antivirus, critical update, and so on).<8>



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

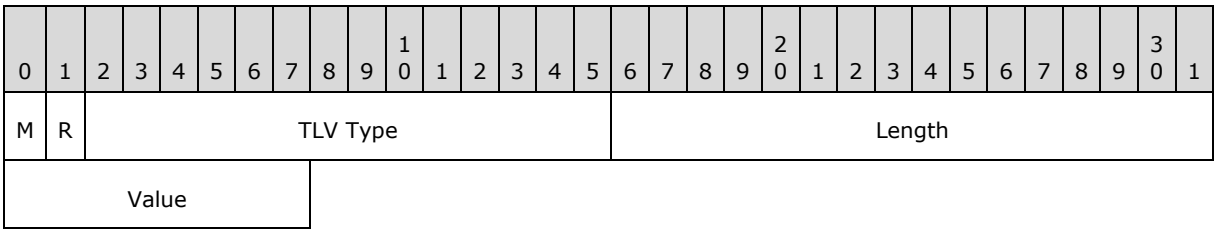
TLV Type (14 bits): The **TLV Type** for this packet type MUST always be 8.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field. For this packet type, MUST be set to 1.

Value (1 byte): An 8-bit field that MUST specify the type of health check that the SHA is performing (firewall, antivirus, critical update, and so on). An example can be found in the MSSHA implementation in WSHA SoH and WSHV SoHR.

2.2.3.5.7 Optional TLV 9: Software-Version

The Optional TLV 9: Software-Version packet specifies the version of the software installed on the client computer.<9>



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

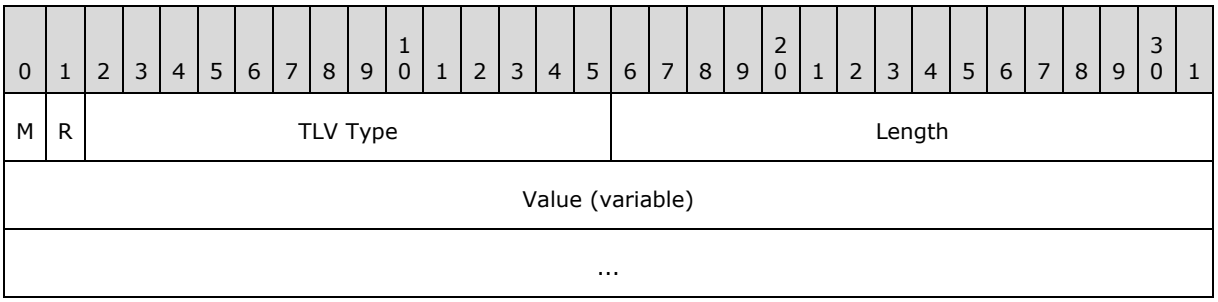
TLV Type (14 bits): The **TLV Type** for this packet type MUST always be 9.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field. For this packet type, MUST be set to 1.

Value (1 byte): An 8-bit field specifying the version of the software installed on the client computer.

2.2.3.5.8 Optional TLV 10: Product-Name

The Optional TLV 10: Product-Name packet specifies the name of the product installed on the client computer.<10>



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

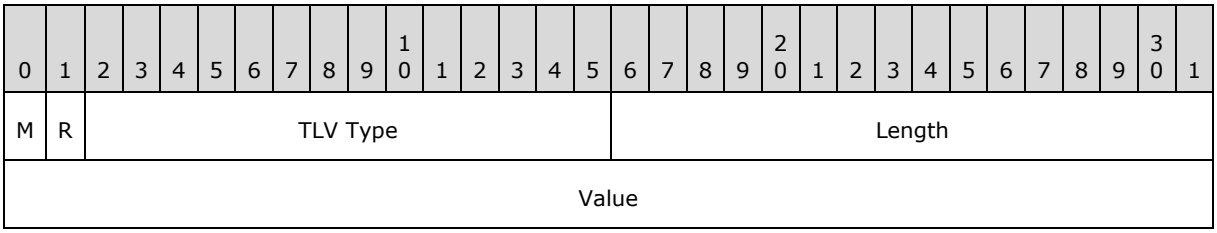
TLV Type (14 bits): The **TLV Type** for this packet type MUST always be 10.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field.

Value (variable): A null-terminated **UNICODE** string that MUST specify the name of the product installed on the client computer.

2.2.3.5.9 Optional TLV 11: Health Class Status

The Optional TLV 11: Health Class Status packet specifies the Status code for the health-class type given by the Health-Class TLV.<11>



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

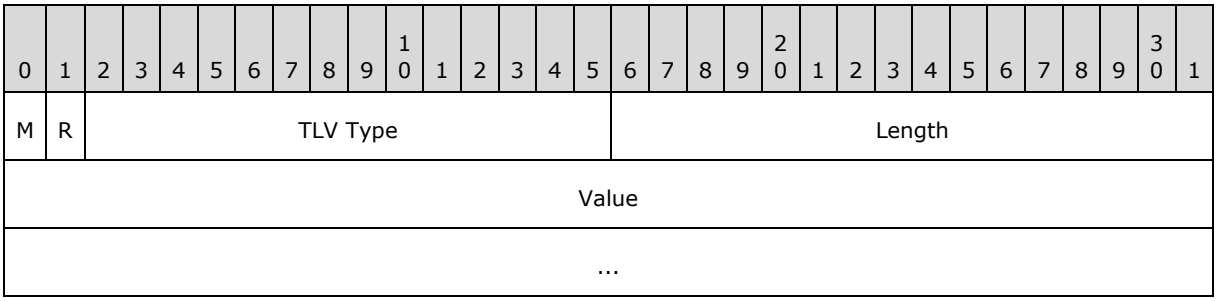
TLV Type (14 bits): The **TLV Type** for this packet type MUST always be 11.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field. For this packet type, the value of **Length** MUST be set to 4.

Value (4 bytes): A 32-bit field that MUST specify the status code for the health-class type given by the Health-Class TLV. An example can be found in the MSSHA implementation in WSHA SoH and WSHV SoHR.

2.2.3.5.10 Optional TLV 12: SOH Generation Time

The Optional TLV 12: SOH Generation Time packet specifies the UTC time when the SoH message was generated.<12>



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

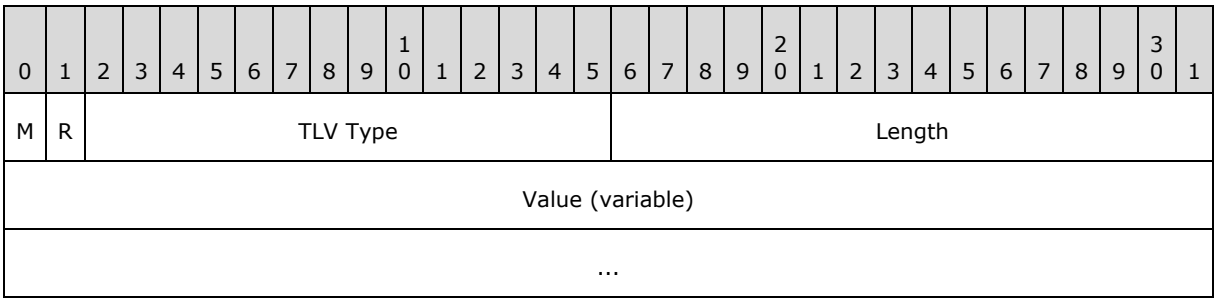
TLV Type (14 bits): The **TLV Type** for this packet type MUST always be 12.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field. For this packet type, MUST be set to 8.

Value (8 bytes): Specifies the UTC time when the SoH message was generated. This 64-bit value MUST represent the number of 100-nanosecond intervals since January 1, 1601 (UTC).

2.2.3.5.11 Optional TLV 13: Error Codes

The Optional TLV 13: Error Codes packet returns a set of error codes of type HRESULT.<13>



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

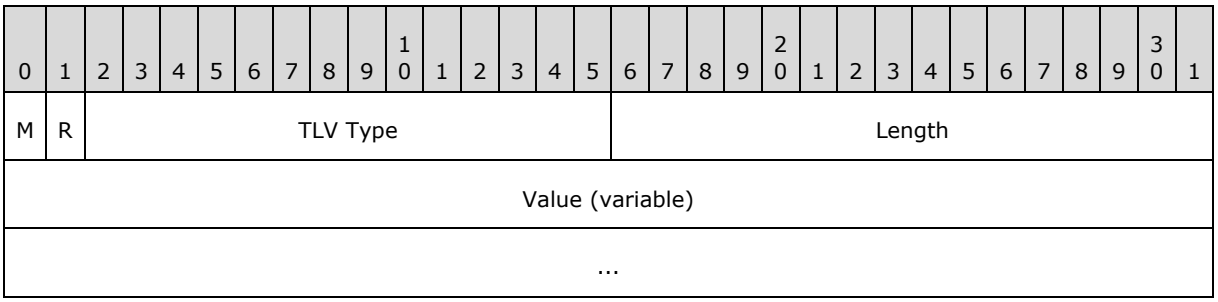
TLV Type (14 bits): The **TLV Type** for this packet type MUST always be 13.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field.

Value (variable): An array of HRESULT values.

2.2.3.5.12 Optional TLV 15: IPv6 Fix-up Servers

The Optional TLV 15: IPv6 Fix-up Servers packet specifies the addresses of the IPv6 Fix-up Servers. An SoH message SHOULD NOT contain this attribute. An SoHR message MAY contain this attribute. When this attribute appears in an SoH or SoHR message, it MAY appear multiple times. An SHA can use this information to perform remediation.



M (1 bit): The **M** bit MUST be set to one of the following values.

Value	Meaning
0	This is a nonmandatory TLV.
1	This is a mandatory TLV.

R (1 bit): The **R** bit is reserved, and MUST be set to zero and ignored on receipt.

TLV Type (14 bits): The **TLV Type** for this packet type MUST always be 15.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field.

Value (variable): MUST be an array of 16-byte IPv6 addresses of the fix-up servers.

2.2.4 SSoHAttribute and SSoHRAttribute

The SSoHAttribute/SSoHRAttribute elements are used to construct valid SSoH and SSoHRTLVs. SSoHAttribute/SSoHRAttribute elements MUST be contained in the **Value** field of a TV (section 2.2.2).

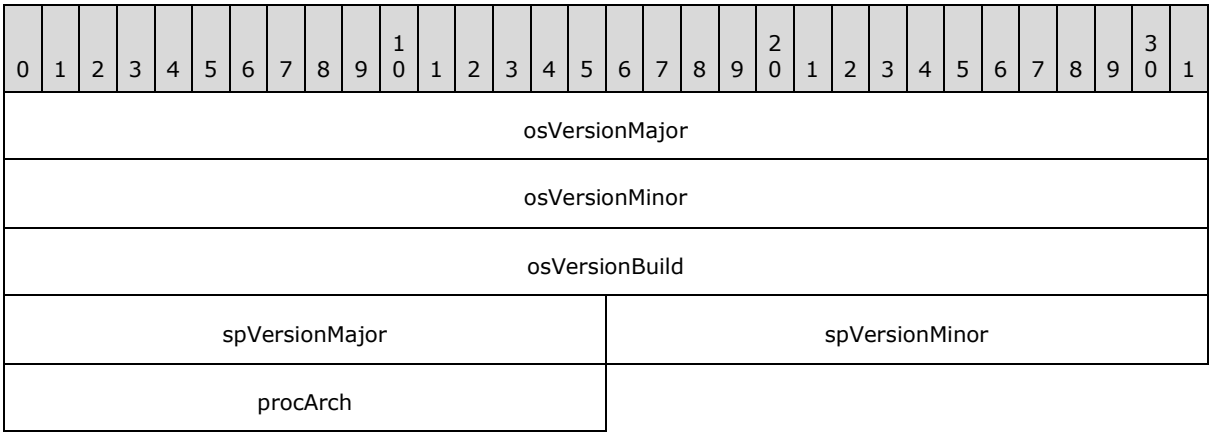
When using a TV to contain an SSoH/SSoHR attribute, the TV types which have values from 0 through 255 are reserved for use in the Statement of Health for NAP Protocol, and MUST NOT be used for other SSoH/SSoHR attributes. The following TV types are for use within this protocol.

TV type	Meaning
1	MS-Machine-Inventory
2	MS-Quarantine-State
3	MS-Packet-Info
4	MS-SystemGenerated-Ids
5	MS-MachineName
6	MS-CorrelationId
7	MS-Installed-Shvs
8	MS-Machine-Inventory-Ex

2.2.4.1 MS-Machine-Inventory Packet

The MS-Machine-Inventory attribute is used to communicate information about the host operating system and its processor architecture.<14>

The attribute MUST be present in an SSoH message, and MAY be present in an SSoHR message<15>.



osVersionMajor (4 bytes): A 32-bit unsigned integer that MUST specify the major version of the host operating system. Some examples are as follows.

Value	Meaning
0x00000004	When combined with an **osMinorVersion** value of 0x00000000, the operating system is Windows NT 4.0.
0x00000005	When combined with an **osMinorVersion** value of 0x00000000, the operating system is Windows 2000. When combined with an **osMinorVersion** value of 0x00000001, the operating system is Windows XP. When combined with an **osMinorVersion** value of 0x00000002, the operating system is Windows Server 2003 or Windows Server 2003 R2.
0x00000006	When combined with an **osMinorVersion** value of 0x00000000, the operating system is Windows Vista or Windows Server 2008. When combined with an **osMinorVersion** value of 0x00000001, the operating system is Windows 7 or Windows Server 2008 R2.

osVersionMinor (4 bytes): A 32-bit unsigned integer that MUST specify the minor version of the host operating system. Some examples are as follows.

Value	Meaning
0x00000000	When combined with an **osMajorVersion** value of 0x00000004, the operating system is Windows NT 4.0. When combined with an **osMajorVersion** value of 0x00000005, the operating system is Windows 2000. When combined with an **osMajorVersion** value of 0x00000006, the operating system is Windows Vista or Windows Server 2008.
0x00000001	When combined with an **osMajorVersion** value of 0x00000005, the operating system is Windows XP. When combined with an **osMajorVersion** value of

Value	Meaning
	0x00000006, the operating system is Windows 7 or Windows Server 2008 R2.
0x00000002	When combined with an **osMajorVersion** value of 0x00000005, the operating system is Windows Server 2003 or Windows Server 2003 R2.

The following table identifies the operating system version corresponding to the specified values for the osMajorVersion and osMinorVersion fields.

osMajorVersion	osMinorVersion	Meaning
0x00000004	0x00000000	The operating system is Windows NT 4.0.
0x00000005	0x00000000	The operating system is Windows 2000.
0x00000005	0x00000001	The operating system is Windows XP.
0x00000005	0x00000002	The operating system is Windows Server 2003 or Windows Server 2003 R2.
0x00000006	0x00000000	The operating system is Windows Vista or Windows Server 2008.
0x00000006	0x00000001	The operating system is Windows 7 or Windows Server 2008 R2.

osVersionBuild (4 bytes): A 32-bit unsigned integer that MUST specify the build number of the host operating system.

spVersionMajor (2 bytes): A 16-bit unsigned integer that MUST specify the major version of the service pack installed on the host operating system. For example, for service pack 3, the major version number is 3. If no service pack has been installed, the value is zero.

spVersionMinor (2 bytes): A 16-bit unsigned integer that MUST specify the minor version of the service pack installed on the host operating system. For example, for service pack 3, the minor version number is 0.

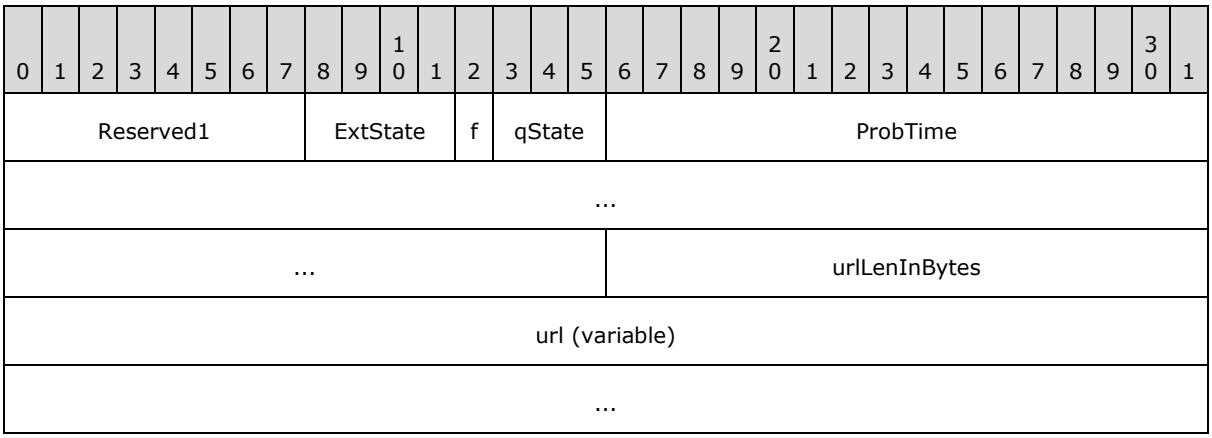
procArch (2 bytes): A 16-bit unsigned integer that MUST specify the processor architecture of the host. Some examples are shown in the following table.

Value	Meaning
0x0000	x86 architecture.
0x0006	Intel Itanium Processor Family (IPF).
0x0009	x64 (AMD or Intel) architecture.
0xffff	Unknown processor.

2.2.4.2 MS-Quarantine-State Packet

The MS-Quarantine-State attribute is used to communicate information about the wanted or resulting permission to a requested network resource for a host. This attribute MUST be present in both the SSoH message and the SSoHR message.

The first 16 bits is a field called **Flags**. This field contains the first four fields: **Reserved1**, **ExtState**, **f**, and **qState**.



Reserved1 (1 byte): An 8-bit reserved field that MUST be set to zero and ignored on receipt.

ExtState (4 bits): A 4-bit field that MUST have one of the following values.

Value	Meaning
0	No evaluation was done (this will be used if the **network policy server (NPS)**policy does not return an extended state).
1	Machine is transitioning from one state to another.
2	Machine is infected, implying a bad health state.
3	Evaluation was done, but extended information could not be determined.

f (1 bit): 1-bit field indicating the health policy server requires that the host MUST remediate any issues before attempting to access its resource again.

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

qState (3 bits): A 3-bit field that MUST be one of the following 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 (8 bytes): A 64-bit field used to represent the time in which the client will be on probation. Probation allows an implementation to grant a client temporary authorization for a

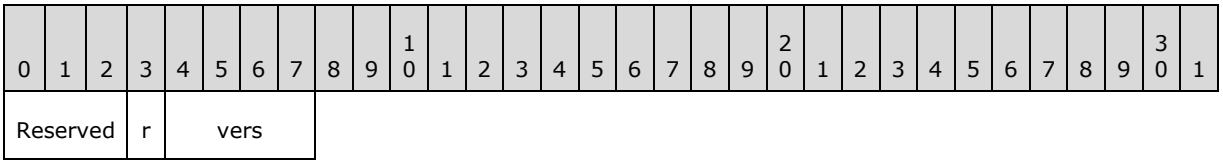
period even when the health check fails. At the end of the probation period, the client SHOULD revalidate its health. The behavior is implementation dependent and SHOULD be policy driven. The value MUST be formatted as a 64-bit value representing the number of 100-nanosecond intervals since January 1, 1601 (UTC).<16>

urlLenInBytes (2 bytes): A 16-bit field that MUST specify the length of the **url** field. The value of **urlLenInBytes** includes the NULL terminating character.

url (variable): UTF-8 (which MUST be as specified in [RFC2781]) representation of a **URL**.

2.2.4.3 MS-Packet-Info Packet

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



Reserved (3 bits): The three bits of the **Reserved** field are reserved, MUST be set to zero when sending and ignored upon receipt.

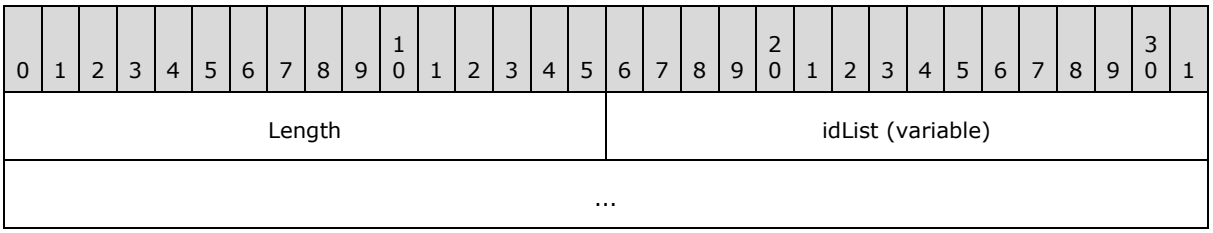
r (1 bit): A 1-bit response/request flag. The value indicates if the attribute contains a request or a response message. The field MUST contain one of the following values.

Value	Meaning
0	Response
1	Request

vers (4 bits): The 4-bit protocol version. MUST be set to 1. This is not to be confused with the version number that is set in the header.

2.2.4.4 MS-SystemGenerated-Ids Packet

The MS-SystemGenerated-Ids attribute contains a list of identifiers corresponding to SoHReportEntry values that contain error information as opposed to information about host state. This attribute MAY be present in an SSoH, and SHOULD NOT be present in an SSoHR.<17>

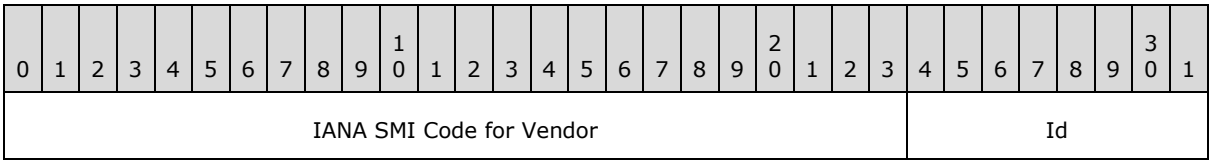


Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **idList** field.

idList (variable): MUST be an array of identifiers for the components that generated the SoHs or SoHRs in the message that contains this attribute. The identifiers MUST be formatted as specified in section 2.2.4.4.1.

2.2.4.4.1 MS-SystemGenerated-Ids Subpacket

The MS-SystemGenerated-Ids subpacket for the MS-SystemGenerated-Ids Packet idList field:

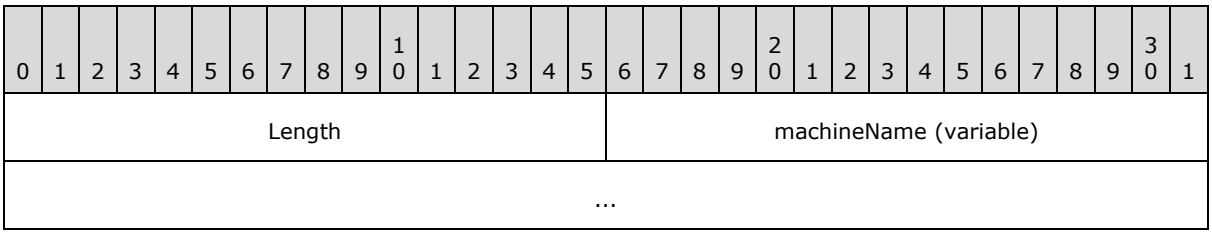


IANA SMI Code for Vendor (3 bytes): A 24-bit unsigned integer that MUST contain the IANA SMI code for the vendor whose component produced the message.

Id (1 byte): An 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 value of the **IANA SMI Code for Vendor** field, allows the routing of an SoHReportEntry set from a client component to the corresponding server-side component that can deal with it. Similarly, the SoHRReportEntry set is routed to the originator based on this ID. The value of 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 ID fully identifies the source and destination of each SoHReportEntry and SoHRReportEntry set.

2.2.4.5 MS-MachineName Packet

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.

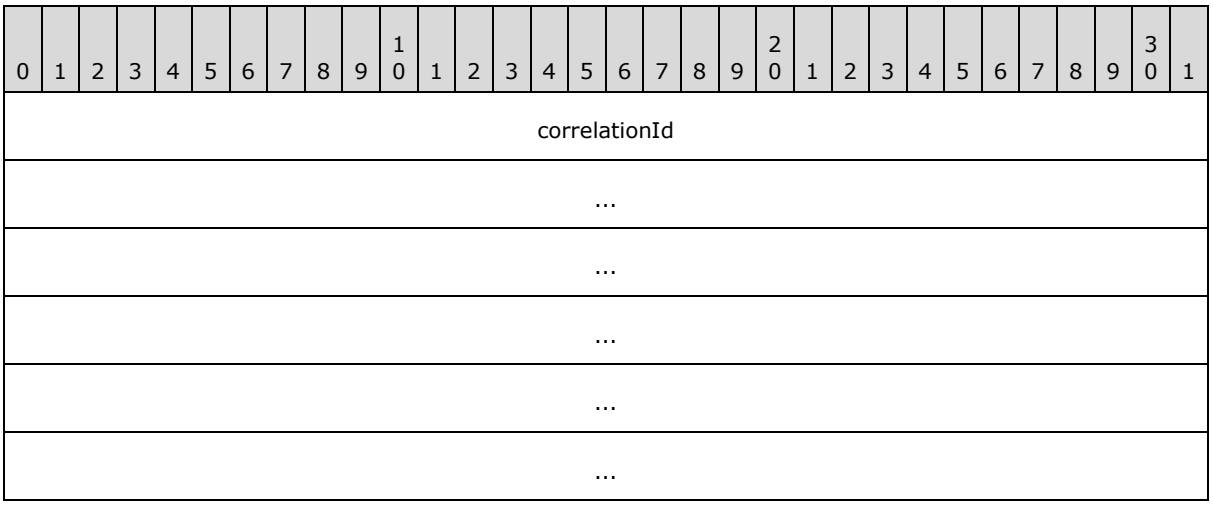


Length (2 bytes): A 16-bit field that MUST specify the length of the machineName field.

machineName (variable): A null-terminated UTF-8 encoded string field that MUST represent the **computer name** of the computer that generated the message.<18>

2.2.4.6 MS-CorrelationId Packet

The MS-CorrelationId attribute is used for diagnostic purposes to facilitate correlating messages related to a single transaction together. This attribute MUST be present in both the SSoH and the SSoHR.



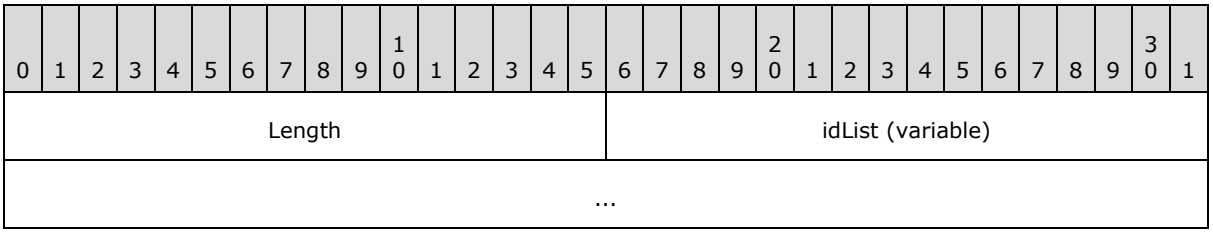
correlationId (24 bytes): A 192-bit field that MUST represent a unique transaction identifier shared across SSoH and SSoHR messages. The format of the **correlationId** is implementation specific.<19>

2.2.4.7 MS-Installed-Shvs Packet

The MS-Installed-Shvs packet is a list of identifiers of services that can evaluate SoH messages on the SoH server.<20>

These identifiers of services can be used as hints to determine what **health messages** to send.<21>

This attribute SHOULD be present in the SSoHR and MAY be present in SSoH.

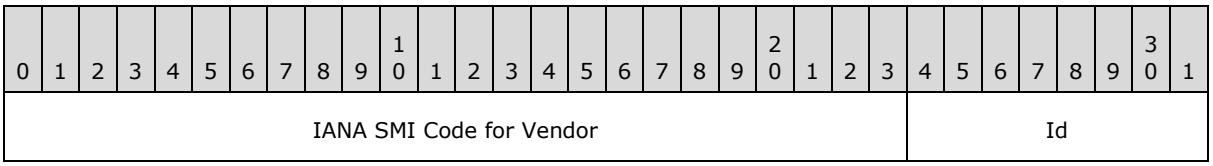


Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **idList** field.

idList (variable): MUST be an array of identifiers for the components that generated the SoHR messages on the server. The identifiers MUST be formatted as specified in section 2.2.4.7.1.

2.2.4.7.1 MS-Installed-Shvs Subpacket

The MS-Installed-Shvs subpacket provides the information about the identifiers contained in the **idList** field of the MS-Installed-Shvs packet.

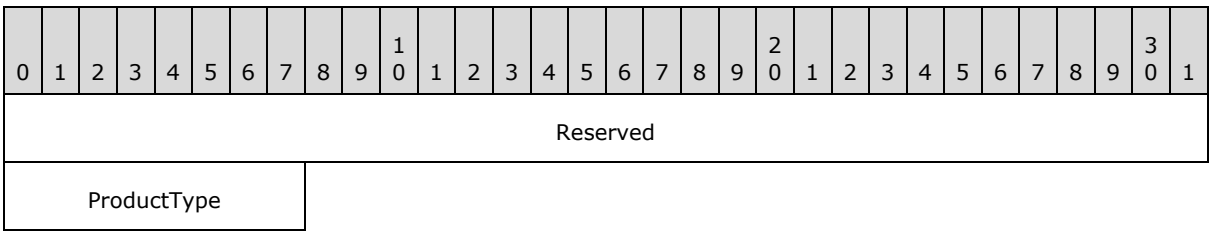


IANA SMI Code for Vendor (3 bytes): A 24-bit unsigned integer that MUST contain the IANA SMI code for the vendor whose component produced the message.

Id (1 byte): An 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.

2.2.4.8 MS-Machine-Inventory-Ex Packet

The MS-Machine-Inventory-Ex packet is used to communicate additional information about the system sending the attribute. This attribute MUST be present in the SSoH and MAY be present in the SSoHR.<22>



Reserved (4 bytes): A 32-bit reserved value. MUST be ignored on receipt.

ProductType (1 byte): An 8-bit field used to represent the type of the operating system. It MUST have one of the following values.

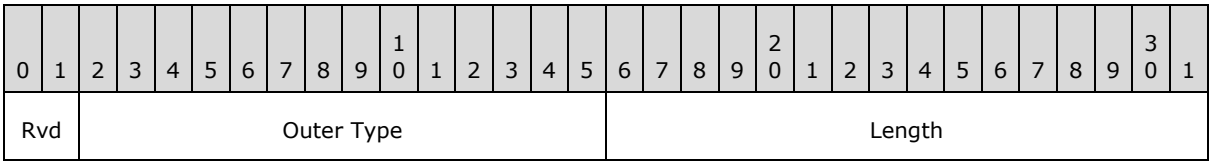
Value	Meaning
0x01	The system is a client.
0x02	The system is a domain controller running on a Windows server operating system.
0x03	The system is a server.

2.2.5 SoH

The SoH message is used to represent a host's claims about its health state. It contains a header followed by a body which includes the remainder of the message content.

2.2.5.1 SoH Header

This is the SoH Header packet for the SoH message.



IANA SMI Code	
Inner Type	Inner Length

Rvd (2 bits): The **Rvd** field is reserved, and MUST be set to zero and ignored on receipt.

Outer Type (14 bits): A 14-bit unsigned integer that MUST be set to 7.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the IANA SMI**Code** field, **Inner Type** field, **Inner Length** field, and the SoH Body (section 2.2.5.2).

IANA SMI Code (4 bytes): A 32-bit unsigned integer that MUST be set to 0x00000137 (see section 1.9). This value, in combination with the value of the **Inner Type** field, allows implementations to identify that these messages belong to the Statement of Health for NAP Protocol. This is useful when implementations at either the client side or the server side get other messages formatted similarly, as EAP TLVs.

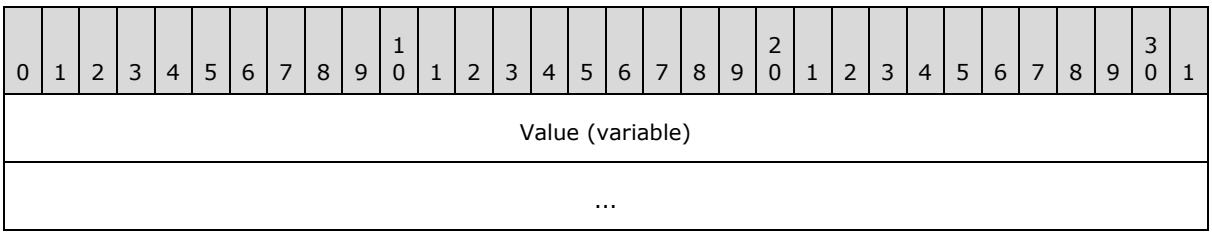
Inner Type (2 bytes): A 16-bit unsigned integer that MUST have the value 0x0001 or 0x0002. This determines the version of the message content, and dictates the format of the data in the SoH Body.

Value	Meaning
0x0001	SSoH SoHReportEntry (0 plus)
0x0002	SoH Mode Subheader SSoH SoHReportEntry (0 plus)

Inner Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the SoH Body.

2.2.5.2 SoH Body

This is the SoH Body packet for the SoH message. The SoH Body MUST follow an SoH Header (section 2.2.5.1) and MUST contain a set of type-length-values (TLVs).



Value (variable): A variable-length field that MUST contain data as follows:

If the **Inner Type** field of the SoH Header is 0x0001:

- SSoH (section 2.2.8)

- SoHReportEntry (section 2.2.5.3) (0 plus)

If the **Inner Type** field of the SoH Header is 0x0002:

- SoH Mode Subheader (section 2.2.7)
- SSoH
- SoHReportEntry (0 plus)

2.2.5.3 SoHReportEntry

The SoHReportEntry message is used to represent a set of SoHAttributes; it has no header of its own and is simply constructed as a set of SoHAttributes.

The System-Health-IDSoH attribute MUST be the first SoH attribute. After this, any set of SoHAttributes can be present (see section 2.2).

The SoHReportEntry message MUST contain one or more additional SoHAttributes as follows (see section 2.2):

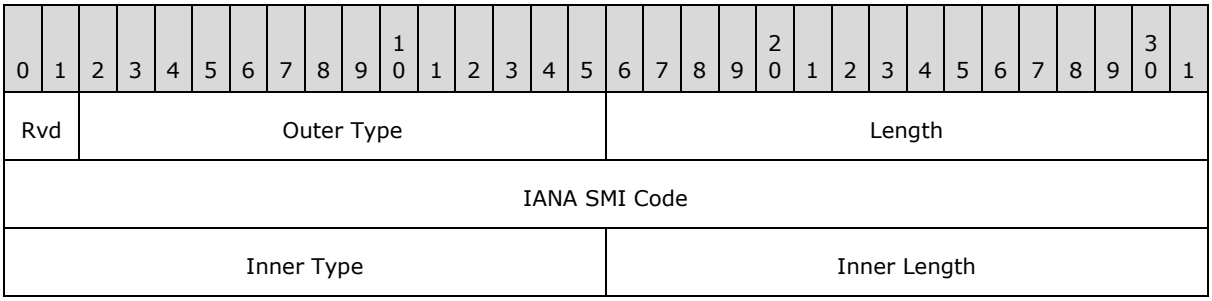
- System-Health-ID
- SoHAttribute (0 plus)

2.2.6 SoHR

The SoHR message is used to transport information about the result of the evaluation of an SoH message by the health policy server. It contains a header followed by a body which includes the remainder of the message content.

2.2.6.1 SoHR Header

This is the SoHR Header for the SoHR packet.



Rvd (2 bits): The **Rvd** field is reserved, and MUST be set to zero and ignored on receipt.

Outer Type (14 bits): A 14-bit unsigned integer that MUST be set to 7.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the IANA SMI**Code** field, **Inner Type** field, **Inner Length** field, and the SoHR Body (section 2.2.6.2).

IANA SMI Code (4 bytes): A 32-bit unsigned integer that MUST be set to 0x00000137 (see section 1.9). This value, in combination with the value of the **Inner Type** field, allows implementations to identify that these messages belong to the Statement of Health for NAP

Protocol. This is useful when implementations at either the client side or the server side get other messages formatted similarly, as EAP TLVs.

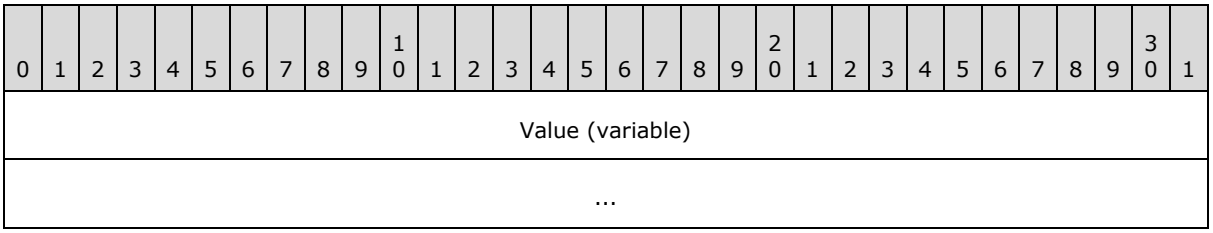
Inner Type (2 bytes): A 16-bit unsigned integer that MUST be set to 0x0001 or 0x0002. This value MUST be the same as the **Inner Type** value in the corresponding SoH message that the server received. This determines the version of the message content and dictates the format of the data in the SoHR Body.

Value	Meaning
0x0001	SSoHR SoHRReportEntry (0 plus)
0x0002	SoH Mode Subheader SSoHR SoHRReportEntry (0 plus)

Inner Length (2 bytes): A 16-bit unsigned integer that MUST indicate the length, in bytes, of the SoHR Body.

2.2.6.2 SoHR Body

This is the SoHR Body packet for the SoHR message. The SoHR Body MUST follow an SoHR Header (section 2.2.6.1) and MUST contain a set of type-length-values (TLVs).



Value (variable): A variable-length field that MUST contain data as follows:

If the value of the **Inner Type** field of the SoHR Header is 0x0001:

- SSoHR (section 2.2.9)
- SoHRReportEntry (section 2.2.6.3) (0 plus)

If the value of the **Inner Type** field of the SoHR Header is 0x0002:

- SoH Mode Subheader (section 2.2.7)
- SSoHR
- SoHRReportEntry (0 plus)

2.2.6.3 SoHRReportEntry

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

The System-Health-Id packet MUST be the first SoHR attribute.

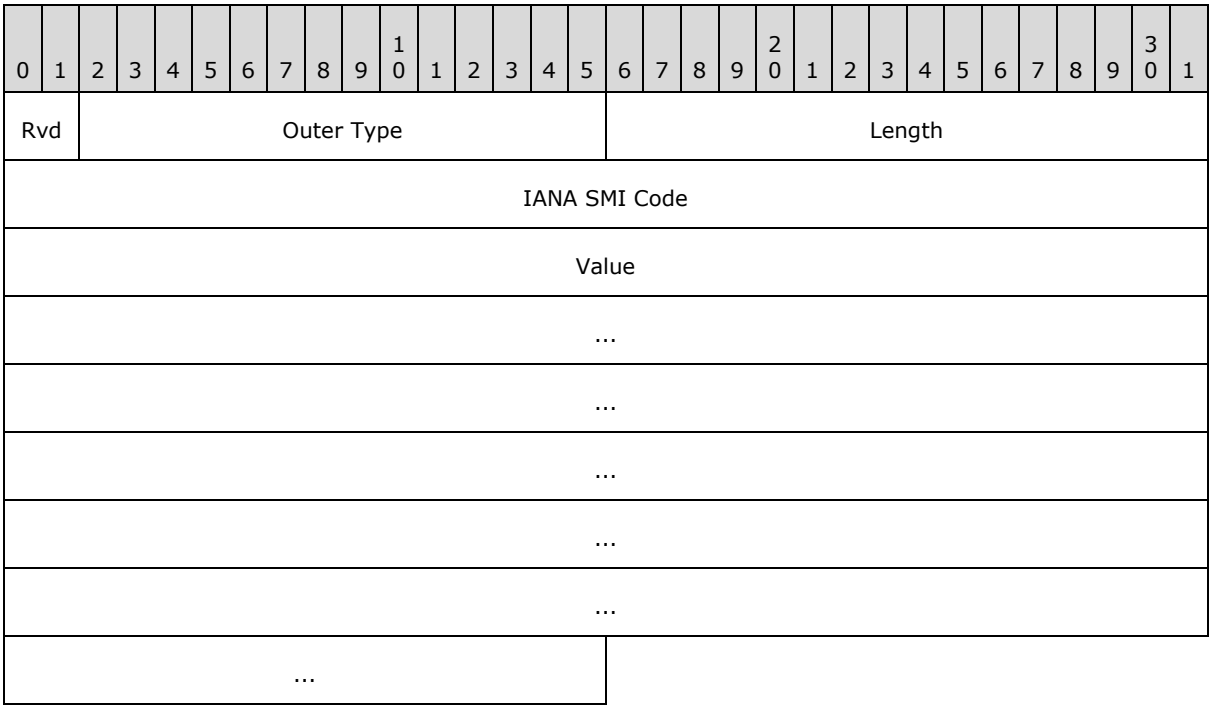
The System-Health-Id MUST be followed by either a Compliance-Result-Codes packet, a Failure Category packet, or both.

The SoHRReportEntry message MUST contain two or more additional SoHR attributes as follows (see section 2.2):

- System-Health-Id
- SoHRAttribute (1 plus) (MUST include a Compliance-Result-Codes, a Failure Category, or both.)

2.2.7 SoH Mode Subheader

The SoH Mode Subheader is used to represent information about the information following it.



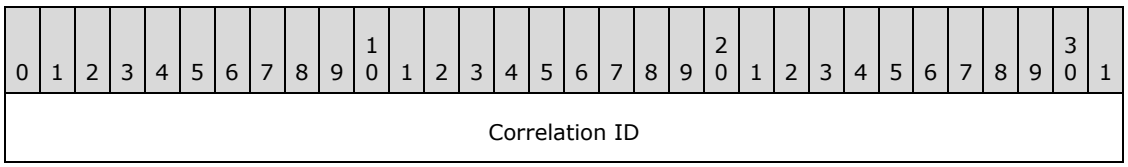
Rvd (2 bits): The **Rvd** field is reserved, and MUST be set to zero and ignored on receipt.

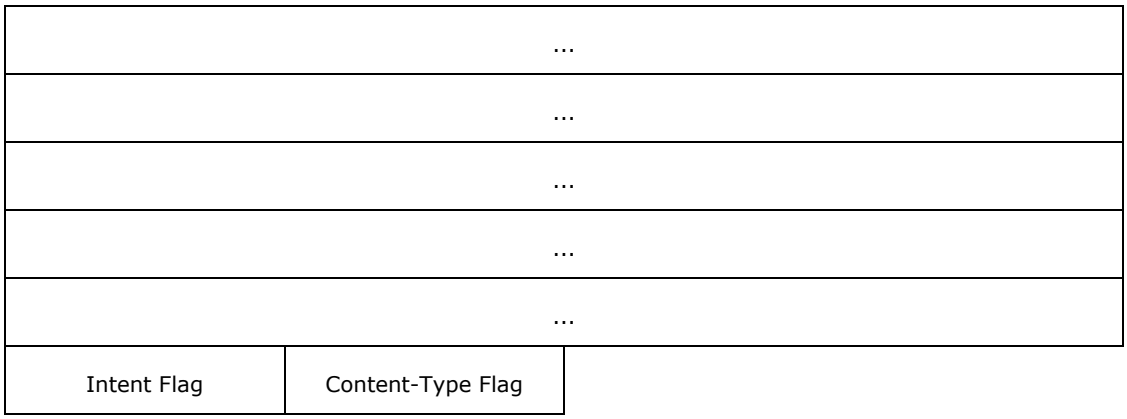
Outer Type (14 bits): A 14-bit unsigned integer that MUST be set to 7.

Length (2 bytes): A 16-bit unsigned integer that MUST specify the length, in bytes, of the **Value** field and **IANA SMI Code**.

IANA SMI Code (4 bytes): A 32-bit unsigned integer that MUST be set to 0x00000137 (see section 1.9).

Value (26 bytes): A field that MUST contain 26 bytes distributed as follows:





Correlation ID (24 bytes): 24 bytes that MUST have the same value as the **Value** field of the MS-CorrelationId message.

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

Value	Meaning
0x01	SoH request message
0x00	SoHR response message

Content-Type Flag (1 byte): 1 byte that MUST be 0x00. This field is intended to help in Statement of Health for NAP Protocol enhancements in the future.

2.2.8 SSoH

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

The SSoH message has no header of its own and is simply constructed as the following ordered sequence of SoHAttributes and SSoHAttributes:

- **System-Health-Id** attribute (see section 2.2.3.1). The value of this attribute MUST be decimal 79616 (0x00013700), created in accordance with the specifications in section 2.2.1.
- **Vendor-Specific** attribute (see section 2.2.3.3). The **Vendor-Specific** attribute MUST have the following fields set to the specified values:
  - **Vendor ID**: IANA SMI code, as specified in [IANA-ENT] set to 0x00000137.
  - **Value**: MUST contain the following SSoHAttributes:
    - **MS-Machine-Inventory** (see section 2.2.4.1).
    - **MS-Quarantine-State** (see section 2.2.4.2).
    - **MS-Packet-Info** (see section 2.2.4.3).
    - **MS-MachineName** (see section 2.2.4.5).

- **MS-CorrelationId** (see section 2.2.4.6).

  The **Value** field MAY contain the following SSoHAttributes:

- **MS-SystemGenerated-Ids** (optional; see section 2.2.4.4).
- **MS-Machine-Inventory-Ex** (optional; see section 2.2.4.8).<23>

2.2.9 SSoHR

The SSoHR message is used to represent generic information about the Health Policy Server.

The SSoHR message has no header of its own and is simply constructed as the following ordered sequence of SoHR attributes and SSoHR attributes:

- **System-Health-Id** attribute (see section 2.2.3.1). The value of this attribute MUST be decimal 79616 (0x00013700), created in accordance to the specifications in section 2.2.1.
- **Vendor-Specific** attribute (see section 2.2.3.3). The **Vendor-Specific** attribute MUST have the following fields set to the specified values:
  - **Vendor ID**: IANA SMI code, as specified in [IANA-ENT] set to 0x00000137.
  - **Value**: MUST contain the following SSoHR attributes:
    - **MS-Packet-Info** (see section 2.2.4.3).
    - **MS-MachineName** (see section 2.2.4.5).
    - **MS-CorrelationId** (see section 2.2.4.6).
    - **MS-Quarantine-State** (see section 2.2.4.2).

      The **Value** field SHOULD contain the following SSoHR attributes:

    - **MS-Installed-Shvs** (optional; see section 2.2.4.7).<24>

3 Protocol Details

The following sections specify details of the Statement of Health for NAP Protocol, including abstract data models and message processing rules.

3.1 Common Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

The abstract data models for client and server are specified in sections 3.2.1 and 3.3.1 respectively.

3.1.2 Timers

None.

3.1.3 Initialization

The Statement of Health for NAP Protocol does not require explicit initialization; however, the transports that carry the protocol can require explicit initialization.

3.1.4 Higher-Layer Triggered Events

IsValidSoH: An abstract interface that evaluates the SoH message for syntax correctness.

```
HRESULT IsValidSoH([in] SoH message);
```

The interface receives an SoH message (section 2.2.5) and returns an **HRESULT (section 2.2.18)** value as follows:

Value	Label	Meaning
0x00000000	ERROR_SUCCESS	The message syntax was successfully evaluated and found compliant to the definitions in section 2.2.5.
0x0000000D	ERROR_INVALID_DATA	The message failed to comply with the expected syntax or data value.

Additional implementation-specific error codes MAY be returned.

IsValidSoHR: An abstract interface that evaluates the SoHR message for syntax correctness.

```
HRESULT IsValidSoHR([in] SoHR message);
```

The interface receives an SoHR message (section 2.2.6) and returns an **HRESULT** value as follows:

Value	Label	Meaning
0x00000000	ERROR_SUCCESS	The message syntax was successfully evaluated and found compliant to the definitions in section 2.2.6.
0x0000000D	ERROR_INVALID_DATA	The message failed to comply with the expected syntax or data value.

Additional implementation-specific error codes MAY be returned.

3.1.5 Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Client-Specific Details

3.2.1 Abstract Data Model

The Statement of Health for NAP Protocol requires a state to be tracked on the SoH client.

ShaTable: The SoH client maintains a list of all registered SHAs. The list includes the callback interface **INapSystemHealthAgentCallback** that is part of the **SHA API** and which is used to interact with each SHA.

LastResult: Represents the last known system health in respect to a certain SHA and includes the following fields:

Field	Meaning
id	The health ID of the SHV that sent the result.
FailureCategory	The classification of the type of failure, if any, that last occurred. A list of possible values is located in section 2.2.3.4. The default value is zero and implies that no failure occurred.
ComplianceResultCodes	A list of HRESULT values that is used to specify whether the client computer is compliant with policy. The default is an empty list.

LastResults-cache: The SoH client maintains a cache of all **LastResult** entries related to each SHA.

QecConnection: An entry which represents the **enforcement client's** connections. It includes the following fields:

Field	Meaning
MS-CorrelationId	A unique identifier that is used to ensure that a received SoHR message

Field	Meaning
(section 2.2.4.6)	corresponds to a sent SoH message, as specified in section 3.2.5.4.
MS-Quarantine-State (section 2.2.4.2)	The value of the last received MS-Quarantine-State, as specified in section 2.2.4.2.
SoHRequest	An opaque buffer containing the SoH request message. The field contains two subfields: **data**, which points to an array of bytes containing the buffer, and **size**, which specifies the buffer size.
SoHResponse	An opaque buffer containing the SoH response message. The field contains two subfields: **data**, which points to an array of bytes containing the buffer, and **size**, which specifies the buffer size.

Note The **SoHRequest** and **SoHResponse** fields are considered empty when the corresponding **size** subfield is set to 0 and the associated **data** subfield is set to NULL; otherwise, both subfields are considered to be set.

QecConnection-cache: The SoH client maintains a cache of all **QecConnection** entries related to each enforcement client.

3.2.2 Timers

3.2.2.1 Probation Timer

The SoH client uses the Probation Timer. Probation grants the SoH client temporary authorization for a period of time even when the health check fails. At the end of the probation time period, the SoH client SHOULD revalidate its health.

3.2.3 Initialization

The **ShaTable** ADM element (section 3.2.1) is initially empty.

3.2.4 Higher-Layer Triggered Events

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

- A user reboots a machine.
- The status of the client changes. For example, the firewall on the client is turned off.

In addition, events specific to the transport mechanism that carries the SoH messages can result in SoH messages being sent by the SoH client. For example, if DHCP is used to carry SoH messages, the renewal of the client IP address can result in an SoH message being sent to the server.

The SoH client MAY be called by the following abstract interface to retrieve compliance or failure information for each SHA.

```
HRESULT GetLastResult (    [in] DWORD  healthId,    [out]  LastResult result);
```

The **GetLastResult** abstract interface receives the health ID of a certain SHA and retrieves the **LastResult** ADM element (section 3.2.1). When the interface is called, the SoH client searches for the relevant entry in the **LastResults-cache** ADM element and returns it to the caller. Upon success, the interface SHOULD return ERROR_SUCCESS (0x00000000). When any failure to find

information occurs, the interface returns any value other than (0x00000000). The interface can be used by client user interface applications, such as netstat.

3.2.4.1 GetFixupServers

The following abstract interface MAY be called to retrieve the list of fix-up servers.

```
HRESULT GetFixupServers(    [in] SoHR message,    [out] IPv4List ip4servers,    [out]
IPv6List ip6servers);
```

The **GetFixupServers** abstract interface receives an **SoHR** message (section 2.2.6). When the interface is called, the SoH client parses the SoHR message.

The SoH client extracts the **IPv4 Fix-up Servers** packet (section 2.2.3.5.3) and updates the *ip4servers* parameter with the list of **IPv4** entries, where each entry is 4 bytes.

The SoH client extracts the **IPv6 Fix-up Servers** packet (section 2.2.3.5.12) and updates the *ip6servers* parameter with the list of **IPv6** entries, where each entry is 16 bytes.

The interface SHOULD return ERROR_SUCCESS (0x00000000) upon success, or any other value upon failure.

3.2.5 Processing Events and Sequencing Rules

The processing of SoH and SoHR messages on the client is performed by the Microsoft Windows® SHA as described in [MS-WSH] or via an implementation-specific SHA developed by a third-party using the COM interface **INapSystemHealthAgentCallback**[MSDN-INapSysHA] which allows plug-ins to evaluate the required resources on the client machine.

An SoH contains data (contained in the SoHReportEntry values) that reports the client's current status to the health policy server. The value of this data is typically provided by software on the client that provides security services, such as an antivirus client or a security update client. Likewise, the validation of this data is typically provided by an antivirus server or a security update server. The Statement of Health for NAP Protocol itself simply provides the mechanism for this data to be supplied and evaluated.

An SoH client MUST do the following:

1. Create valid SoH messages containing host status information in accordance with locally configured policy by using one or more suitable SHAs.<25> The message version SHOULD be version 2.

2. The SoH client informs all registered enforcement clients of a change in the health state by calling **the INapEnforcementClientCallback::NotifySoHChange**, which is part of the **NAP EC API**. Upon such notifications, the enforcement clients SHOULD query the SoH client for an SoH message using the **INapEnforcementClientBinding::GetSoHRequest** method that is part of the NAP EC API, and then call **INapEnforcementClientConnection::GetSoHRequest** that is part of the NAP EC API, in order to retrieve the SoH message as an opaque buffer from the SoH client. The enforcement clients SHOULD send the SoH message to the server.

3. Receive an SoHR message from the registered enforcement client that was used to send the SoH (for example, HCEP). The enforcement client SHOULD call **INapEnforcementClientConnection::SetSoHResponse** that is part of NAP EC API and pass the SoHR as an opaque buffer, and then call the **INapEnforcementClientBinding::ProcessSoHResponse** that is part of the NAP EC API.

4. Process the SoHRs received.<26>

3.2.5.1 Sending SoHs

An SoH client MUST ensure that all SoHs sent contain unique values for the **correlationId** subfield in the MS-CorrelationId value of the SSoH included in the SoH.<27> The SoH client MUST update the **MS-CorrelationId** field in the appropriate **QecConnection** entry accordingly.

Before sending the SoH message to each registered enforcement client, the SoH client MUST test the appropriate **QecConnection** entry in the **QecConnection-cache** ADM element to determine whether the **SoHRequest** field is set and the **SoHResponse** is empty, and therefore, a pending SoH request exists for the enforcement client. If such a request exists, The SoH client MUST remove the SoH from the **MS-CorrelationId Cache** ADM (section 3.2.1) and MUST notify the system health agents (SHAs) about an orphan request. Notification to the system health agents (SHAs) SHOULD be done using the **INapSystemHealthAgentCallback::NotifyOrphanedSoHRequest** method which is part of SHA API.

The values of the MS-Quarantine-State packet (section 2.2.4.2) SHOULD be copied from the **MS-Quarantine-State** field of the appropriate **QecConnection** entry, as specified in sections 3.2.1 and 3.2.7.

The values of the MS-MachineName packet (section 2.2.4.5) SHOULD be set to identify the machine. The manner in which the machine name is obtained is implementation-specific.<28> The **machineName** subfield SHOULD contain the machine name and the **Length** subfield MUST be set to the number of bytes contained in **machineName** including the terminating null character.

The value of the **productType** subfield in the MS-Machine-Inventory-Ex packet (section 2.2.4.8) SHOULD be set to client, server, or domain-controller to identify the type of the operating system as described in section 2.2.4.8. The manner in which the product type is obtained is implementation-specific.<29>

The values for the subfields **Reserved**, **r**, and **vers** in MS-Packet-Info packet (section 2.2.4.3), as specified in section 2.2.4.3, MUST be set to zero, 1, and 1 respectively.

In all optional TLVs (section 2.2.3.5), the **M** bitfield SHOULD be set to 0 and the **R** bitfield MUST be set to 0.

The following optional TLVs are not directly set by the SoH client and MAY be added by SHAs:<30>

- Time-of-Last-Update as specified in section 2.2.3.5.4.
- Health-Class as specified in section 2.2.3.5.6.
- Software-Version as specified in section 2.2.3.5.7.
- Product-Name as specified in section 2.2.3.5.8.
- SOH Generation Time as specified in section 2.2.3.5.10.

3.2.5.2 Receiving SoHs

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

3.2.5.3 Sending SoHRs

An SoH client MUST NOT send an SoHR message.

3.2.5.4 Receiving SoHRs

The SoH client MUST NOT make any assumptions about the timing of when the SoHR is received and also MUST NOT rely on receiving an SoHR at all. All message timing issues are handled by the transport protocol that conveys the SoHs/SoHRs, which also handles the situation when the SoHR is missing. For more information, see [MS-HCEP] sections 3.1.5.2 and 3.1.8.

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 SoHAttributes value in the SoHR contains at least a Compliance-Result-Codes attribute or a **Failure Category** 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 value in the SSoHR attribute that does not exist in the **QecConnection** entry associated with this enforcement client, and therefore, does not correspond to an MS-CorrelationId value previously sent in an SoH message.

A compliant SoHR MUST be passed to the system health agents (SHA). The SoHR SHOULD be passed to the SHAs using the **INapSystemHealthAgentCallback::ProcessSoHResponse** method, which is part of the SHA API.

The SoH client SHOULD examine the MS-Quarantine-State (section 2.2.4.2) packet and if the packet state indicates that probation is in effect, the SoH client SHOULD initiate the Probation Timer (section 3.2.2.1) using the value specified in the **ProbTime** field of the packet. The SoH client MUST update the **MS-Quarantine-State** field in the appropriate **QecConnection** accordingly, as specified in section 3.2.1.

The SoH client SHOULD iterate over the SoHAttributes in the SoHR and extract the following from each:

- The health ID from the **System-Health-Packet** (section 2.2.3.1).
- The value from the **Failure Category** (section 2.2.3.4).
- The values from the **Compliance-Results-Codes** (section 2.2.3.2).

The SoH client then builds a **LastResult** ADM element (section 3.2.1) from these attributes and adds the **LastResult** to the **LastResults-cache** ADM element (section 3.2.1). If the **LastResults-cache** already contains an entry for the indicated health ID, that entry SHOULD first be removed from the **LastResults-cache** ADM element before adding the new entry.

3.2.6 Timer Events

When the Probation Timer (section 3.2.2.1) expires, an event is triggered. When this event is triggered, the SoH client SHOULD revalidate its health.

3.2.7 Other Local Events

The SoH client SHOULD be called using the **INapEnforcementClientBinding::NotifyConnectionStateDown** method, which is part of the NAP EC API, in order to be notified about dropped connections. Upon such event the SoH client MUST test the appropriate **QecConnection** entry in the **QecConnection-cache** ADM element to determine whether the **SoHRequest** field is set and the **SoHResponse** is empty, and therefore, a pending SoH request exists for the enforcement client. If such a request exists, The SoH client MUST

remove the SoH from the **MS-CorrelationId Cache** ADM element, as specified in section 3.2.1, and MUST notify the system health agents (SHAs) about an orphan request. Notification to the system health agents (SHAs) SHOULD be done using the **INapSystemHealthAgentCallback::NotifyOrphanedSoHRequest** method, which is part of SHA API.

The SoH client SHOULD be called using the **INapEnforcementClientBinding::CreateConnection** method, which is part of the NAP EC API, in order to be notified about new connections. Upon such event the SoH client MUST add a new **QecConnection** entry to the **QecConnection-cache** ADM element, as specified in section 3.2.1. Before any SoHR is received and no quarantine state is set – the MS-Quarantine-State (section 2.2.4.2) field SHOULD be reset as follows:

Subfield in MS-Quarantine-State	Initial value
qState	1
ExtState	0
f	0
ProbTime	0
urlLenInByes	0
url	empty

The **SoHResponse** field in the **QecConnection** entry MUST be reset as follows:

Subfield in SoHResponse	Initial value
size	0
data	NULL

To register and unregister SHAs, the SoH client SHOULD be called using the **INapSystemHealthAgentBinding::Initialize** and **INapSystemHealthAgentBinding::Uninitialize** methods respectively, which are part of the SHA API. Upon receiving the method call, the SoH client SHOULD update the entries in the **ShaTable** ADM element specified in section 3.2.1.

To pass the SoH message gathered from the SHAs (section 3.2.5) to the enforcement clients, the SoH client SHOULD be called by the enforcement clients using the **INapEnforcementClientBinding::GetSoHRequest** method that is part of the NAP EC API. This method is used by enforcement clients to retrieve the SoH that was previously set to the appropriate **QecConnection** entry in the **QecConnection-cache** ADM element and also to set the SoH response on the appropriate entry in **QecConnection-cache**. When this method is called, the SoH client MUST set the **SoHRequest** field of the appropriate entry in **QecConnection-cache**.

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

1. A new IP address is configured or assigned to the SoH client.
2. A new SoH transport protocol completes initialization.
3. A certificate expires.
4. A DHCP lease expires (for more information, see [RFC2131] section 4.4).

5. A DHCP lease renews (for more information, see [RFC2131] section 4.4).

6. 802.1x sessionauthentication is started on the client.

The SoH client uses the **INapEnforcementClientCallback::NotifySoHChange** which is part of the NAP EC API, in order to notify the enforcement clients about a change in the health state. Upon such events the enforcement clients SHOULD call the **INapEnforcementClientBinding::GetSoHRequest** which is part of the NAP EC API, and then call the **INapEnforcementClientConnection::GetSoHRequest** which is part of the NAP EC API, in order to retrieve the SoH message as an opaque buffer from the SoH client.

The enforcement client SHOULD call **INapEnforcementClientBinding::ProcessSoHResponse**, which is part of the NAP EC API, to initiate processing of the SoH response by the SoH client. Processing MUST be performed after the enforcement client has set the SoH response on the appropriate **QecConnection** entry using **INapEnforcementClientConnection::SetSoHResponse**. Upon such event, the SoH client SHOULD process the SoHR message, as specified in section 3.2.5.4.

3.3 Server-Specific Details

3.3.1 Abstract Data Model

The processing of SoH and SoHR messages on the server is performed by the Microsoft Windows® SHV as described in [MS-WSH] or via an implementation-specific SHV developed by a third party.<31>

System-Health-ID Mapping: The server must maintain a mapping that establishes the correspondence between the value of the SHV**System-Health-ID** packet and the SHV responsible for processing the SoHReportEntry. The mapping is a list of tuple, 32-bit unsigned integers representing the SHV**System-Health-ID** and a reference to the SHV. For example, the Windows SHV**System-Health-ID** value is 0x00013780 as specified in [MS-WSH] section 2.2.4, and a pointer to a function is provided as the reference. For initialization information about this ADM element, see section 3.3.3.

ShvTimeoutInMsec: A DWORD ([MS-DTYP] section 2.2.9) that specifies the time-out value for the call by the NAP validator to the SHV, in milliseconds. The default value is 2000.

NapSystemHealthValidationRequest: An ADM element that is used to pass information between the SoH server and the SHVs. The ADM element includes the following subfields.

Subfield in NapSystemHealthValidationRequest	Description
SoHRequest	An SoH request as specified in section 2.2.5.
SoHResponse	An SoHR response as specified in section 2.2.6.

3.3.2 Timers

None.

3.3.3 Initialization

The **System-Health-ID Mapping** ADM element is updated by SHVs when each SHV registers with the server. During the registration process for each SHV, this ADM element sets the SHVSystem-Health-ID value and a reference to the SHV APIs.<32>The SHV**System-Health-ID** is a static

number that never changes for a given SHV. If multiple implementations of SHVs used the same **System-Health-ID** value and are registering to the same server, the last SHV registered will be used.

For more information about the **System-Health-ID Mapping** ADM element, see section 3.3.1.

3.3.4 Higher-Layer Triggered Events

3.3.4.1 GetMachineInventory

The SoH server MAY be called with the following abstract interface to retrieve machine information from the client that sent the SoH message:

```
HRESULT GetMachineInventory(    [in] SoH message,    [out] STRING machineName,    [out] DWORD
osVersionMajor,    [out] DWORD osVersionMinor,    [out] DWORD osVersionBuild,    [out] WORD
spVersionMajor,    [out] WORD spVersionMinor,    [out] WORD procArch,    [out] BYTE
ProductType );
```

The GetMachineInventory abstract interface receives an SoH message (section 2.2.5). When the interface is called, the SoH server parses the SoH message, extracts the packets, and uses the various subfields as described in this section.

The SoH server extracts the MS-MachineName packet (section 2.2.4.5) and applies the value of the **machineName** subfield to the value of the interface parameter of the same name. The value of the **Length** subfield in the packet is used to determine the length of the **machineName** subfield.

The SoH server extracts the MS-Machine-Inventory packet (section 2.2.4.1) and applies the values of the **osVersionMajor**, **osVersionMinor**, **osVersionBuild**, **spVersionMajor**, **spVersionMinor** and **procArch** subfields to the values of the interface parameters of the same names, respectively.

The SoH server extracts the MS-Machine-Inventory-Ex packet (section 2.2.4.8) and applies the value of the **ProductType** subfield to the value of the interface parameter of the same name.

Upon success, the abstract interface SHOULD return ERROR_SUCCESS (0x00000000). When any failure to find information occurs, the interface returns any value other than ERROR_SUCCESS (0x00000000).

3.3.4.2 GetQuarantineState

The SoH server MAY be called with the following abstract interface to obtain the quarantine state:

```
HRESULT GetQuarantineState(    [in] SoH message,    [out] DWORD ExtState,    [out] BOOL f,
[out] DWORD qState,    [out] LONGLONG ProbTime,    [out] String url    [out] DWORD
urlLenInBytes)
```

The **GetQuarantineState** abstract interface receives an SoH message (section 2.2.5). When the interface is called, the SoH server parses the SoH message and extracts the MS-Quarantine-State packet (section 2.2.4.2). The SoH server applies the values of the packet subfields **ExtState**, **f**, **qState**, **ProbTime**, **urlLenInBytes**, and **url** to the values of the interface parameters that have the same names, respectively. Upon success, the interface SHOULD return ERROR_SUCCESS (0x00000000). When a failure to find any information occurs, the interface returns any value other than (0x00000000).

3.3.4.3 SetSoH

The **SetSoH** abstract interface is called to provide an SoH message to be processed by the SoH server:

```
HRESULT SetSoH([in] SoH message);
```

The **SetSoH** abstract interface receives an SoH message (section 2.2.5). When the interface is called, the SoH server processes the SoH message as described in section 3.3.5 and creates an SoHR.

3.3.5 Processing Events and Sequencing Rules

An SoHR contains data (contained in the SoHRReportEntry values) that reports the results of an evaluation of the client's current status. The value of this data is typically provided by other servers that provide security services, such as an antivirus server or a security update server. The Statement of Health for NAP Protocol itself simply provides the mechanism for the client status to be supplied so that the health of the client can be evaluated.

A health policy server MUST do the following:

- Receive and process SoHs.<33>
- Create SoHRs.<34>
- Send SoHRs.<35>

3.3.5.1 Sending SoHs

The health policy server MUST NOT send an SoH.

3.3.5.2 Receiving SoHs

1. The SoH server MUST ensure that the received SoH is properly formed by validating the length of each attribute. If the received SoH is malformed, then the SoH server MUST discard this SoH request.

   The SoH server MUST verify that the **r** and **vers** fields in MS-Packet-Info packet (section 2.2.4.3) (as specified in section 2.2.4.3) have the value of 1. If the value of the **r** and **vers** fields is not 1, the server MUST discard the SoH message.

2. The server MUST create the SoHR. The SSoHR entry MUST be stored in the SoHR (as specified in 2.2.9). The value of the MS-CorrelationId attribute MUST be set to the value of the same attribute in the SoH. The value of the MS-Quarantine-State attribute will be set as specified in the following processing rules.

3. The SoH server attempts to match every SoHReportEntry in the SoH to the SHV using the value of the System-Health-ID Packet and **System-Health-ID Mapping**.

   - If the match is not found, the server MUST ignore the SoHReportEntry.
   - If the match is found, the server then requests the SHV to evaluate declarations of client health stored in the SoHReportEntry. To do that, the SoH server creates a **NapSystemHealthValidationRequest** ADM element and sets the value of its **SoHRequest** subfield to the appropriate request message and the value of its **SoHResponse** subfield to

empty. After the ADM element has been created, the SHV SHOULD be called using the **INapSystemHealthValidator::Validate** method, which is part of the **SHV API**. The following parameters SHOULD be passed to this call:

- The newly created **NapSystemHealthValidationRequest** ADM element. The SHV SHOULD update the **SoHResponse** subfield of this ADM element with an SoHRReportEntry (section 2.2.6.3) containing the health evaluation results.<36>
- The **ShvTimeoutInMsec** ADM element as a time-out hint.
- A pointer to the **INapServerCallback::OnComplete** callback method described in section 3.3.7.

If the call returns with a value of S_OK, the SoH server MUST continue to the subsequent processing step (3c) pertaining to when the SHV is successfully returned. If the call returns a value of E_PENDING, the SoH server MUST trigger the ShvProcessingTimer timer described in section 3.3.6 and set the timer to expire after the duration specified in the **ShvTimeoutInMsec** ADM element. The SoH server MUST halt processing of the SHV until either the timer expires or the callback is called. When any other value is returned by the call, the SoH server SHOULD continue to the final processing step in this series (3d) pertaining to when the SHV fails.

- If the SHV successfully returned its health evaluation results, then the SoH server MUST store these results in the corresponding SoHRReportEntry in the SoHR. This SoHRReportEntry MUST include a valid Compliance-Result-Codes attribute in its **SoHRAttributeSet**.
- If the SHV fails the requested health state evaluation, or any error occurs during the health evaluation process, then the SoH server MUST create a corresponding SoHRReportEntry for this SHV. This SoHRReportEntry MUST include a **Failure Category** attribute in the **SoHRAttributeSet**.

4. After all the SoHReportEntries of the SoH are processed, the SoH server MUST evaluate the compliance of the SoH against policy, based on the results returned by the SHVs. The server MUST then set the value of the **ExtState**, **f**, **qState**, **ProbTime**, **urlLenInBytes**, and **url** subfields of the MS-Quarantine-State attribute in the SSoHR based on the result of policy evaluation.
5. Finally, the SoH server MUST send the SoHR back to its corresponding SoH client using a **PEP channel**, as specified in section 3.3.5.3.

3.3.5.3 Sending SoHRs

The SoH server MUST ensure that every SSoHR that it sends is properly formed by validating the length of each attribute. The SoH server MUST include at least a valid Compliance-Result-Codes attribute or a **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 **correlationId** subfield in the **MS-CorrelationId** attribute in the SoHR with the value of the **correlationId** subfield in the **MS-CorrelationId** attribute in the SoH to which this SoHR is a response.

The values of the MS-MachineName packet (section 2.2.4.5) SHOULD be set to identify the machine. The manner in which the machine name is obtained is implementation-specific.<37> The **machineName** subfield SHOULD contain the machine name and the **Length** subfield MUST be set to the number of bytes contained in **machineName** including the terminating null character.

The values for subfields **Reserved**, **r** and **vers** in MS-Packet-Info packet (section 2.2.4.3), as specified in section 2.2.4.3, MUST be set to zero, zero, and 1 respectively.

The following optional TLVs are not set by the SoH server and MAY be added by SHVs: <38>

- IPv4 Fix-up Servers as specified in section 2.2.3.5.3.
- IPv6 Fix-up Servers as specified in section 2.2.3.5.12.
- Health Class Status as specified in section 2.2.3.5.9.
- Error Codes as specified in section 2.2.3.5.11.

The actual sending of the SoHR is handled by an RNAP server or an EAP-supporting RADIUS server, and is described in [MS-NAPSO] section 10. The SoH server uses the **SetSoHR** abstract interface described in [MS-NAPSO] section 10.3.2 to send the SoHR to the task. The interface parameters are set as follows:

- **SoHR** is set to the SoHR message.
- **quarantineState** is set to the value of **qState**.
- **extendedQuarantineState** is set to the value of **ExtState**.

3.3.5.4 Receiving SoHRs

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

3.3.6 Timer Events

ShvProcessingTimer: When this timer expires, the SoH server SHOULD fail the SHV validation and continue processing the SHV as described in section 3.3.5.2, step 3d, pertaining to when the SHV fails. If the **INapServerCallback::OnComplete** callback (section 3.3.7) is received from the SHV, the callback MUST be ignored.

3.3.7 Other Local Events

SHVs MAY call the SoH server with the **INapServerCallback::OnComplete** callback, which is part of the SHV API. Upon such event, the SoH server SHOULD resume processing of the SHV as specified in section 3.3.5.2, steps 3c and 3d.

The following event MUST result in an SoHR being sent by the SoH server:

- A RADIUS packet containing a Statement of Health for NAP Protocol attribute is received.

4 Protocol Examples

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 in the following.

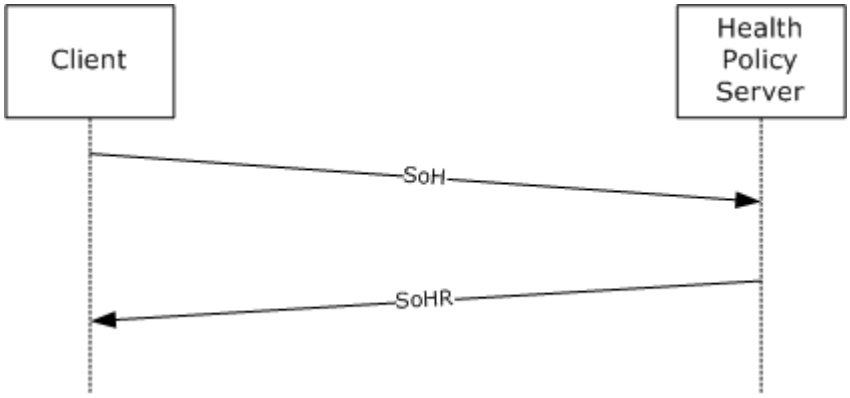


Figure 4: Client SoH request and health policy server response

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. For specifics about the flow, see section 3.

When the SoH is being sent, it is likely that the client is requesting access to some service and is being required to prove its 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 (that is, the SoHR) to the client by means of the original receiver of the SoH.

Generally, the receipt of an SoHR by the client, allows access to the service 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 receive remedy. After the client is restored to good health, the client can initiate the protocol again.

5 Security

The following sections specify security considerations for implementers of the Statement of Health for NAP Protocol.

5.1 Security Considerations for Implementers

Security for health messages should be provided by the transport layer protocol. The transport protocol should guard against replay and tampering, and provide privacy of health messages. Health messages should not be transmitted unencrypted even if the transport protocol itself does not encrypt the communication. In such cases, the individual messages should be encrypted, signed, and time-stamped to ensure their integrity and confidentiality, and to prevent usage of an SoH after it no longer represents the state of the computer.

Version 2 of the protocol can support enhancements in a later version. Even though version 2 does not offer additional security over version 1 of the Statement of Health for NAP Protocol, implementers should use version 2 for future compatibility and security enhancements.

Implementers can use the protocol in applications where the messages are carried in a transport protocol that does not provide security (for example, DHCP). In such cases, it is important for the implementation to guide their users (the network administrators) to have infrastructure measures in place that accommodate and compensate for such usage.

The following risks are mitigated when the transport protocol provides security for health messages:

- Passive Observation of Confidential Information. Health messages contain information that may not only disclose personally identifying information of a user but also disclose a current security issue in the system. That is the nature of the message and the service that it provides. For this reason, it is important to preserve the confidentiality of these messages. It should not be possible for a **man in the middle attack (MITM)** to successfully view the contents of health messages as they are transmitted.
- Spoofing. A health message (an SoH) is a token that potentially causes a client to be authorized to access a protected resource. It should not be possible for anyone other than the system that created the SoH to use the health message. 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, **EAP server**, or a client.
- Active Tampering. For the risks discussed previously, it should not be possible for an attacker to modify the SoHR undetected. Similarly, tampering of the SoH can cause a client to be granted access when it is impermissible, or cause a client to be denied access when it is permitted. The security provided by the transport mechanism should prevent tampering with these messages.
- Replaying. A message that causes a client to be granted access can potentially be retransmitted by another client to incorrectly give the client 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 effective measures currently exist. These are documented here to make implementers aware of them.

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

There is always the potential that the host itself is compromised by some kernel mode malicious software (malware). In such cases, the SoHs and SoHAttributes produced by the client cannot be trusted. However, without broad deployment of trusted hardware, currently there are no effective solutions for this.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Windows Vista® operating system
- Windows Server® 2008 operating system
- Windows® 7 operating system
- Windows Server® 2008 R2 operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 1.5: Client and server prerequisites for NAP: The Windows implementation of the SoH Client sends version 2 messages by default, but can be configured to send version 1 messages. The Windows implementation of the SoH server accepts versions 1 and 2 messages.

<2> Section 1.5: Client prerequisites for NAP: The client has to be configured to recognize the access methods that are required to enable the NAP solution.

<3> Section 2.2.3.1: The Statement of Health for NAP Protocol IDs that are used. The following IDs are used in the Windows implementation of the Statement of Health for NAP Protocol: 0x00013700, 0x00013701, 0x00013702, 0x00013703, 0x00013704, 0x00013705, 0x00013706, 0x00013707, 0x00013780, and 0x00013781.

<4> Section 2.2.3.3: An IANA-assigned SMI vendor ID is strongly recommended. Other types of IDs are also acceptable as long as they can uniquely identify the vendor who specifies the data in the Data field. For example, in the Windows implementation, Windows Security Health Agent (WSHA) uses the following vendor ID format: IANA-assigned SMI vendor code plus component ID (same as the Health ID format specified in section 2.2.3.1).

<5> Section 2.2.3.5: Third-parties are allowed to use these optional TLVs in their implementations to construct their own SoH or SoHR messages. The Windows implementation does not check the **Value** fields of the optional TLVs in third-party implementations. However, the Windows implementation does perform consistency checks on the length of the attributes. For example, if Optional TLV 5 (the Time-of-Last-Update TLV) is used in a third-party SoH or SoHR message, the Windows implementation requires that the length of its **Value** field be 8, but does not require specific contents for the **Value** field.

<6> Section 2.2.3.5.4: Third-party SHA/SHV implementations and MSSHA/SHV may use this optional TLV in their SoH/SoHR messages. The Windows implementation of the SoH protocol does not care when or how many of these optional TLVs are used in the SoH/SoHR messages. The detailed implementation by MSSHA/SHV can be found in WSHA SoH and WSHV SoHR[MS-WSH].

<7> Section 2.2.3.5.5: Third-party SHA/SHV implementations and MSSHA/SHV may use this optional TLV in their SoH/SoHR messages. The Windows implementation of the SoH protocol does

not care when or how many of these optional TLVs are used in the SoH/SoHR messages. The detailed implementation by MSSHA/SHV can be found in WSHA SoH and WSHV SoHR[MS-WSH].

<8> Section 2.2.3.5.6: Third-party SHA/SHV implementations and MSSHA/SHV may use this optional TLV in their SoH/SoHR messages. The Windows implementation of the SoH protocol does not care when or how many of these optional TLVs are used in the SoH/SoHR messages. The detailed implementation by MSSHA/SHV can be found in WSHA SoH and WSHV SoHR[MS-WSH].

<9> Section 2.2.3.5.7: Third-party SHA/SHV implementations and MSSHA/SHV may use this optional TLV in their SoH/SoHR messages. The Windows implementation of the SoH protocol does not care when or how many of these optional TLVs are used in the SoH/SoHR messages. The detailed implementation by MSSHA/SHV can be found in WSHA SoH and WSHV SoHR[MS-WSH].

<10> Section 2.2.3.5.8: Third-party SHA/SHV implementations and MSSHA/SHV may use this optional TLV in their SoH/SoHR messages. The Windows implementation of the SoH protocol does not care when or how many of these optional TLVs are used in the SoH/SoHR messages. The detailed implementation by MSSHA/SHV can be found in WSHA SoH and WSHV SoHR[MS-WSH].

<11> Section 2.2.3.5.9: Third-party SHA/SHV implementations and MSSHA/SHV may use this optional TLV in their SoH/SoHR messages. The Windows implementation of the SoH protocol does not care when or how many of these optional TLVs are used in the SoH/SoHR messages. The detailed implementation by MSSHA/SHV can be found in WSHA SoH and WSHV SoHR[MS-WSH].

<12> Section 2.2.3.5.10: Third-party SHA/SHV implementations and MSSHA/SHV may use this optional TLV in their SoH/SoHR messages. The Windows implementation of the SoH protocol does not care when or how many of these optional TLVs are used in the SoH/SoHR messages. The detailed implementation by MSSHA/SHV can be found in WSHA SoH and WSHV SoHR[MS-WSH].

<13> Section 2.2.3.5.11: Third-party SHA/SHV implementations and MSSHA/SHV may use this optional TLV in their SoH/SoHR messages. The Windows implementation of the SoH protocol does not care when or how many of these optional TLVs are used in the SoH/SoHR messages. The detailed implementation by MSSHA/SHV can be found in WSHA SoH and WSHV SoHR[MS-WSH].

<14> Section 2.2.4.1: The MS-Machine-Inventory (section 2.2.4.1) fields that are populated. Windows populates the **osVersionMajor**, **osVersionMinor**, **spVersionMajor**, **spVersionMinor**, and **procArch** fields based on the returns from GetVersionInfoEx and its OSVERSIONINFOEX structure (for more information, see [MSDN-OSVERSIONINFOEX]).

<15> Section 2.2.4.1: The health policy server does not send this TLV in the SSoHR. The client does not require nor read this TLV when the SSoHR is parsed.

<16> Section 2.2.4.2: Client resubmission at end-of-probation time. At the end of the probation time, the client resubmits itself for validation. The resubmission process depends on how the client is deployed. For example, in the case of an HRA deployment, the client certificate expires at the end of probation time and the client enrolls for a new certificate by posting a new HCEP message at that time. When the client resubmits, Windows evaluates its compliance and grants access according to policy.

<17> Section 2.2.4.4: The MS-SystemGenerated-Ids attribute for internal error indication: The Windows client includes the MS-SystemGenerated-Ids attribute when an internal error occurs while the client is attempting to gather state information about the host. The Windows server does not include this attribute.

<18> Section 2.2.4.5: The MS-MachineName (section 2.2.4.5) attribute used to submit the name: The **machineName** attribute is the fully qualified domain name (FQDN) of the computer if it is joined to a Windows domain; otherwise, the computer name is used.

<19> Section 2.2.4.6: The MS-CorrelationId (section 2.2.4.6) for diagnostic purposes: The **correlationId** is a concatenation of the 16-byte connection ID and the **FILETIME** at which the attribute was generated.

<20> Section 2.2.4.7: The SoH Evaluation API: Implemented in Windows Server 2008 and Windows Server 2008 R2. The health policy server in Windows includes an API that enables plug-ins that perform an SoH evaluation to register. The WindowsSoH server only includes the MS-Installed-Shvs (section 2.2.4.7) attribute if such plug-ins are registered.

<21> Section 2.2.4.7: The Windows client always sends all available health messages.

<22> Section 2.2.4.8: The health policy server does not send this TLV in the SSoHR. The client does not require nor read this TLV when the SSoHR is parsed.

<23> Section 2.2.8: MS-SystemGenerated-Ids is a list of component IDs that is unable to provide an SoHReportEntry. The Windows implementation of the Statement of Health for NAP Protocol includes the MS-SystemGenerated-Ids with a list of component IDs that are unable to provide an SoHReportEntry at the time that the SSoH is generated. The Windows implementation always sends the optional MS-Machine-Inventory-Ex attribute in the SSoH.

<24> Section 2.2.9: MS-Installed-Shvs has a list of SHVs that are installed. Implemented in Windows Server 2008 and Windows Server 2008 R2. The Windows implementation of the health policy server includes MS-Installed-Shvs with a list of SHVs that are installed on the server to perform health validation of SoHReportEntry.

<25> Section 3.2.5: The SoH client includes an API to allow plug-ins to report client state. The WindowsSoH client, also called the NAP agent, includes an API that allows plug-ins that report client state to register with the system. These plug-ins are called SHAs. Examples of SHAs include antivirus clients and security update clients. One specific SHA, WSHA, as specified in [MS-WSH], is included in Windows. Each SHA produces an SoHReportEntry for the state that it reports. The NAP agent forms an SoH by appending the collection of the SoHReportEntry from the SHAs to a valid SSoH. The NAP agent creates a new SoH whenever it receives a new SoHReportEntry from an SHA, or whenever a quarantine enforcement client requests one.

<26> Section 3.2.5: When the NAP agent receives the SoHR, it validates the message and notifies the user if the MS-Quarantine-State value indicates that the user's network connectivity is restricted. The NAP agent delivers the SoHReportEntry values in the SoHR to the appropriate SHA for processing according to the System-Health-ID attribute in the SoHRReportEntry.

<27> Section 3.2.5.1: MS-CorrelationId creates a GUID corresponding to the network connection. The NAP agent forms its MS-CorrelationId by filling in the first 16 bytes with the value of a GUID corresponding to the network connection over which the SoH is being transported. These 16 bytes are the same each time the SoH is transported over that connection. The last 8 bytes of MS-CorrelationId is a 64-bit unsigned integer representing the number of 100-nanosecond intervals between January 1, 1601 (UTC) and when the SoH was delivered for transport.

<28> Section 3.2.5.1: In Windows, the machine name can be obtained using the **GetComputerNameEx** Win32 API with a value of ComputerNamePhysicalDnsFullyQualified for the *NameType* parameter when the computer is joined to the domain; otherwise, with a value of ComputerNamePhysicalNetBIOS.

<29> Section 3.2.5.1: In Windows, the product type can be obtained using the **GetVersionEx** Win32 API and by querying the **wProductType** field of the retrieved **OSVERSIONINFOEX** structure.

<30> Section 3.2.5.1: Third-party SHA/SHV implementations can use these optional TLVs in their SoH/SoHR messages. The Windows implementation of the SoH protocol does not process optional TLVs used in SoH/SoHR messages.

<31> Section 3.3.1: Windows Server 2008 and Windows Server 2008 R2 provide an API that allows plug-ins to validate the SoHReportEntry messages sent inside an SoH by SoH clients. This API is a COM interface INapSystemHealthValidator [MSDN-INapSysHV]. SHVs are components that implement this interface. SHVs are always instantiated and used on the same computer as NPS. The **System-Health-Id Mapping** is a list that associates the System-Health-ID value with a pointer to the instance of **INapSystemHealthValidator**.

<32> Section 3.3.3: Windows Server 2008 and Windows Server 2008 R2 provide an API that allows plug-ins to validate the SoHReportEntry messages sent inside an SoH by SoH clients. This API is a COM interface INapSystemHealthValidator [MSDN-INapSysHV]. SHVs are components that implement this interface. SHVs are always instantiated and used on the same computer as NPS. The **System-Health-Id Mapping** is a list that associates the System-Health-ID value with a pointer to the instance of **INapSystemHealthValidator**.

<33> Section 3.3.5: The Windowshealth policy server is part of the NPS. This is implemented in Windows Server 2008 and Windows Server 2008 R2. NPS is the WindowsRADIUS server. It also includes an implementation of SHV, WSHV, as specified in [MS-WSH]. The health policy server validates the format of a received SoH and delivers the SoHReportEntry values to the appropriate SHV for evaluation based on the value of the **SystemHealthId** attribute in the SoHReportEntry. The health policy server then waits for the SHVs to complete their evaluation of the SoHReportEntry values.

<34> Section 3.3.5: SHVs evaluate the SoHReportEntry values by delivering SoHReportEntry values to the health policy server. This is implemented in Windows Server 2008 and Windows Server 2008 R2. The SHVs complete their evaluation of the SoHReportEntry values by delivering SoHReportEntry values to the health policy server. The health policy server forms an SSoHR and populates the **Quarantine-State** attribute therein, according to policy by taking the Compliance-Result-Codes and **Failure Category** attributes as input. The health policy server forms a valid SoHR by using the resulting SSoHR and the collection of SoHReportEntry messages received from the SHVs.

<35> Section 3.3.5: The health policy server delivers the SoH message to the NPS to be processed against policy, and sends the message via RADIUS to the SoH client. This is implemented in Windows Server 2008 and Windows Server 2008 R2. The health policy server delivers the SoHR to the NPS, which processes it against policy and sends it as a vendor-specific attribute (VSA) in RADIUS to a RADIUS client, as specified in [MS-RNAP]. The RADIUS client then delivers the SoHR to the SoH client over the transport mechanism that was originally used to send the SoH. An example of this process is specified in [MS-HCEP] section 1.3.

<36> Section 3.3.5.2: The **INapSoHConstructor** interface that is part of the SHA API and SHV API is a helper class for constructing SoH and SoHR messages (**INapSoHConstructor::Initialize**), appending attributes (**INapSoHConstructor::AppendAttribute**), and retrieving the built message (**INapSoHConstructor::GetSoH**). These helper functions are optional and the SHA and SHV implementer is not required to use them, as long as the created SoH/SoHR message complies with the definitions in this specification.

<37> Section 3.3.5.3: In Windows, the machine name can be obtained using the **GetComputerNameEx** Win32 API with a value of ComputerNamePhysicalDnsFullyQualified for the *NameType* parameter when the computer is joined to the domain; otherwise, with a value of ComputerNamePhysicalNetBIOS.

<38> Section 3.3.5.3: Third-party SHA/SHV implementations can use these optional TLVs in their SoH/SoHR messages. The Windows implementation of the SoH protocol does not process the optional TLVs used in SoH/SoHR messages.

7 Change Tracking

This section identifies changes that were made to the [MS-SOH] protocol document between the May 2011 and June 2011 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the language and formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.

- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated.**

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.2 References	Added explanatory statement regarding the removal of the publishing year from Microsoft Open Specification document references.	N	Content updated.

8 Index