[MS-RSVD]:
Remote Shared Virtual Disk Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
08/08/2013	1.0	New	Released new document.
11/14/2013	2.0	Major	Significantly changed the technical content.

Contents

1 Introduction

The Remote Shared Virtual Disk (RSVD) Protocol is a block-based protocol that is used to access the virtual disk file in the network over SMB3.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. Sections 1.5 and 1.9 are also normative but cannot contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

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

GUID
SCSI
Unicode

The following terms are specific to this document:

SCSI command descriptor block: A block of information that describes the SCSI command.

sense data: Data describing command-completed information that a device server delivers to an application client in the same structure as the status or as parameter data in response to an SCSI command.

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 Specifications documentation 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.

A reference marked "(Archived)" means that the reference document was either retired and is no longer being maintained or was replaced with a new document that provides current implementation details. We archive our documents online [Windows Protocol].

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-SMB2] Microsoft Corporation, "Server Message Block (SMB) Protocol Versions 2 and 3".

[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

[UNICODE] The Unicode Consortium, "Unicode Home Page", 2006, http://www.unicode.org/

1.2.2 Informative References

[MS-FSCC] Microsoft Corporation, "File System Control Codes".

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

[SPC-3] International Committee on Information Technology Standards, "SCSI Primary Commands - 3 (SPC-3)", Project T10/1416-D, May 2005, http://www.t10.org/cgi-bin/ac.pl?t=f&f=/spc3r23.pdf

1.3 Overview

The Remote Shared Virtual Disk Protocol enables a client application to access virtual disk files in a shared fashion on a remote server.

The RSVD protocol supports the following features:

- Allowing a client to open a shared virtual disk on a remote share.
- Reading, writing, or closing shared virtual disk files on the target server.
- Forwarding of raw SCSI commands and receipt of their results.

1.4 Relationship to Other Protocols

This protocol depends on Server Message Block (SMB) Protocol version 3 for its transport, as specified in [MS-SMB2].

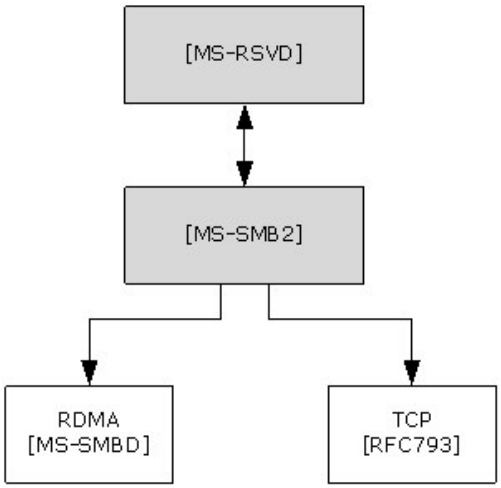


Figure 1: Relationship to other protocols

1.5 Prerequisites/Preconditions

The RSVD Protocol has the following preconditions:

- An SMB client has established a connection to an SMB server. This has to be done before a client can issue Remote Shared Virtual Disk Protocol commands.
- The SMB client and server support the SVHDX_OPEN_DEVICE_CONTEXT create context, as specified in section 2.2.5.2.

1.6 Applicability Statement

The Remote Shared Virtual Disk Protocol is applicable for all scenarios that access a shared virtual disk file between client and server.

1.7 Versioning and Capability Negotiation

The Remote Shared Virtual Disk Protocol has a single version.<1>

Version	Value
RSVD Protocol version 1	0x00000001

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

This protocol shares the standards assignments of Server Message Block Protocol versions 2 and 3, as specified in [MS-SMB2] section 1.9.

2 Messages

2.1 Transport

The following sections specify how RSVD Protocol messages are encapsulated on the wire and common protocol data types.

All RSVD Protocol messages begin with a fixed-length RSVD header that is described in section 2.2.5.1. The RSVD tunnel header contains an OperationCode field indicating the operation code that is requested by the RSVD client or responded to by the RSVD server.

Unless otherwise specified, multiple-byte fields (16-bit, 32-bit, and 64-bit fields) in the RSVD Protocol message MUST be transmitted in little-endian order (least-significant byte first).

Unless otherwise indicated, numeric fields are integers of the specified byte length.

Unless otherwise specified, all textual strings MUST be in **Unicode** version 5.0 format, as specified in [UNICODE], using the 16-bit Unicode Transformation Format (UTF-16) form of the encoding. Textual strings with separate fields identifying the length of the string MUST NOT be null-terminated unless otherwise specified.

Unless otherwise noted, fields marked as "Reserved" MUST be set to 0 when being sent and MUST be ignored when received. These fields are reserved for future protocol expansion and MUST NOT be used for implementation-specific functionality.

The RSVD Protocol uses the SMB 3.02 dialect in SMB Protocol version 3 as its transport. For more information, see [MS-SMB2] section 2.1.

2.2 Message Syntax

2.2.1 Constants

Constant name	Value
RSVD_ECP_CONTEXT_VERSION_1	0x00000001
RSVD_CDB_GENERIC_LENGTH	0x10
RSVD_SCSI_SENSE_BUFFER_SIZE	0x14
RSVD_MAXIMUM_NAME_LENGTH	0x3F
FSCTL_SVHDX_SYNC_TUNNEL_REQUEST	0x00090304
SMB_CCF_APP_INSTANCE_EA_NAME	"ClusteredApplicationInstance"
APP_TRAFFIC_TYPE_EA_NAME	"ApplicationTrafficType"

2.2.2 Operation Codes

The following is a list of all control codes used in shared virtual disk operations.

Name	Value
RSVD_TUNNEL_GET_FILE_INFO_OPERATION	0x02001001

Name	Value
RSVD_TUNNEL_SCSI_OPERATION	0x02001002
RSVD_TUNNEL_CHECK_CONNECTION_STATUS_OPERATION	0x02001003
RSVD_TUNNEL_SRB_STATUS_OPERATION	0x02001004
RSVD_TUNNEL_GET_DISK_INFO_OPERATION	0x02001005
RSVD_TUNNEL_VALIDATE_DISK_OPERATION	0x02001006

2.2.3 Error Code

The following is a list of possible RSVD error codes that can be returned by the server.

Name	Value
STATUS_SVHDX_ERROR_NOT_AVAILABLE	0xC05CFF00
STATUS_SVHDX_UNIT_ATTENTION_AVAILABLE	0xC05CFF01
STATUS_SVHDX_UNIT_ATTENTION_CAPACITY_DATA_CHANGED	0xC05CFF02
STATUS_SVHDX_UNIT_ATTENTION_RESERVATIONS_PREEMPTED	0xC05CFF03
STATUS_SVHDX_UNIT_ATTENTION_RESERVATIONS_RELEASED	0xC05CFF04
STATUS_SVHDX_UNIT_ATTENTION_REGISTRATIONS_PREEMPTED	0xC05CFF05
STATUS_SVHDX_UNIT_ATTENTION_OPERATING_DEFINITION_CHANGED	0xC05CFF06
STATUS_SVHDX_RESERVATION_CONFLICT	0xC05CFF07
STATUS_SVHDX_WRONG_FILE_TYPE	0xC05CFF08
STATUS_SVHDX_VERSION_MISMATCH	0xC05CFF09

2.2.4 SCSI command descriptor block flags

The following is a list of possible **SCSI command descriptor block** flags.

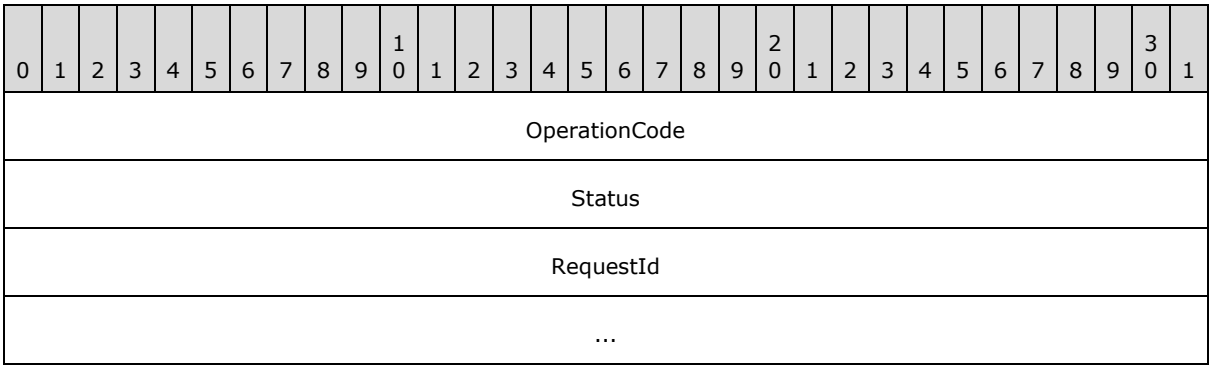
Flag name	Value
SRB_FLAGS_QUEUE_ACTION_ENABLE	0x00000002
SRB_FLAGS_DISABLE_DISCONNECT	0x00000004
SRB_FLAGS_DISABLE_SYNCH_TRANSFER	0x00000008
SRB_FLAGS_BYPASS_FROZEN_QUEUE	0x00000010
SRB_FLAGS_DISABLE_AUTOSENSE	0x00000020
SRB_FLAGS_DATA_IN	0x00000040
SRB_FLAGS_DATA_OUT	0x00000080

Flag name	Value
SRB_FLAGS_NO_DATA_TRANSFER	0x00000000
SRB_FLAGS_UNSPECIFIED_DIRECTION	SRB_FLAGS_DATA_IN | SRB_FLAGS_DATA_OUT
SRB_FLAGS_NO_QUEUE_FREEZE	0x00000100
SRB_FLAGS_ADAPTER_CACHE_ENABLE	0x00000200
SRB_FLAGS_FREE_SENSE_BUFFER	0x00000400
SRB_FLAGS_D3_PROCESSING	0x00000800
SRB_FLAGS_IS_ACTIVE	0x00001000
SRB_FLAGS_ALLOCATED_FROM_ZONE	0x00002000
SRB_FLAGS_SGLIST_FROM_POOL	0x00004000
SRB_FLAGS_BYPASS_LOCKED_QUEUE	0x00008000
SRB_FLAGS_NO_KEEP_AWAKE	0x00010000
SRB_FLAGS_PORT_DRIVER_ALLOCSENSE	0x00020000
SRB_FLAGS_PORT_DRIVER_SENSEHASPORT	0x00040000
SRB_FLAGS_DONT_START_NEXT_PACKET	0x00080000

2.2.5 Structures

2.2.5.1 SVHDX_TUNNEL_OPERATION_HEADER

The RSVD tunnel header is the header of RSVD Protocol operations specified in section 2.2.2.



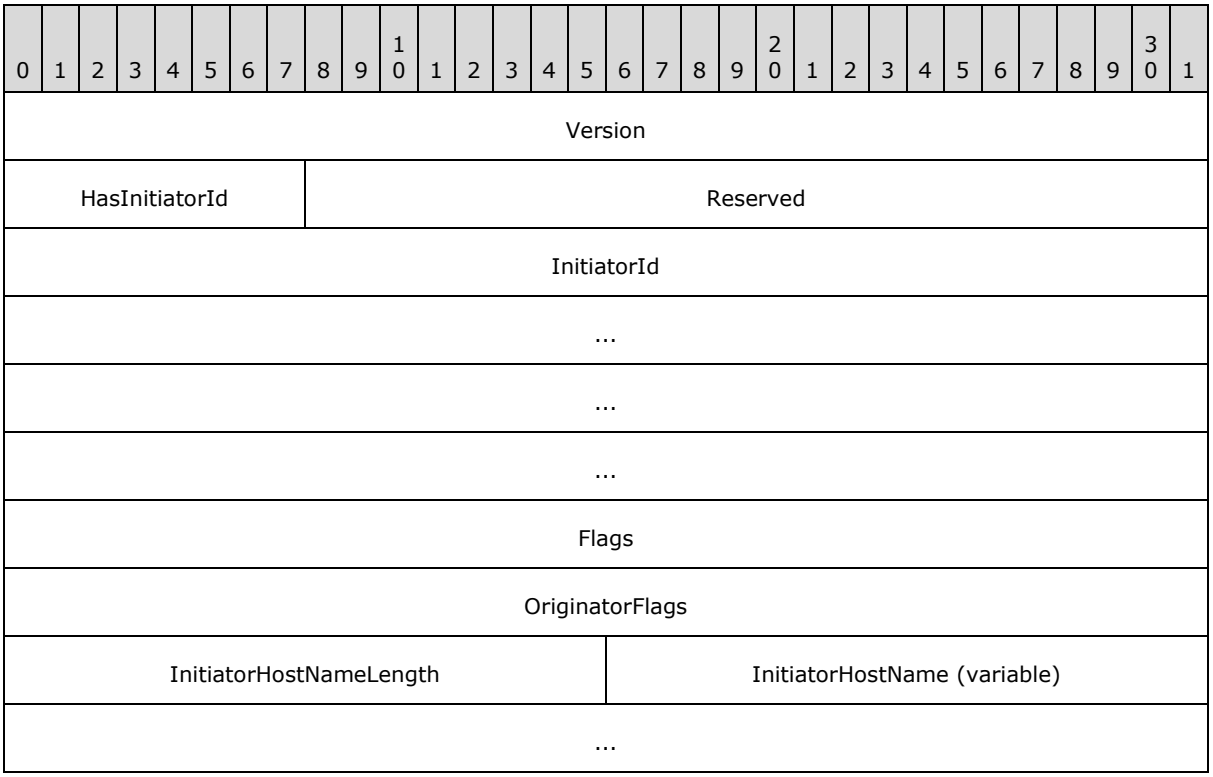
OperationCode (4 bytes): The command code of this packet. This field MUST contain one of values specified in section 2.2.2.

Status (4 bytes): The client SHOULD set this field to zero, and the server MUST ignore it on receipt.

RequestId (8 bytes): A value that uniquely identifies an operation request and response for all requests sent by this client.

2.2.5.2 SVHDX_OPEN_DEVICE_CONTEXT

The SVHDX_OPEN_DEVICE_CONTEXT packet is sent by the client to open the shared virtual disk.



Version (4 bytes): The version of the create context. It MUST be set to the highest supported version of the protocol, as specified in section 1.7.

HasInitiatorId (1 bytes): A Boolean value, where zero represents FALSE and nonzero represents TRUE.

Reserved (3 bytes): This field MUST be set to zero when sent and MUST be ignored on receipt.

InitiatorId (16 bytes): A **GUID** that optionally identifies the initiator of the open request.

Flags (4 bytes): Reserved. The client SHOULD set this field to 0x00000000, and the server MUST ignore it on receipt.

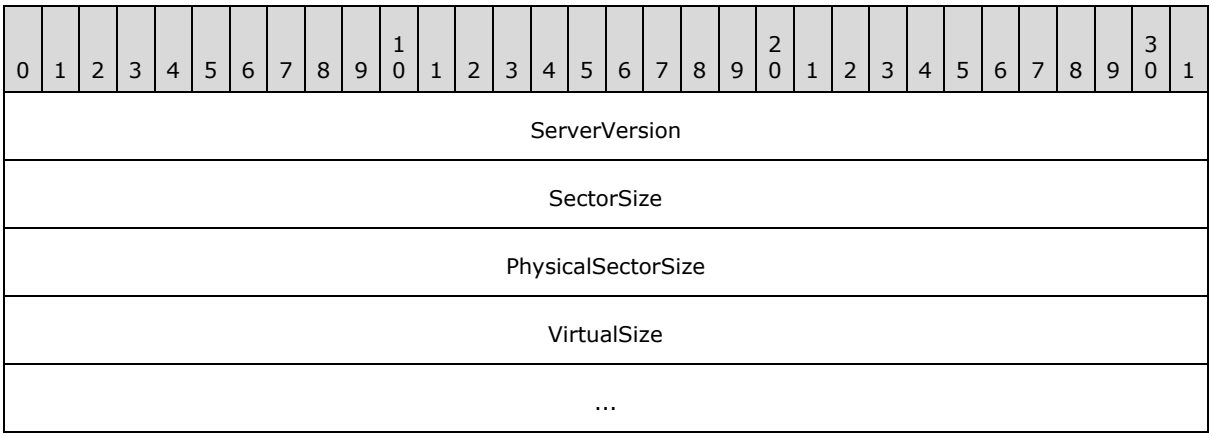
OriginatorFlags(4 bytes): This field is used to indicate which component has originated or issued the operation. This field MUST be set to 0x00000001.

InitiatorHostNameLength(2 bytes): The length, in bytes, of the **InitiatorHostName**, including the null termination.

InitiatorHostName(Variable): A variable-length buffer of up to 126 bytes containing a null-terminated Unicode UTF-16 string that specifies the computer name on which the initiator resides.

2.2.5.3 SVHDX_TUNNEL_FILE_INFO_REQUEST Structure

The SVHDX_TUNNEL_ FILE_INFO_REQUEST packet is sent by the client to get the shared virtual disk file information.



ServerVersion (4 bytes): The current version of the protocol running on the server.

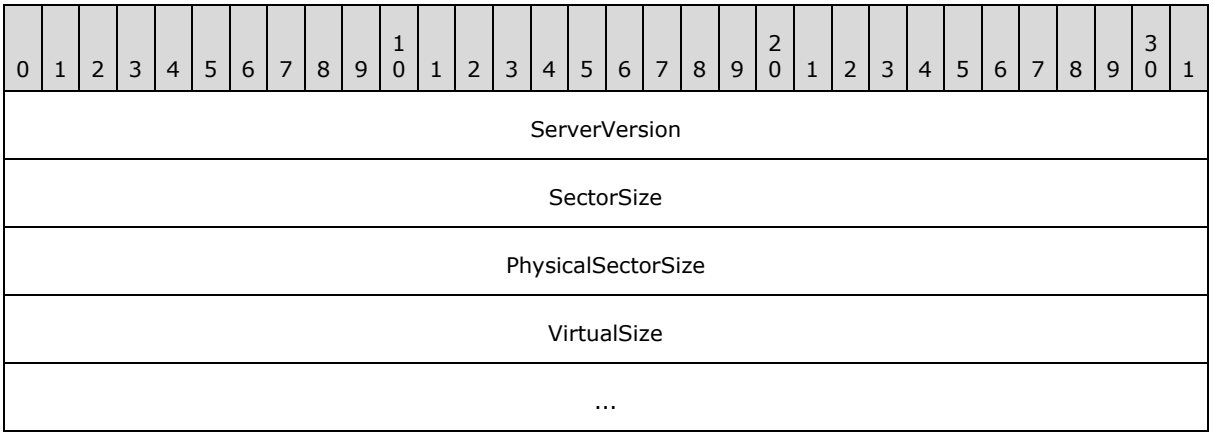
SectorSize (4 bytes): The client SHOULD set this field to zero, and the server MUST ignore it on receipt.

PhysicalSectorSize (4 bytes): The client SHOULD set this field to zero, and the server MUST ignore it on receipt.

VirtualSize (8 bytes): The client SHOULD set this field to zero, and the server MUST ignore it on receipt.

2.2.5.4 SVHDX_TUNNEL_FILE_INFO_RESPONSE Structure

The SVHDX_TUNNEL_FILE_INFO_RESPONSE packet is sent by the server in response to an RSVD_TUNNEL_GET_FILE_INFO_OPERATION packet.



ServerVersion (4 bytes): The current version of the protocol running on the server.

SectorSize (4 bytes): A 32-bit unsigned integer which indicates the sector size of the shared virtual disk.

PhysicalSectorSize (4 bytes): A 32-bit unsigned integer which indicates the physical sector size of the shared virtual disk.

VirtualSize (8 bytes): A 64-bit unsigned integer which indicates the virtual size of the shared virtual disk.

2.2.5.5 SVHDX_TUNNEL_CHECK_CONNECTION_REQUEST Structure

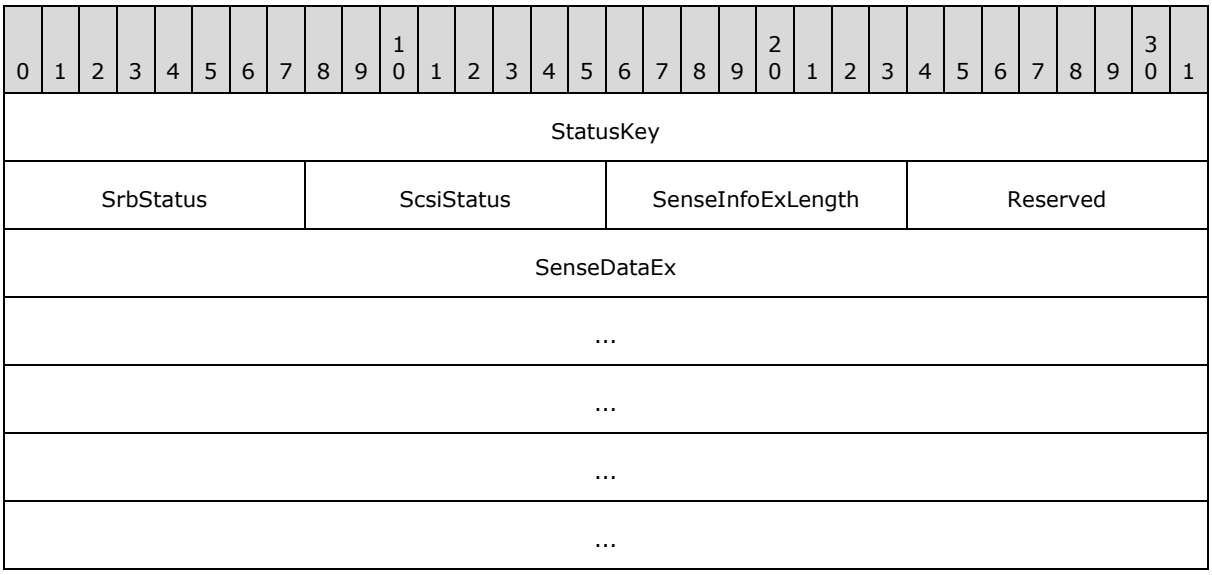
The SVHDX_TUNNEL_CHECK_CONNECTION_REQUEST packet is sent by the client to check the connection status to the shared virtual disk. The request MUST contain only SVHDX_TUNNEL_OPERATION_HEADER, and MUST NOT contain any payload.

2.2.5.6 SVHDX_TUNNEL_CHECK_CONNECTION_RESPONSE Structure

The SVHDX_TUNNEL_CHECK_CONNECTION_RESPONSE packet is sent by the server in response to the operation RSVD_TUNNEL_CHECK_CONNECTION_STATUS_OPERATION. The response MUST contain only SVHDX_TUNNEL_OPERATION_HEADER, and MUST NOT contain any payload.

2.2.5.7 SVHDX_TUNNEL_SRB_STATUS_REQUEST

The SVHDX_TUNNEL_SRB_STATUS_REQUEST packet is sent by the client to get the sense error code of a previously failed request.



StatusKey (4 bytes): The client MUST set this field to the least significant bytes of the status code value given by the server for the failed request.

SrbStatus (1 byte): This field MUST NOT be used and MUST be reserved. The client MUST set this field to zero, and the server MUST ignore it on receipt.

ScsiStatus (1 byte): This field MUST NOT be used and MUST be reserved. The client MUST set this field to zero, and the server MUST ignore it on receipt.

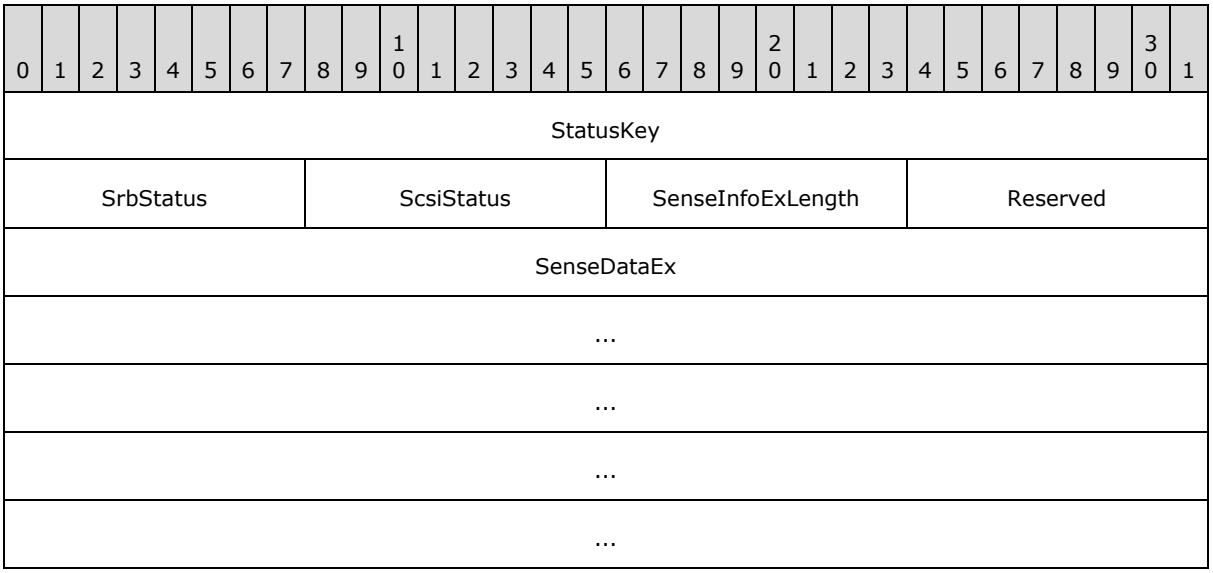
SenseInfoExLength (1 byte): This field MUST NOT be used and MUST be reserved. The client MUST set this field to zero, and the server MUST ignore it on receipt.

Reserved (1 byte): This field MUST be set to zero, and MUST be ignored on receipt.

SenseDataEx (20 bytes): This field MUST NOT be used and MUST be reserved. The client MUST set this field to zero, and the server MUST ignore it on receipt.

2.2.5.8 SVHDX_TUNNEL_SRB_STATUS_RESPONSE

The SVHDX_TUNNEL_SRB_STATUS_RESPONSE packet is sent by the server in a response to the RSVD_TUNNEL_SRB_STATUS_OPERATION.



StatusKey (4 bytes): The server MUST set this field to the status key value given by the client for the failed request.

SrbStatus (1 byte): An 8-bit field used to communicate error messages from the server to the client.

ScsiStatus (1 byte): An 8-bit field used to communicate the **SCSI** status that was returned by the shared virtual disk.

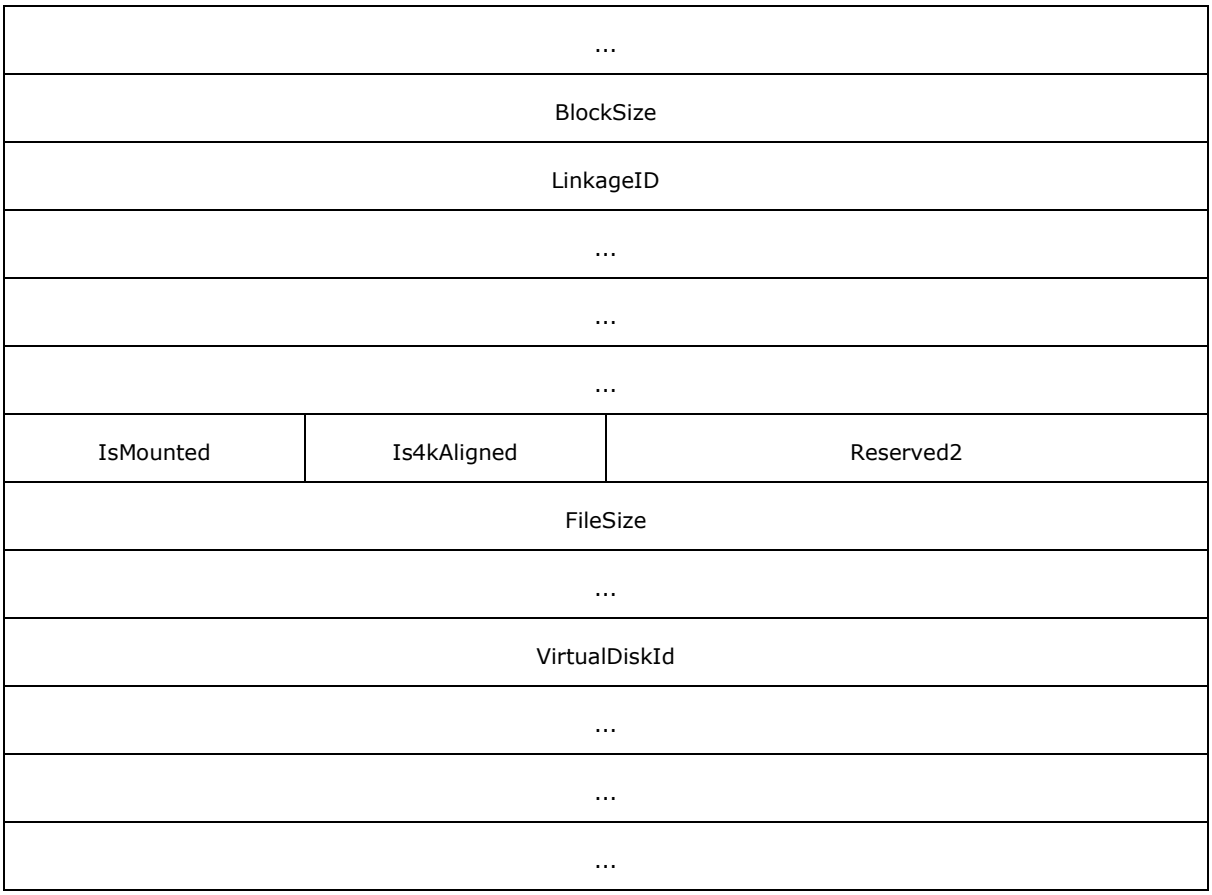
SenseInfoExLength (1 byte): The length, in bytes, of the request sense data buffer.

Reserved (1 byte): This field MUST be set to zero, and the client MUST ignore it on receipt.

SenseDataEx (20 bytes): A variable-length buffer that contains the **sense data**.

2.2.5.9 SVHDX_TUNNEL_DISK_INFO_REQUEST

The SVHDX_TUNNEL_DISK_INFO_REQUEST packet is sent by the client to get shared virtual disk information.



Reserved1 (8 byte): The client MUST set this field to zero, and the server MUST ignore it on receipt.

BlockSize (4 bytes): The client MUST set this field to zero, and the server MUST ignore it on receipt.

LinkageID (16 bytes): The client MUST set this field to zero, and the server MUST ignore it on receipt.

IsMounted (1 byte): The client MUST set this field to zero, and the server MUST ignore it on receipt.

Is4kAligned (1 byte): The client MUST set this field to zero, and the server MUST ignore it on receipt.

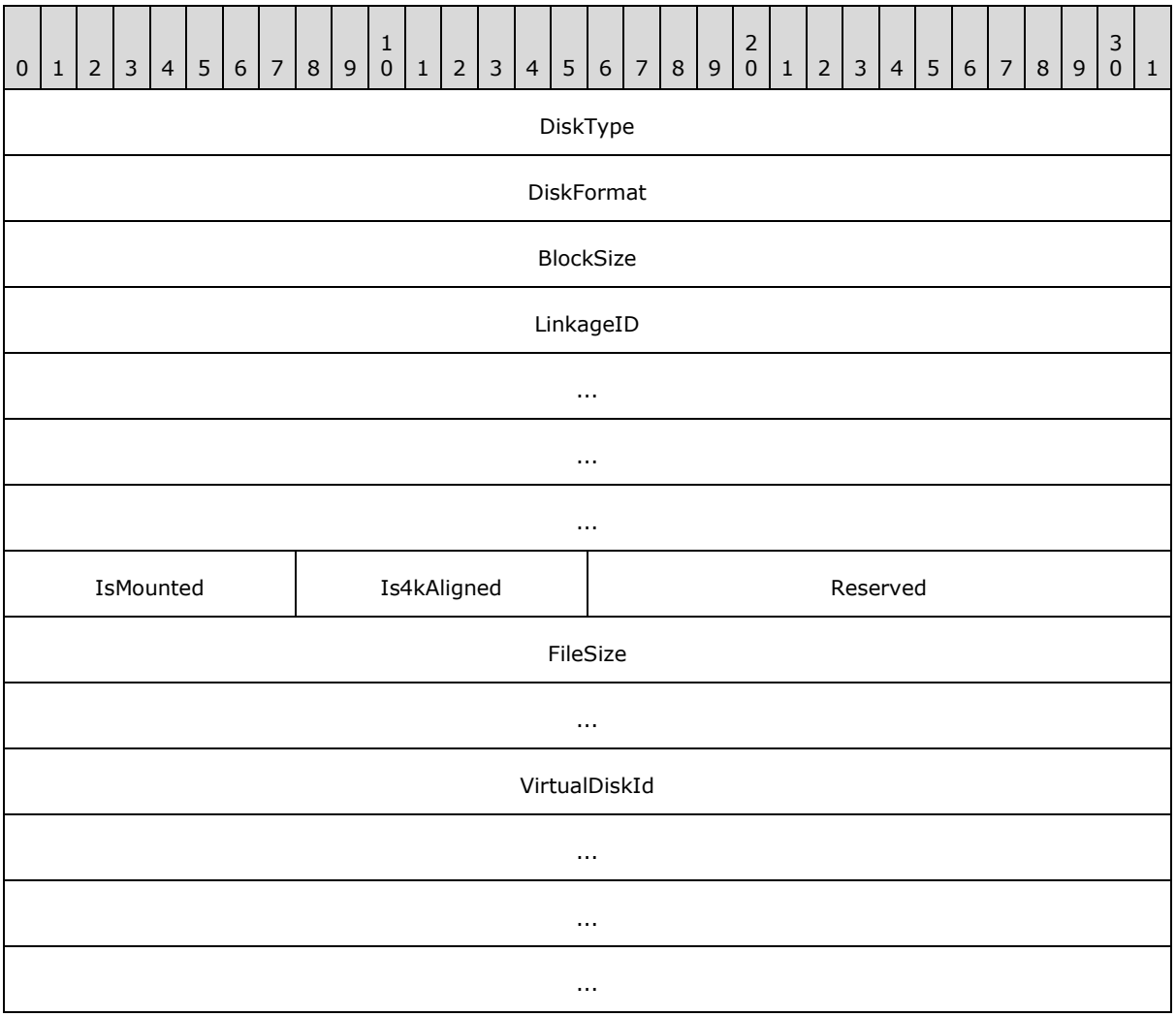
Reserved2 (2 bytes): The client MUST set this field to zero, and the server MUST ignore it on receipt.

FileSize (8 bytes): The client MUST set this field to zero, and the server MUST ignore it on receipt.

VirtualDiskId (16 bytes): This field MUST NOT be used and MUST be reserved. The client MUST set this field to zero, and the server MUST ignore it on receipt.

2.2.5.10 SVHDX_TUNNEL_DISK_INFO_RESPONSE

The SVHDX_TUNNEL_DISK_INFO_RESPONSE packet is sent by the server in response to an RSVD_TUNNEL_VALIDATE_DISK_OPERATION.



DiskType (4 bytes): Indicates the type of disk type. This field MUST contain exactly one of the following values.

Value	Meaning
VHD_TYPE_FIXED 0x00000002	Indicates that the type of the disk is fixed.
VHD_TYPE_DYNAMIC 0x00000003	Indicates that the type of the disk is dynamic.

DiskFormat (4 bytes): Indicates the type of the disk. This field MUST contain exactly one of the following values.

Value	Meaning
VIRTUAL_STORAGE_TYPE_DEVICE_VHDX 0x00000003	Indicates that the type of the disk is shared virtual disk.

BlockSize (4 bytes): Specifics the disk block size in bytes.

LinkageID (16 bytes): A GUID that specifies the linkage identification of the disk.

IsMounted (1 byte): A Boolean value. Zero represents FALSE (0x00), indicating that the disk is not ready for read or write operations. One represents TRUE (0x01), indicating that the disk is mounted and ready for read or write operations.

Is4kAligned (1 byte): A Boolean value. Zero represents FALSE, indicating disk sectors are not aligned to 4 kilobytes. One represents TRUE, indicating disk sectors are aligned to 4 kilobytes.

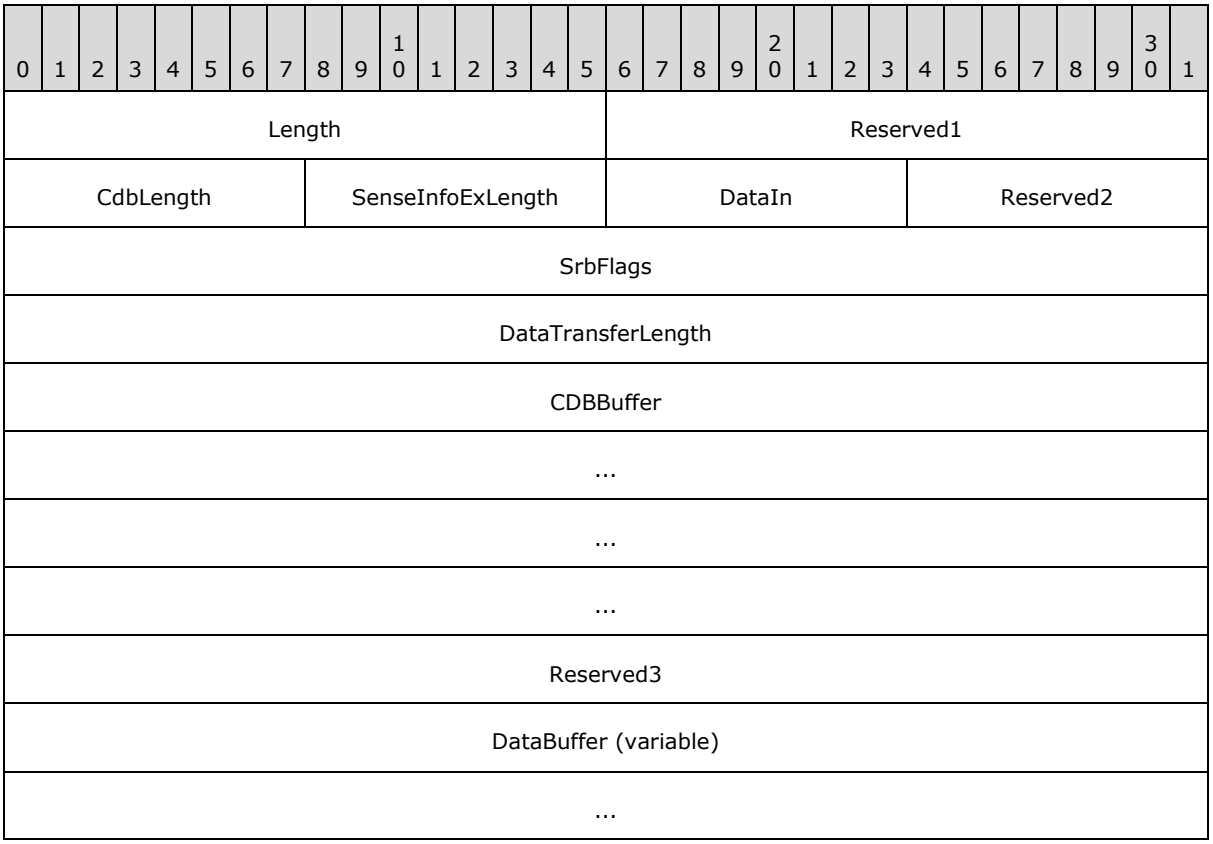
Reserved (2 bytes): This field MUST NOT be used and MUST be reserved. The client SHOULD set this field to zero, and the server MUST ignore it on receipt.

FileSize (8 bytes): The size of shared virtual disk, in bytes.

VirtualDiskId (16 bytes): A GUID that specifies the identification of the disk.

2.2.5.11 SVHDX_TUNNEL_SCSI_REQUEST

The SVHDX_TUNNEL_SCSI_REQUEST packet is sent by the client to process the SCSI request.



Length (2 bytes): Specifies the size, in bytes, of the SVHDX_TUNNEL_SCSI_REQUEST structure.

Reserved1 (2 byte): The client SHOULD set this field to zero, and the server MUST ignore it on receipt.

CdbLength (1 byte): The length, in bytes, of the SCSI command descriptor block. This value MUST be less than or equal to RSVD_CDB_GENERIC_LENGTH.

SenseInfoExLength (1 byte): The length, in bytes, of the request sense data buffer. This value MUST be less than or equal to RSVD_SCSI_SENSE_BUFFER_SIZE.

DataIn (1 byte): A Boolean, indicating the SCSI command descriptor block transfer type. The value TRUE (0x01) indicates that the operation is to store the data onto the disk. The value FALSE (0x00) indicates that the operation is to retrieve the data from the disk.

Reserved2 (1 byte): This field MUST be set to 0x00, and MUST be ignored on receipt.

SrbFlags (4 bytes): An optional, application-provided flag to indicate the options of the SCSI request, as specified in section 2.2.4.

DataTransferLength (4 bytes): The length, in bytes, of the additional data placed in the **DataBuffer** field.

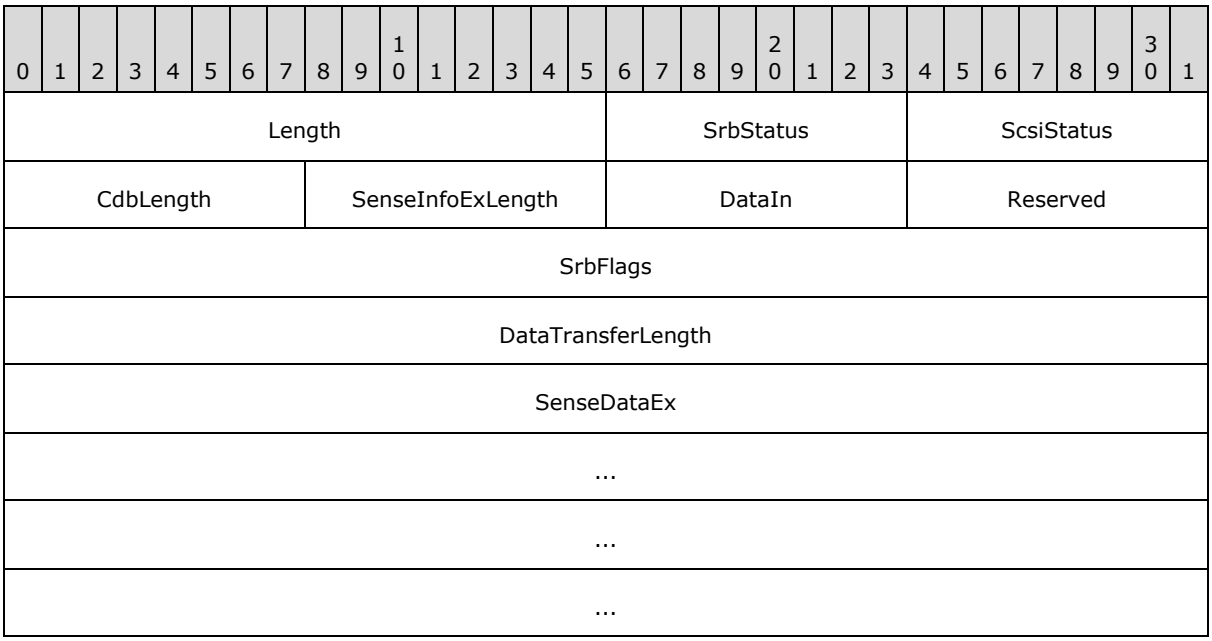
CDBBuffer (16 bytes): A buffer that contains the SCSI command descriptor block.

Reserved3 (4 bytes): This field SHOULD be set to 0x00000000; it MUST be ignored on receipt.

DataBuffer (variable): A variable-length buffer that contains the additional buffer, as described by the **DataTransferLength** field.

2.2.5.12 SVHDX_TUNNEL_SCSI_RESPONSE

The SVHDX_TUNNEL_SCSI_RESPONSE packet is sent by the server in response to the operation RSVD_TUNNEL_SCSI_OPERATION.



DataBuffer (variable)
...

Length (2 bytes): Specifies the size, in bytes, of the SVHDX_TUNNEL_SCSI_RESPONSE structure.

SrbStatus (1 byte): An 8-bit field used to communicate error messages from the server to the client, as specified in [SPC-3] section 4.18.

ScsiStatus (1 byte): An 8-bit field used to communicate the SCSI status that was returned by the target device.

CdbLength (1 byte): The length, in bytes, of the SCSI command descriptor block.

SenseInfoExLength (1 byte): The length, in bytes, of the request sense data buffer.

DataIn (1 byte): A Boolean used to indicate the command descriptor block data-transfer type. TRUE (0x01), indicates that the operation is to store the data onto the disk. The value FALSE (0x00) indicates that the operation is to retrieve the data from the disk.

Reserved (1 byte): This field MUST be set to zero, and MUST be ignored on receipt.

SrbFlags (1 byte): Special flags to indicate options of the SCSI response, as specified in section 2.2.4.

DataTransferLength (4 bytes): The length, in bytes, of the additional data placed in the **DataBuffer** field.

SenseDataEx (20 bytes): A variable-length buffer that contains the command descriptor buffer.

DataBuffer (variable): A variable-length buffer that contains the additional buffer, as described by the **DataTransferLength** field.

2.2.5.13 SVHDX_TUNNEL_VALIDATE_DISK_REQUEST Structure

The **SVHDX_TUNNEL_VALIDATE_DISK_REQUEST** packet is sent by the client to validate the shared virtual disk. The request MUST contain only SVHDX_TUNNEL_OPERATION_HEADER, and MUST NOT contain any payload.

2.2.5.14 SVHDX_TUNNEL_VALIDATE_DISK_RESPONSE Structure

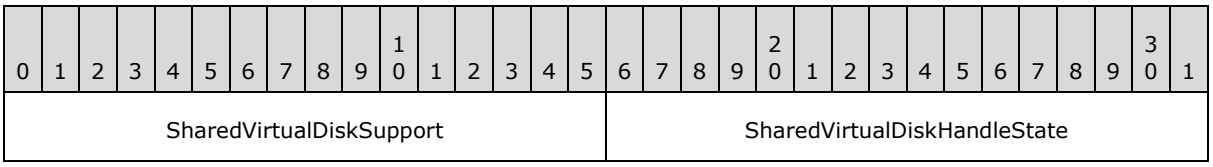
The **SVHDX_TUNNEL_VALIDATE_DISK_RESPONSE** packet is sent by the server in a response to the operation RSVD_TUNNEL_VALIDATE_DISK_OPERATION. The response MUST contain only SVHDX_TUNNEL_OPERATION_HEADER, and MUST NOT contain any payload.

2.2.5.15 SVHDX_SHARED_VIRTUAL_DISK_SUPPORT_REQUEST

The **SVHDX_SHARED_VIRTUAL_DISK_SUPPORT_REQUEST** packet is sent by the client to verify the status of shared virtual disk support on the Open. The request MUST contain only SVHDX_TUNNEL_OPERATION_HEADER, and MUST NOT contain any payload.

2.2.5.16 SVHDX_SHARED_VIRTUAL_DISK_SUPPORT_RESPONSE

The SVHDX_SHARED_VIRTUAL_DISK_SUPPORT_RESPONSE packet is sent by the server in a response to an FSCTL_QUERY_SHARED_VIRTUAL_DISK_SUPPORT request.



SharedVirtualDiskSupport (2 bytes): This field is used to indicate whether the server supports shared virtual disks. This field MUST be 0x0001.

SharedVirtualDiskHandleState(2 bytes): This field is used to indicates the state of the shared virtual disk Open. This field MUST contain one of the following values.

Value	Meaning
HandleStateNone 0x0000	The Open is not opened as a shared virtual disk .
HandleStateFileShared 0x0001	The shared virtual disk file is opened as a shared virtual disk by another Open.
HandleStateShared 0x0003	The shared virtual disk file is opened as a shared virtual disk by this Open.

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

3.1.1.1 Global

The client MUST implement the following:

ClientServiceVersion: The highest protocol version supported by the client.

RequestIdentifier: An unsigned 64-bit value assigned by the client for an outgoing request.

3.1.2 Timers

None.

3.1.3 Initialization

The client MUST initialize **ClientServiceVersion** to the highest protocol version supported by the client.<2>

RequestIdentifier: SHOULD<3> be initialized to an implementation-specific value.

3.1.4 Higher-Layer Triggered Events

3.1.4.1 Sending Any Outgoing Message

The Client MUST increment **RequestIdentifier** by 1, for every outgoing tunnel operation specified in section 2.2.2.

3.1.4.2 Application Requests Opening a Shared Virtual Disk

The application provides the following:

- The name of the server to connect to.
- The name of the share to connect to.
- InitiatorId to uniquely identify the initiator (optional).
- User credentials, an opaque implementation-specific entity that identifies the credentials to be used when authenticating to the remote server.
- The shared Virtual Disk file name to open.
- The flags for open command (optional).
- The Initiator host name.

Upon successful completion, the client MUST return a handle to the application.

The client MUST construct an SVHDX_OPEN_DEVICE_CONTEXT as specified in section 2.2.5.2 and MUST be initialized as follows:

- The **Version** field is set to **ClientServiceVersion**.
- The **InitiatorId** field is set to the application-provided initiatorID, if any. Otherwise, InitiatorId is set to zero.
- The **HasInitiatorId** field is set to TRUE if the application provides initiator ID; otherwise set to FALSE.
- The **Flags** field is set to application-provided flags.
- The **OriginatorFlags** set to 0x00000001.
- The **InitiatorHostNameLength** set to **InitiatorHostName** length, in bytes, including the terminating null character.
- The **InitiatorHostName** set to application-provided initiator name.

The client MUST construct two FILE_FULL_EA_INFORMATION structures as specified in [MS-FSCC] section 2.4.15.

The client MUST initialize application instance EA as follows:

- The **NextEntryOffset** field is set to zero.
- The **Flags** field is set to zero.
- The **EaNameLength** field is set length of SMB_CCF_APP_INSTANCE_EA_NAME, in bytes.
- The **EaValueLength** field is set to length of InitiatorId, in bytes.
- The **EaName** field is set to SMB_CCF_APP_INSTANCE_EA_NAME.
- The **EaValue** field is set to InitiatorId.

The client MUST initialize traffic EA as follows:

- The **NextEntryOffset** field is set to zero.
- The **Flags** field is set to zero
- The **EaNameLength** field is set length of *APP_TRAFFIC_TYPE_EA_NAME*, in bytes.
- The **EaValueLength** field is set to length of InitiatorId, in bytes.
- The **EaName** field is set to *APP_TRAFFIC_TYPE_EA_NAME*.
- The **EaValue** field is set to InitiatorId.

The client MUST append ":SharedVirtualDisk" at the end of the **VHDXFileName**.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.30, to open the shared virtual disk file, supplying the following input parameters:

- The application-provided server name.
- The spplication-provided share name.
- The spplication-provided user credentials.

- The client MUST append ":SharedVirtualDisk" at the end of the application-provided file name, and supply the result as the file name parameter.
- The extended Attributes list.
- The SVHDX_OPEN_DEVICE_CONTEXT Create context.

If there are any errors from the preceding call, return the error to the caller.

3.1.4.3 Application Requests Closing a Shared Virtual Disk

The application provides:

- A handle to the **Open** identifying the shared virtual disk file.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.5 to close the open on the shared virtual disk file, supplying the application-provided handle.

3.1.4.4 Application Requests Reading From a Shared Virtual Disk

The application provides:

- A handle to the **Open** identifying the shared virtual disk file.
- The offset, in bytes, from the beginning of the virtual disk from which to read data.
- The number of bytes to read.
- The minimum number of bytes to be read (optional).
- The buffer to receive the data.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.6, supplying the application-provided parameters

3.1.4.5 Application Requests Writing To a Shared Virtual Disk

The application provides:

- A handle to the **Open** identifying a shared virtual disk file.
- The offset, in bytes, from the beginning of the virtual disk where data should be written.
- The number of bytes to write.
- A buffer containing the bytes to be written.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.7, supplying the application-provided parameters.

3.1.4.6 Application Requests Virtual Disk File information

The application provides:

- A handle to the **Open** identifying a shared virtual disk file.
- The maximum output buffer size that it will accept.

The client MUST construct an SVHDX_TUNNEL_FILE_INFO _REQUEST structure, as specified in section 2.2.5.3 as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** MUST be set to RSVD_TUNNEL_GET_FILE_INFO_OPERATION.
- The **Status** field MUST be set to zero.
- The **RequestId** field MUST be set to **RequestIdentifier**.

The SVHDX_TUNNEL_FILE_INFO_REQUEST Request MUST be initialized as follows:

- The **ServerVersion** field MUST be set to zero.
- The **SectorSize** field MUST be set to zero.
- The **PhysicalSectorSize** field MUST be set to zero.
- The **VirtualSize** field MUST be set to zero.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.20.11, supplying the following input parameters:

- Application-provided handle to identify the **Open**.
- Control code: FSCTL_SVHDX_SYNC_TUNNEL_REQUEST.
- SVHDX_TUNNEL_FILE_INFO_REQUEST structure as payload.
- The maximum output buffer size that it will accept.

3.1.4.7 Application Requests Connection Status

The application provides:

- A handle to the **Open** identifying a shared virtual disk file.

The client MUST construct an SVHDX_TUNNEL_CHECK_CONNECTION_REQUEST structure, as specified in section 2.2.5.5 as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** MUST be set to RSVD_TUNNEL_CHECK_CONNECTION_STATUS_OPERATION.
- The **Status** field MUST be set to zero.
- The **RequestId** field MUST be set to **RequestIdentifier**.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.20.11, supplying the following input parameters:

- Application-provided handle to identify the **Open**.
- Control code: FSCTL_SVHDX_SYNC_TUNNEL_REQUEST.
- SVHDX_TUNNEL _OPERATION_HEADER_ as payload.

3.1.4.8 Application Requests Shared Virtual Disk Information

The application provides:

- A handle to the **Open** identifying a shared virtual disk file.

The client MUST construct an SVHDX_TUNNEL_DISK_INFO_REQUEST structure, as specified in section 2.2.5.9, as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** MUST be set to RSVD_TUNNEL_GET_DISK_INFO_OPERATION.
- The **Status** field MUST be set to zero.
- The **RequestId** field MUST be set to **RequestIdentifier**.

The SVHDX_TUNNEL_DISK_INFO_REQUEST Request MUST be initialized as follows:

- The **Reserved** field MUST be set to zero.
- The **BlockSize** field MUST be set to zero.
- The **LinkageId** field MUST be set to zero.
- The **IsMounted** field MUST be set to zero.
- The **Is4kAligned** field MUST be set to zero.
- The **FileSize** field MUST be set to zero.
- The **VirtualDiskId** field MUST be set to zero.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.20.11, supplying the following input parameters:

- Application-provided handle to identify the **Open**.
- Control code: FSCTL_SVHDX_SYNC_TUNNEL_REQUEST.
- SVHDX_TUNNEL_DISK_INFO_REQUEST packet, as payload.

3.1.4.9 Application Requests Execution of SCSI Command

The application provides:

- A handle to the **Open** identifying a shared virtual disk file.
- Initiator ID of the initiator.
- SCSI Common Descriptor Block (CDB) command length in bytes.
- Sense info length in bytes.
- SCSI command flags, indicates options about the SCSI request, as specified in section 2.2.4. Buffer containing the SCSI CDB command being requested.
- Length of any additional data (optional).

- Additional data buffer (optional).

The client MUST construct an SVHDX_TUNNEL_SCSI_REQUEST structure, as specified in section 2.2.5.11as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** field is set to RSVD_TUNNEL_SCSI_OPERATION.
- The **Status** is set to zero.
- The **RequestId** field MUST be set to RequestIdentifier.

The SVHDX_TUNNEL_SCSI_REQUEST MUST be initialize as follows:

- The **Length** is set to the size, in bytes, of the SVHDX_TUNNEL_SCSI_REQUEST structure that is constructed following the syntax specified in section 2.2.5.16 with the client input parameters as follows:
  - The **CDBLength** is set to the application-provided CDB command-length value.
  - The **SenseInfoExLength** set to the application-provided sense info length value.
  - The **DataIn** set to TRUE (0x01) if the SCSI command type is Data-in, that is, the CDB buffer specified is to receive data from the server, as specified in [SPC-3] section 3.2. Otherwise, set to FALSE (0x00).)
  - The **Reserved** field MUST be set to zero.
  - The **SrbFlags** field MUST be set to the application-provided special flag values.
  - The **DataBufferLength** field MUST be set to the application-provided value. If the application doesn't provide a data buffer length, the client MUST set this field to zero.
  - The **CDBBuffer** field MUST be set to the application-provided CDB buffer. If the application doesn't provide the buffer, the client MUST set this field to empty.
  - The **DataBuffer** field MUST be set to the application-provided additional data buffer. If the application doesn't provide the buffer, the client MUST set this field to empty.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.20.11, supplying the following input parameters:

- Application-provided handle to identify the **Open**.
- Control code: FSCTL_SVHDX_SYNC_TUNNEL_REQUEST.
- SVHDX_TUNNEL_SCSI_REQUEST structure as payload.

3.1.4.10 Application Requests to Validate a Shared Virtual Disk

The application provides:

- A handle to the **Open** identifying a shared virtual disk file.

The client MUST construct an SVHDX_TUNNEL_VALIDATE_DISK_REQUEST structure, as specified in section 2.2.5.13, as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** MUST be set to RSVD_TUNNEL_VALIDATE_DISK_OPERATION.
- The **Status** field MUST be set to zero.
- The **RequestId** field MUST be set to **RequestIdentifier**.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.20.11, supplying the following input parameters:

- Application-provided handle to identify the **Open**.
- Control code: FSCTL_SVHDX_SYNC_TUNNEL_REQUEST.
- SVHDX_TUNNEL_OPERATION_HEADER structure as payload.

3.1.4.11 Application Requests Querying Shared Virtual Disk Support

The application provides:

- A handle to the **Open** identifying a shared virtual disk file.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.20.11 supplying the following input parameters:

- Application-provided handle to identify the **Open.**
- Control code: FSCTL_QUERY_SHARED_VIRTUAL_DISK_SUPPORT

3.1.5 Message Processing Events and Sequencing Rules

3.1.5.1 Receiving an Open Response

If the response returned by the SMB server indicates an error, the client MUST return the received error code to the calling application.

Otherwise, the client MUST return success and the **Open.ContextHandle** to the calling application.

3.1.5.2 Receiving a Close Response

The client MUST return the received response to the calling application.

3.1.5.3 Receiving a Read Response

If the response returned by the SMB server indicates success, the client MUST return success and the data buffer that contains the data read for the response.

If the response indicates an error, and the received error code is specified in the section 2.2.3, the server MUST return the error code.

If the received error code is not specified in the section 2.2.3, the client MUST construct an SVHDX_TUNNEL_SRB_STATUS_REQUEST structure as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** MUST be set to RSVD_TUNNEL_SRB_STATUS_OPERATION.

- The **Status** field MUST be set to zero.
- The **RequestId** field MUST be set to **RequestIdentifier**.

The SVHDX_TUNNEL_SRB_STATUS_REQUEST structure MUST be initialized as follows:

- The **StatusKey** field MUST be set to the least significant bytes of the status code value given by the server for this request.
- The **SrbStatus** field MUST be set to zero.
- The **ScsiStatus** field MUST be set to zero.
- The **SenseInfoExLength** field MUST be set to zero.
- The **Reserved** field MUST be set to zero.
- The **SenseDataEx** field MUST be set to empty.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.20.11, supplying the following input parameters:

- Application-provided handle used for Read request.
- Control code: FSCTL_SVHDX_SYNC_TUNNEL_REQUEST.
- SVHDX_TUNNEL_SRB_STATUS_REQUEST structure as payload.

The client MUST return the received sense error response to the calling application.

3.1.5.4 Receiving a Write Response

If the response returned by the SMB server indicates success, the client MUST return success to the calling application.

If the response indicates an error, and the received error code is specified in the section 2.2.3, the server MUST return the error code.

If the received error code is not specified in the section 2.2.3, the client MUST construct an SVHDX_TUNNEL_SRB_STATUS_REQUEST structure as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** MUST be set to RSVD_TUNNEL_SRB_STATUS_OPERATION.
- The **Status** field MUST be set to zero.
- The **RequestId** field MUST be set to **RequestIdentifier**.

The SVHDX_TUNNEL_SRB_STATUS_REQUEST structure MUST be initialized as follows:

- The **StatusKey** field MUST be set to the least significant bytes of the status code value given by the server for this request.
- The **SrbStatus** field MUST be set to zero.
- The **ScsiStatus** field MUST be set to zero.
- The **SenseInfoExLength** field MUST be set to zero.

- The **Reserved** field MUST be set to zero.
- The **SenseDataEx** field MUST be set to empty.

The client MUST call the interface specified in [MS-SMB2] section 3.2.4.20.11, supplying the following input parameters:

- Application-provided handle used for Write request.
- Control code: FSCTL_SVHDX_SYNC_TUNNEL_REQUEST.
- SVHDX_TUNNEL_SRB_STATUS_REQUEST structure as payload.

The client MUST return the received sense error response to the calling application.

3.1.5.5 Receiving Virtual Disk File information Response

If the response returned by the SMB server indicates an error, the client MUST return the received error code to the calling application.

If the response indicates success, the client MUST return success and the disk file information to the calling application.

3.1.5.6 Receiving Connection Status Response

The client MUST return the received status code to the calling application.

3.1.5.7 Receiving Shared Virtual Disk Information

If the response returned by the SMB server indicates an error, the client MUST return the received status code to the calling application.

If the response indicates success, the client MUST return success and the virtual disk information to the calling application.

3.1.5.8 Receiving SCSI Command Response

If the response returned by the SMB server indicates an error, the client MUST return the received error code to the calling application.

If the response indicates success, the client MUST return success and response data to the calling application.

3.1.5.9 Receiving Validate Disk Response

The client MUST return the received status to the calling application.

3.1.5.10 Receiving Shared Virtual Disk Support Response

The client MUST return the received status and response data to the calling application.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

3.2.1 Abstract Data Model

3.2.1.1 Global

The server MUST implement the following:

OpenTable: A table of opened shared virtual disk files, as specified in section 3.2.1.2, indexed by InitiatorId.

ServerServiceVersion: The highest protocol version supported by the server.

3.2.1.2 Per Open

Open.LocalOpen: An **Open** of a shared virtual disk file in the local resource that is used to perform the local operations, such as reading or writing, to the underlying object.

Open.FileName: A variable-length string that contains the Unicode file name supplied by the client for opening the shared virtual disk.

Open.SenseErrorSequence: An unsigned 8-bit identifier indicating the sense error sequence which counts from 0 through 255.

Open.SenseErrorDataList: A list of sense errors indexed by the **Open.SenseErrorsequence**, as defined in section 3.2.1.3.

3.2.1.3 Per SenseError in SenseErrorDataList

SenseError.StatusKey: A 32-bit value assigned by the server for this sense error.

SenseError.SrbStatus: An 8-bit value indicating the status SCSI request block.

SenseError.ScsiStatus: An 8-bit value indicating the SCSI status that was returned by the shared virtual disk.

SenseError.SenseData: A pointer to the sense data.

3.2.2 Timers

None.

3.2.3 Initialization

When the server is started:

- It MUST notify the SMB server that it is ready to process client requests, as specified in [MS-SMB2] section 3.3.4.25.
- The server MUST initialize **ServerServiceVersion** to the highest protocol version supported by the server.<4>

3.2.4 Higher-Layer Triggered Events

3.2.5 Message Processing Events and Sequencing Rules

3.2.5.1 Receiving an Open Request

When the server receives a request to open a shared virtual disk file, by providing the file name and SVHDX_OPEN_DEVICE_CONTEXT, the server MUST verify the following:

If the file name does not contain":SharedVirtualDisk" at the end, the server MUST fail the request with error STATUS_INVALID_PARAMETER.

If the **Version** is not equal to the RSVD_ECP_CONTEXT_VERSION_1, the server MUST fail the request with STATUS_INVALID_PARAMETER.

If **HasInitiatorId** is neither FALSE (0x00) nor TRUE (0x01), the server MUST fail the request with STATUS_INVALID_PARAMETER.

The server MUST pass the request to the underlying object store to open the file with read and write access permissions.

If the underlying object store returns a failure indicating that the attempted open operation failed, the server MUST return the received error code from the object store.

If the underlying object store returns success, the server MUST initialize the **Open** as follows, and return STATUS_SUCCESS to the client.

- **Open.LocalOpen** is set to the **Open** of the object in the local resource received as part of the local creates operation.
- **Open.FileName** MUST be set to the application-provided file name.
- **Open.SenseErrorDataList** MUST be set to empty.
- **Open.SenseErrorSequence** MUST be set to 0x00.

3.2.5.2 Receiving a Close Request

When the server receives a request to close a shared virtual disk file, the server MUST verify the following:

The server MUST locate the **Open** in the **OpenTable** where **Open.LocalOpen** matches the **Open** provided by the SMB server, as specified in [MS-SMB2] section 3.3.4.17.

The server MUST close the underlying object store open.

If under the underlying object store return success, the server MUST remove the **Open** from the **OpenTable**, free the **Open** object, and MUST return STATUS_SUCCESS to the client.

Otherwise, the server MUST return the received error code to the client.

3.2.5.3 Receiving a Read Request

When the server receives a read request, the server MUST locate the Open in the OpenTable, where Open.LocalOpen matches the Open provided by the SMB2 server, as specified in [MS-SMB2] section 3.2.5.12.

If no **Open** is found, the server MUST fail the request with the STATUS_INVALID_PARAMETER code.

If **Open** is found, the server MUST issue a read to the underlying object store in an implementation-specific manner.

If the underlying object store indicates the read is successful, the server MUST return the read data buffer and success to the client.

If the underlying object store indicates that the read failed, and the received error code is specified in the section 2.2.3, the server MUST return the error code.

If the error code is not specified in the section 2.2.3, the server MUST store the received sense data in **Open.SenseErrorDataList** as follows:

- If **Open.SenseErrorSequence** is 0xFF, the server MUST reset **Open.SenseErrorSequence** to 0x00. Otherwise, the server MUST increment the **Open.SenseErrorSequence** by 0x01.
- The server MUST update **SenseError** in **Open.SenseErrorDataList** at index **Open.SenseErrorSequence** as follows:
  - **SenseError.StatusKey** MUST be set to **Open.SenseErrorSequence**.
  - **SenseError.SrbStatus** MUST be set to the value provided by the object store.
  - **SenseError.ScsiStatus** MUST be set to the value provided by the object store
  - **SenseError.SenseData** MUST be set to the data provided by the object store.

The server MUST return the error (STATUS_SVHDX_ERROR_STORED | **SenseError.StatusKey**) to the client.

3.2.5.4 Receiving a Write Request

When the server receives a write request, the server MUST locate the **Open** in the **OpenTable** where **Open.LocalOpen** matches the Open provided by the SMB2 server, as specified in [MS-SMB2] section 3.3.5.13.

If no **Open** is found, the server MUST fail the request with STATUS_INVALID_PARAMETER code.

If an **Open** is found, the server MUST issue a write to the underlying object store in an implementation-specific manner

If the underlying object store indicates that the write was successful, the server MUST return success to the client.

If the underlying object store indicates a persistent reservation denied the write, the server MUST fail the request with error STATUS_SVHDX_RESERVATION_CONFLICT.

Otherwise, if the received error code is specified in the section 2.2.3, the server MUST return the error code.

If the error code is not specified in the section 2.2.3, the server MUST store the received sense data in to **Open.SenseErrorDataList** as following.

- If **Open.SenseErrorSequence** is 0xFF, the server MUST reset **Open.SenseErrorSequence** to 0x00. Otherwise, the server MUST increment the **Open.SenseErrorSequence** by 0x01.

- The server MUST update **SenseError** in **Open.SenseErrorDataList** at index **Open.SenseErrorSequence** as follows:
  - **SenseError.StatusKey** MUST be set to **Open.SenseErrorSequence**.
  - **SenseError.SrbStatus** MUST be set to the value provided by the object store.
  - **SenseError.ScsiStatus** MUST be set to the value provided by the object store
  - **SenseError.SenseData** MUST be set to the data provided by the object store.

The server MUST return the error STATUS_SVHDX_ERROR_STORED or **SenseError.StatusKey** to the client.

3.2.5.5 Receiving Tunnel Operation request

When the server receives a tunnel operation request, the server MUST locate the **Open** in the **OpenTable** where **Open.LocalOpen** matches the **Open** provided by the SMB2 server, as specified in [MS-SMB2] section 3.3.5.13.

If no **Open** is found, the server MUST fail the request and return STATUS_INVALID_PARAMETER.

If an **Open** is found, the server MUST verify the **OperationCode** of the request.

If the **OperationCode** value does not exist in the tunnel operation list, specified in section 2.2.2, the server MUST fail the request with STATUS_SVHDX_VERSION_MISMATCH.

Processing for a specific **OperationCode** is as specified in subsequent sections.

3.2.5.5.1 Receiving Virtual Disk File Information Request

When the server receives a request **OperationCode** equal to RSVD_TUNNEL_GET_FILE_INFO_OPERATION, the request handling proceeds as follows:

The server MUST issue a file information request to the underlying object store in an implementations-specific manner, and update the received response to the client.

If the underlying object store indicates an error, the server MUST fail the request with the error received by placing it into the **Status** field of the SVHDX_TUNNEL_OPERATION_HEADER, and return the header to the client.

Otherwise, the server MUST construct an SVHDX_TUNNEL_FILE_INFO_RESPONSE structure as specified in section 2.2.5.4 with the following values:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- **OperationCode** MUST be set to the OperationCode value of the request.
- **Status** MUST be set to STATUS_SUCCESS
- The **RequestId** field MUST be set to the value received in the request.

The SVHDX_TUNNEL_FILE_INFO_RESPONSE structure MUST be initialized as follows:

- **ServerVersion** MUST be set to the **ServerServiceVersion**.
- **SectorSize** MUST set to the sector size of the shared virtual disk received from the object store.

- **PhysicalSectorSize** MUST set to the physical sector size of the shared virtual disk received from the object store.
- **VirtualSize** MUST set to the virtual size of the shared virtual disk received from the object store.

The response MUST be sent to the client.

3.2.5.5.2 Receiving Connection Status Request

When the server receives a request with an **OperationCode** equal to RSVD_TUNNEL_CHECK_CONNECTION_STATUS_OPERATION, the request handling proceeds as follows:

The server MUST construct an SVHDX_TUNNEL_CHECK_CONNECTION_RESPONSE structure as specified in section 2.2.5.6 with the following values:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- **OperationCode** MUST be set to the OperationCode value of the request.
- **Status** MUST be set to STATUS_SUCCESS.
- The **RequestId** field MUST be set to the value received in the request.

The response MUST be sent to the client.

3.2.5.5.3 Receiving Sense Code Request

When the server receives a request with **OperationCode** equal to RSVD_TUNNEL_SRB_STATUS_OPERATION, the request handling proceeds as follows:

If **SenseInfoExLength** field of the request is not equal to the size of the **SenseDataEx**, the server MUST fail the request with STATUS_INVALID_PARAMETER.

The server MUST locate the **SenseError** in **Open.SenseErrorDataList** where **SenseError.StatusKey** matches the **StatusKey** of the request.

If **SenseError** is not found, the server MUST return STATUS_SVHDX_ERROR_NOT_AVAILABLE.

If **SenseError** is found, the server MUST construct the SVHDX_TUNNEL_SRB_STATUS_RESPONSE structure as specified in section 2.2.5.8 with the following values:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** MUST be set to the OperationCode value of the request.
- The **Status** MUST be set to STATUS_SUCCESS.
- The **RequestId** field MUST be set to the value received in the request.

The SVHDX_TUNNEL_SRB_STATUS_RESPONSE packet MUST be initialized as follows:

- The **StatusKey** field MUST be set to the value received in the request.
- The **SrbStatus** MUST be set to **SenseError.SrbStatus**.
- The **ScsiStatus** MUST be set to **SenseError.ScsiStatus**.

- The **SenseInfoExLength** MUST be set to the length of the **SenseError.SenseData**, in bytes
- The **Reserved** field MUST be set to zero.
- The **SenseDataEx** MUST be set to **SenseError.SenseData**.

The response MUST be sent to the client.

3.2.5.5.4 Receiving Shared Virtual Disk Information Request

When the server receives a request with an **OperationCode** equal to RSVD_TUNNEL_GET_DISK_INFO_OPERATION, the request handling proceeds as follows:

The server MUST issue a query disk information request to the underlying object store in an implementation-specific manner and update the received response to the client.

If the underlying object store indicates an error, the server MUST fail the request with the error received, placing into the **Status** field of the SVHDX_TUNNEL_OPERATION_HEADER, and return the header to the client.

Otherwise, the server MUST construct an SVHDX_TUNNEL_DISK_INFO_RESPONSE structure as specified in section 2.2.5.8 with the following values:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- **OperationCode** MUST be set to the OperationCode value of the request.
- **Status** MUST be set to STATUS_SUCCESS.
- The **RequestId** field MUST be set to the value received in the request.

The SVHDX_TUNNEL_DISK_INFO_RESPONSE structure MUST be initialized as follows:

- **DiskType** MUST be set to the value received from the object store.
- **DiskFormat** MUST be set to VIRTUAL_STORAGE_TYPE_DEVICE_VHDX.
- **BlockSize** MUST be set to the number of bytes received from object store.
- **LinkageId** MUST be set to the linkage GUID received from object store.
- **IsMounted** MUST be set to TRUE if the object store indicates that the disk is is ready for read or write operations. Otherwise, the server MUST set this field to FALSE.
- **Is4kAligned** MUST be set to TRUE if the object store indicates that the disk sectors are aligned to 4 kilobytes. Otherwise, the server MUST set this field to FALSE.
- **Reserved** MUST be set to zero.
- **FileSize** MUST be set to the value received from the object store.
- **VirtualDiskId** MUST be set to the virtual disk GUID received from object store.

The response MUST be sent to the client.

3.2.5.5.5 Receiving SCSI Command Request

When the server receives a request with an **OperationCode** equal to RSVD_TUNNEL_SCSI_OPERATION, the request handling proceeds as follows:

If the **Length** field of the request is not equal to the size of the SVHDX_TUNNEL_SCSI_REQUEST, the server MUST fail the request with STATUS_INVALID_PARAMETER.

If **CdbLength** is greater than size of **CDBBuffer**, the server MUST fail the request with STATUS_INVALID_PARAMETER.

The server MUST issue a CDB command to the underlying object store in an implementation-specific manner and update the received response to the client.

If the underlying object store indicates an error, the server MUST fail the request with the error received, placing into the **Status** field of the SVHDX_TUNNEL_OPERATION_HEADER, and return the header to the client.

Otherwise, the server MUST construct a SVHDX_TUNNEL_SCSI_RESPONSE structure as specified in section 2.2.5.12 with the following values:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** field MUST be set to the OperationCode value of the request.
- The **Status** field MUST be set to STATUS_SUCCESS.
- The **RequestId** field MUST be set to the value received in the request.

The SVHDX_TUNNEL_SCSI_RESPONSE MUST be initialized as follows:

- The **SrbStatus** field MUST be set to value received from the underlying object store.
- The **SCSIStatus** field MUST be set to value received from the underlying object store.
- The **DataIn** field MUST be set to the value received in the request.
- The **Length** field is set to the size, in bytes, of the SVHDX_TUNNEL_SCSI_RESPONSE structure that is constructed following the syntax specified in section 2.2.5.12.

The response MUST be sent to the client.

3.2.5.5.6 Receiving Validate Disk Request

When the server receives a request with an **OperationCode** equal to RSVD_TUNNEL_VALIDATE_DISK_OPERATION, the request handling proceeds as follows:

The server MUST issue a validate request to the underlying object store in an implementation-specific manner and update the received response to the client.

The server MUST construct an SVHDX_TUNNEL_VALIDATE_DISK_RESPONSE structure as specified in section 2.2.5.12 with the following values:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

OperationCode MUST be set to the OperationCode value of the request.

If the underlying object store indicates command success, the **Status** MUST be set to STATUS_SUCCESS; otherwise, it MUST be set to the error code provided by the underlying object store.

The **RequestId** field MUST be set to the value received in the request.

The response MUST be sent to the client.

3.2.5.6 Receiving Query Shared Virtual Disk Support Request

The calling application provides.

- A handle identifying the **Open**.
- File name to identify the shared virtual disk file.

The server MUST search the **OpenTable** where **Open.FileName** matches the file name.

If no **Open** is found, the server MUST set **SharedVirtualDiskHandleState** to **HandleStateNone** (0x0000).

If any **Open** is found for which **Open.LocalOpen** matches the application-provided handle, the server MUST set **SharedVirtualDiskHandleState** to **HandleStateShared** (0x0003).

Otherwise, the server MUST set **SharedVirtualDiskHandleState** to **HandleStateFileShared** (0x0001).

The response MUST be sent to the client.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

None.

5 Security

5.1 Security Considerations for Implementers

None.

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 Server 2012 R2 operating system
- Windows 8.1 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 1.7: The following table illustrates Windows operating system versions that support RSVD clients and RSVD servers.

RSVD client	RSVD server
Windows Server 2012 R2 Windows 8.1	Windows Server 2012 R2

<2> Section 3.1.3: Windows 8.1 and Windows Server 2012 R2 set **ClientServiceVersion** to RSVD protocol version 1(0x00000001).

<3> Section 3.1.3: Windows-based RSVD clients initialize to zero.

<4> Section 3.2.3: Windows Server 2012 R2 sets **ServerServiceVersion** to RSVD Protocol version 1 (0x00000001).

7 Change Tracking

This section identifies changes that were made to the [MS-RSVD] protocol document between the August 2013 and November 2013 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
2.2.5.2 SVHDX_OPEN_DEVICE_CONTEXT	69484 Clarified that the InitiatorId value is optional.	N	Content updated.
3.2.5.5.5 Receiving SCSI Command Request	69407 Updated the processing rules for the OperationCode field.	Y	Content updated.

8 Index