[MS-RSVD-Diff]:

Remote Shared Virtual Disk Protocol

Revision Summary

Date	Comments
04/28/15	Released Preview Document.

Table of Contents

1 Introduction

The Remote Shared Virtual Disk (RSVD) Protocol is a block-based protocol that is used to access ~~the~~ virtual ~~disk file~~disks in ~~the~~a shared fashion across a network over Server Message Block (SMB) Protocol version 3 (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 [RFC2119]. 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:

persistent reservation: A SCSI feature supporting control operations for shared devices ([SPC-3] section 5.6).

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.

virtual hard disk set (VHD set): A type of virtual disk that stores snapshot details.

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

1.2 References

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.

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

[MS-SRVS] Microsoft Corporation, "Server Service Remote Protocol".

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

1.2.2 Informative References

[~~MS-GLOS~~MSKB-3025091] Microsoft Corporation, "A shared Hyper-V virtual disk is inaccessible when it is located in Storage Spaces on a server that is running Windows ~~Protocols Master Glossary".~~Server 2012 R2", April 2015, https://support.microsoft.com/en-us/kb/3025091

[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

Note: Some of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure. For information about specific differences between versions, see the behavior notes that are provided in the Product Behavior appendix.

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.

The Remote Shared Virtual Disk Protocol version 2 additionally enables a client application to create and manage snapshots of shared virtual disk files.

1.4 Relationship to Other Protocols

This protocol depends on Server Message Block (SMB) Protocol version 3 (SMB3) for its transport, as specified in [MS-SMB2].

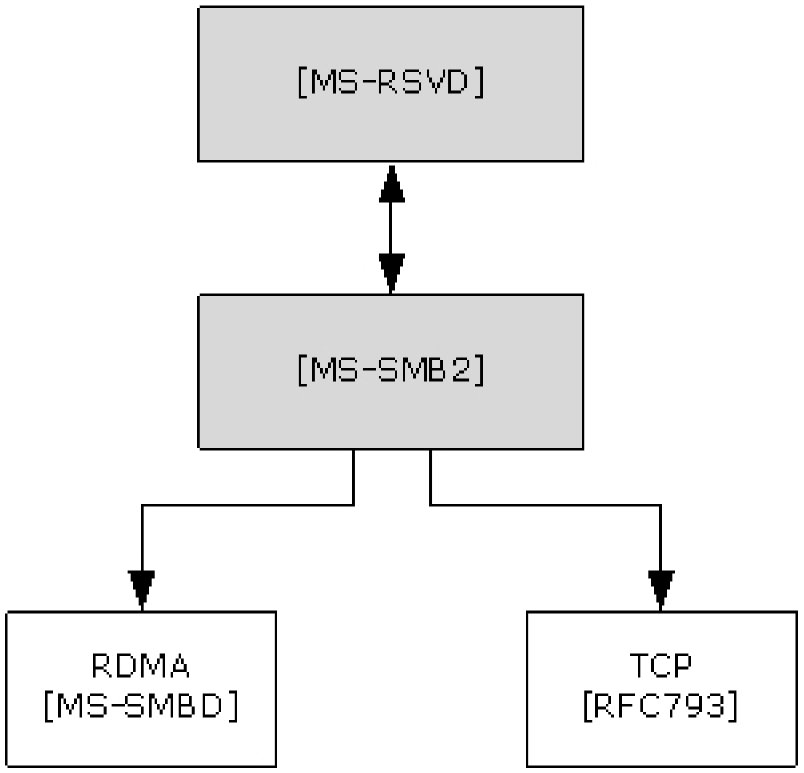

Figure 1: Relationship to other protocols

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 2.2.4.12.

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

Note: Some of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure. For information about specific differences between versions, see the behavior notes that are provided in the Product Behavior appendix.

The Remote Shared Virtual Disk Protocol ~~has a single version.<~~defines the following two versions.<1>

Version	Value
RSVD protocol version 1	0x00000001
RSVD protocol version 2	0x00000002

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

Note: Some of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure. For information about specific differences between versions, see the behavior notes that are provided in the Product Behavior appendix.

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 2.2.4.11. 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 3.0.2 or SMB 3.1.1 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	Meaning
RSVD_CDB_GENERIC_LENGTH 0x10	Generic length of command descriptor block
RSVD_SCSI_SENSE_BUFFER_SIZE 0x14	SENSE buffer size
RSVD_MAXIMUM_NAME_LENGTH 0x7E	Maximum length of the InitiatorHostName field in section 2.2.4.12
FSCTL_SVHDX_SYNC_TUNNEL_REQUEST 0x00090304	Control code for SYNC TUNNEL REQUEST
FSCTL_QUERY_SHARED_VIRTUAL_DISK_SUPPORT 0x00090300	Control code for querying shared virtual disk support
FSCTL_SVHDX_ASYNC_TUNNEL_REQUEST	Control code for ASYNC TUNNEL REQUEST

Constant name	Meaning
0x00090364	

2.2.2 Operation Codes

Note: Some of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure. For information about specific differences between versions, see the behavior notes that are provided in the Product Behavior appendix.

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

Name	Meaning
RSVD_TUNNEL_GET_~~FILE~~INITIAL_INFO_OPERATION 0x02001001~~0x001~~	Query shared virtual disk file and server information
RSVD_TUNNEL_SCSI_OPERATION ~~0x002~~0x02001002	Perform SCSI operation
RSVD_TUNNEL_CHECK_CONNECTION_STATUS_OPERATION ~~0x003~~0x02001003	Query shared virtual disk connection status
RSVD_TUNNEL_SRB_STATUS_OPERATION ~~0x004~~0x02001004	Query sense error code of previously failed request
RSVD_TUNNEL_GET_DISK_INFO_OPERATION ~~0x005~~0x02001005	Query disk information
RSVD_TUNNEL_VALIDATE_DISK_OPERATION ~~0x006~~0x02001006	Perform shared virtual disk validation
RSVD_TUNNEL_META_OPERATION_START 0x02002101	Start a meta-operation
RSVD_TUNNEL_META_OPERATION_QUERY_PROGRESS 0x02002002	Query the progress of an ongoing meta-operation
RSVD_TUNNEL_VHDSET_QUERY_INFORMATION 0x02002005	Query the information about a **virtual hard disk set (VHD set)**
RSVD_TUNNEL_DELETE_SNAPSHOT 0x02002006	Delete a previously created snapshot
RSVD_TUNNEL_CHANGE_TRACKING_GET_PARAMETERS 0x02002008	Get change-tracking parameters
RSVD_TUNNEL_CHANGE_TRACKING_START 0x02002009	Start change tracking
RSVD_TUNNEL_CHANGE_TRACKING_STOP 0x0200200A	Stop change tracking

2.2.3 Error Code

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

Name	Meaning
STATUS_SVHDX_ERROR_STORED 0xC05C0000	Sense error data was stored on server
STATUS_SVHDX_ERROR_NOT_AVAILABLE 0xC05CFF00	Sense error data is not available
STATUS_SVHDX_UNIT_ATTENTION_AVAILABLE 0xC05CFF01	Unit Attention data is available for the initiator to query
STATUS_SVHDX_UNIT_ATTENTION_CAPACITY_DATA_CHANGED 0xC05CFF02	The data capacity of the device has changed, resulting in a Unit Attention condition
STATUS_SVHDX_UNIT_ATTENTION_RESERVATIONS_PREEMPTED 0xC05CFF03	A previous operation resulted in this initiator's reservations being preempted, resulting in a Unit Attention condition
STATUS_SVHDX_UNIT_ATTENTION_RESERVATIONS_RELEASED 0xC05CFF04	A previous operation resulted in this initiator's reservations being released, resulting in a Unit Attention condition
STATUS_SVHDX_UNIT_ATTENTION_REGISTRATIONS_PREEMPTED 0xC05CFF05	A previous operation resulted in this initiator's registrations being preempted, resulting in a Unit Attention condition
STATUS_SVHDX_UNIT_ATTENTION_OPERATING_DEFINITION_CHANGED 0xC05CFF06	Represents the data storage format of the device has changed, resulting in a Unit Attention condition
STATUS_SVHDX_RESERVATION_CONFLICT 0xC05CFF07	The current initiator is not allowed to perform the SCSI command because of a reservation conflict
STATUS_SVHDX_WRONG_FILE_TYPE 0xC05CFF08	File on which open is performed is of wrong type
STATUS_SVHDX_VERSION_MISMATCH 0xC05CFF09	Protocol version in request is not equal to 1

2.2.4 Structures

2.2.4.1 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.

2.2.4.2 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.

2.2.4.3 SVHDX_TUNNEL_SRB_STATUS_REQUEST Structure

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

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

StatusKey (1 byte): The client MUST set this field to the least significant byte of the error code of a previously failed request.

Reserved (27 bytes): This field MUST be set to zero, and MUST be ignored on receipt.

2.2.4.4 SVHDX_TUNNEL_SRB_STATUS_RESPONSE Structure

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

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
StatusKey								A	SrbStatus							ScsiStatus								SenseInfoExLength							
SenseDataEx (variable)																															
...																															
...																															

StatusKey (1 byte): The server MUST set this field to the status key value received in the request.

A - SenseInfoAutoGenerated (1 bit): A 1-bit field used to indicate that **sense data** was automatically generated by the virtual **SCSI** disk.

SrbStatus (7 bits): A 7-bit field used to communicate error messages from the server to the client. This field MUST contain one of the values in section 2.2.5.

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.

SenseDataEx (variable): A buffer of maximum size 20 bytes that contains the sense data.

2.2.4.5 SVHDX_TUNNEL_DISK_INFO_REQUEST Structure

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

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Reserved1																															
...																															
BlockSize																															
LinkageID																															
...																															
...																															
...																															

IsMounted	Is4kAligned	Reserved2
FileSize		
...		
VirtualDiskId		
...		
...		
...		

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.

2.2.4.6 SVHDX_TUNNEL_DISK_INFO_RESPONSE Structure

The SVHDX_TUNNEL_DISK_INFO_RESPONSE packet is sent by the server in response to an ~~RSVD~~SVHDX_TUNNEL_~~VALIDATE_~~DISK_~~OPERATION~~INFO_REQUEST.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
DiskType																															
DiskFormat																															

BlockSize		
LinkageID		
...		
...		
...		
IsMounted	Is4kAligned	Reserved
FileSize		
...		
VirtualDiskId		
...		
...		
...		

DiskType (4 bytes): Indicates the type of disk. 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 format 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, in bytes, of the file opened as the shared virtual disk~~, in bytes~~.

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

2.2.4.7 SVHDX_TUNNEL_SCSI_REQUEST Structure

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

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Length																Reserved1															
CDBLength								SenseInfoExLength								DataIn								Reserved2							
SrbFlags																															
DataTransferLength																															
CDBBuffer																															
...																															
...																															
...																															
Reserved3																															
DataBuffer (variable)																															
...																															

Length (2 bytes): Specifies the size, in bytes, of the SVHDX_TUNNEL_SCSI_REQUEST structure excluding the **DataBuffer** field. The client MUST set this field to 36.

Reserved1 (2 byte): The client MUST set this field to zero.

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~~ This field indicates the SCSI command descriptor block transfer type~~. The value TRUE (0x01) indicates that~~ and MUST be set to one of 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.~~following values:

Value	**Meaning**
0x00	Indicates that the client is requesting data from the server
0x01	Indicates that the client is sending data to the server
0x02	Indicates that the client is neither sending nor requesting an additional data buffer

Reserved2 (1 byte): This field MUST be set to 0x00.

SrbFlags (4 bytes): An optional, application-provided flag to indicate the options of the SCSI request. This field MUST contain zero or more of the following values:

Name	**Meaning**
SRB_FLAGS_DATA_IN 0x00000040	The application is sending data to the server
SRB_FLAGS_DATA_OUT 0x00000080	The application is requesting data from the server

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 MUST be set to 0x00000000.

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

2.2.4.8 SVHDX_TUNNEL_SCSI_RESPONSE Structure

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

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Length																A	SrbStatus							ScsiStatus							

CDBLength	SenseInfoExLength	DataIn	Reserved
SrbFlags			
DataTransferLength			
SenseDataEx (variable)			
...			
...			
...			
DataBuffer (variable)			
...			

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

A - SenseInfoAutoGenerated (1 bit): A 1-bit field used to indicate that sense data was automatically generated by the virtual SCSI disk.

SrbStatus (1 byte): A 7-bit field used to communicate error messages from the server to the client. This field MUST contain one of the values in section 2.2.5.

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~~ This field indicates the SCSI command descriptor block ~~data~~ transfer type~~. TRUE (0x01), indicates that~~ and MUST be set to one of 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.~~following values:

Value	**Meaning**
0x00	Indicates that the client requested data from the server
0x01	Indicates that the client sent data to the server
0x02	Indicates that the client neither sent nor requested an additional data buffer

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

SrbFlags (4 bytes): Special flags to indicate options of the SCSI response. This field MUST contain zero or more of the following values:

Name	**Meaning**
SRB_FLAGS_DATA_IN 0x00000040	The client sent data to the server.
SRB_FLAGS_DATA_OUT 0x00000080	The server is sending data to the client.

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

SenseDataEx (variable): A buffer of maximum size 20 bytes that contains the ~~command descriptor buffer~~sense data.

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

2.2.4.9 SVHDX_TUNNEL_VALIDATE_DISK_REQUEST Structure

The **SVHDX_TUNNEL_VALIDATE_DISK_REQUEST** packet is sent by the client to validate the shared virtual disk. This request contains 56 bytes. ~~The client MUST set all 56 bytes to zero and the server MUST ignore them on receipt.~~

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Reserved (56 bytes)																															
...																															
...																															

Reserved (56 bytes): The client MUST set all 56 bytes to zero and the server MUST ignore them on receipt.

2.2.4.10 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.

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

IsValidDisk (1 byte): A Boolean value that, if set, indicates that the disk is valid.

2.2.4.11 SVHDX_TUNNEL_OPERATION_HEADER Structure

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

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
~~ProtocolId~~				~~ProtocolVersion~~					OperationCode																						
Status																															
RequestId																															
...																															

~~**ProtocolId (1 byte):** The protocol identifier. The client and server MUST set this value to 2.~~

~~**ProtocolVersion(12 bits):** The protocol version. The client and server MUST set this value to 1.~~

OperationCode (~~12 bits~~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.

2.2.4.12 SVHDX_OPEN_DEVICE_CONTEXT Structure

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

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Version																															
HasInitiatorId								Reserved																							
InitiatorId																															
...																															
...																															
...																															

Flags	
OriginatorFlags	
OpenRequestId	
...	
InitiatorHostNameLength	InitiatorHostName (126 bytes)
...	
...	

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 one of the following values:

Name	Meaning
SVHDX_ORIGINATOR_PVHDPARSER 0x00000001	Shared virtual disk file to be opened as a virtual SCSI disk device
SVHDX_ORIGINATOR_VHDMP 0x00000004	Shared virtual disk file to be opened in underlying object store

OpenRequestId (8 bytes): A 64-bit value assigned by the client for an outgoing request. The server MUST ignore it on receipt.

InitiatorHostNameLength (2 bytes): The length, in bytes, of the **InitiatorHostName**. This value MUST be less than or equal to RSVD_MAXIMUM_NAME_LENGTH.

InitiatorHostName (126 bytes): A 126-byte buffer containing a null-terminated Unicode UTF-16 string that specifies the computer name on which the initiator resides.

2.2.4.13 SVHDX_TUNNEL_~~FILE~~INITIAL_INFO_REQUEST Structure

The SVHDX_TUNNEL_~~FILE~~INITIAL_INFO_REQUEST packet is sent by the client to get the shared virtual disk file information. The request MUST contain only SVHDX_TUNNEL_OPERATION_HEADER, and MUST NOT contain any payload.

2.2.4.14 SVHDX_TUNNEL_~~FILE~~INITIAL_INFO_RESPONSE Structure

The SVHDX_TUNNEL_INITIAL_INFO_RESPONSE packet is sent by the server in response to an RSVD_TUNNEL_GET_~~FILE~~INITIAL_INFO_OPERATION packet.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
ServerVersion																															
SectorSize																															
PhysicalSectorSize																															
Reserved																															
VirtualSize																															
...																															

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

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

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

Reserved (4 bytes): This field MUST be set to zero, and the client MUST ignore it on receipt.

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

2.2.4.15 SVHDX_SHARED_VIRTUAL_DISK_SUPPORT_REQUEST Structure

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 NOT contain any payload.

2.2.4.16 SVHDX_SHARED_VIRTUAL_DISK_SUPPORT_RESPONSE Structure

Note: Some of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure. For information about specific differences between versions, see the behavior notes that are provided in the Product Behavior appendix.

The SVHDX_SHARED_VIRTUAL_DISK_SUPPORT_RESPONSE packet is sent by the server in a response to an ~~FSCTL_QUERY~~SVHDX_SHARED_VIRTUAL_DISK_SUPPORT_REQUEST request.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
SharedVirtualDiskSupport																															
SharedVirtualDiskHandleState																															

SharedVirtualDiskSupport (4 bytes): This field is used to indicate ~~whether~~the capabilities supported by the server~~ supports shared virtual disks.~~. This field MUST ~~be 0x00000001~~contain one of the following values.

Value	**Meaning**
SharedVirtualDiskSupported 0x00000001	The server supports shared virtual disks.
SharedVirtualDiskVer2OperationsSupported 0x00000003	The server supports shared virtual disks and version 2 operations.

SharedVirtualDiskHandleState (4 bytes): This field is used to ~~indicates~~indicate the state of the shared virtual disk Open. This field MUST contain one of the following values.

Value	Meaning
HandleStateNone 0x00000000	The Open is not opened as a shared virtual disk.
HandleStateFileShared 0x00000001	The shared virtual disk file is opened as a shared virtual disk by another Open.
HandleStateShared 0x00000003	The shared virtual disk file is opened as a shared virtual disk by this Open.

2.2.4.17 SVHDX_META_OPERATION_START_REQUEST Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_META_OPERATION_START_REQUEST** packet is sent by the client to start a meta-operation on the shared virtual disk file.

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

...
...
...
OperationType
Padding
Data (variable)
...

TransactionId (16 bytes): Indicates the transaction ID for the operation. The field MUST be set to a globally unique ID for each meta-operation.

OperationType (4 bytes): Indicates the type of the operation. This field MUST contain one of the following values.

Value	Meaning
SvhdxMetaOperationTypeCreateSnapshot 0x00000001	The meta-operation is part of a snapshot creation process.
SvhdxMetaOperationTypeOptimize 0x00000002	The meta-operation requests that the server optimize the target file.
SvhdxMetaOperationTypeExtractVHD 0x00000003	The meta-operation requests that the server extract a differencing VHD from the target file.
SvhdxMetaOperationTypeConvertToVHDSet 0x00000004	The meta-operation requests that the given virtual disk file be converted to a VHD set.
SvhdxMetaOperationTypeResize 0x00000005	The meta-operation requests to resize the target file.

Padding (4 bytes): This value MUST be set to 0 by the client and MUST be ignored by the server.

Data (variable): A variable-length field that contains the additional input specified by the **OperationType** field.

If the **OperationType** is **SvhdxMetaOperationTypeCreateSnapshot**, this field is provided in the format SVHDX_META_OPERATION_CREATE as specified in section 2.2.4.17.1.

If the **OperationType** is **SvhdxMetaOperationTypeExtractVHD**, this field is provided in the format SVHDX_META_OPERATION_EXTRACT as specified in section 2.2.4.17.2.

If the **OperationType** is **SvhdxMetaOperationTypeOptimize**, this field MUST be empty.

If the **OperationType** is **SvhdxMetaOperationTypeConvertToVHDSe**t, this field is provided in the format SVHDX_META_OPERATION_CONVERT_TO_VHDSET, as specified in section 2.2.4.17.3.

If the **OperationType** is **SvhdxMetaOperationTypeResize**, this field is provided in the format SVHDX_META_OPERATION _RESIZE_VIRTUAL_DISK as specified in section 2.2.4.17.4.

2.2.4.17.1 SVHDX_META_OPERATION_CREATE_SNAPSHOT Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The SVHDX_META_OPERATION_CREATE_SNAPSHOT structure is used to send the additional parameters for snapshot creation.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
SnapshotType																															
Flags																															
Stage1																															
Stage2																															
Stage3																															
Stage4																															
Stage5																															
Stage6																															
SnapshotId																															
...																															
...																															
...																															
ParametersPayloadSize																															
CdpParameters (Variable)																															

...

SnapshotType (4 bytes): The type of snapshot. This field MUST contain one of the values defined in section 2.2.6.

Flags (4 bytes): This field MUST be set to zero or a combination of the following values:

Name	Meaning
SVHDX_SNAPSHOT_DISK_FLAG_ENABLE_CHANGE_TRACKING 0x00000001	A request is made for change tracking to be enabled when snapshot is taken.

Stage1 (4 bytes): The first stage. This field MUST contain one of the following values.

Name	Meaning
SvhdxSnapshotStageInitialize 0x00000001	Perform any required initialization so that the appropriate type of snapshot can be taken.
SvhdxSnapshotStageBlockIO 0x00000002	Temporarily pause all IO against the target virtual device.
SvhdxSnapshotStageSwitchObjectStore 0x00000003	Switch aspects of the underlying object store so that the appropriate snapshot is taken.
SvhdxSnapshotStageUnblockIO 0x00000004	Allow further IO against the target virtual device.
SvhdxSnapshotStageFinalize 0x00000005	Tear down any state associated with a snapshot of the target virtual device.

Stage2 (4 bytes): The second stage. This field MUST be any of the values identified in **Stage1** or in **SvhdxSnapshotStageInvalid**, defined as follows.

Name	Meaning
SvhdxSnapshotStageInvalid 0x00000000	No stage present in this field.

Stage3 (4 bytes): The third stage. This field MUST be any of the values identified in **Stage1** or **SvhdxSnapshotStageInvalid.**

Stage4 (4 bytes) The fourth stage. This field MUST be any of the values identified in **Stage1** or **SvhdxSnapshotStageInvalid.**

Stage5 (4 bytes): The fifth stage. This field MUST be any of the values identified in **Stage1** or **SvhdxSnapshotStageInvalid.**

Stage6 (4 bytes): The sixth stage. This field MUST be any of the values identified in **Stage1** or **SvhdxSnapshotStageInvalid.**

SnapshotId (16 bytes): The ID to assign to the snapshot.

ParametersPayloadSize (4 bytes): The size of any parameters included with the create snapshot request. If the **SnapshotType** is SvhdxSnapshotTypeVM, then this field MUST be set to 0. If the **SnapshotType** is **SvhdxSnapshotTypeCDP**, then this field MUST NOT be set to 0.

CdpParameters (variable): Parameters supplied with the **continuous data protection snapshot (CDP snapshot)** operation. This field is provided in the format SVHDX_META_OPERATION_CREATE_CDP_PARAMETER as specified in section 2.2.4.17.1.1.

2.2.4.17.1.1 SVHDX_META_OPERATION_CREATE_CDP_PARAMETER Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

SVHDX_META_OPERATION_CREATE_CDP_PARAMETER is used to send additional CDP parameters.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
LogFileNameOffset																															
LogFileId																															
...																															
...																															
...																															
LogFileName(Variable)																															
...																															

LogFileNameOffset (4 bytes): The offset, in bytes, of the **LogFileName** from the **CdpParameters** field specified in section 2.2.4.17.1.

LogFileId (16 Bytes): A GUID associated with the log file.

LogFileName (Variable): A log file name containing a null-terminated Unicode UTF-16 string.

2.2.4.17.2 SVHDX_META_OPERATION_EXTRACT Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_META_OPERATION_EXTRACT** packet is issued by a server as part of a **SVHDX_META_OPERATION_START_REQUEST** operation, when the meta-operation type indicates an **SvhdxMetaOperationTypeExtractVHD**. The server issues such a request to export the differences between different VM snapshots.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
SnapshotType																															
Padding																															
Flags																															
SourceSnapshotId																															
...																															
...																															
...																															
SourceLimitSnapshotId																															
...																															
...																															
...																															
DestinationVhdSetNameLength																															
DestinationVhdSetName (Variable)																															
...																															

SnapshotType (4 bytes): The type of snapshot. This field MUST contain one of the values defined in section 2.2.6.

Padding (4 bytes): This value MUST be set to 0 by the client and MUST be ignored by the server.

Flags (4 bytes): This field MUST be set to zero or more of the following values:

Name	**Meaning**
SVHDX_EXTRACT_SNAPSHOTS_FLAG_DELETE_ON_CLOSE 0x00000001	Request for the VHD set to be deleted on close after exporting the differences between specified snapshots.

SourceSnapshotId (16 Bytes): A GUID that indicates the ID of the first snapshot to include in the extract request.

SourceLimitSnapshotId (16 Bytes): A GUID that indicates the ID of the last snapshot to include in the extract request.

DestinationVhdSetNameLength (4 Bytes): The length, in bytes, including the NULL terminating character, of the VHD set name to extract the data differences to.

DestinationVhdSetName (Variable): A buffer containing a null-terminated Unicode UTF-16 string that indicates the VHD set name to extract the data differences to.

2.2.4.17.3 SVHDX_META_OPERATION_CONVERT_TO_VHDSET Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The SVHDX_CONVERT_TO_VHDSET packet is issued by a server as part of a SVHDX_META_OPERATION_START_REQUEST operation, when the meta-operation type indicates a **SvhdxMetaOperationTypeConvertToVHDSet**. The server issues such a request to convert a VHD file to a snapshot-capable VHD set.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
DestinationVhdSetNameLength																															
DestinationVhdSetName (Variable)																															
...																															

DestinationVhdSetNameLength (4 Bytes): The length, in bytes, including the NULL terminating character, of the **DestinationVhdSetName**.

DestinationVhdSetName (Variable): A buffer containing a null-terminated Unicode UTF-16 string that indicates the name for the new VHD set that is to be created.

2.2.4.17.4 SVHDX_META_OPERATION_RESIZE_VIRTUAL_DISK Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The SVHDX_CONVERT_TO_VHDSET packet is issued by a server as part of a SVHDX_META_OPERATION_START_REQUEST operation, when the meta-operation type indicates a

SvhdxMetaOperationTypeResize. The server issues such a request to resize the shared virtual disk.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
NewSize																															
...																															
ExpandOnly								AllowUnsafeVirtualSize								ShrinkToMinimumSafeSize								Reserved							

NewSize (8 bytes): This specifies the new size of the shared virtual disk.

ExpandOnly (1 byte): A nonzero value indicates that the shared virtual disk size can only expand.

AllowUnsafeVirtualSize (1 byte): A nonzero value indicates that the shared virtual disk size can be less than the data it currently contains.

ShrinkToMinimumSafeSize (1 byte): A nonzero value indicates that the shared virtual disk size can be shrunk to the data it currently contains.

Reserved (1 byte): This value MUST be set to 0 by the client and MUST be ignored by the server.

2.2.4.18 SVHDX_META_OPERATION_REPLY Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_META_OPERATION_REPLY** packet is issued by a server in reply to an **SVHDX_META_OPERATION_START_REQUEST** operation. When the meta-operation type indicates an **SvhdxMetaOperationTypeCreateSnapshot** with snapshot type **SvhdxSnapshotTypeCDP**, this structure is returned. Otherwise, this structure is not present.

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

ChangeTrackingErrorStatus (4 bytes): Indicates an error condition when attempting to perform change-tracking operations. This MUST be one of the following values.

Name	Meaning
SVHDX_TUNNEL_CHANGE_TRACKING_STATUS_SUCCESS 0x00000000	Change-tracking is active on the virtual disk without any error.
SVHDX_TUNNEL_CHANGE_TRACKING_NOT_INITIALIZED 0xC03A0020	Change-tracking is not initialized on the virtual disk.

Name	**Meaning**
SVHDX_TUNNEL_CHANGE_TRACKING_LOGSIZE_EXCEEDED_MAXSIZE_ 0xC03A0021	The log file size has exceeded the client specified maximum size.
SVHDX_TUNNEL_CHANGE_TRACKING_VHD_CHANGED_OFFLINE_ 0xC03A0022	A virtual disk write was detected to be missing from the current log file.
SVHDX_TUNNEL_CHANGE_TRACKING_INVALID_TRACKING_STATE_ 0xC03A0023	A change-tracking operation cannot be performed on the virtual disk in its current state.
SVHDX_TUNNEL_CHANGE_TRACKING_INCONSISTENT_TRACKING_FILE_ 0xC03A0024	An inconsistent log file detected.

2.2.4.19 SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_REQUEST Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_REQUEST** packet is sent by a client as part of an **RSVD_TUNNEL_VHDSET_QUERY_INFORMATION** request.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
VHDSetInformationType																															
SnapshotType																															
SnapshotId																															
...																															
...																															
...																															

VHDSetInformationType (4 bytes): The information type requested. This MUST be one of the following values.

Name	**Meaning**
SvhdxVHDSetInformationTypeSnapshotList 0x00000002	Returns a list of snapshots.
SvhdxVHDSetInformationTypeSnapshotEntry 0x00000005	Returns the details about a specific snapshot entry.
SvhdxVHDSetInformationTypeOptimizeNeeded 0x00000008	Queries whether the target VHD set needs optimization.
SvhdxVHDSetInformationTypeCdpSnapshotRoot 0x00000009	Returns the oldest CDP snapshot present in the target VHD set.
SvhdxVHDSetInformationTypeCdpSnapshotActiveList 0x0000000A	Returns the list of CDP snapshot IDs that are active for the VHD set.
SvhdxVHDSetInformationTypeCdpSnapshotInactiveList 0x0000000C	Returns the list of CDP snapshot IDs that are inactive in the VHD set.

SnapshotType (4 bytes): The snapshot type queried by this operation. This MUST be one of the **SvhdxSnapshotType** values identified.

SnapshotId (16 bytes): The snapshot ID relevant to the particular request.

2.2.4.20 SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_SNAPSHOT_LIST_RESPONSE Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_SNAPSHOT_LIST_RESPONSE** packet is sent by a server in response to an **RSVD_TUNNEL_VHDSET_QUERY_INFORMATION** request where the VHDSetInformationType is **SvhdxVHDSetInformationTypeSnapshotList**.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
VHDSetInformationType																															
Padding																															
SnapshotsFilled								Reserved																							
NumberOfSnapshots																															
SnapshotIds (Variable)																															
...																															

VHDSetInformationType (4 bytes): The information type. The server MUST set this to SvhdxVHDSetInformationTypeSnapshotList.

Padding (4 bytes): This field is set to any value by the server and MUST be ignored by the client.

SnapshotsFilled (1 byte): Indicates if the reply contains snapshot IDs in the **SnapshotIds** field. Zero (0x00) indicates that the **SnapshotIds** field is not present. One (0x01) indicates that the **SnapshotIds** field is present.

Reserved (3 bytes): This field MUST be set to 0 by the server and MUST be ignored by the client.

NumberOfSnapshots (4 bytes): The number of snapshots contained in the **SnapshotIds** field.

SnapshotIds (Variable): A list of IDs of snapshots of a particular type. If the **SnapshotsFilled** field is TRUE, then this field contains a list of GUIDs that define each snapshot ID. The number of snapshot IDs in the field is specified by the **NumberOfSnapshots** fields.

2.2.4.21 SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_SNAPSHOT_ENTRY_RESPONSE Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_SNAPSHOT_ENTRY_RESPONSE** packet is sent by a server in response to an **RSVD_TUNNEL_VHDSET_QUERY_INFORMATION** request where the **VHDSetInformationType** is **SvhdxVHDSetInformationTypeSnapshotEntry**.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
VHDSetInformationType																															
SnapshotCreationTime																															
...																															
SnapshotType																															
IsValidSnapshot																															
SnapshotId																															
...																															
...																															

...
ParentSnapshotId
...
...
...
LogFileId
...
...
...

VHDSetInformationType (4 bytes): The information type. The server MUST set this to SvhdxSetInformationTypeSnapshotEntry.

SnapshotCreationTime (8 bytes): The time, in milliseconds since Jan 1, 1970, when the snapshot was created.

SnapshotType (4 bytes): The type of snapshot. This MUST be one of the **SvhdxSnapshotType** values defined in section 2.2.6.

IsValidSnapshot (4 bytes): Set to 1 when the snapshot is valid; set to 0 when the snapshot is invalid.

SnapshotId (16 bytes): The globally unique ID of the snapshot.

ParentSnapshotId (16 bytes): The parent snapshot ID. This field will be set for CDP snapshots.

LogFileId (16 bytes): The ID of the log file associated with this snapshot. This field will be set for CDP snapshots.

2.2.4.22 SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_OPTIMIZE_RESPONSE Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_OPTIMIZE_RESPONSE** packet is sent by a server in response to an **RSVD_TUNNEL_VHDSET_QUERY_INFORMATION** request where the **VHDSetInformationType** is **VHDSvhdxSetInformationTypeOptimizeNeeded**.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
VHDSetInformationType																															
OptimizeNeeded																															

VHDSetInformationType (4 bytes): The information type. The server MUST set this to **SvhdxVHDSetInformationTypeOptimizeNeeded**.

OptimizeNeeded (4 bytes): Indicates whether optimization is needed for the target VHD set.

2.2.4.23 SVHDX_META_OPERATION_QUERY_PROGRESS_REQUEST Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_META_OPERATION_QUERY_PROGRESS_REQUEST** packet is sent by the client to query the progress of an ongoing meta-operation.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
TransactionId																															
...																															
...																															
...																															

TransactionId (16 bytes): Indicates the transaction ID for the operation. This is the transaction ID used in the RSVD_TUNNEL_META_OPERATION_START request.

2.2.4.24 SVHDX_META_OPERATION_QUERY_PROGRESS_RESPONSE Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_META_OPERATION_QUERY_PROGRESS_RESPONSE** packet is sent by the server in response to a RSVD_TUNNEL_META_OPERATION_QUERY_PROGRESS request.

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

...
CompleteValue
...

CurrentProgressValue (8 bytes): A server-defined progress value that indicates how far along the meta-operation has proceeded.

CompleteValue (8 bytes): The maximum progress value for the completed operation. That is, when **CurrentProgressValue** is equal to **CompleteValue**, the operation is complete.

2.2.4.25 SVHDX_CHANGE_TRACKING_GET_PARAMETERS_RESPONSE Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_CHANGE_TRACKING_GET_PARAMETERS_RESPONSE** packet is sent by the server in response to a RSVD_TUNNEL_CHANGE_TRACKING_GET_PARAMETERS operation.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
ChangeTrackingStatus																															
...																															
LogFileSize																															
...																															

ChangeTrackingStatus (8 bytes): The current status of change tracking on the server. This MUST be one of the error codes specified in section 2.2.3.

LogFileSize (8 bytes): The number of bytes consumed by the log file used to conduct change tracking.

2.2.4.26 SVHDX_TUNNEL_DELETE_SNAPSHOT_REQUEST Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_TUNNEL_DELETE_SNAPSHOT_REQUEST** packet is sent by a client to delete a snapshot from a shared virtual disk file.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
SnapshotId																															
...																															
...																															
...																															
PersistReference																															
SnapshotType																															

SnapshotId (16 bytes): The snapshot ID relevant to the particular request.

PersistReference (4 bytes): A flag to indicate if the snapshot needs to be persisted. One represents TRUE (0x00000001), indicating that the snapshot will be stored as a reference without any data. Zero represents FALSE (0x00000000), indicating that the snapshot will be deleted. This field MUST be set to FALSE if **SnapshotType** is **SvhdxSnapshotTypeVM**.

SnapshotType (4 bytes): The type of snapshot. This MUST be one of the **SvhdxSnapshotType** values defined in section 2.2.6.

2.2.4.27 SVHDX_CHANGE_TRACKING_START_REQUEST Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_CHANGE_TRACKING_START_REQUEST** packet is sent by the client to start change tracking on the server. The packet contains the following fields.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
TransactionId																															
...																															
...																															
...																															
LogFileNameLength																															

LogFileId
...
...
...
MaxLogFileSize
AppendData
LogFileName (Variable)
...

TransactionId (16 bytes): The client MUST set this to a globally unique transaction ID for each operation.

LogFileNameLength (4 bytes): The length, in bytes, of the name string. This length MUST include a NULL terminating character.

LogFileId (16 bytes): A globally unique Id used to refer to this log file.

MaxLogFileSize (8 bytes): The maximum allowed size, in bytes, of the underlying object store for this particular change-tracking log file.

AppendData (4 byte): When set to TRUE, change tracking will be resumed, and further tracked data will be appended to the existing log file. The client should set this to FALSE for the first change-tracking request issued against the target device.

LogFileName (Variable): The name of the log file containing a null-terminated Unicode UTF-16 string.

2.2.4.28 SVHDX_CHANGE_TRACKING_START_RESPONSE Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_CHANGE_TRACKING_START_RESPONSE** packet is sent by the server in response to an RSVD_TUNNEL_CHANGE_TRACKING_START operation. This response MUST NOT contain any payload.

2.2.4.29 SVHDX_CHANGE_TRACKING_STOP_REQUEST Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_CHANGE_TRACKING_STOP_REQUEST** packet is sent by the client to stop change tracking on the server. This request MUST NOT contain any payload.

2.2.4.30 SVHDX_CHANGE_TRACKING_STOP_RESPONSE Structure

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The **SVHDX_CHANGE_TRACKING_STOP_RESPONSE** packet is sent by the server in response to an RSVD_TUNNEL_CHANGE_TRACKING_STOP operation. This response MUST NOT contain any payload.

2.2.4.31 SVHDX_OPEN_DEVICE_CONTEXT_RESPONSE Structure

The SVHDX_OPEN_DEVICE_CONTEXT_RESPONSE packet is sent by the server in response to open the shared virtual disk request.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Version																															
HasInitiatorId								Reserved																							
InitiatorId																															
...																															
...																															
...																															
Flags																															
OriginatorFlags																															
OpenRequestId																															
...																															
InitiatorHostNameLength																InitiatorHostName (126 bytes)															
...																															
...																															

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 one of the following values.

Name	Meaning
SVHDX_ORIGINATOR_PVHDPARSER 0x00000001	Shared virtual disk file to be opened as a virtual SCSI disk device
SVHDX_ORIGINATOR_VHDMP 0x00000004	Shared virtual disk file to be opened in underlying object store

OpenRequestId (8 bytes): A 64-bit value assigned by the client for an outgoing request. The server MUST ignore it on receipt.

InitiatorHostNameLength (2 bytes): The length, in bytes, of the **InitiatorHostName**. This value MUST be less than or equal to RSVD_MAXIMUM_NAME_LENGTH.

InitiatorHostName (126 bytes): A 126-byte buffer containing a null-terminated Unicode UTF-16 string that specifies the computer name which initiated the request.

2.2.5 SRB Status Code

The following is a list of possible SRB error messages used to communicate from server to client.

Value	Meaning
0x01	The request was completed successfully.
0x02	The request was aborted.
0x06	The Shared Virtual Disk does not support the given request.
0x08	The Shared Virtual Disk device is no longer available.
0x0A	The SCSI device selection timed out.
0x12	A data overrun or underrun error occurred.
0x04	The request completed with any other error.

2.2.6 Snapshot Types

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

Following are the possible snapshot types.

Name	**Meaning**
SvhdxSnapshotTypeVM 0x01	A snapshot created as part of routine Virtual Machine operations.
SvhdxSnapshotTypeCDP 0x03	A snapshot created as part of a continuous data protection (CDP) process.

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.
- A Boolean that, if set, requires the shared virtual disk file to be opened as virtual SCSI disk.

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 2.2.4.12 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.
- If the application requires the shared virtual disk file to be opened as a virtual SCSI disk device, the client MUST set **OriginatorFlags** to SVHDX_ORIGINATOR_PVHDPARSER. Otherwise, it MUST be set to SVHDX_ORIGINATOR_VHDMP.
- The **OpenRequestId** is set to **RequestIdentifier**.
- The **InitiatorHostNameLength** is set to **InitiatorHostName** length, in bytes.
- The **InitiatorHostName** is set to the application-provided initiator name. If **InitiatorHostNameLength** is less than 126 bytes, the remaining bytes in the buffer MUST be set to zero.

The client MUST append ":SharedVirtualDisk" at the end of the file name.

The client MUST establish a connection to the server by calling the interface specified in [MS-SMB2] section 3.2.4.2 and providing the following input parameters:

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

If the connection is successfully established, the client MUST open the shared virtual disk file by calling the interface specified in [MS-SMB2] section 3.2.4.3 and provide the following input parameters:

- File name
- The SVHDX_OPEN_DEVICE_CONTEXT Create context.
- **CreateOptions** as specified in [MS-SMB2] section 2.2.13 with the FILE_NO_INTERMEDIATE_BUFFERING bit set.

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_INITIAL ~~FILE~~_INFO_REQUEST structure, as specified in section 2.2.4.13 as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** MUST be set to RSVD_TUNNEL_GET_INITIAL_INFO_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.
- 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 2.2.4.1 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 2.2.4.5, 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 client MUST call the interface specified in [MS-SMB2] section 3.2.4.20.11, supplying the following input parameters:

~~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.~~
- 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.
- The ID of the initiator.
- The length of the SCSI Command Descriptor Block (CDB) buffer, in bytes.
- The length of the sense information, in bytes.
- ~~A Boolean value indicating the SCSI CDB transfer type, as specified in section 2.2.4.3.~~
- ~~SCSI command flags~~SrbFlags that indicate options about the SCSI request~~.~~, as specified in section 2.2.4.7<4>.
- The SCSI CDB buffer.
- The length of any additional data (optional).
- Any additional data buffer (optional).

The client MUST construct an SVHDX_TUNNEL_SCSI_REQUEST structure, as specified in section 2.2.4.7 as 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 initialized as follows:

- The **Length** field is set to 36.
- The **Reserved1**, **Reserved2**, and **Reserved3** fields are set to zero.
- The **CDBLength** is set to the application-provided CDB length value.

- The **SenseInfoExLength** set to the application-provided sense info length value.
- ~~The **DataIn** value is set to the application-provided Boolean value.~~
- The **SrbFlags** field MUST be set to the application-provided ~~special flag values.~~ **SrbFlags**.
- If **SrbFlags** includes SRB_FLAGS_DATA_OUT, set **DataIn** to 0x00. If **SrbFlags** includes SRB_FLAGS_DATA_IN, **DataIn** SHOULD<5> be set to 0x01. If **SrbFlags** includes neither SRB_FLAGS_DATA_OUT nor SRB_FLAGS_DATA_IN, set **DataIn** to 0x02.
- The **DataTransferLength** field MUST be set to the application-provided value for the **Length** of the additional 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.
- 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 2.2.4.9, 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

3.1.4.12 Application Requests Creating a Virtual Machine Snapshot

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a VHD set.
- **TransactionId**: A GUID used to uniquely identify the start operation.
- Stage 1 value
- Stage 2 value
- Stage 3 value
- Stage 4 value
- Stage 5 value
- Stage 6 value
- **SnapshotId**: A GUID to uniquely identify the snapshot.

The client MUST construct an SVHDX_META_OPERATION_START_REQUEST structure, as specified in section 2.2.4.17 as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

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

The SVHDX_META_OPERATION_START_REQUEST MUST be initialized as follows:

- The **TransactionId** field is set to the application-provided **TransactionId**.
- The **OperationType** field is set to **SvhdxMetaOperationTypeCreateSnapshot**.
- The **Data** field is set to an SVHDX_META_OPERATION_CREATE_SNAPSHOT structure initialized as follows:
  - The **SnapshotType** field is set to the snapshot type provided by the application
  - The **Stage1** field is set to the stage 1 value provided by the application.
  - The **Stage2** field is set to the stage 2 value provided by the application.

- The **Stage3** field is set to the stage 3 value provided by the application.
- The **Stage4** field is set to the stage 4 value provided by the application.
- The **Stage5** field is set to the stage 5 value provided by the application.
- The **Stage6** field is set to the stage 6 value provided by the application.
- The **SnapshotId** field is set to the SnapshotId provided by the application.
- The **ParametersPayloadSize** is set to 0**.**

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_ASYNC_TUNNEL_REQUEST.
- SVHDX_META_OPERATION_START_REQUEST structure as payload.

3.1.4.13 Application Requests Creating a CDP Snapshot

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a VHD set.
- **TransactionId**: A GUID used to uniquely identify the start operation.
- Stage 1 value
- Stage 2 value
- Stage 3 value
- Stage 4 value
- Stage 5 value
- Stage 6 value
- **SnapshotId**: A GUID to uniquely identify the snapshot.
- **LogFileID**: A GUID to identify the log file.
- **FileName**: A log file name.

The client MUST construct an SVHDX_META_OPERATION_START_REQUEST structure, as specified in section 2.2.4.17, as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** MUST be set to RSVD_TUNNEL_META_OPERATION_START.
- The **Status** field MUST be set to zero.

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

The SVHDX_META_OPERATION_START_REQUEST MUST be initialized as follows:

- The **TransactionId** field is set to the application-provided **TransactionId**.
- The **OperationType** field is set to **SvhdxMetaOperationTypeCreateSnapshot**.
- The **Data** field is set to an SVHDX_META_OPERATION_CREATE_SNAPSHOT structure initialized as follows:
  - The **SnapshotType** field is set to **SvhdxSnapshotTypeCDP**.
  - The **Stage1** field is set to the stage 1 value provided by the application.
  - The **Stage2** field is set to the stage 2 value provided by the application.
  - The **Stage3** field is set to the stage 3 value provided by the application.
  - The **Stage4** field is set to the stage 4 value provided by the application.
  - The **Stage5** field is set to the stage 5 value provided by the application.
  - The **Stage6** field is set to the stage 6 value provided by the application.
  - The **SnapshotId** field is set to the **SnapshotId** provided by application.
  - The **ParametersPayloadSize** is set to the size, in bytes, of the **CdpParameters** field.
  - The **CdpParameters** field is set to an SVHDX_META_OPERATION_CREATE_CDP_PARAMETER structure initialized as follows:
    - The **LogFileId** field is set to the **LogFileID** provided by the application.
    - The **LogFileName** field is set to the **FileName** provided by the application.

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_ASYNC_TUNNEL_REQUEST.
- SVHDX_META_OPERATION_START_REQUEST structure as payload.

3.1.4.14 Application Requests Optimizing the Target VHD set

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a VHD set.
- **TransactionId**: A GUID used to uniquely identify the start operation.

The client MUST construct an SVHDX_META_OPERATION_START_REQUEST structure, as specified in section 2.2.4.17, as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

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

The SVHDX_META_OPERATION_START_REQUEST MUST be initialized as follows:

- The **TransactionId** field is set to the application-provided **TransactionId**.
- The **OperationType** field is set to **SvhdxMetaOperationTypeOptimize**.

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_ASYNC_TUNNEL_REQUEST.
- SVHDX_META_OPERATION_START_REQUEST structure as payload.

3.1.4.15 Application Requests Extracting a Differencing VHD

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a VHD set.
- **TransactionId**: A GUID used to uniquely identify the start operation.
- **SnapshotType**: The type of the snapshot.
- **Flags**: The flags to be passed to the server.
- **SnapshotId**: A GUID to identify the first snapshot data to include in the extract.
- **SnapshotLimitId**: A GUID to identify the last snapshot data to include in the extract.
- **DestinationVhdSetName**: A VHD set name into which the difference data will be written.

The client MUST construct an SVHDX_META_OPERATION_START_REQUEST structure, as specified in section 2.2.4.17, as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

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

The SVHDX_META_OPERATION_START_REQUEST MUST be initialized as follows:

- The **TransactionId** field is set to the application-provided **TransactionId**.

- The **OperationType** field is set to **SvhdxMetaOperationTypeExtractVHD**.
- The **Data** field to a **SVHDX_META_OPERATION_EXTRACT** structure initialized as follows:
  - The **SnapshotType** field is set to the **SnapshotType** provided by the application.
  - The **Flags** field is set to the **Flags** provided by the application.
  - The **SourceSnapshotId** field is set to the **SnapshotId** provided by the application.
  - The **SourceLimitSnapshotId** field is set to the **SnapshotLimitId** provided by the application.
  - The **DestinationVhdSetNameLength** field is set to the size, in bytes, of **DestinationVhdSetName**.
  - The **DestinationVhdSetName** field is set to the **DestinationVhdSetName** provided by the application.

  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_ASYNC_TUNNEL_REQUEST.
  - SVHDX_META_OPERATION_START_REQUEST structure as payload.

3.1.4.16 Application Requests Converting a Virtual Disk to VHD Set

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a shared virtual disk.
- **TransactionId**: A GUID used to uniquely identify the start operation.
- **DestinationVhdSetName**: A name for the VHD set.

The client MUST construct an SVHDX_META_OPERATION_START_REQUEST structure, as specified in section 2.2.4.17, as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

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

The SVHDX_META_OPERATION_START_REQUEST MUST be initialized as follows:

- The **TransactionId** field is set to the application-provided **TransactionId**.
- The **OperationType** field is set to **SvhdxMetaOperationTypeExtractVHD**.

- The **Data** field to a SVHDX_META_OPERATION_CONVERT_TO_VHDSET structure is initialized as follows:
  - The **DestinationVhdSetNameLength** field is set to the size, in bytes, of **DestinationVhdSetName**.
  - The **DestinationVhdSetName** field is set to the **DestinationVhdSetName** provided by the application.

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_ASYNC_TUNNEL_REQUEST.
- SVHDX_META_OPERATION_START_REQUEST structure as payload.

3.1.4.17 Application Requests Resizing a Shared Virtual Disk

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a shared virtual disk.
- **TransactionId**: A GUID used to uniquely identify the start operation.
- **NewVirtualSize**: The new size of the virtual disk.
- **ExpandOnly**: Indicates that a size can only be expanded.
- **AllowUnsafeVirtualSize**: Indicates that the new size can be less than the data on the disk.
- **ShrinkToMinimumSafeSize**: Indicates that the server should automatically shrink the virtual disk to the minimum safe virtual size.

The client MUST construct an SVHDX_META_OPERATION_START_REQUEST structure, as specified in section 2.2.4.17, as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

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

The SVHDX_META_OPERATION_START_REQUEST MUST be initialized as follows:

- The **TransactionId** field is set to the application-provided **TransactionId**.
- The **OperationType** field is set to **SvhdxMetaOperationTypeExtractVHD**.
- The **Data** field to a SVHDX_META_OPERATION_ RESIZE_VIRTUAL_DISK structure is initialized as follows:

- The **NewSize** field is set to the **NewVirtualSize** provided by the application.
- The **ExpandOnly** field is set to the **ExpandOnly** provided by the application.
- The **AllowUnsafeVirtualSize** field is set to **AllowUnsafeVirtualSize** provided by the application.
- The **ShrinkToMinimumSafeSize** field is set to **ShrinkToMinimumSafeSize** provided by the application.

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_ASYNC_TUNNEL_REQUEST.
- SVHDX_META_OPERATION_START_REQUEST structure as payload.

3.1.4.18 Application Requests Querying Meta Operation Progress

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a VHD set.
- **TransactionId**: A GUID used to uniquely identify the query operation.

The client MUST construct an SVHDX_META_OPERATION_QUERY_PROGRESS_REQUEST structure, as specified in section 2.2.4.23, as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

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

The SVHDX_META_OPERATION_QUERY_PROGRESS_REQUEST MUST be initialized as follows:

- The **TransactionId** is set to the **TransactionId** provided by the application.

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_META_OPERATION_QUERY_PROGRESS_REQUEST structure as payload.

3.1.4.19 Application Requests Querying VHD Set Information

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a VHD set.
- **InformationType**: The type of information requested by the client.
- **SnapshotType**: The type of the snapshot.
- **SnapshotId**: A snapshot identifier relevant to the handle.

The client MUST construct an SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_REQUEST structure, as specified in section 2.2.4.22, as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

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

The SVHDX_TUNNEL_VHDSET_ QUERY_INFORMATION_REQUEST MUST be initialized as follows:

- The **VHDSetInformationType** is set to the **InformationType** provided by the application.
- The **SnapshotType** field is set to the **SnapshotType** provided by the client.
- The **SnapshotId** field is set to the **SnapshotId** provided by the application.

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_VHDSET_ QUERY_INFORMATION_REQUEST structure as payload.

3.1.4.20 Application Requests Deleting a Snapshot

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a VHD set.
- **SnapshotID**: An identifier of the snapshot to be deleted.
- **PersistReference**: A flag; if TRUE, indicates that the snapshot reference should be persisted.
- **SnapshotType**: Type of the snapshot.

The client MUST construct an SVHDX_TUNNEL_DELETE_SNAPSHOT_REQUEST structure, as specified in section 2.2.4.26, as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

- The **OperationCode** MUST be set to RSVD_TUNNEL_DELETE_SNAPSHOT

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

The SVHDX_TUNNEL_DELETE_SNAPSHOT_REQUEST MUST be initialized as follows:

- The **SnapshotId** is set to the **SnapshotID** provided by the application.
- The **PersistReference** is set to the **PersistReference** provided by the application.
- The **SnapshotType** is set to the **SnapshotType** provided by the application.

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_DELETE_SNAPSHOT_REQUEST structure as payload.

3.1.4.21 Application Requests Querying Change Tracking Parameters

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a VHD set.

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

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

3.1.4.22 Application Requests Starting a Change Tracking Operation

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a VHD set.
- **TransactionId**: An identifier of the snapshot to be deleted.

- **LogFileId**: A GUID that identifies the log file.
- **MaxLogFileSize**: Maximum size of the log file.
- **AppendData**: A flag; if TRUE, indicates that the tracked data will be appended to the existing log file.
- **LogFileName**: A log file name.

The client MUST construct an **SVHDX_CHANGE_TRACKING_START_REQUEST** structure, as specified in section 2.2.4.27, as follows:

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

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

The **SVHDX_CHANGE_TRACKING_START_REQUEST** MUST be initialized as follows:

- The **TransactionId** is set to the **TransactionId** provided by the application.
- The **LogFileNameLength** is set to the size, bytes, of the application-provided LogFileName.
- The **MaxLogFileSize** is set to the **MaxLogFileSize** provided by the application.
- The **AppendData** is set to the **AppendData** provided by the application.
- The **LogFileName** is set to the **LogFileName** provided by the application.

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_CHANGE_TRACKING_START_REQUEST structure as payload.

3.1.4.23 Application Requests Stopping a Change Tracking Operation

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The application provides:

- A handle to the **Open** identifying a VHD set.

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized as follows:

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

3.1.5 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 byte of the received error code.
- The **Reserved** 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 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 byte of the received error code.
- The **Reserved** 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 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 a 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 a Connection Status Response

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

3.1.5.7 Receiving a Shared Virtual Disk Information ~~Request~~Response

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 a 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 a Validate Disk Response

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

3.1.5.10 Receiving a Shared Virtual Disk Support Response

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

3.1.5.11 Receiving a Meta-Operation Start Response

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

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

3.1.5.12 Receiving a Meta-Operation Progress Response

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

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.

3.1.5.13 Receiving a Query VHD Set Information Response

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

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.

3.1.5.14 Receiving a Delete Snapshot Response

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

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

3.1.5.15 Receiving a Change Tracking Parameter Response

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

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.

3.1.5.16 Receiving a Start Change Tracking Response

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

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

3.1.5.17 Receiving a Stop Change Tracking Response

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

The client MUST return the received status 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:

IsSVHDXSupported: A Boolean value that, if set, indicates support for operations on shared virtual disks.

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

Note: Some of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure. For information about specific differences between versions, see the behavior notes that are provided in the Product Behavior appendix.

The server MUST implement the following:

InitiatorId: A GUID that identifies the initiator of the open request.

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 that 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.

Open.IsVirtualSCSIDisk: A Boolean that, if set, indicates that the file is a virtual SCSI disk.

Open.CreateOptions: The create options, in the format specified in [MS-SMB2] section 2.2.13.

If the server implements **RSVD protocol version 2**, it MUST implement the following:

Open.IsVHDSET: A Boolean that, if set, indicates that the virtual disk is a VHD set.

Open.PendingDelete: A Boolean that, if set, indicates that the VHD set is marked to be deleted on close.

3.2.1.3 Per SenseError in SenseErrorDataList

The server MUST implement the following:

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

SenseError.SrbStatus: A 7-bit value indicating the status of the SCSI request block as specified in section 2.2.5.

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

The server MUST enumerate the shares by calling **NetrShareEnum** as specified in [MS-SRVS] section 3.1.4.8. In the enumerated list, if any of the shares have shi*_type set to

STYPE_CLUSTER_SOFS, as specified in [MS-SRVS] section 2.2.2.4, the server MUST set **IsSVHDXSupported** to TRUE.

When the server is started:

- ~~It~~If **IsSVHDXSupported** is TRUE, the server 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.<6>

3.2.4 Higher-Layer Triggered Events

3.2.4.1 Handling a Tunnel Operation Request

The calling application provides:

- A handle identifying the **Open**.
- Tunnel Operation request.
- The maximum size of the response that will be accepted by the client, indicated by **MaxOutputResponse**.

The server MUST process this request as specified in section 3.2.5.5.

3.2.4.2 Handling a Shared Virtual Disk Support Request

The calling application provides:

- A handle identifying the **Open**.
- File name to identify the shared virtual disk file.
- The maximum size of the response data buffer that will be accepted by the client, indicated by **MaxOutputResponse**.

The server MUST process this request as specified in section 3.2.5.6.

3.2.5 Message Processing Events and Sequencing Rules

For this section, if **IsSVHDXSupported** is FALSE, the server MUST fail the request with STATUS_INVALID_DEVICE_REQUEST.

3.2.5.1 Receiving an Open Request

Note: Some of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure. For information about specific differences between versions, see the behavior notes that are provided in the Product Behavior appendix.

If **Open.LocalOpen** is not NULL, the server MUST look up an **Open** in **OpenTable** where **Open.LocalOpen** matches the **Open** provided by the SMB2 server, as specified in [MS-SMB2] section 3.3.5.9.12. If an **Open** is found, the server MUST stop processing the request. If no **Open** is found, the server MUST fail the request with STATUS_INVALID_HANDLE.

If **Open.LocalOpen** is NULL, this indicates that the server received a request to open a shared virtual disk file. The server MUST process the request as described in the steps that follow, providing the file name and SVHDX_OPEN_DEVICE_CONTEXT.

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

If the **OriginatorFlags** field in the request is set to SVHDX_ORIGINATOR_VHDMP, 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 fails with STATUS_SHARING_VIOLATION, the server MUST fail the request with STATUS_VHD_SHARED.

If the **OriginatorFlags** field in the request is not set to SVHDX_ORIGINATOR_VHDMP, the server SHOULD<~~5~~7> pass the request to the virtual SCSI disk.

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

If the underlying object store or virtual SCSI disk 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~~create operation.
- If **HasInitiatorId** is 0x00, **Open.InitiatorId** is set to zero. Otherwise, it is set to **InitiatorId**.
- **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.
- **Open.IsVirtualSCSIDisk** SHOULD<8> be set to TRUE if **OriginatorFlags** is not set to SVHDX_ORIGINATOR_VHDMP. Otherwise, it MUST be set to FALSE.
- **Open.CreateOptions** MUST be set to the **CreateOptions** received as part of the local create operation.
- **Open.IsVHDSET** MUST be set to TRUE if the application-provided file name has the .vhds extension. FALSE otherwise.
- **Open.PendingDelete** MUST be set to FALSE.

The server MUST construct a SVHDX_OPEN_DEVICE_CONTEXT_RESPONSE structure using the received SVHDX_OPEN_DEVICE_CONTEXT.

The server SHOULD<9> return the constructed SVHDX_OPEN_DEVICE_CONTEXT_RESPONSE and STATUS_SUCCESS to the client.

3.2.5.2 Receiving a Close Request

Note: Some of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure. For information about specific differences between versions, see the behavior notes that are provided in the Product Behavior appendix.

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 **Open.PendingDelete** is TRUE, the server MUST pass the request to the underlying object store to delete the file.

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 the **Open** is found and **Open.InitiatorId** is zero, the server MUST ~~fail the request with the STATUS_INVALID_HANDLE code.~~process 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**.
  - Update other fields of **SenseError** with an implementation-specific<10> value.
- The server MUST return the error (STATUS_SVHDX_ERROR_STORED | **SenseError.StatusKey**) to the client.

If the **Open** is found and the FILE_NO_INTERMEDIATE_BUFFERING bit is not set in **Open.CreateOptions**, the server MUST fail the request with the STATUS_NOT_SUPPORTED code.

If the **Open** is found and **Open.IsVirtualSCSIDisk** is FALSE, the server MUST issue a read to the underlying object store; otherwise the server MUST pass the request to the virtual SCSI disk in an implementation-specific manner.

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

If the underlying object store or virtual SCSI disk 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 underlying object store is the virtual SCSI disk and the error code is not specified in section 2.2.3, the server MUST store the received sense data and return an error 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 virtual SCSI disk.
  - **SenseError.ScsiStatus** MUST be set to the value provided by the virtual SCSI disk.
  - **SenseError.SenseData** MUST be set to the data provided by the virtual SCSI disk~~.~~.<11>
- 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 the STATUS_INVALID_PARAMETER error code.

If the **Open** is found and **Open.InitiatorId** is zero, the server MUST ~~fail the request with the STATUS_INVALID_HANDLE code.~~process 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**.
  - Update other fields of **SenseError** with an implementation-specific<12> value.
- The server MUST return the error (STATUS_SVHDX_ERROR_STORED | **SenseError.StatusKey**) to the client.

If the **Open** is found and the FILE_NO_INTERMEDIATE_BUFFERING bit is not set in **Open.CreateOptions**, the server MUST fail the request with the STATUS_NOT_SUPPORTED code.

If the **Open** is found and **Open.IsVirtualSCSIDisk** is FALSE, the server MUST issue a write to the underlying object store; otherwise the server MUST pass the request to the virtual SCSI disk in an implementation-specific manner.

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

If the underlying object store or virtual SCSI disk indicates the write was denied due to a **persistent reservation** conflict, as described in [SPC-3] section 5.6~~, 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 underlying object store is the virtual SCSI disk and the error code is not specified in section 2.2.3, the server MUST store the received sense data and return an error 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 virtual SCSI disk.
  - **SenseError.ScsiStatus** MUST be set to the value provided by the virtual SCSI disk.
  - **SenseError.SenseData** MUST be set to the data provided by the virtual SCSI disk~~.~~.<13>
- The server MUST return the error (STATUS_SVHDX_ERROR_STORED | **SenseError.StatusKey**) to the client.

3.2.5.5 Receiving a 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 the size of the tunnel operation request including the header is less than 16, the server MUST fail the request with STATUS_BUFFER_TOO_SMALL.

If an **Open** is found and if ~~ProtocolId is~~the bitwise AND of the **OperationCode** value and 0xFF000000 is not equal to ~~0~~0x02000000, the server MUST fail the request with STATUS_INVALID_DEVICE_REQUEST.

If any of the following conditions are TRUE, the server MUST construct the SVHDX_TUNNEL_OPERATION_HEADER with the ~~ProtocolId, ProtocolVersion,~~ OperationCode~~,~~ and RequestId fields set to the value received in the request, and with the Status field set as follows:

~~▪~~ If **ServerServiceVersion** is equal to RSVD protocol version 1(0x00000001) and the ~~**ProtocolId**~~bitwise AND of the **OperationCode** value and 0x00FFF000 is not equal to ~~0x01 or 0x02, the **Status** field MUST be set to STATUS_NOT_IMPLEMENTED.~~

~~▪ If the **ProtocolId** value is equal to 0x01, the **Status** field MUST be set to STATUS_INVALID_DEVICE_REQUEST.~~

- ~~If the **ProtocolVersion** value is not equal to 1~~0x00001000, the **Status** field MUST be set to STATUS_SVHDX_VERSION_MISMATCH.

- If **ServerServiceVersion** is equal to RSVD protocol version 2(0x00000002) and the bitwise AND of the **OperationCode** value and 0x00FFF000 is not equal to 0x00001000 or 0x00002000, the **Status** field MUST be set to STATUS_SVHDX_VERSION_MISMATCH.

- If the **OperationCode** value does not exist in the tunnel operation list as specified in section 2.2.2, the **Status** field MUST be set to STATUS_INVALID_PARAMETER.

- ~~If the **OperationCode** is equal to RSVD_TUNNEL_SCSI_OPERATION and **Open.InitiatorId** is zero, the **Status** field MUST be set to STATUS_INVALID_HANDLE.~~

The server MUST return SVHDX_TUNNEL_OPERATION_HEADER to the client.

~~If the size of the tunnel operation request including header is less than 16, the server MUST fail the request with STATUS_BUFFER_TOO_SMALL.~~

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

3.2.5.5.1 Receiving a Virtual Disk File Information Request

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

If **MaxOutputResponse** is less than 40 (size of SVHDX_TUNNEL_OPERATION_HEADER + size of SVHDX_TUNNEL_~~FILE~~INITIAL_INFO_RESPONSE), the server MUST fail the request with STATUS_BUFFER_TOO_SMALL.

The server MUST issue a file information request to the virtual SCSI disk in an implementations-specific manner, and update the received response to the client.

If the virtual SCSI disk 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~~INITIAL_INFO_RESPONSE structure as specified in section 2.2.4.14 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~~INITIAL_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 virtual SCSI disk.

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

- **VirtualSize** MUST set to the virtual size of the shared virtual disk received from the virtual SCSI disk.

The response MUST be sent to the client.

3.2.5.5.2 Receiving a 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:

If **MaxOutputResponse** is less than 16 (size of SVHDX_TUNNEL_OPERATION_HEADER), the server MUST fail the request with STATUS_BUFFER_OVERFLOW.

The server MUST construct an SVHDX_TUNNEL_CHECK_CONNECTION_RESPONSE structure as specified in section 2.2.4.2 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 a ~~Sense Code~~Status Request for a Prior Operation

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

If **MaxOutputResponse** is less than 40 (size of SVHDX_TUNNEL_OPERATION_HEADER + size of SVHDX_TUNNEL_SRB_STATUS_RESPONSE), the server MUST fail the request with STATUS_INVALID_PARAMETER.

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 2.2.4.4 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_SRB_STATUS_RESPONSE packet MUST be initialized as follows:

- The **StatusKey** field MUST be set to the value received in the request.
- The **SenseInfoAutoGenerated** field MUST be set to the value received from the virtual SCSI disk.
- The **SrbStatus** field MUST be set to **SenseError.SrbStatus**.

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

The response MUST be sent to the client.

3.2.5.5.4 Receiving a 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:

If **MaxOutputResponse** is less than 72 (size of SVHDX_TUNNEL_OPERATION_HEADER + size of SVHDX_TUNNEL_DISK_INFO_RESPONSE), the server MUST fail the request with STATUS_BUFFER_TOO_SMALL.

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

If the virtual SCSI disk 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 2.2.4.4 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_DISK_INFO_RESPONSE structure MUST be initialized as follows:

- **DiskType** MUST be set to the value received from the virtual SCSI disk.
- **DiskFormat** MUST be set to VIRTUAL_STORAGE_TYPE_DEVICE_VHDX.
- **BlockSize** MUST be set to the number of bytes received from virtual SCSI disk.
- **LinkageId** MUST be set to the linkage GUID received from virtual SCSI disk.
- **IsMounted** MUST be set to TRUE if the virtual SCSI disk indicates that the disk is ready for read or write operations. Otherwise, the server MUST set this field to FALSE.
- **Is4kAligned** MUST be set to TRUE if the virtual SCSI disk 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 virtual SCSI disk
- **VirtualDiskId** MUST be set to the virtual disk GUID received from virtual SCSI disk.

The response MUST be sent to the client.

3.2.5.5.5 Receiving a 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 **MaxOutputResponse** is less than 52 (size of SVHDX_TUNNEL_OPERATION_HEADER + size of SVHDX_TUNNEL_SCSI_RESPONSE), the server MUST fail the request with STATUS_INVALID_PARAMETER.

If any of the following conditions is TRUE, the server MUST generate an error response as specified below:

- Size of the request is less than 36
- **Length** field of the request is not equal to 36 bytes
- **SenseInfoExLength** is greater than (size of CDBBuffer + 4)
- **CDBLength** is greater than the size of **CDBBuffer**
- **DataIn** is 0x01 and **DataTransferLength** is less than the size of the **DataBuffer** field
- **Open.InitiatorId** is zero

The SVHDX_TUNNEL_OPERATION_HEADER MUST be initialized. ~~**ProtocolId**, **ProtocolVersion**,~~ **OperationCode** and **RequestId** fields MUST be set to the values received in the request. If **Open.InitiatorId** is zero, the **Status** field MUST be set to STATUS_INVALID_HANDLE. Otherwise, the **Status** field MUST be set to STATUS_INVALID_PARAMETER. The SVHDX_TUNNEL_SCSI_RESPONSE MUST be set to the data received in SVHDX_TUNNEL_SCSI_REQUEST structure of the request. The server MUST send SVHDX_TUNNEL_OPERATION_HEADER and SVHDX_TUNNEL_SCSI_RESPONSE to the client and SHOULD NOT<14> add additional data to the response.

Otherwise, the server MUST issue a CDB command to the virtual SCSI disk in an implementation-specific manner.

If **DataIn** is 0 and the data returned by the virtual SCSI disk is greater than **DataTransferLength** in the request, the server MUST fail the request with STATUS_INVALID_PARAMETER.

The server MUST construct a SVHDX_TUNNEL_SCSI_RESPONSE structure as specified in section 2.2.4.8 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 the value received from the virtual SCSI disk.
- The **RequestId** field MUST be set to the value received in the request.

The SVHDX_TUNNEL_SCSI_RESPONSE MUST be initialized as follows:

- The **SenseInfoAutoGenerated** field MUST be set to the value received from the virtual SCSI disk.

- The **SrbStatus** field MUST be set to one of the values specified in section 2.2.5 reflecting the status indicated by the virtual SCSI disk.
- The **SCSIStatus** field MUST be set to value received from the virtual SCSI disk.
- The **DataIn** field MUST be set to the value received in the request.
- **SrbFlags** MUST be set to the value received in the request.
- The **CDBLength** 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 2.2.4.8.
- The **SenseInfoExLength** field is set to the length of the sense information received from the virtual SCSI disk, if any.
- The **SenseDataEx** field is set to the sense information received from the virtual SCSI disk, if any.
- The **DataTransferLength** is set to the length of the additional data returned by the virtual SCSI disk, if any.
- The **DataBuffer** is set to the additional data returned by the virtual SCSI disk, if any.

The response MUST be sent to the client.

3.2.5.5.6 Receiving a 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:

If **MaxOutputResponse** is less than 17 (size of SVHDX_TUNNEL_OPERATION_HEADER + size of SVHDX_TUNNEL_VALIDATE_DISK_RESPONSE), the server MUST fail the request with STATUS_BUFFER_TOO_SMALL.

The server MUST issue a validate request to the virtual SCSI disk in an implementation-specific manner.

The server MUST construct an SVHDX_TUNNEL_VALIDATE_DISK_RESPONSE structure, as specified in section 2.2.4.10, 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 virtual SCSI disk indicates command success, the **Status** MUST be set to STATUS_SUCCESS; otherwise, it MUST be set to the error code provided by the virtual SCSI disk.

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

The **IsValidDisk** field MUST be set to TRUE if the virtual SCSI disk indicates that the disk is valid. Otherwise, the server MUST set this field to FALSE.

The response MUST be sent to the client.

3.2.5.5.7 Receiving a Start Meta-Operation Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

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

If the **OperationType** is not one of **SvhdxMetaOperationTypeCreateSnapshot**, **SvhdxMetaOpeartionTypeOptimize**, or **SvhdxMetaOperationTypeExtractVHD**, the operation is failed with the STATUS_INVALID_PARAMETER.

Processing for a specific **OperationType** is as specified in sections 3.2.5.~~6~~5.7.1, 3.2.5.5.7.2, and 3.2.5.5.7.3.

3.2.5.5.7.1 Receiving a Create Snapshot Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

When the server receives a request with **OperationType** set to **SvhdxMetaOperationTypeCreateSnapshot**, the request processing proceeds as follows:

If the input buffer is less than the 76 + **ParametersPayloadSize** bytes, the server MUST fail the request with STATUS_BUFFER_TOO_SMALL.

If the **SnapshotType** is not **SvhdxSnapshotTypeVM** or **SvhdxSnapshotTypeCDP**, the operation is failed with STATUS_INVALID_PARAMETER_1 (0xc00000ef).

If **Stage1** is **SvhdxSnapshotStageInvalid**, the operation is failed with STATUS_INVALID_PARAMETER_2 (0xc00000f0).

If any of the following conditions is TRUE, the server MUST fail the operation with STATUS_INVALID_PARAMETER_5 (0xc00000f3):

- If the **Flags** field is nonzero and **SnapshotType** is not **SvhdxSnapshotTypeVM**
- If the **Flags** field is set to SVHDX_SNAPSHOT_DISK_FLAG_ENABLE_CHANGE_TRACKING and **Stage1** is not **SvhdxSnapshotStageInitialize**.
- If **SnapshotType** is **SvhdxSnapshotTypeVM** and the **Flags** field is a value other than 0 or SVHDX_SNAPSHOT_DISK_FLAG_ENABLE_CHANGE_TRACKING.

Each **Stage2** through **Stage6** field is checked with the following:

- If the current stage value is less than or equal to that of the prior stage, the operation is failed with STATUS_INVALID_PARAMETER_3(0xc00000f1).
- If the current stage value is not **SvhdxSnapshotStageInvalid** and the server has observed an **SvhdxSnapshotStageInvalid** for this request, the operation is failed with STATUS_INVALID_PARAMETER4 (0xc00000f2).

If **Open.IsVHDSET** is FALSE, the server MUST fail the request with STATUS_INVALID_DEVICE_REQUEST.

The server MUST issue a Create Snapshot request to the virtual SCSI disk in an implementation-specific manner.

The server MUST construct an **SVHDX_META_OPERATION_REPLY** structure as specified in section 2.2.4.4, 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_META_OPERATION_REPLY structure MUST be initialized as follows:

- The **ChangeTrackingErrorStatus** field MUST be set to the value received from the virtual SCSI disk.

The server MUST send the constructed **SVHDX_META_OPERATION_REPLY** structure to the client.

3.2.5.5.7.2 Receiving an Optimize Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

When the server receives a request in which the **OperationType** is **SvhdxMetaOperationTypeOptimize**, the request processing proceeds as follows:

The server MUST issue a VHD set optimize request to the virtual SCSI disk in an implementation-specific manner.

The server MUST set the **Status** field of the SVHDX_TUNNEL_OPERATION_HEADER as ERROR_SUCCESS, and return the header to the client.

3.2.5.5.7.3 Receiving an ExtractVHD Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

When the server receives a request in which the **OperationType** is **SvhdxMetaOperationTypeExtractVHD**, the request processing proceeds as follows:

If the input buffer is less than the 68 + **ParametersPayloadSize** bytes, the server MUST fail the request with STATUS_BUFFER_TOO_SMALL.

If any of the following conditions is TRUE, the server MUST fail the request with STATUS_INVALID_PARAMETER:

- **ParametersPayloadSize** is less than 60 bytes.
- **DestinationFileLengthBytes** is less than 12.
- The UNICODE string in the **DestinationVhdSetName** field is not a valid name.
- The **DestinationVhdSetName** field contains any directories.
- The **SourceSnapshotId** or **SourceLimitSnapshotId** is not a valid snapshot ID.

If any of the following conditions is TRUE, the server MUST fail the operation with STATUS_INVALID_PARAMETER_1 (0xc00000f3):

- If the **Flags** field is nonzero and **SnapshotType** is not **SvhdxSnapshotTypeVM**.

- If **SnapshotType** is **SvhdxSnapshotTypeVM** and the **Flags** field is a value other than 0 or SVHDX_EXTRACT_SNAPSHOTS_FLAG_DELETE_ON_CLOSE.

The server MUST issue an extract VHD request to the virtual SCSI disk in an implementation-specific manner.

If SVHDX_EXTRACT_SNAPSHOTS_FLAG_DELETE_ON_CLOSE is set in the **Flags** field, the server MUST set **Open.PendingDelete** to TRUE.

The server MUST set the **Status** field of the SVHDX_TUNNEL_OPERATION_HEADER as the error returned by the virtual SCSI disk, and return the header to the client.

3.2.5.5.7.4 Receiving a Convert to VHD Set Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

When the server receives a request in which the **OperationType** is **SvhdxMetaOperationTypeConvertToVHDSet**, the request processing proceeds as follows:

If the **DestinationVhdSetName** field doesn't contain the .vhds extension, the server MUST fail the request with STATUS_INVALID_PARAMETER.

The server MUST issue a convert to VHD set request to the virtual SCSI disk in an implementation-specific manner.

The server MUST set the **Status** field of the SVHDX_TUNNEL_OPERATION_HEADER to the error returned by the virtual SCSI disk, and return the header to the client.

3.2.5.5.7.5 Receiving a Resize Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

When the server receives a request in which the **OperationType** is **SvhdxMetaOperationTypeResize**, the request processing proceeds as follows:

If **ExpandOnly** is nonzero and **NewSize** is less than current virtual disk size, the server MUST fail the request with ERROR_INVALID_PARAMETER.

The server MUST issue a resize VHD set request to the virtual SCSI disk in an implementation-specific manner.

The server MUST set the **Status** field of the SVHDX_TUNNEL_OPERATION_HEADER to the error returned by the virtual SCSI disk, and return the header to the client.

3.2.5.5.8 Receiving a Query Meta-Operation Progress Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

When the server receives a request in which **OperationCode** is equal to RSVD_TUNNEL_META_OPERATION_QUERY_PROGRESS, the request handling proceeds as follows:

The server MUST issue a Query Meta-operation request to the virtual SCSI disk in an implementation-specific manner.

If the virtual SCSI disk indicates an error, the server MUST fail the request with the error received, placing it into the **Status** field of the SVHDX_TUNNEL_OPERATION_HEADER, and return the header to the client.

Otherwise, the server MUST construct a VHDX_META_OPERATION_QUERY_PROGRESS_RESPONSE structure as specified in section 2.2.4.4, 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_META_OPERATION_QUERY_PROGRESS_RESPONSE structure MUST be initialized as follows:

- **CurrentProgressValue** MUST be set to the value received from the virtual SCSI disk.
- **CompleteValue** MUST be set to the value received from the virtual SCSI disk.

The response MUST be sent to the client.

3.2.5.5.9 Receiving a Query VHD Set Information Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

When the server receives a request in which **OperationCode** is equal to RSVD_TUNNEL_VHDSET_QUERY_INFORMATION, the request handling proceeds as follows:

If the provided input buffer is not equal to the size of the SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_REQUEST structure, then the server MUST fail the request with STATUS_BUFFER_TOO_SMALL.

If the **Open.IsVHDSET** is FALSE, the server MUST fail the request with STATUS_INVALID_DEVICE_REQUEST.

If **VHDSetInformationType** is not one of the valid information types specified in section 2.2.4.19, the server MUST fail the request with STATUS_INVALID_PARAMETER_1(0xc00000ef).

If **VHDSetInformationType** is **SvhdxVHDSetInformationTypeSnapshotEntry** and the supplied **SnapshotType** is not **SvhdxSnapshotTypeVM** or **SvhdxSnapshotTypeCDP**, the server MUST fail the request with STATUS_INVALID_PARAMETER_1.

If **VHDSetInformationType** is **SvhdxVHDSetInformationTypeSnapshotList** and the supplied **SnapshotType** is not **SvhdxSnapshotTypeVM**, the server MUST fail the request with STATUS_INVALID_PARAMETER_1.

If **VHDSetInformationType** is neither **SvhdxVHDSetInformationTypeSnapshotEntry** nor **SvhdxVHDSetInformationTypeSnapshotList** and **SnapshotType** is not **SvhdxSnapshotTypeInvalid**, the server MUST fail the request with STATUS_INVALID_PARAMETER**,**

The server MUST issue a query VHD set information request to the virtual SCSI disk in an implementation-specific manner.

If the virtual SCSI disk indicates an error, the server MUST fail the request with the error received, placing it into the **Status** field of the SVHDX_TUNNEL_OPERATION_HEADER, and return the header to the client.

Otherwise, the server MUST process as follows:

- If **VHDSetInformationType** is **SvhdxVHDSetInformationTypeSnapshotEntry**, the server MUST construct a **SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_SNAPSHOT_ENTRY_RESPONSE** structure as specified in section 2.2.4.21, 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_VHDSET_QUERY_INFORMATION_SNAPSHOT_ENTRY_RESPONSE** structure MUST be initialized as follows:
    - **VHDSetInformationType** MUST be set to **SvhdxVHDSetInformationTypeSnapshotEntry**.
    - **SnapshotType** MUST be set to the value received from the virtual SCSI disk.
    - **IsValidSnapshot** MUST be set to the value received from the virtual SCSI disk.
    - **SnapshotId** MUST be set to the **SnapshotId** provided in the request.
    - **ParentSnapshotId** MUST be set to zero.
    - **LogFileId** MUST be set to zero.
  - The response MUST be sent to the client.
- If **VHDSetInformationType** is **SvhdxVHDSetInformationTypeSnapshotList**, **SvhdxVHDSetInformationTypeCdpSnapshotActiveList**, or **SvhdxVHDSetInformationTypeCdpSnapshotInactiveList**, the server MUST construct a **SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_SNAPSHOT_LIST** structure as specified in section 2.2.4.20, 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_VHDSET_QUERY_INFORMATION_SNAPSHOT_LIST** structure MUST be initialized as follows:
    - **VHDSetInformationType** MUST be set to **SvhdxVHDSetInformationTypeSnapshotList**.

        - **SnapshotsFilled** MUST be set to TRUE if this response contains a list of snapshot IDs.
        - **SnapshotsFilled** MUST be set to the value received from the virtual SCSI disk.
        - **NumberOfSnapshots** MUST be set to the value received from the virtual SCSI disk.
        - If **SnapshotsFilled** is TRUE, **SnapshotIds** MUST be set to the value received from the virtual SCSI disk.
    - The response MUST be sent to the client.
- If **VHDSetInformationType** is **SvhdxVHDSetInformationTypeCdpSnapshotRoot**, the server MUST construct a **SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_SNAPSHOT_ENTRY_RESPONSE** structure as specified in section 2.2.4.21, 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_VHDSET_QUERY_INFORMATION_SNAPSHOT_ENTRY_RESPONSE** structure MUST be initialized as follows:
        - **VHDSetInformationType** MUST be set to **SvhdxVHDSetInformationTypeCdpSnapshotRoot**.
        - **SnapshotType** MUST be set to the value received from the virtual SCSI disk.
        - **IsValidSnapshot** MUST be set to the value received from the virtual SCSI disk.
        - **SnapshotId** MUST be set to the **SnapshotId** provided in the request.
        - **ParentSnapshotId** MUST be set to the value received from the virtual SCSI disk.
        - **LogFileId** MUST be set to the value received from the virtual SCSI disk.
    - The response MUST be sent to the client.
- If **VHDSetInformationType** is **SvhdxVHDSetInformationTypeOptimizeNeeded**, the server MUST construct an **SVHDX_TUNNEL_VHDSET_QUERY_INFORMATION_OPTIMIZE_RESPONSE** structure as specified in section 2.2.4.22, 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 response MUST be sent to the client.

3.2.5.5.10 Receiving a Delete Snapshot Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

When the server receives a request in which **OperationCode** is equal to RSVD_TUNNEL_DELETE_SNAPSHOT, the request handling proceeds as follows:

If PersistReference is not equal to zero and **SnapshotType** is not equal to **SvhdxSnapshotTypeVM**, the server MUST return an implementation specific error.

The server MUST issue a delete snapshot request to the virtual SCSI disk in an implementation-specific manner.

The server MUST initialize the SVHDX_TUNNEL_OPERATION_HEADER as follows:

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

If the virtual SCSI disk indicates command success, the **Status** MUST be set to STATUS_SUCCESS; otherwise, it MUST be set to the error code provided by the virtual SCSI disk.

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

The response MUST be sent to the client.

3.2.5.5.11 Receiving a Change Tracking Get Parameter Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

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

The server MUST issue a get tracking **Parameter** request to the virtual SCSI disk in an implementation-specific manner.

If the virtual SCSI disk indicates an error, the server MUST fail the request with the error received, 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_DISK_INFO_RESPONSE structure as specified in section 2.2.4.4, 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_CHANGE_TRACKING_GET_PARAMETERS_RESPONSE structure MUST be initialized as follows:

- Set **ChangeTrackingStatus** as the status returned by the virtual SCSI disk.
- Set **LogFileSize** returned by the virtual SCSI disk.

The response MUST be sent to the client.

3.2.5.5.12 Receiving a Change Tracking Start Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

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

If **AppendData** is FALSE and **LogFileNameLengthInBytes** is 0, the server MUST fail the request with STATUS_INVALID_PARAMETER.

The server MUST issue a start tracking request to the virtual SCSI disk in an implementation-specific manner.

The server MUST initialize the SVHDX_TUNNEL_OPERATION_HEADER as follows:

- The **OperationCode** field MUST be set to the **OperationCode** value of the request.
- The **Status** field MUST be set to the value received from the virtual SCSI disk.
- The **RequestId** field MUST be set to the value received in the request.

The response MUST be sent to the client.

3.2.5.5.13 Receiving a Change Tracking Stop Request

Note: All of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure.

When the server receives a request in which **OperationCode** is equal to RSVD_TUNNEL_CHANGE_TRACKING_STOP, the request handling proceeds as follows:

The server MUST issue a stop tracking request to the virtual SCSI disk in an implementation-specific manner.

The server MUST initialize the SVHDX_TUNNEL_OPERATION_HEADER as follows:

- The **OperationCode** field MUST be set to the OperationCode value of the request.
- The **Status** field MUST be set to the value received from the virtual SCSI disk.
- 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 a Query Shared Virtual Disk Support Request

Note: Some of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure. For information about specific differences between versions, see the behavior notes that are provided in the Product Behavior appendix.

If **MaxOutputResponse** is less than 8 (size of SVHDX_SHARED_VIRTUAL_DISK_SUPPORT_RESPONSE), the server MUST fail the request with STATUS_BUFFER_TOO_SMALL.

If **IsSVHDXSupported** is TRUE, the server MUST set **SharedVirtualDiskSupport** to 1. Otherwise, the server SHOULD<15> set **SharedVirtualDiskSupport** to 0.

If **ServerServiceVersion** is equal to RSVD protocol version 1(0x00000001), the server MUST set **SharedVirtualDiskSupport** to **SharedVirtualDiskSupported**. Otherwise the server MUST set **SharedVirtualDiskSupport** to **SharedVirtualDiskVer2OperationsSupported**.

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**.

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

Otherwise, the server MUST set **SharedVirtualDiskHandleState** to **HandleStateFileShared**.

The response MUST be sent to the client.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

The following section describes common scenarios that indicate normal traffic flow in order to illustrate the function of the Remote Shared Virtual Disk (RSVD) Protocol.

4.1 Retrieving Virtual Disk File Information

The following diagram demonstrates the steps taken to open a shared virtual disk file, retrieve virtual disk file information, and close it.

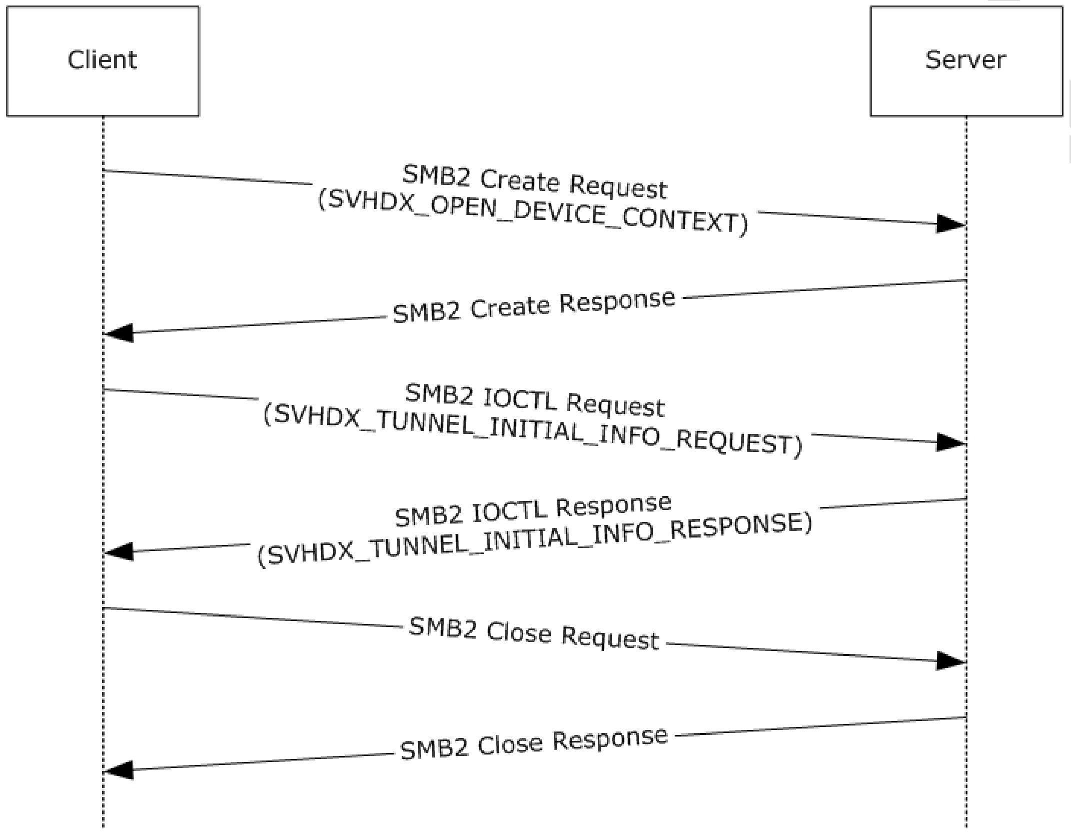
Client
Server
SMB2 Create Request (SVHDX_OPEN_DEVICE_CONTEXT)
SMB2 Create Response
SMB2 IOCTL Request (SVHDX_TUNNEL_INITIAL_INFO_REQUEST)
SMB2 IOCTL Response (SVHDX_TUNNEL_INITIAL_INFO_RESPONSE)
SMB2 Close Request
SMB2 Close Response

Figure 2: Retrieving virtual disk file information

1. The client sends an SMB2 CREATE Request with the SVHDX_OPEN_DEVICE_CONTEXT create context to open a shared virtual disk file.

```
Version: 1 (0x00000001)
HasInitiatorId: 1
Reserved: 0 (0x000000)
InitiatorId: (0x07770D201F2740834579D46F5AC43B73)
Flags: 0 (0x00000000)
OriginatorFlags: 1 (0x00000001)
OpenRequestId: (0x000000001EC7871E)
InitiatorHostNameLength: 16 (0x0010)
InitiatorHostName: client01 (0x003100300074006E00650069006C0063)
```

2. The server responds with an SMB2 CREATE Response giving the handle to the open identifying shared virtual disk file.

3. The client sends an SMB2 IOCTL Request with SVHDX_TUNNEL_~~FILE~~INITIAL_INFO_REQUEST to retrieve the virtual disk information.

```
OperationCode: RSVD_TUNNEL_GET_INITIAL_INFO_OPERATION (0x02001001)
Status: 0 (0x00000000)
RequestId: (0x000000001Ec7871E)
ServerVersion: 0 (0x00000000)
SectorSize: 0 (0x00000000)
PhysicalSectorSize: 0 (0x00000000)
VirtualSize: 0 (0x0000000000000000)
```

4. The server sends an SMB2 IOCTL Response with SVHDX_TUNNEL_INITIAL~~FILE~~_INFO_RESPONSE containing the virtual disk information.

```
OperationCode: RSVD_TUNNEL_GET_INITIAL_INFO_OPERATION (0x02001001)
Status: 0 (0x00000000)
RequestId: (0x000000001Ec7871E)
ServerVersion: 1 (0x00000001)
SectorSize: 512 (0x00000200)
PhysicalSectorSize: 256 (0x00000100)
VirtualSize: (0x4000000000000000)
Reserved: 0 (0x00000000)
```

5. The client sends an SMB2 CLOSE Request to close the shared virtual disk file.

6. The server sends an SMB2 CLOSE Response indicating the close was successful.

4.2 Executing a SCSI Command

The following diagram demonstrates the steps taken to open a shared virtual disk file, execute a SCSI command, and close it.

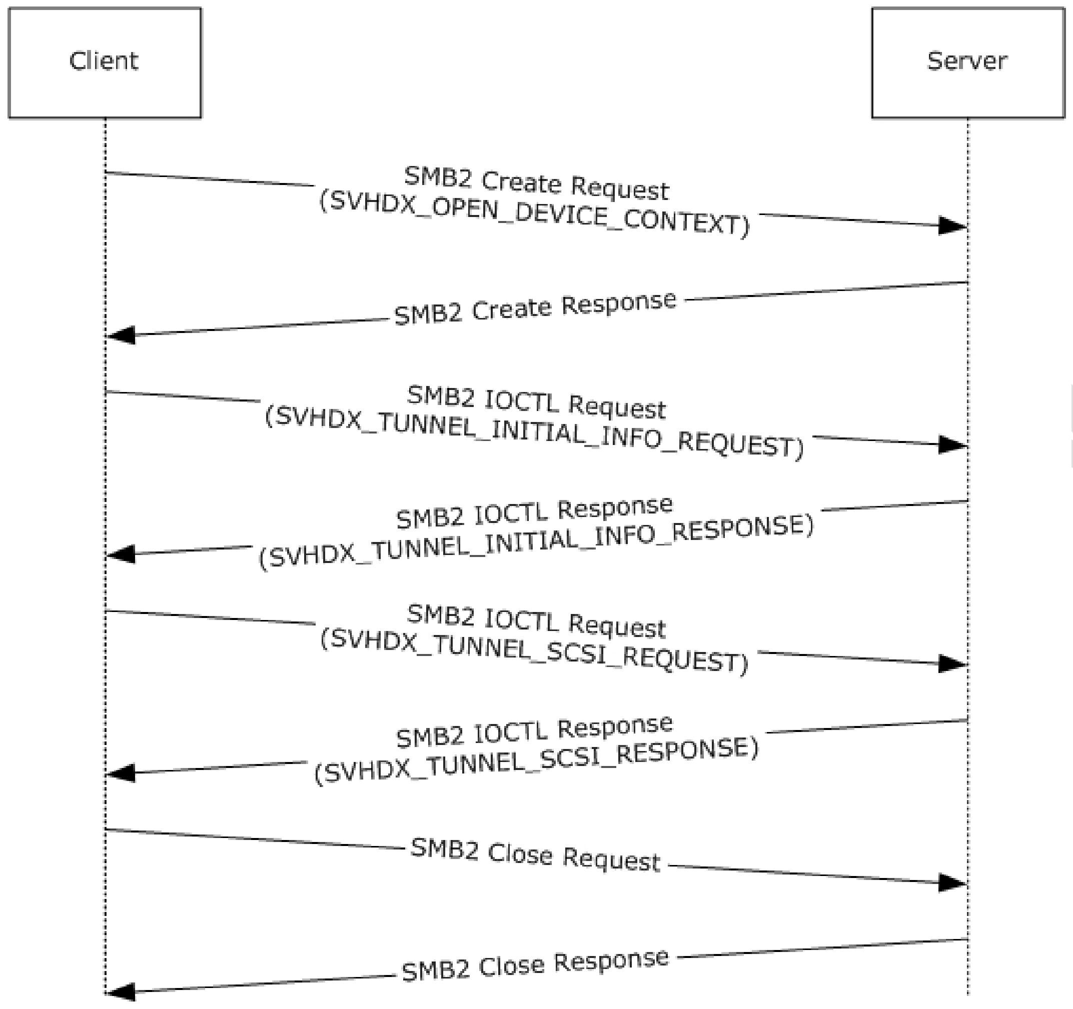
Client
Server
SMB2 Create Request (SVHDX_OPEN_DEVICE_CONTEXT)
SMB2 Create Response
SMB2 IOCTL Request (SVHDX_TUNNEL_INITIAL_INFO_REQUEST)
SMB2 IOCTL Response (SVHDX_TUNNEL_INITIAL_INFO_RESPONSE)
SMB2 IOCTL Request (SVHDX_TUNNEL_SCSI_REQUEST)
SMB2 IOCTL Response (SVHDX_TUNNEL_SCSI_RESPONSE)
SMB2 Close Request
SMB2 Close Response

Figure 3: Executing an SCSI command

1. The client sends an SMB2 CREATE Request with the SVHDX_OPEN_DEVICE_CONTEXT create context to open a shared virtual disk file.

```
Version: 1(0x00000001)
HasInitiatorId: 1
Reserved: 0 (0x000000)
InitiatorId: (0x07770D201F2740834579D46F5AC43B73)
Flags: 0 (0x00000000)
OriginatorFlags: 1 (0x00000001)
OpenRequestId: (0x000000001EC7871E)
InitiatorHostNameLength: 16 (0x0010)
InitiatorHostName: client01 (0x003100300074006E00650069006C0063)
```

2. The server responds with an SMB2 CREATE Response giving the handle to the open identifying shared virtual disk file.

3. The client sends an SMB2 IOCTL Request with SVHDX_TUNNEL_~~FILE~~INITIAL_INFO_REQUEST to retrieve the virtual disk information.

```
OperationCode: RSVD_TUNNEL_GET_INITIAL_INFO_OPERATION (0x02001001)
Status: 0 (0x00000000)
RequestId: (0x000000001Ec7871E)
ServerVersion: 0 (0x00000000)
SectorSize: 0 (0x00000000)
PhysicalSectorSize: 0 (0x00000000)
VirtualSize: 0 (0x0000000000000000)
```

4. The server sends an SMB2 IOCTL Response with SVHDX_TUNNEL_~~FILE~~INITIAL_INFO_RESPONSE containing the virtual disk information.

```
OperationCode: RSVD_TUNNEL_GET_~~FILE~~INITIAL_INFO_OPERATION (0x02001001)
Status: 0 (0x00000000)
RequestId: (0x000000001Ec7871E)
ServerVersion: 1 (0x00000001)
SectorSize: 512 (0x00000200)
PhysicalSectorSize: 256 (0x00000100)
VirtualSize: (0x4000000000000000)
Reserved: 0 (0x00000000)
```

5. The client sends an SMB2 IOCTL Request with SVHDX_TUNNEL_ SCSI_REQUEST to execute a SCSI command.

```
OperationCode: RSVD_TUNNEL_SCSI_OPERATION (0x02001002)
Status: 0 (0x00000000)
RequestId: (0x000000001Ec7871E)
Length: 36 (0x0024)
Reserved1: 0 (0x0000)
CDBLength: 16 (0x10)
SenseInfoExLength: 20 (0x14)
DataIn: 1 (0x01)
Reserved2: 0 (0x00)
SrbFlags: (0x0020014A)
DataTransferLength: 8 (0x00000008)
CDBBuffer: 37 (0x00000000000000000000000000000025)
Reserved3: 0 (0x00000000)
DataBuffer: 0 (0x0000000000000000)
```

6. The server sends an SMB2 IOCTL Response with SVHDX_TUNNEL_SCSI_RESPONSE containing the response.

```
OperationCode: RSVD_TUNNEL_SCSI_OPERATION (0x02001002)
Status: 0 (0x00000000)
RequestId: (0x000000001Ec7871E)
Length: 36 (0x0024)
SrbStatus: 0 (0x00)
ScsiStatus: 0 (0x00)
CDBLength: 16 (0x10)
SenseInfoExLength: 20 (0x14)
DataIn: 1 (0x01)
Reserved: 0 (0x00)
SrbFlags: (0x0020014A)
DataTransferLength: 8 (0x00000008)
SenseDataEx: 37 (0x0000000000000000000000000000000000000025)
```

```
    DataBuffer: 0 (0x0000000000000000)
```

7. The client sends an SMB2 CLOSE Request to close the shared virtual disk file.
8. The server sends an SMB2 CLOSE Response indicating the close was successful.

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~~:~~.

Note: Some of the information in this section is subject to change because it applies to a preliminary implementation of the protocol or structure. For information about specific differences between versions, see the behavior notes that are provided in the Product Behavior appendix.

- Windows Server 2012 R2 operating system
- Windows 8.1 operating system
- Windows vNext operating system
- Windows Server vNext 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 8.1 Windows Server 2012 R2 Windows vNext Windows Server vNext	Windows Server 2012 R2 Windows Server vNext

<2> Section 3.1.3: Windows 8.1 and Windows Server 2012 R2 set **ClientServiceVersion** to RSVD protocol version 1(0x00000001). Windows vNext and Windows Server vNext set **ClientServiceVersion** to RSVD protocol version 2(0x00000002).

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

<4> Section 3.1.4.9: Microsoft Windows applications set unspecified bits in **SrbFlags** in addition to the specified bits in section 2.2.4.7. All bits set in **SrbFlags** are ignored for processing by Windows servers as specified in section 3.2.5.5.5.

<5> Section 3.1.4.9: Windows-based RSVD clients set **DataIn** to 0x01 when **SrbFlags** includes both SRB_FLAGS_DATA_IN and SRB_FLAGS_DATA_OUT.

<6> Section 3.2.3: Windows Server 2012 R2 sets **ServerServiceVersion** to RSVD protocol version 1 (0x00000001). Windows Server vNext sets **ServerServiceVersion** to RSVD protocol version 2(0x00000002).

<~~5~~7> Section 3.2.5.1: ~~If~~ When the ~~**OriginatorFlags** in the request is 0x00000008~~shared virtual disk file does not previously exist, Windows Server 2012 R2 attempts to open with a create

disposition which allows an empty file to be created. This in turn will ~~not consider it as a virtual SCSI disk device and will incorrectly pass the request to the underlying object store~~cause the RSVD request to return STATUS_FILE_CORRUPT_ERROR.

<8> Section 3.2.5.1: If the **OriginatorFlags** in the request is 0x00000008, Windows~~ Server 2012 R2~~-based RSVD servers will incorrectly set **Open.IsVirtualSCSIDisk** to FALSE.

<9> Section 3.2.5.1: Windows Server 2012 R2 without [MSKB-3025091] doesn't return SVHDX_OPEN_DEVICE_CONTEXT_RESPONSE.

<10> Section 3.2.5.3: Windows-based RSVD servers set **SenseError.SrbStatus** to 0x02, **SenseError.ScsiStatus** to 0x02, and **SenseError.SenseData** to 0xF0000000000000A000000000000000000000000.

<11> Section 3.2.5.~~5~~3: Windows Server 2012 R2 ~~appends~~sets the returned sense data to an arbitrary value.

<12> Section 3.2.5.4: Windows-based RSVD servers set **SenseError.SrbStatus** to 0x02, **SenseError.ScsiStatus** to 0x02, and **SenseError.SenseData** to 0xF0000000000000A000000000000000000000000.

<13> Section 3.2.5.4: Windows Server 2012 R2 sets the returned sense data to an arbitrary value.

<14> Section 3.2.5.5.5: Windows Server 2012 R2 and Windows Server vNext append [**MaxOutputResponse** – (size of SVHDX_TUNNEL_OPERATION_HEADER + size of SVHDX_TUNNEL_SCSI_RESPONSE)] number of bytes, all set to zeroes, at the end of the response.

<15> Section 3.2.5.6: Windows Server 2012 R2 always sets **SharedVirtualDiskSupport** to 1.