[MS-SMB2]: Server Message Block (SMB) Protocol Versions 2 and 3

This topic lists the Errata found in [MS-SMB2] since it was last published. Since this topic is updated frequently, we recommend that you subscribe to these RSS or Atom feeds to receive update notifications.

Errata are subject to the same terms as the Open Specifications documentation referenced.

RSS
Atom

Errata below are for Protocol Document Version V58.0 – 2019/04/30.

Errata Published*	Description
2019/09/02	In Section 2.2.3.1.4, SMB2_NETNAME_NEGOTIATE_CONTEXT_ID, the following has been changed from: The SMB2_NETNAME_NEGOTIATE_CONTEXT_ID context is specified in an SMB2 NEGOTIATE request to indicate the server name the client connects to. The server MUST ignore this context. The format of the data in the Data field of this SMB2_NEGOTIATE_CONTEXT is as follows. Changed to: The SMB2_NETNAME_NEGOTIATE_CONTEXT_ID context is specified in an SMB2 NEGOTIATE request to indicate the server name the client connects to. The format of the data in the Data field of this SMB2_NEGOTIATE_CONTEXT is as follows. In Section 3.2.5.2, Receiving an SMB2 NEGOTIATE Response, the following has been changed from: • If the NegotiateContextList contains more than one SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context, the client MUST return an error to the calling application. Changed to: • If the NegotiateContextList does not contain exactly one SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context, the client MUST return an error to the calling application. The following was added: • For each context in the received NegotiateContextList, if the context is any negotiate context other than SMB2_PREAUTH_INTEGRITY_CAPABILITIES, SMB2_COMPRESSION_CAPABILITIES, and SMB2_ENCRYPTION_CAPABILITIES negotiate context, the client MUST ignore the negotiate context. In Section 3.3.5.4, Receiving an SMB2 NEGOTIATE Request, the following has been changed from: • If the NegotiateContextList contains more than one SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context, the server MUST fail the negotiate request with STATUS_INVALID_PARAMETER.

Errata Published*	Description
	Changed to: ● If the NegotiateContextList does not contain exactly one SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context, the server MUST fail the negotiate request with STATUS_INVALID_PARAMETER. The following was added: ● For each context in the received NegotiateContextList , if the context is SMB2_NETNAME_NEGOTIATE_CONTEXT_ID or any negotiate context other than SMB2_PREAUTH_INTEGRITY_CAPABILITIES, SMB2_COMPRESSION_CAPABILITIES, and SMB2_ENCRYPTION_CAPABILITIES negotiate context, the server MUST ignore the negotiate context.
2019/08/19	In Section 2.2.3.1.4, SMB2_NETNAME_NEGOTIATE_CONTEXT_ID, the NetName field has been changed from: NetName (variable): A null-terminated Unicode string containing the server name and specified by the client application. Changed to: NetName (variable): A Unicode string containing the server name and specified by the client application. In Section 3.2.1.2, Per SMB2 Transport Connection, the description of Connection.ServerName has been changed from: Connection.ServerName: A null-terminated Unicode UTF-16 fully qualified domain name, a NetBIOS name, or an IP address of the server machine. Changed to: Connection.ServerName: A Unicode UTF-16 fully qualified domain name, a NetBIOS name, or an IP address of the server machine. In Section 3.2.1.9, Per Server, the description of ServerName has been changed from: ServerName: A fully qualified domain name, a NetBIOS name, or an IP address of the server machine. Changed to: ServerName: A Unicode UTF-16 fully qualified domain name, a NetBIOS name, or an IP address of the server machine. In Section 3.2.4.2.2.2, SMB2-Only Negotiate, the last bullet point has been changed from: ● NetName MUST be set to the application-provided ServerName formatted as null-terminated Unicode string. Changed to:

Errata Published*	Description
	• NetName MUST be set to the application-provided ServerName.
2019/08/05	In Section 2.2.2.1, SMB2 ERROR Context Response, the description of ErrorId has been changed from: ErrorId (4 bytes): An identifier for the error context. This field MUST be set to the following value. Changed to: ErrorId (4 bytes): An identifier for the error context. This field MUST be set to one of the following values.
2019/08/05	In Section 3.1.4.4, Compressing the Message, the following has been changed from: The sender MUST perform the following: Set Offset to the length, in bytes, if any, of the uncompressed part of the message. Otherwise, set Offset to zero. Set OriginalCompressedSegmentSize to the uncompressed length, in bytes, of the portion of the message that is compressed. Assemble the outgoing message as a concatenation of the SMB2 COMPRESSION_TRANSFORM_HEADER followed by the uncompressed portion of the message followed by the compressed payload. The sender MUST compress the data using the CompressionAlgorithm as specified in [MS-XCA] section 2. If the size of the resulting SMB2 message is less than OriginalCompressedSegmentSize, the sender MUST replace the SMB2 message with the concatenated SMB2 COMPRESSION_TRANSFORM_HEADER and compressed SMB2 message. Changed to: The sender MUST perform the following: If the entire SMB2 message is being compressed, then set Offset to zero; otherwise, set Offset to the length, in bytes, of the uncompressed part of the message. Set OriginalCompressedSegmentSize to the uncompressed length, in bytes, of the portion of the message that is being compressed. The sender MUST compress the data using the CompressionAlgorithm as specified in [MS-XCA] section 2. If the size of the compressed data is less than OriginalCompressedSegmentSize, the sender MUST perform the following: If Offset is zero, the sender MUST replace the SMB2 message with the SMB2 COMPRESSION_TRANSFORM_HEADER followed by the compressed SMB2 message. Otherwise, the sender MUST replace the portion of the SMB2 message selected for compression with the compressed part and prepend the SMB2 message with the SMB2 COMPRESSION_TRANSFORM_HEADER.
2019/08/05	In this document, multiple sections have been updated to handle multichannel scenarios. For details on these changes, see the PDF doc here.
2019/06/24	In Section 3.3.5.4, Receiving an SMB2 NEGOTIATE Request, information about how the compression algorithms are handled has been added.

Errata Published*	Description
	Changed from: The server SHOULD <234> set Connection.CompressionIds to all the algorithms in the CompressionAlgorithms field, in the order they are received. If the server does not support any of the algorithms provided by the client, Connection.CompressionIds MUST be set to an empty list. Changed to: The server SHOULD <234> set Connection.CompressionIds to all the supported compression algorithms common to both client and server in the CompressionAlgorithms field, in the order they are received. If the server does not support any of the algorithms provided by the client, Connection.CompressionIds MUST be set to an empty list. In this same section, the following step has been removed from the processing rules: ● If CompressionAlgorithm received in the request is "NONE".
2019/06/10	In Section 3.3.5.15, Receiving an SMB2 IOCTL Request, we added some clarifying information. Changed from: If InputCount is not equal to zero, the server MUST fail the request with STATUS_INVALID_PARAMETER in the following cases: ▪ If InputOffset is greater than zero but less than (size of SMB2 header + size of the SMB2 IOCTL request not including Buffer) or if InputOffset is greater than (size of SMB2 header + size of the SMB2 IOCTL request). ▪ If OutputOffset is greater than zero but less than (size of SMB2 header + size of the SMB2 IOCTL request not including Buffer) or if OutputOffset is greater than (size of SMB2 header + size of the SMB2 IOCTL request). ▪ If (InputOffset + InputCount) is greater than (size of SMB2 header + size of the SMB2 IOCTL request). ▪ If (OutputOffset + OutputCount) is greater than (size of SMB2 header + size of the SMB2 IOCTL request). ▪ If OutputCount is greater than zero and OutputOffset is less than (InputOffset + InputCount). Changed to: If InputCount is not equal to zero, the server MUST fail the request with STATUS_INVALID_PARAMETER in the following cases: ▪ If InputOffset is greater than zero but less than (size of SMB2 header + size of the SMB2 IOCTL request not including Buffer). ▪ If InputOffset is not a multiple of 8 bytes. ▪ If InputOffset is greater than size of SMB2 Message. ▪ If (InputOffset + InputCount) is greater than size of SMB2 Message. If InputCount is equal to zero and InputOffset is greater than size of SMB2 Message, the server MAY<320> fail the request with STATUS_INVALID_PARAMETER. The server SHOULD<321> ignore OutputOffset and OutputCount fields.

Errata Published*	Description
	<320> Section 3.3.5.15: Windows 8 and later and Windows Server 2012 and later do not fail the request. <321> Section 3.3.5.15: Windows Vista, Windows Server 2008, Windows 7, and Windows Server 2008 R2 fail the request with STATUS_INVALID_PARAMETER in the following cases: ▪ If OutputCount is not equal to zero and OutputOffset is greater than zero but less than (size of SMB2 header + size of the SMB2 IOCTL request not including Buffer). ▪ If OutputCount is not equal to zero and OutputOffset is greater than size of SMB2 Message. ▪ If OutputCount is not equal to zero and OutputOffset is not rounded up to a multiple of 8 bytes. ▪ If (OutputOffset + OutputCount) is greater than size of SMB2 Message. ▪ If OutputCount is greater than zero and OutputOffset is less than (InputOffset + InputCount). Windows 7 and Windows Server 2008 R2 fail the request with STATUS_INVALID_PARAMETER if OutputOffset or OutputCount is greater than size of SMB2 Message.
2019/04/29	In Section 2.2.3.1, SMB2 NEGOTIATE_CONTEXT Request Values, the value of SMB2_COMPRESSION_CAPABILITIES has been changed from: 0x0004 Changed to: 0x0003 In Section 6, Appendix A: Product Behavior, the following behavior notes have been changed from: <13> Section 2.2.3.1: Windows 10 v1809 operating system and prior, Windows Server v1809 operating system and prior, and Windows Server 2019 and prior do not support compression. <14> Section 2.2.3.1: Windows 10 v1809 and prior, Windows Server v1809 and prior, and Windows Server 2019 and prior do not support SMB2 _NETNAME_NEGOTIATE_CONTEXT_ID. <53> Section 2.2.19: Windows 10 v1809 and prior and Windows Server v1809 and prior do not support SMB2_READFLAG_REQUEST_COMPRESSED flag. <72> Section 2.2.42: Windows 10 v1809 and prior and Windows Server v1809 and prior do not support compression and SMB2 COMPRESSION_TRANSFORM_HEADER. Changed to: <13> Section 2.2.3.1: Windows 10 v1809 operating system and prior, Windows Server v1809 operating system and prior, and Windows Server 2019 and prior do not send or process SMB2_COMPRESSION_CAPABILITIES. <14> Section 2.2.3.1: Windows 10 v1809 and prior, Windows Server v1809 and prior, and Windows Server 2019 and prior do not send or process SMB2 _NETNAME_NEGOTIATE_CONTEXT_ID. <53> Section 2.2.19: Windows 10 v1809 and prior and Windows Server v1809 and prior do not send or process SMB2_READFLAG_REQUEST_COMPRESSED flag. <72> Section 2.2.42: Windows 10 v1809 and prior and Windows Server v1809 and prior do not send or process SMB2 COMPRESSION_TRANSFORM_HEADER.

Errata Published*	Description
2019/04/29	This document has been updated to add the DataLength validation step for the processing of SMB2_ENCRYPTION_CAPABILITIES negotiate context. In Section 3.3.5.2, Receiving Any Message, the following has been changed from: If the Connection.Dialect is "3.1.1", then the server MUST process the negotiate context list that is specified by the request's NegotiateContextOffset and NegotiateContextCount fields as follows: Processing the SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context: If the negotiate context list does not contain exactly one SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context, then the server MUST fail the negotiate request with STATUS_INVALID_PARAMETER. If the SMB2_PREAUTH_INTEGRITY_CAPABILITIES HashAlgorithms array does not contain any hash algorithms that the server supports, then the server MUST fail the negotiate request with STATUS_SMB_NO_PREAUTH_INTEGRITY_HASH_OVERLAP (0xC05D0000). Changed to: If the Connection.Dialect is "3.1.1", then the server MUST process the NegotiateContextList that is specified by the request's NegotiateContextOffset and NegotiateContextCount fields as follows: If the NegotiateContextList contains more than one SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context, the server MUST fail the negotiate request with STATUS_INVALID_PARAMETER. If the NegotiateContextList contains more than one SMB2_ENCRYPTION_CAPABILITIES negotiate context, the server MUST fail the negotiate request with STATUS_INVALID_PARAMETER. If the NegotiateContextList contains more than one SMB2_COMPRESSION_CAPABILITIES negotiate context, the server MUST fail the negotiate request with STATUS_INVALID_PARAMETER. Processing the SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context: If the DataLength of the negotiate context is less than the size of SMB2_PREAUTH_INTEGRITY_CAPABILITIES structure, the server MUST fail the negotiate request with STATUS_INVALID_PARAMETER. If the SMB2_PREAUTH_INTEGRITY_CAPABILITIES HashAlgorithms array does not contain any hash algorithms that the server supports, the server MUST fail the negotiate request with STATUS_SMB_NO_PREAUTH_INTEGRITY_HASH_OVERLAP (0xC05D0000). ... Changed from: If the negotiate context list contains more than one SMB2_ENCRYPTION_CAPABILITIES negotiate context, then the server MUST fail the negotiate request with STATUS_INVALID_PARAMETER. The server MUST set Connection.CipherId to one of the ciphers in the client's SMB2_ENCRYPTION_CAPABILITIES Ciphers array in an implementation-specific manner. If the client and server have no common cipher, then the server MUST sets Connection.CipherId to 0. Changed to:

Errata Published*	Description
	If the DataLength of the negotiate context is less than the size of the SMB2_ENCRYPTION_CAPABILITIES structure, the server MUST fail the negotiate request with STATUS_INVALID_PARAMETER. The server MUST set Connection.CipherId to one of the ciphers in the client's SMB2_ENCRYPTION_CAPABILITIES Ciphers array in an implementation-specific manner. If the client and server have no common cipher, the server MUST set Connection.CipherId to 0. In Section, 3.3.5.4, Receiving an SMB2 NEGOTIATE Request, the following has been changed from: If Connection.Dialect is "3.1.1", then the server MUST build a negotiate context list for its negotiate response as follows: Building an SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context: The server MUST add an SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context to the response's negotiate context list. HashAlgorithmCount MUST be set to 1. SaltLength MUST be set to an implementation-specific<239> number of Salt bytes. HashAlgorithms[0] MUST be set to Connection.PreauthIntegrityHashId. The Salt buffer MUST be filled with SaltLength unique bytes that are generated for this response by a cryptographic secure pseudo-random number generator. Building an SMB2_ENCRYPTION_CAPABILITIES negotiate response context: If the server received an SMB2_ENCRYPTION_CAPABILITIES negotiate context in the client's negotiate request, then the server MUST add an SMB2_ENCRYPTION_CAPABILITIES negotiate context to the response's negotiate context list. Note that the server MUST send an SMB2_ENCRYPTION_CAPABILITIES context even if the client and server have no common cipher. This is done so that the client can differentiate between a server that does not support encryption (no SMB2_ENCRYPTION_CAPABILITIES context in the response's negotiate context list) and a server that supports encryption but does not share a cipher with the client (an SMB2_ENCRYPTION_CAPABILITIES context in the response's negotiate context list that indicates a cipher of 0). Changed to: If Connection.Dialect is "3.1.1", then the server MUST build a NegotiateContextList for its negotiate response as follows: Building an SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context: The server MUST add an SMB2_PREAUTH_INTEGRITY_CAPABILITIES negotiate context to the response's NegotiateContextList. HashAlgorithmCount MUST be set to 1. SaltLength MUST be set to an implementation-specific<239> number of Salt bytes. HashAlgorithms[0] MUST be set to Connection.PreauthIntegrityHashId. The Salt buffer MUST be filled with SaltLength unique bytes that are generated for this response by a cryptographic secure pseudo-random number generator. Building an SMB2_ENCRYPTION_CAPABILITIES negotiate response context: If the server received an SMB2_ENCRYPTION_CAPABILITIES negotiate context in the client's negotiate request, the server MUST add an SMB2_ENCRYPTION_CAPABILITIES negotiate context to the response's NegotiateContextList. Note that the server MUST send an SMB2_ENCRYPTION_CAPABILITIES context even if the client and server have no common cipher. This is done so that the client can differentiate between a server that does not support encryption (no SMB2_ENCRYPTION_CAPABILITIES context in the response's NegotiateContextList) and a server that supports encryption but does not share a cipher with the client (an SMB2_ENCRYPTION_CAPABILITIES context in the response's NegotiateContextList that indicates a cipher of 0).

Errata Published*	Description
2019/04/29	This document has been updated to handle the case when an invalid ProtocolId is received in the header of the message. In Section 2.2.1.1, SMB2 Packet Header – ASYNC, the following has been changed from: ProtocolId (4 bytes): The protocol identifier. The value MUST be (in network order) 0xFE, 'S', 'M', and 'B'. Changed to: ProtocolId (4 bytes): The protocol identifier. The value MUST be set to 0x424D53FE, also represented as (in network order) 0xFE, 'S', 'M', and 'B'. In Section 2.2.1.2, SMB2 Packet Header – SYNC, the following has been changed from: ProtocolId (4 bytes): The protocol identifier. The value MUST be (in network order) 0xFE, 'S', 'M', and 'B'. Changed to: ProtocolId (4 bytes): The protocol identifier. The value MUST be set to 0x424D53FE, also represented as (in network order) 0xFE, 'S', 'M', and 'B'. In Section 2.2.41, SMB2 TRANSFORM_HEADER, the following has been changed from: ProtocolId (4 bytes): The protocol identifier. The value MUST be (in network order) 0xFD, 'S', 'M', and 'B'. Changed to: ProtocolId (4 bytes): The protocol identifier. The value MUST be set to 0x424D53FD, also represented as (in network order) 0xFD, 'S', 'M', and 'B'. In Section 2.2.42, SMB2 COMPRESSION_TRANSFORM_HEADER, the following has been changed from: ProtocolId (4 bytes): The protocol identifier. The value MUST be (in network order) 0xFC, 'S', 'M', and 'B'. Changed to: ProtocolId (4 bytes): The protocol identifier. The value MUST be set to 0x424D53FC, also represented as (in network order) 0xFC, 'S', 'M', and 'B'. In Section 3.2.5.1, Receiving Any Message, the following has been changed from: If the server implements the SMB 3.x dialect family and ProtocolId in the header of the received message is 0x424d53FD, the client MUST decrypt the request as specified in section 3.2.5.1.1 before performing the following steps.

Errata Published*	Description
	If the server implements the SMB 3.1.1 dialect and ProtocolId in the header of the received message is 0x424d53FC, the client MUST decompress the request as specified in section 3.2.5.1.10 before performing the following steps. Unless specifically noted in a subsequent section, the following logic MUST be applied to any response message that is received from the server by the client. If the status code in the SMB2 header is not equal to STATUS_SUCCESS, the client SHOULD<143> retry the operation, in an implementation-specific manner, on the same or different channel. The client MUST ignore the CreditCharge field in the SMB2 header. If the message size received exceeds Connection.MaxTransactSize, the client MUST disconnect the connection. Changed to: If the client implements the SMB 3.x dialect family and ProtocolId in the header of the received message is 0x424D53FD, the client MUST decrypt the request as specified in section 3.2.5.1.1 before performing the following steps. If the client implements the SMB 3.1.1 dialect and ProtocolId in the header of the received message is 0x424D53FC, the client MUST decompress the request as specified in section 3.2.5.1.10 before performing the following steps. If ProtocolId in the header of the received message is 0x424D53FE, the client MUST perform the following: Unless specifically noted in a subsequent section, the following logic MUST be applied to any response message that is received from the server by the client. If the status code in the SMB2 header is not equal to STATUS_SUCCESS, the client SHOULD<143> retry the operation, in an implementation-specific manner, on the same or different channel. The client MUST ignore the CreditCharge field in the SMB2 header. If the message size received exceeds Connection.MaxTransactSize, the client MUST disconnect the connection. Otherwise the client MUST disconnect the connection. In Sections 3.2.5.1.1, Decrypting the Message, and 3.2.5.1.10, Decompressing the Message, all instances of "0x424d53FE" have been changed to "0x424D53FE" In Section 3.3.5.2, Receiving Any Message, the following has been changed from: If the server implements the SMB 3.x dialect family, and the ProtocolId in the header of the received message is 0x424d53FD, the server MUST decrypt the message as specified in section 3.3.5.2.1 before performing the following steps. If the server supports compression and the ProtocolId in the header of the received message is 0x424d53FC, the server MUST decompress the message as specified in section 3.3.5.2.13 before performing the following steps. Changed to: If ProtocolId in the header of the received message is 0x424D53FF and the command received is SMB_COM_NEGOTIATE, the client MUST process the request as specified in section 3.3.5.3. If the server implements the SMB 3.x dialect family, and the ProtocolId in the header of the received message is 0x424D53FD, the server MUST decrypt the message as specified in section 3.3.5.2.1 before performing the following steps. If the server supports compression and the ProtocolId in the header of the received message is 0x424D53FC, the server MUST decompress the message as specified in section 3.3.5.2.13 before performing the following steps. If ProtocolId in the header of the received message is 0x424D53FE, the server MUST perform the following:

Errata Published*	Description
	... Otherwise, the server MUST disconnect the connection. In Section 3.3.5.2.1, Decrypting the Message, "0x424d53FC" has been changed to "0x424D53FC" and "0x424d53FE" has been changed to "0x424D53FE" In Section 3.3.5.2.6, Handling Incorrectly Formatted Requests, the following has been removed: The ProtocolId field in the SMB2 header is not equal to 0xFE, 'S', 'M', and 'B' (in network order). In Section 3.3.5.2.13, Decompressing the message, "0x424d53FE" has been changed to "0x424D53FE" In Section 3.3.5.3, Receiving an SMB_COM_NEGOTIATE, the following has been changed from: If the request does not have a valid SMB2 header following the syntax specified in section 2.2.1, the server MUST check to see if it has received an SMB_COM_NEGOTIATE as described in section 1.7. This request is defined in [MS-SMB] section 2.2.4.5.1, with the SMB header defined in section 2.2.3.1. If the request matches the format described there, and Connection.NegotiateDialect is 0xFFFF, processing MUST continue as specified in 3.3.5.3.1. Otherwise, the server MUST disconnect the connection, as specified in section 3.3.7.1, without sending a response. Changed to: If Connection.NegotiateDialect is 0xFFFF, processing MUST continue as specified in 3.3.5.3.1. Otherwise, the server MUST disconnect the connection, as specified in section 3.3.7.1, without sending a response. For details on the above changes, see the PDF doc here.
2019/04/15	In two sections, the logic for checking against Dialect and Channel has been modified. In Section 3.3.5.12, Receiving an SMB2 READ Request, the following has been changed from: Connection.Dialect is "3.0.2" or "3.1.1" and Channel is not equal to SMB2_CHANNEL_RDMA_V1 or SMB2_CHANNEL_NONE. Connection.Dialect is "3.0" and Channel is not equal to SMB2_CHANNEL_RDMA_V1_INVALIDATE. Changed to: Connection.Dialect is "3.0.2" or "3.1.1" and Channel is not equal to SMB2_CHANNEL_RDMA_V1_INVALIDATE or SMB2_CHANNEL_RDMA_V1 or SMB2_CHANNEL_NONE. Connection.Dialect is "3.0" and Channel is not equal to SMB2_CHANNEL_RDMA_V1 or SMB2_CHANNEL_NONE.

Errata Published*	Description
	In Section 3.3.5.13, Receiving an SMB2 WRITE Request, the following has been changed from: Connection.Dialect is "3.0.2" or "3.1.1" and Channel is not equal to SMB2_CHANNEL_RDMA_V1 or SMB2_CHANNEL_NONE. Connection.Dialect is "3.0" and Channel is not equal to SMB2_CHANNEL_RDMA_V1_INVALIDATE. Changed to: Connection.Dialect is "3.0.2" or "3.1.1" and Channel is not equal to SMB2_CHANNEL_RDMA_V1_INVALIDATE or SMB2_CHANNEL_RDMA_V1 or SMB2_CHANNEL_NONE. Connection.Dialect is "3.0" and Channel is not equal to SMB2_CHANNEL_RDMA_V1 or SMB2_CHANNEL_NONE.
2019/04/15	In Section 2.2.31.1, SRV_COPYCHUNK_COPY, the first paragraph has been changed from: The SRV_COPYCHUNK_COPY packet is sent in an SMB2 IOCTL Request by the client to initiate a server-side copy of data. It is set as the contents of the input data buffer. This packet consists of the following: … Changed to: The SRV_COPYCHUNK_COPY packet is sent to the server in an SMB2 IOCTL Request using FSCTL_SRV_COPYCHUNK or FSCTL_SRV_COPYCHUNK_WRITE by the client to initiate a server-side copy of data. It is set as the contents of the input data buffer. This packet consists of the following: … In Section 2.2.32.1, SRV_COPYCHUNK_RESPONSE, the first paragraph has been changed from: The SRV_COPYCHUNK_RESPONSE packet is sent in an SMB2 IOCTL Response by the server to return the results of a server-side copy operation . It is placed in the Buffer field of the SMB2 IOCTL Response packet. This packet consists of the following: … Changed to: The SRV_COPYCHUNK_RESPONSE packet is sent to the client by the server in an SMB2 IOCTL Response for FSCTL_SRV_COPYCHUNK or FSCTL_SRV_COPYCHUNK_WRITE requests to return the results of a server-side copy operation . It is placed in the Buffer field of the SMB2 IOCTL Response packet. This packet consists of the following: …
2019/04/15	Several sections have been updated to modify information about an input buffer. In Section 2.2.31, SMB2 IOCTL Request, the InputOffset field has been changed from:

Errata Published*	Description
	InputOffset (4 bytes): The offset, in bytes, from the beginning of the SMB2 header to the input data buffer. If no input data is required for the FSCTL/IOCTL command being issued, the client SHOULD set this value to 0.<57> <57> Section 2.2.31: If no input data is required for the FSCTL/IOCTL command being issued, Windows-based clients set this field to any value. Changed to: InputOffset (4 bytes): The offset, in bytes, from the beginning of the SMB2 header to the input data buffer. If no input data is required for the FSCTL/IOCTL command being issued, this field can be set to any value by the client and MUST be ignored by the server. In Section 3.2.4.20.1, Application Requests Enumeration of Previous Versions, the following has been changed from: The SMB2 IOCTL Request MUST be initialized as follows: ● The CtlCode field is set to FSCTL_SRV_ENUMERATE_SNAPSHOTS. ● The FileId field is set to Open.FileId. ● The InputOffset field SHOULD<131> be set to 0. ● The InputCount field is set to 0. ● The OutputOffset field SHOULD<132> be set to zero. ● The OutputCount field is set to 0. ● The MaxInputResponse field is set to 0. … Changed to: The SMB2 IOCTL Request MUST be initialized as specified in section 2.2.31, with the exception of the following values: ● The CtlCode field is set to FSCTL_SRV_ENUMERATE_SNAPSHOTS. ● The FileId field is set to Open.FileId. ● The InputCount field is set to 0. ● The MaxInputResponse field is set to 0. … In Section 3.2.4.20.2.1, Application Requests a Source File Key, the following has been changed from: The SMB2 IOCTL Request MUST be initialized as follows: ● The CtlCode field is set to the FSCTL_SRV_REQUEST_RESUME_KEY. ● The FileId field is set to Open.FileId. ● The InputOffset field SHOULD<133> be set to 0. ● The InputCount field is set to 0. ● The OutputOffset field SHOULD<134> be set to 0. ● The OutputCount field is set to 0. … Changed to:

Errata Published*	Description
	The SMB2 IOCTL Request MUST be initialized as specified in section 2.2.31, with the exception of the following values: ● The CtlCode field is set to the FSCTL_SRV_REQUEST_RESUME_KEY. ● The FileId field is set to Open.FileId. ● The InputCount field is set to 0. ● The OutputOffset field SHOULD<134> be set to 0. … In Section 3.2.4.20.5, Application Requests a Peek at Pipe Data, the following has been changed from: The SMB2 IOCTL Request MUST be initialized as follows: ● The CtlCode field is set to FSCTL_PIPE_PEEK. ● The FileId field is set to Open.FileId. ● The InputOffset field SHOULD<139> be set to 0. ● The InputCount field is set to 0. ● The OutputOffset field SHOULD<140> be set to zero. ● The OutputCount field is set to 0. … Changed to: The SMB2 IOCTL Request MUST be initialized as specified in section 2.2.31, with the exception of the following values: ● The CtlCode field is set to FSCTL_PIPE_PEEK. ● The FileId field is set to Open.FileId. ● The InputCount field is set to 0. ● The OutputOffset field SHOULD<140> be set to zero. … In Section 3.2.4.20.10, Application Requests Querying Server's Network Interfaces, has been changed from: The SMB2 IOCTL Request MUST be initialized as specified in section 2.2.31, with the exception of the following values: ● The CtlCode field is set to FSCTL_QUERY_NETWORK_INTERFACE_INFO. ● The FileId field is set to { 0xFFFFFFFFFFFFFFFF, 0xFFFFFFFFFFFFFFFF }. ● The MaxInputResponse field is set to 0. ● The MaxOutputResponse field is set to an implementation-specific<145> value. ● SMB2_0_IOCTL_IS_FSCTL is set to TRUE in the Flags field. ● The InputOffset field is set to the offset to the Buffer, in bytes, from the beginning of the SMB2 header. … Changed to: The SMB2 IOCTL Request MUST be initialized as specified in section 2.2.31, with the exception of the following values: ● The CtlCode field is set to FSCTL_QUERY_NETWORK_INTERFACE_INFO.

Errata Published*	Description
	● The FileId field is set to { 0xFFFFFFFFFFFFFFFF, 0xFFFFFFFFFFFFFFFF }. ● The InputCount field is set to 0. ● The MaxInputResponse field is set to 0. ● The MaxOutputResponse field is set to an implementation-specific<145> value. ● SMB2_0_IOCTL_IS_FSCTL is set to TRUE in the Flags field. … In Section 3.3.5.15, Receiving an SMB2 IOCTL Request, the following has been changed from: The server MUST fail the request with STATUS_INVALID_PARAMETER in the following cases: … Changed to: If InputCount is not equal to zero, the server MUST fail the request with STATUS_INVALID_PARAMETER in the following cases: …

*Date format: YYYY/MM/DD