[MS-FSA]: File System Algorithms

This topic lists the Errata found in the MS-FSA document 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

June 24, 2019 – Download

Errata below are for Protocol Document Version V29.0 – 2019/05/30.

Errata Published*	**Description**
2019/12/16	In Section 2.1.5.1.1, Creation of a New File, described when to initialize UsnReason to USN_REASON_FILE_CREATE and when to set UsnReason to USN_REASON_OBJECT_ID_CHANGE and USN_REASON_STREAM_CHANGE. Also, clarified how the object store posts a USN change. Changed from: Pseudocode for the operation is as follows: • If StreamTypeToOpen is DirectoryStream and DesiredFileAttributes.FILE_ATTRIBUTE_TEMPORARY is set, the operation MUST be failed with STATUS_INVALID_PARAMETER. • If DesiredFileAttributes.FILE_ATTRIBUTE_READONLY and CreateOptions.FILE_DELETE_ON_CLOSE are both set, the operation MUST be failed with STATUS_CANNOT_DELETE. • If Open.RemainingDesiredAccess.ACCESS_SYSTEM_SECURITY is set and Open.GrantedAccess.ACCESS_SYSTEM_SECURITY is not set and SecurityContext.PrivilegeSet does not contain "SeSecurityPrivilege", the operation MUST be failed with STATUS_ACCESS_DENIED. • If StreamTypeToOpen is DataStream and Open.GrantedAccess.FILE_ADD_FILE is not set and AccessCheck(SecurityContext,Open.Link.ParentFile.SecurityDescriptor, FILE_ADD_FILE) returns FALSE and Open.HasRestoreAccess is FALSE, the operation MUST be failed with STATUS_ACCESS_DENIED. • If StreamTypeToOpen is DirectoryStream and Open.GrantedAccess.FILE_ADD_SUBDIRECTORY is not set and AccessCheck(SecurityContext,Open.Link.ParentFile.SecurityDescriptor, FILE_ADD_SUBDIRECTORY) returns FALSE and Open.HasRestoreAccess is FALSE, the operation MUST be failed with STATUS_ACCESS_DENIED. • If the object store implements encryption and DesiredFileAttributes.FILE_ATTRIBUTE_ENCRYPTED is TRUE: • If UserCertificate is empty, the operation MUST be failed with STATUS_CS_ENCRYPTION_NEW_ENCRYPTED_FILE. • EndIf • The object store MUST build a new File object with fields initialized as follows:... • If TunnelCacheEntry.ObjectIdInfo.ObjectId is not empty: • If TunnelCacheEntry.ObjectIdInfo.ObjectId is not unique on File.Volume: • The object store MUST construct a FILE_OBJECTID_INFORMATION structure (as specified in [MS-FSCC] section 2.4.28.1) ObjectIdInfo as follows:

Errata Published*	Description
	• ObjectIdInfo.FileReference set to File.FileId64. • ObjectIdInfo.ObjectId set to TunnelCacheEntry.ObjectIdInfo.ObjectId. • ObjectIdInfo.BirthVolumeId set to TunnelCacheEntry.ObjectIdInfo.BirthVolumeId. • ObjectIdInfo.BirthObjectId set to TunnelCacheEntry.ObjectIdInfo.BirthObjectId. • ObjectIdInfo.DomainId set to TunnelCacheEntry.ObjectIdInfo.DomainId. • Send directory change notification as specified in section 2.1.4.1, with Volume equal to File.Volume, Action equal to FILE_ACTION_ID_NOT_TUNNELLED, FilterMatch equal to FILE_NOTIFY_CHANGE_FILE_NAME, FileName equal to "\$Extend\$ObjId", NotifyData equal to ObjectIdInfo, and NotifyDataLength equal to sizeof(FILE_OBJECTID_INFORMATION). • Else: • Set File.ObjectId to TunnelCacheEntry.ObjectIdInfo.ObjectId. • Set File.BirthVolumeId to TunnelCacheEntry.ObjectIdInfo.BirthVolumeId. • Set File.BirthObjectId to TunnelCacheEntry.ObjectIdInfo.BirthObjectId. • Set File.DomainId to TunnelCacheEntry.ObjectIdInfo.DomainId. • EndIf... • If StreamTypeToOpen is DataStream, then the object store MUST create a new data stream for the file as follows:<53> • Build a new Stream object with all fields initially set to zero. • Set Stream.StreamType to DataStream. • Set Stream.Name to StreamNameToOpen. • Set Stream.File to File. • Add Stream to File.StreamList. • Set Open.Stream to Stream. • Else the object store MUST create a new directory stream as follows: • Build a new Stream object with all fields initially set to zero. • Set Stream.StreamType to DirectoryStream. • Set Stream.File to File. • Add Stream to File.StreamList. • Set Open.Stream to Stream. • EndIf • If the object store implements encryption and File.FileAttributes.FILE_ATTRIBUTE_ENCRYPTED is TRUE: • If File.FileType is DataFile, set Stream.IsEncrypted to TRUE. • EndIf • The object store MUST update the duplicated information as specified in section 2.1.4.18 with Link equal to Link. • The object store MUST set Open.File to File. • The object store MUST set Open.Link to Link. • The object store MUST insert Link into File.LinkList. • The object store MUST insert Link into Link.ParentFile.DirectoryList.

Errata Published*	Description
	• The object store MUST update Link.ParentFile.LastModificationTime, Link.ParentFile.LastChangeTime, and Link.ParentFile.LastAccessTime to the current system time. • If the Oplock member of the DirectoryStream in Link.ParentFile.StreamList (hereinafter referred to as ParentOplock) is not empty, the object store MUST check for an oplock break on the parent according to the algorithm in section 2.1.4.12, with input values as follows:…. Changed to: Pseudocode for the operation is as follows: • If StreamTypeToOpen is DirectoryStream and DesiredFileAttributes.FILE_ATTRIBUTE_TEMPORARY is set, the operation MUST be failed with STATUS_INVALID_PARAMETER. • If DesiredFileAttributes.FILE_ATTRIBUTE_READONLY and CreateOptions.FILE_DELETE_ON_CLOSE are both set, the operation MUST be failed with STATUS_CANNOT_DELETE. • If Open.RemainingDesiredAccess.ACCESS_SYSTEM_SECURITY is set and Open.GrantedAccess.ACCESS_SYSTEM_SECURITY is not set and SecurityContext.PrivilegeSet does not contain "SeSecurityPrivilege", the operation MUST be failed with STATUS_ACCESS_DENIED. • If StreamTypeToOpen is DataStream and Open.GrantedAccess.FILE_ADD_FILE is not set and AccessCheck(SecurityContext,Open.Link.ParentFile.SecurityDescriptor, FILE_ADD_FILE) returns FALSE and Open.HasRestoreAccess is FALSE, the operation MUST be failed with STATUS_ACCESS_DENIED. • If StreamTypeToOpen is DirectoryStream and Open.GrantedAccess.FILE_ADD_SUBDIRECTORY is not set and AccessCheck(SecurityContext,Open.Link.ParentFile.SecurityDescriptor, FILE_ADD_SUBDIRECTORY) returns FALSE and Open.HasRestoreAccess is FALSE, the operation MUST be failed with STATUS_ACCESS_DENIED. • If the object store implements encryption and DesiredFileAttributes.FILE_ATTRIBUTE_ENCRYPTED is TRUE: • If UserCertificate is empty, the operation MUST be failed with STATUS_CS_ENCRYPTION_NEW_ENCRYPTED_FILE. • EndIf • Initialize UsnReason to zero. • Set UsnReason.USN_REASON_FILE_CREATE to TRUE. • The object store MUST build a new File object with fields initialized as follows:... • If TunnelCacheEntry.ObjectIdInfo.ObjectId is not empty: • If TunnelCacheEntry.ObjectIdInfo.ObjectId is not unique on File.Volume: • The object store MUST construct a FILE_OBJECTID_INFORMATION structure (as specified in [MS-FSCC] section 2.4.28.1) ObjectIdInfo as follows: • ObjectIdInfo.FileReference set to File.FileId64. • ObjectIdInfo.ObjectId set to TunnelCacheEntry.ObjectIdInfo.ObjectId. • ObjectIdInfo.BirthVolumeId set to TunnelCacheEntry.ObjectIdInfo.BirthVolumeId. • ObjectIdInfo.BirthObjectId set to TunnelCacheEntry.ObjectIdInfo.BirthObjectId. • ObjectIdInfo.DomainId set to TunnelCacheEntry.ObjectIdInfo.DomainId.

Errata Published*	Description
	● Send directory change notification as specified in section 2.1.4.1, with Volume equal to File.Volume, Action equal to FILE_ACTION_ID_NOT_TUNNELLED, FilterMatch equal to FILE_NOTIFY_CHANGE_FILE_NAME, FileName equal to "\$Extend\$ObjId", NotifyData equal to ObjectIdInfo, and NotifyDataLength equal to sizeof(FILE_OBJECTID_INFORMATION). ● Else: ● Set File.ObjectId to TunnelCacheEntry.ObjectIdInfo.ObjectId. ● Set File.BirthVolumeId to TunnelCacheEntry.ObjectIdInfo.BirthVolumeId. ● Set File.BirthObjectId to TunnelCacheEntry.ObjectIdInfo.BirthObjectId. ● Set File.DomainId to TunnelCacheEntry.ObjectIdInfo.DomainId. ● Set UsnReason.USN_REASON_OBJECT_ID_CHANGE to TRUE. ● EndIf... ● If StreamTypeToOpen is DataStream, then the object store MUST create a new data stream for the file as follows:<53> ● Build a new Stream object with all fields initially set to zero. ● Set Stream.StreamType to DataStream. ● Set Stream.Name to StreamNameToOpen. ● Set Stream.File to File. ● Add Stream to File.StreamList. ● Set Open.Stream to Stream. ● If Stream.Name is not empty, set UsnReason.USN_REASON_STREAM_CHANGE to TRUE. ● Else the object store MUST create a new directory stream as follows: ● Build a new Stream object with all fields initially set to zero. ● Set Stream.StreamType to DirectoryStream. ● Set Stream.File to File. ● Add Stream to File.StreamList. ● Set Open.Stream to Stream. ● EndIf ● If the object store implements encryption and File.FileAttributes.FILE_ATTRIBUTE_ENCRYPTED is TRUE: ● If File.FileType is DataFile, set Stream.IsEncrypted to TRUE. ● EndIf ● The object store MUST update the duplicated information as specified in section 2.1.4.18 with Link equal to Link. ● The object store MUST set Open.File to File. ● The object store MUST set Open.Link to Link. ● The object store MUST insert Link into File.LinkList. ● The object store MUST insert Link into Link.ParentFile.DirectoryList. ● The object store MUST post a USN change as specified in section 2.1.4.11 with File equal to File, Reason equal to UsnReason, and FileName equal to Link.Name. ● The object store MUST update Link.ParentFile.LastModificationTime, Link.ParentFile.LastChangeTime, and Link.ParentFile.LastAccessTime to the current system time.

Errata Published*	Description
	• If the Oplock member of the DirectoryStream in Link.ParentFile.StreamList (hereinafter referred to as ParentOplock) is not empty, the object store MUST check for an oplock break on the parent according to the algorithm in section 2.1.4.12, with input values as follows:...
2019/12/16	In Section 2.1.4.17, Algorithm for Noting That a File Has Been Modified, added product behavior note <42> with information about when file systems choose to defer processing for a file that has been modified to a later time. Changed from: The inputs for this algorithm are as follows: • Open: The Open through which the file was modified. The pseudocode for the algorithm is as follows: • If Open.UserSetModificationTime is FALSE, set Open.File.LastModificationTime to the current system time. • If Open.UserSetChangeTime is FALSE, set Open.File.LastChangeTime to the current system time. • If Open.UserSetAccessTime is FALSE, set Open.File.LastAccessTime to the current system time. • Set Open.File.FileAttributes.FILE_ATTRIBUTE_ARCHIVE to TRUE. Changed to: The inputs for this algorithm are as follows: • Open: The Open through which the file was modified. The pseudocode for the algorithm is as follows: • The object store SHOULD<42>: • If Open.UserSetModificationTime is FALSE, set Open.File.LastModificationTime to the current system time. • If Open.UserSetChangeTime is FALSE, set Open.File.LastChangeTime to the current system time. • If Open.UserSetAccessTime is FALSE, set Open.File.LastAccessTime to the current system time. • Set Open.File.FileAttributes.FILE_ATTRIBUTE_ARCHIVE to TRUE. <42> Section 2.1.4.17: File systems may choose to defer processing for a file that has been modified to a later time, favoring performance over accuracy. The NTFS file system on versions prior to Windows 10 v1809 operating system, Windows Server v1809 operating system, and Windows Server 2019, and non-NTFS file systems on all versions of Windows, defer this processing until the Open gets closed. Added new Section 2.1.4.19, Algorithm for Noting That a File Has Been Accessed: 2.1.4.19 Algorithm for Noting That a File Has Been Accessed The inputs for this algorithm are as follows: • Open: The Open through which the file was accessed. The pseudocode for the algorithm is as follows: • The object store SHOULD<43>: • If Open.UserSetAccessTime is FALSE, set Open.File.LastAccessTime to the current system time.

Errata Published*	Description
	<43> Section 2.1.4.19: File systems may choose to defer processing for a file that has been accessed to a later time, favoring performance over accuracy. The NTFS file system on versions prior to Windows 10 v1809, Windows Server v1809, and Windows Server 2019, and non-NTFS file systems on all versions of Windows, defer this processing until the Open gets closed. In Section 2.1.5.2, Server Requests a Read, clarified the object store behavior when the file has been accessed. Changed from: ... Pseudocode for the operation is as follows: ... ● If (ByteOffset >= Open.Stream.ValidDataLength): ● If Open.Mode.FILE_SYNCHRONOUS_IO_ALERT is TRUE or Open.Mode.FILE_SYNCHRONOUS_IO_NONALERT is TRUE, the object store MUST set Open.CurrentByteOffset to (ByteOffset + ByteCount). ● If Open.File.UserSetAccessTime is FALSE, the object store MUST update Open.File.LastAccessTime to the current system time. ● The object store MUST return: ● BytesRead set to ByteCount. ● OutputBuffer filled with ByteCount zero(s). ● Status set to STATUS_SUCCESS. ● EndIf ● If ((ByteOffset + ByteCount) >= Open.Stream.ValidDataLength), truncate ByteCount to (Open.Stream.ValidDataLength - ByteOffset). ● Set BytesToRead to BlockAlign(ByteCount,Open.File.Volume.LogicalBytesPerSector). ● Read BytesToRead bytes from the disk at offset ByteOffset for this stream into OutputBuffer. If the read from the disk failed, the operation MUST be failed with the same error status. ● If RequestedByteCount > ByteCount, zero out OutputBuffer between ByteCount and RequestedByteCount. ● If Open.Mode.FILE_SYNCHRONOUS_IO_ALERT is TRUE or Open.Mode.FILE_SYNCHRONOUS_IO_NONALERT is TRUE, the object store MUST set Open.CurrentByteOffset to (ByteOffset + RequestedByteCount). ● If Open.File.UserSetAccessTime is FALSE, the object store MUST update Open.File.LastAccessTime to the current system time. ● Upon successful completion of the operation, the object store MUST return: ● BytesRead set to RequestedByteCount. ● Status set to STATUS_SUCCESS. ● Else ● Read ByteCount bytes at offset ByteOffset from the cache for this stream into OutputBuffer. ● If Open.Mode.FILE_SYNCHRONOUS_IO_ALERT is TRUE or Open.Mode.FILE_SYNCHRONOUS_IO_NONALERT is TRUE, the object store MUST set Open.CurrentByteOffset to (ByteOffset + ByteCount). ● If Open.File.UserSetAccessTime is FALSE, the object store MUST update Open.File.LastAccessTime to the current system time. ● Upon successful completion of the operation, the object store MUST return: ● BytesRead set to ByteCount.

Errata Published*	Description
	• Status set to STATUS_SUCCESS. • EndIf Changed to: ... Pseudocode for the operation is as follows: ... • If (ByteOffset >= Open.Stream.ValidDataLength): • If Open.Mode.FILE_SYNCHRONOUS_IO_ALERT is TRUE or Open.Mode.FILE_SYNCHRONOUS_IO_NONALERT is TRUE, the object store MUST set Open.CurrentByteOffset to (ByteOffset + ByteCount). • The object store MUST note that the file has been accessed as specified in section 2.1.4.19 with Open equal to Open. • The object store MUST return: • BytesRead set to ByteCount. • OutputBuffer filled with ByteCount zero(s). • Status set to STATUS_SUCCESS. • EndIf • If ((ByteOffset + ByteCount) >= Open.Stream.ValidDataLength), truncate ByteCount to (Open.Stream.ValidDataLength - ByteOffset). • Set BytesToRead to BlockAlign(ByteCount,Open.File.Volume.LogicalBytesPerSector). • Read BytesToRead bytes from the disk at offset ByteOffset for this stream into OutputBuffer. If the read from the disk failed, the operation MUST be failed with the same error status. • If RequestedByteCount > ByteCount, zero out OutputBuffer between ByteCount and RequestedByteCount. • If Open.Mode.FILE_SYNCHRONOUS_IO_ALERT is TRUE or Open.Mode.FILE_SYNCHRONOUS_IO_NONALERT is TRUE, the object store MUST set Open.CurrentByteOffset to (ByteOffset + RequestedByteCount). • The object store MUST note that the file has been accessed as specified in section 2.1.4.19 with Open equal to Open. • Upon successful completion of the operation, the object store MUST return: • BytesRead set to RequestedByteCount. • Status set to STATUS_SUCCESS. • Else • Read ByteCount bytes at offset ByteOffset from the cache for this stream into OutputBuffer. • If Open.Mode.FILE_SYNCHRONOUS_IO_ALERT is TRUE or Open.Mode.FILE_SYNCHRONOUS_IO_NONALERT is TRUE, the object store MUST set Open.CurrentByteOffset to (ByteOffset + ByteCount). • The object store MUST note that the file has been accessed as specified in section 2.1.4.19 with Open equal to Open. • Upon successful completion of the operation, the object store MUST return: • BytesRead set to ByteCount. • Status set to STATUS_SUCCESS. • EndIf

Errata Published*	**Description**
	In Section 2.1.5.5.3, Directory Information Queries, clarified the object store behavior when the file has been accessed. Changed from: ... Pseudocode for the algorithm is as follows: ... ● If Open.File.UserSetAccessTime is FALSE, the object store MUST update Open.File.LastAccessTime to the current system time. ● The object store MUST return: ● Status set to StatusToReturn. ● OutputBuffer containing an array of as many entries that match the query as will fit in OutputBufferSize. ● BytesReturned containing the number of bytes filled in OutputBuffer. Changed to: ... Pseudocode for the algorithm is as follows: ... ● The object store MUST note that the file has been accessed as specified in section 2.1.4.19 with Open equal to Open. ● The object store MUST return: ● Status set to StatusToReturn. ● OutputBuffer containing an array of as many entries that match the query as will fit in OutputBufferSize. ● BytesReturned containing the number of bytes filled in OutputBuffer.
2019/11/25	In Section 2.1.5.11.30, FileNormalizedNameInformation, changed the pseudocode from BuildNormalizedRelativeName to BuildRelativeName. Changed from: ... Pseudocode for the operation is as follows: ... ● Set FileName to BuildNormalizedRelativeName(Open.Link, Open.File.Volume.RootDirectory). Changed to: ... Pseudocode for the operation is as follows: ... ● Set FileName to BuildRelativeName(Open.Link, Open.File.Volume.RootDirectory). Removed Section 2.1.4.19, BuildNormalizedRelativeName -- Algorithm for Building the Normalized Relative Path Name for a Link.

Errata Published*	Description
	<Deleted content> The inputs for this algorithm are: ● Link: A Link whose relative path name is being created. ● RootDirectory: A DirectoryFile indicating how far to walk up the directory hierarchy when creating the relative path name. This algorithm returns a Unicode string representing the portion of a Link's path name from the RootDirectory to the Link itself, inclusive. The returned string starts with a backslash and uses backslashes as path separators. If Link is not a descendant of RootDirectory, the algorithm returns an empty string to indicate the error. Pseudocode for the algorithm is as follows: ● If Link.File equals RootDirectory: ● Return "\". ● Else If Link.File equals Link.File.Volume.RootDirectory: ● Return an empty string. ● Else If Link.ParentFile equals RootDirectory: ● Return "\" + Link.Name. ● Else ● Set ParentRelativeName to BuildRelativeName(Link.ParentFile, RootDirectory). ● If ParentRelativeName is empty: ● Return an empty string. ● Else ● Return ParentRelativeName + "\" + Link.Name. ● EndIf ● EndIf
2019/10/16	In Section 2.1.5.14.2, FileBasicInformation, the timestamp behavior has been clarified by updating the pseudocode for the operation. Changed from: ... Pseudocode for the operation is as follows: ● If InputBufferSize is less than sizeof(FILE_BASIC_INFORMATION), the operation MUST be failed with STATUS_INFO_LENGTH_MISMATCH. ● The operation MUST be failed with STATUS_INVALID_PARAMETER under any of the following conditions: ● If InputBuffer.CreationTime is less than -1. ● If InputBuffer.LastAccessTime is less than -1. ● If InputBuffer.LastWriteTime is less than -1. ● If InputBuffer.ChangeTime is less than -1.<148> ... ● If InputBuffer.ChangeTime != 0: ● The object store MUST set Open.UserSetChangeTime to TRUE. ● If InputBuffer.ChangeTime != -1: ● Set BreakParentOplock to TRUE. ● If InputBuffer.ChangeTime !=Open.File.LastChangeTime, the object store MUST set UsnReason.USN_REASON_BASIC_INFO_CHANGE to TRUE.

Errata Published*	Description
	● The object store MUST set Open.File.LastChangeTime to InputBuffer.ChangeTime. ● EndIf ● EndIf ● If InputBuffer.CreationTime != 0 and InputBuffer.CreationTime != -1:... ● If InputBuffer.LastAccessTime != 0: ● The object store MUST set Open.UserSetAccessTime to TRUE. ● If InputBuffer.LastAccessTime != -1: ● Set BreakParentOplock to TRUE. ● If InputBuffer. LastAccessTime != Open.File.LastAccessTime, the object store MUST set UsnReason.USN_REASON_BASIC_INFO_CHANGE to TRUE. ● The object store MUST set Open.File.LastAccessTime to InputBuffer. LastAccessTime. ● The object store MUST set Open.File.PendingNotifications.FILE_NOTIFY_CHANGE_LAST_ACCESS to TRUE. ● If Open.UserSetChangeTime is FALSE and InputBuffer.ChangeTime != -1, the object store MUST set Open.File.LastChangeTime to CurrentTime. ● EndIf ● EndIf ● If InputBuffer.LastWriteTime != 0: ● The object store MUST set Open.UserSetModificationTime to TRUE. ● If InputBuffer.LastWriteTime != -1: ● Set BreakParentOplock to TRUE. ● If InputBuffer. LastWriteTime != Open.File.LastModificationTime, the object store MUST set UsnReason.USN_REASON_BASIC_INFO_CHANGE to TRUE. ● The object store MUST set Open.File.LastModificationTime to InputBuffer. LastWriteTime. ● The object store MUST set Open.File.PendingNotifications.FILE_NOTIFY_CHANGE_LAST_WRITE to TRUE. ● If Open.UserSetChangeTime is FALSE and InputBuffer.ChangeTime != -1, the object store MUST set Open.File.LastChangeTime to CurrentTime. ● EndIf ● EndIf... Changed to: ... Pseudocode for the operation is as follows: ● If InputBufferSize is less than sizeof(FILE_BASIC_INFORMATION), the operation MUST be failed with STATUS_INFO_LENGTH_MISMATCH. ● The operation MUST be failed with STATUS_INVALID_PARAMETER under any of the following conditions: ● If InputBuffer.CreationTime is less than -2. ● If InputBuffer.LastAccessTime is less than -2. ● If InputBuffer.LastWriteTime is less than -2. ● If InputBuffer.ChangeTime is less than -2.<148>... ● If InputBuffer.ChangeTime != 0: ● If InputBuffer.ChangeTime != -2: ● The object store MUST set Open.UserSetChangeTime to TRUE.

Errata Published*	Description
	• If InputBuffer.ChangeTime != -1: • Set BreakParentOplock to TRUE. • If InputBuffer.ChangeTime !=Open.File.LastChangeTime, the object store MUST set UsnReason.USN_REASON_BASIC_INFO_CHANGE to TRUE. • The object store MUST set Open.File.LastChangeTime to InputBuffer.ChangeTime. • EndIf • Else • The object store MUST set Open.UserSetChangeTime to FALSE. • EndIf • EndIf • If InputBuffer.CreationTime != 0 and InputBuffer.CreationTime != -1 and InputBuffer.CreationTime != -2: ... • If InputBuffer.LastAccessTime != 0: • If InputBuffer.LastAccessTime != -2: • The object store MUST set Open.UserSetAccessTime to TRUE. • If InputBuffer.LastAccessTime != -1: • Set BreakParentOplock to TRUE. • If InputBuffer. LastAccessTime != Open.File.LastAccessTime, the object store MUST set UsnReason.USN_REASON_BASIC_INFO_CHANGE to TRUE. • The object store MUST set Open.File.LastAccessTime to InputBuffer. LastAccessTime. • The object store MUST set Open.File.PendingNotifications.FILE_NOTIFY_CHANGE_LAST_ACCESS to TRUE. • If Open.UserSetChangeTime is FALSE and InputBuffer.ChangeTime != -1, the object store MUST set Open.File.LastChangeTime to CurrentTime. • EndIf • Else: • The object store MUST set Open.UserSetAccessTime to FALSE. • EndIf • EndIf • If InputBuffer.LastWriteTime != 0: • If InputBuffer.LastWriteTime != -2: • The object store MUST set Open.UserSetModificationTime to TRUE. • If InputBuffer.LastWriteTime != -1: • Set BreakParentOplock to TRUE. • If InputBuffer. LastWriteTime != Open.File.LastModificationTime, the object store MUST set UsnReason.USN_REASON_BASIC_INFO_CHANGE to TRUE. • The object store MUST set Open.File.LastModificationTime to InputBuffer. LastWriteTime. • The object store MUST set Open.File.PendingNotifications.FILE_NOTIFY_CHANGE_LAST_WRITE to TRUE. • If Open.UserSetChangeTime is FALSE and InputBuffer.ChangeTime != -1, the object store MUST set Open.File.LastChangeTime to CurrentTime. • EndIf • Else: • The object store MUST set Open.UserSetModificationTime to FALSE.

Errata Published*	**Description**
	● EndIf ● EndIf …
2019/09/02	In Section 2.1.1.1, Per Volume, a new ADM element has been added: ● ReservedSpace: A 64-bit unsigned integer specifying the amount of free space of the volume in bytes that is reserved for implementation specific use and not available to callers. This value MUST be a multiple of ClusterSize and MUST be less than or equal to Volume.FreeSpace. In Section 2.1.5.9.11, FSCTL_GET_NTFS_VOLUME_DATA, the following bullet point has been changed from: OutputBuffer.TotalReserved set to an implementation-specific value. Changed to: OutputBuffer.TotalReserved set to Open.File.Volume.ReservedSpace / Open.File.Volume.ClusterSize. In Section 2.1.5.9.12, FSCTL_GET_REFS_VOLUME_DATA, the following bullet point has been changed from: OutputBuffer.TotalReserved set to an implementation-specific value. Changed to: OutputBuffer.TotalReserved set Open.File.Volume.ReservedSpace / Open.File.Volume.ClusterSize. In Section 2.1.5.12.3, FileFsSizeInformation, the following bullet points have been changed from: OutputBuffer.AvailableAllocationUnits set to Open.File.Volume.FreeSpace / Open.File.Volume.ClusterSize. If RemainingQuota < Open.File.Volume.FreeSpace: Changed to: OutputBuffer.AvailableAllocationUnits set to (Open.File.Volume.FreeSpace - Open.File.Volume.ReservedSpace) / Open.File.Volume.ClusterSize. If RemainingQuota < (Open.File.Volume.FreeSpace - Open.File.Volume.ReservedSpace): In Section 2.1.5.12.7, FileFsFullSizeInformation, the following bullet points have been changed from:

Errata Published*	Description
	OutputBuffer.CallerAvailableAllocationUnits set to Open.File.Volume.FreeSpace / Open.File.Volume.ClusterSize. OutputBuffer.ActualAvailableAllocationUnits set to Open.File.Volume.FreeSpace / Open.File.Volume.ClusterSize. If RemainingQuota < Open.File.Volume.FreeSpace: Changed to: OutputBuffer.CallerAvailableAllocationUnits set to (Open.File.Volume.FreeSpace - Open.File.Volume.ReservedSpace) / Open.File.Volume.ClusterSize. OutputBuffer.ActualAvailableAllocationUnits set to (Open.File.Volume.FreeSpace - Open.File.Volume.ReservedSpace) / Open.File.Volume.ClusterSize. If RemainingQuota < (Open.File.Volume.FreeSpace - Open.File.Volume.ReservedSpace):
2019/07/08	In Section 2.1.5.11.28, FileStandardLinkInformation, the error code was changed from: This operation is not supported and MUST be failed with STATUS_INVALID_INFO_CLASS. Changed to: This operation is not supported and MUST be failed with STATUS_NOT_SUPPORTED.

*Date format: YYYY/MM/DD