Similar updates will appear a future release of MS-SMB2:

3.3.1.10 Per Open

Open.ResumeKey: A 24-byte key that identifies a source file in Server side Data Copy Operation.

3.3.5.15.5 Handling a Source File Key Request

…

The server MUST provide a 24-byte value that is used to uniquely identify the open. The server SHOULD use **Open.DurableFileId**, or alternately, MAY use an internally generated value that is unique for all opens on the server.<283>The server MUST set **Open.ResumeKey** and **ResumeKey** in SRV_REQUEST_RESUME_KEY Response to the generated value.

3.3.5.15.6 Handling a Server-Side Data Copy Request

When the server receives a request with an SMB2 header with a **Command** value equal to SMB2 IOCTL, and a **CtlCode** of FSCTL_SRV_COPYCHUNK or FSCTL_SRV_COPYCHUNK_WRITE, message handling proceeds as follows:

The server MUST locate the source open from where data will be read by locating the open ~~using the~~ ~~**SourceFile** key~~where **Open.ResumeKey** matches **SourceKey** received in the SRV_COPYCHUNK_COPY structure received in the buffer described by **InputCount** and **InputOffset** of the SMB2 IOCTL Request. If the open is not found, the server MUST fail the request with STATUS_OBJECT_NAME_NOT_FOUND.

If **OutputCount** in SMB2 IOCTL request is less than the size of SRV_COPYCHUNK_RESPONSE structure, the server MUST fail the SMB2 IOCTL request with STATUS_INVALID_PARAMETER.

If **OutputCount** in SMB2 IOCTL request is greater or equal to the size of SRV_COPYCHUNK_RESPONSE structure and under any of the following conditions, the server MUST send an SMB2 IOCTL Response as specified in section 3.3.5.15.6.2:

- **InputCount** in SMB2 IOCTL request is less than the size of Buffer field containing SRV_COPYCHUNK_COPY structure
- **ChunkCount** is greater than **ServerSideCopyMaxNumberofChunks**
- **Length** in a single chunk is greater than **ServerSideCopyMaxChunkSize** or equal to zero
- Sum of **Length**s in all chunks is greater than **ServerSideCopyMaxDataSize**
- **TargetOffset** in any chunk is less than zero but not equal to 0xFFFFFFFFFFFFFFFF.
- **Open.TreeConnect** of the source or destination file is on a named pipe filesystem

If **Open.GrantedAccess** of the destination file does not include FILE_WRITE_DATA or FILE_APPEND_DATA, then the request MUST be failed with STATUS_ACCESS_DENIED. If **Open.GrantedAccess** of the ~~destination~~source file does not include FILE_READ_DATA access ~~and the~~

~~**CtlCode** is FSCTL_SRV_COPYCHUNK~~, then the request MUST be failed with STATUS_ACCESS_DENIED.

If **Open.TreeConnect.Session** of the destination file is not equal to **Open.TreeConnect.Session** of the source file, the server MUST fail the request with STATUS_OBJECT_NAME_NOT_FOUND.

~~If the server is configured to limit the amount of data copied in a single server side copy operation, and the size of data sent in the request exceeds the limit, the server MUST send an SMB2 IOCTL Response as specified in Sending an Invalid Parameter Server-Side Copy Response (section 3.3.5.15.6.2).~~

Starting with the first chunk received in the Chunks field and for each chunk, the server MUST apply the following processing rules:

- ~~For each range,~~ The server MUST issue a read using the **SourceOffset** and **Length** from the source file.<284> If the **SourceOffset** or **SourceOffset** + **Length** extends beyond the end of file, the server SHOULD<285> treat this as a STATUS_END_OF_FILE error. If the read fails, the server MUST map the error code returned to a valid status code as described in section 2.2, and MUST send an SMB2 IOCTL response as specified in Sending a Copy Failure Server-Side Copy Response (section 3.3.5.15.6.1).

- If the read operation is successful, the server MUST issue a write of the data read using the **TargetOffset** and **Length** in the range against the destination file.<286> If the write fails, the server MUST send an SMB2 IOCTL response as specified in Sending a Copy Failure Server-Side Copy Response (section 3.3.5.15.6.1).

If all ranges are copied successfully, the server MUST construct an SMB2 IOCTL Response following the syntax specified in the section 2.2.32, with the following values:

- **CtlCode** MUST be set to FSCTL_SRV_COPYCHUNK or FSCTL_SRV_COPYCHUNK_WRITE.
- **FileId.Persistent** MUST be set to **Open.DurableFileId**. **FileId.Volatile** MUST be set to **Open.FileId**.
- **InputOffset** SHOULD be set to the offset, in bytes, from the beginning of the SMB2 header to the **Buffer[]** field of the response.
- **InputCount** SHOULD be set to zero.
- **OutputOffset** MUST be set to **InputOffset** + **InputCount**, rounded up to a multiple of 8.
- **OutputCount** MUST be set to 12.
- **Flags** MUST be set to zero.
- The server MUST copy a SRV_COPYCHUNK_RESPONSE following the syntax specified in section 2.2.32.1 into the **Buffer** field at the **OutputOffset** computed above. **ChunksWritten** MUST be set to the number of chunks processed. **ChunkBytesWritten** MUST be set to zero. **TotalBytesWritten** MUST be set to the total number of bytes written to the destination file across all chunk writes.

The response MUST be sent to the client.

3.3.5.15.6.1 Sending a Copy Failure Server-Side Copy Response

If a range is encountered that is not copied successfully, the server MUST construct an SMB2 IOCTL Response following the syntax specified in section 2.2.32, with the following values:

- **Status** in the SMB2 header MUST be set to the error returned during processing specified in section 3.3.5.15.6~~one of the error codes listed in section 3.3.5.15~~.
- **CtlCode** MUST be set to the CtlCode in SMB2 IOCTL Request~~FSCTL_SRV_COPYCHUNK or FSCTL_SRV_COPYCHUNK_WRITE~~.
- **FileId.Persistent** MUST be set to **Open.DurableFileId**. **FileId.Volatile** MUST be set to **Open.FileId**.
- **InputOffset** SHOULD be set to the offset, in bytes, from the beginning of the SMB2 header to the **Buffer[]** field of the response.
- **InputCount** SHOULD be set to zero.
- **OutputOffset** MUST be set to **InputOffset** + **InputCount**, rounded up to a multiple of 8.
- **OutputCount** MUST be set to 12.
- **Flags** MUST be set to zero.
- The server MUST copy a SRV_COPYCHUNK_RESPONSE following the syntax specified in section 2.2.32.1 into the **Buffer** field at the **OutputOffset** computed above. **ChunksWritten** MUST be set to the number of chunks successfully written. If the error was encountered partway through a write, **ChunkBytesWritten** MUST be set to the number of bytes written in the final, partial write. Otherwise, **ChunkBytesWritten** MUST be set to 0. **TotalBytesWritten** MUST be set to the total number of bytes written to the destination file across all chunk writes.

The response MUST be sent to the client.

3.3.5.15.6.2 Sending an Invalid Parameter Server-Side Copy Response

~~If the server determines that the total chunk count is more than **ServerSideCopyMaxNumberofChunks**, or the size of any chunk is more than **ServerSideCopyMaxChunkSize**, or the total size of all chunks exceeds **ServerSideCopyMaxDataSize**,~~ the server MUST construct an SMB2 IOCTL Response, following the syntax specified in section 2.2.32, with the following values:

- **Status** in the SMB2 header MUST be set to STATUS_INVALID_PARAMETER.
- **CtlCode** MUST be set to the CtlCode in SMB2 IOCTL Request~~FSCTL_SRV_COPYCHUNK or FSCTL_SRV_COPYCHUNK_WRITE~~.
- **FileId.Persistent** MUST be set to **Open.DurableFileId**. **FileId.Volatile** MUST be set to **Open.FileId**.
- **InputOffset** SHOULD be set to the offset, in bytes, from the beginning of the SMB2 header to the **Buffer[]** field of the response.
- **InputCount** SHOULD be set to zero.
- **OutputOffset** MUST be set to **InputOffset** + **InputCount**, rounded up to a multiple of 8.

- **OutputCount** MUST be set to 12.
- Flags MUST be set to zero.
- The server MUST copy a SRV_COPYCHUNK_RESPONSE, following the syntax specified in section 2.2.32.1, into the **Buffer** field at the **OutputOffset** computed above, with the following differences. **ChunksWritten** MUST be set **ServerSideCopyMaxNumberofChunks**. **ChunkBytesWritten** MUST be set **ServerSideCopyMaxChunkSize**. **TotalBytesWritten** MUST be set to **ServerSideCopyMaxDataSize**.

The response MUST be sent to the client.