# Microsoft Networks

# SMB File Sharing Protocol Extensions

## Document Version 3.4

April 18, 2021

μ

# 1. Introduction

This document is meant as amendment to the existing SMB documents and as such does not discuss the SMB protocol in general. Refer to Version 3.3 for a thorough description of the protocol. This document only discusses those changes to the protocol required for compatibility with the Lanman 2.1 protocol dialect.

# 2. Negotiate Protocol

## 2.1 NegProt Resp SMB

The dialects requested for the LM2.1 protocol are

"LANMAN2.1" for the OS/2 clients and
"DOS LANMAN2.1" for DOS clients.

When one of the above protocols is negotiated, a new response is returned to the NegProt SMB. The new response is represented by the data structure below (* indicates change from earlier protocol).

| | | |
|---|---|---|
| | byte | smb_wct |
| | word | smb_index |
| | word | smb_secmode |
| | word | smb_maxxmt |
| | word | smb_maxmux |
| | word | smb_maxvcs |
| | word | smb_blkmode |
| | dword | smb_sesskey |
| | word | smb_srv_time |
| | word | smb_srv_date |
| | word | smb_srv_tzone |
| * | word | smb_cryptkeylen |
| * | word | smb_rsvd |
| | word | smb_bcc |
| | byte | smb_cryptkey |
| * | byte | smb_domain[] |

In earlier versions of this protocol **smb_rsvd** was a dword, and the encryption key length was indicated by **smb_bcc**. There is a new field, **smb_domain**, that is a null terminated string that contains the name of the domain that this server is a member of. With the addition of this field, **smb_bcc** now represents the total bytes present due to both **smb_cryptkey** and **smb_domain** . To account for this **smb_rsvd** was split and the new field **smb_cryptkeylen** was created to represent the length of the **smb_cryptkey** field.

# 3. Session Setup and X

## 3.1 SessSetupX Request

The SessSetupX request SMB for the LM2.1 protocol has an additional 3 fields. These fields are used to indicate, the name of the domain on which the client was authenticated (**smb_domain**), the type of operating system the client machine is using (**smb_nativeos**) and what kind of LAN Manager software the client is using (**smb_nativelm**). All theses fields are null terminated strings and their orientation in the SMB is indicated in the data structure below (* indicates new field).

```
        byte    smb_wct          /* value = 10 */
        byte    smb_com2         /* secondary (X) command, 0xFF = none */
        byte    smb_reh2         /* reserved, MBZ */
        word    smb_off2         /* offset (from SMB header) to next cmd (@smb_wct) */
        word    smb_bufsize      /* the consumers max buffer size */
        word    smb_mpxmax       /* actual max multiplexed pending requests */
        word    smb_vc_num       /* 0 = first only, non zero - additional VC number */
        dword   smb_sesskey      /* Session key (valid only if smb_vc_num != 0) */
        word    smb_apasslen     /* size of account password (smb_apasswd) */
        dword   smb_rsvd         /*  reserved */
        word    smb_bcc          /* minimum value = 0 */
        byte    smb_apasswrd[*]/* account password (* = smb_apasslen value) */
        byte    smb_aname[]      /* account name string*/
*       byte    smb_domain[]     /* name of domain that client was authenticated on */
*       byte    smb_nativeos[]   /* native operating system of client */
*       byte    smb_nativelm[]   /* native LAN Manager type */
```

Some examples of the **smb_nativeos** field might be,
"OS/2 1.0", "OS/2 1.21", "MS-DOS 5.0", "Unix BSD 4.0", ... etc.

Examples of **smb_nativelm** field might be,
"LAN Manager 2.1", "LAN Manager 2.1", "LAN Server 2.0", ... etc.

## 3.1 SessSetupX Response

The SessSetupX response SMB for the LM 2.1 protocol contains 2 additional fields. These fields are the servers corresponding smb_nativeos and smb_nativelm fields. These fields are null terminated strings. There orientation in the SMB are indicated in the data structure below (* indicates new field).

```
        byte    smb_wct          /* value = 3 */
        byte    smb_com2         /* secondary (X) command, 0xFF = none */
        byte    smb_res2         /* reserved (pad to word) */
        word    smb_off2         /* offset (from SMB header) to next cmd (@smb_wct) */
        word    smb_action       /* request mode: bit0 = logged on successfully - but as guest */
        word    smb_bcc          /* min value = 0 */
*       byte    smb_nativeos[]   /* server's native operating system */
*       byte    smb_nativelm[]   /* server's native LM type */
```

# 4. Tree Connect and X SMB

## 4.1 TreeConnX Response

The response to the TreeConnX SMB returns 2 new fields for the LM2.1 dialect. With the addition of some new features in various operating systems it has become necessary to differentiate which of these features can be taken advantage of by the consumers on a connection basis. The exclusive search feature of OS/2 1.3 is an example of such features. The **smb_optsupp** field now returned by the server can be used to determine what if any of these features are available. In addition to this field the **smb_nativefs** field has been added to help the consumer determine the type of file system this connection is to. The **smb_optsupp** field is a word of bit masks and the **smb_nativefs** field is a null terminated string. There orientation is the smb is indicated below (* indicates new field).

```
        byte    smb_wct         /* value = 3 */
        byte    smb_com2        /* secondary (X) command, 0xFF = none */
        byte    smb_res2        /* reserved (pad to 0) */
        word    smb_off2        /* offset (from SMB header) to next cmd (@smb_wct) */
*       word    smb_optsupp     /* bit mask indicating advanced OS features available */
                                /* bit0 = 1, exclusive search bits supported */
        word    smb_bcc         /* minimum value = 3 */
        byte    smb_nativefs[]  /* native file system for this connection */
```

Server's that can support the new search bits defined below will identify themselves by setting bit0 of **smb_oppsupp** to 1. Note that this allows the server to optionally support these features on a per connection basis.

For servers that negotiate support of the exclusive search bits, the new search bits will be passed along in the **smb_attr** field of the Find, FindUnique, Search, Trans2 and FindFirst SMBs. This change will not be further documented individually later in this spec. All these SMBs will interpret the new bits of the **smb_attr** field as follows. New bits are 8 - 13.

| **Bit** | **Meaning** |
|---|---|
| 13 | If set, only files marked as archive are included |
| 12 | If set, only directories are included |
| 11 | Meaningless |
| 10 | If set, only files marked as system are included |
| 9 | If set, only files marked as hidden are include |
| 8 | If set, only files marked as read only are included |

# 5. Locking and X SMB

## 5.1 LockingX SMB Request

The LockingX request SMB has modified the meaning of the previous **smb_locktype** field to take advantage of new locking features provided in various operating systems. The new LockingX request SMB is detailed below (* indicates changes from earlier protocols).

```
            byte        smb_wct             /* value = 8 */
            byte        smb_com2            /* secondary (X) command, 0xFF = none */ byte        smb_reh2            /* reserved
(must be zero) */ word      smb_off2            /* offset (from SMB hdr start) to next cmd (@smb_wct) */     word
            smb_fid             /* file handle */ /* word      smb_lockflags     /* locking mode: */                  /* bit0 = 0,
Lock out all access;  bit0 = 1,  Read ok while locked */                  /* bit1 = 1,  Single user total file unlock (OpLock
Break) */
                        /* bit2 = 1, Requesting change lock type on supplied smb_lockrng[] */
                        /* bit3 = 1, Requesting cancel of lock specified in smb_unlkrng[] */     dword    smb_timeout;      /*
number of milliseconds to attempt each lock */         word      smb_unlocknum; /* number of unlock range structures
following */       word      smb_locknum;     /* number of lock range structures following */         word      smb_bcc;
            /* total bytes following */ struct      smb_unlkrng[*];  /* unlock range structures (* = smb_unlocknum) */   struct
            smb_lockrng[*];  /* lock range structures (* = smb_locknum) */
```

The consumer may take advantage of the feature supplied in OS/2 2.0 of being able to convert the type of lock held on a region without first unlocking the range by using Bit2 of **smb_lockflags** in the following manner. The consumer should specify the new lock type by setting Bit0 appropriately for the lock range specified in the **smb_lockrng** field. The range specified should already be locked or the Server will return an error. As normal the consumer should expect to wait **smb_timeout** milliseconds in the case of a lock conflict.

By using Bit3 of **smb_lockflags** the consumer may take advantage of the ability to cancel a lock request in OS/2 2.0. The consumer specifies the range to apply the cancel to in the **smb_unlkrng** field and sets Bit3 to 1.

## 6. Unlock SMB correction

It has been determined that previous versions of the SMB spec have incorrectly stated that an Unlock SMB sent on a range that was not previously locked would result in a No-Op. This is incorrect. This will result in a Locking Violation error returned by the server.

## 7. T2FindFirst SMB

In the SMB specification for LM 2.1 (SMB File Sharing Protocol Extensions Version 3.0) it was specified that a consumer using the T2FindFirst SMB could only identify non-8.3 filenames. This limitation has been partially removed in this version of the SMB protocol. Consumers may now request 8.3 only file names with a T2FindFirst SMB as long as the search level is 1.

## 8. Extension for Open Modes on the OpenX and T2Open SMBs

For the OpenX and T2Open request SMBs the open mode (**smb_mode**) has been extended in the following way. Bit 12 and Bits 8-10 are now interpreted as follows.

| Bit12 | Cache Usage Bit |
|---|---|
| 0 | This open is treated as any other opened file |
| 1 | The consumer is advising not to cache the data in this file. |

| Bits8-10 | Locality of Reference Bits |
|---|---|
| 000 | Locality of reference is not known |
| 001 | Mainly Sequential Access |
| 010 | Mainly Random Access |
| 011 | Random with some Locality |
| 1XX | Currently Undefined |

These Bits have no mandated functionality and are passed to assist the server in optimizing memory usage. They should originate form the consumer performing the open and should provide "hints" on the consumers intended usage of the file.