|  |  |  |  |
| --- | --- | --- | --- |
| Server Message Block (SMB) Protocol Specification (MS-SMB) | | | |
| Test Suite Version | 1.0.0.0 | TD (Technical Document) Version and date | 23.0, v20100406 |
| <Test Design Spec> | | | |

**Based on Template version 3.0**

[1 Document Review 2](#_Toc280363094)

[1.1 Introduction 2](#_Toc280363095)

[1.1.1 Overview 2](#_Toc280363096)

[1.1.2 Relationship to Other Protocols 2](#_Toc280363097)

[1.1.3 Message 3](#_Toc280363098)

[1.2 Protocol Properties 6](#_Toc280363099)

[1.3 Test Approach 6](#_Toc280363100)

[1.4 Adapter Method 7](#_Toc280363101)

[2 Test Suite Planning 8](#_Toc280363102)

[2.1 Assumptions, Scope and Constraints 8](#_Toc280363103)

[2.1.1 Assumptions 8](#_Toc280363104)

[2.1.2 Scope 8](#_Toc280363105)

[2.1.3 Out of Scope 8](#_Toc280363106)

[2.1.4 Constraints 9](#_Toc280363107)

[2.2 Test Suite Approach 9](#_Toc280363108)

[2.2.1 Test Suite Overview 9](#_Toc280363109)

[2.2.2 Model Plan 10](#_Toc280363110)

[2.3 Technical Feasibility of Message Generation/Adapter Approach 13](#_Toc280363111)

[2.4 Considerations 14](#_Toc280363112)

[2.4.1 Encryption Considerations 14](#_Toc280363113)

[2.4.2 Other Considerations 14](#_Toc280363114)

[2.5 Scenarios 14](#_Toc280363115)

[3 Test Suite Design 18](#_Toc280363116)

[3.1 Model or Test Design Description 18](#_Toc280363117)

[3.1.1 Model Design 18](#_Toc280363118)

[3.1.2 Scenario Machine Design 20](#_Toc280363119)

[3.2 Test Cases 21](#_Toc280363120)

[3.3 Adapter Design 28](#_Toc280363121)

[3.3.1 Protocol Adapter Design 28](#_Toc280363122)

[3.3.2 SUT Control Adapter Design 29](#_Toc280363123)

[3.3.3 Sequence Diagram 29](#_Toc280363124)

[4 References 32](#_Toc280363125)

[4.1 Related Links 32](#_Toc280363126)

# Document Review

## Introduction

### Overview

[Common Internet File System (CIFS)](http://www.microsoft.com/about/legal/intellectualproperty/protocols/BSTD/CIFS/draft-leach-cifs-v1-spec-02.txt) defines an open cross-platform mechanism for client systems to request file and print services from server systems over network.

Server Message Block (SMB) Protocol supports the sharing of files, print services and interprocess communication (IPC) mechanisms such as namepipe and mailslot. It defines extensions to the existing Common Internet File System (CIFS), and extends the CIFSwith additional security, file, and disk management support. These extensions introduce new structures, new information levels, and extensions of CIFS request and response messages.

*Figure 1‑1: Classification of SMB Supported Shares*

The SMB Protocol provides support for the following features:

* File & File System
* Enumeration to previous versions of files.
* Access to previous versions of files.
* Server-side data copy method by which can be done entirely on the server-side without consuming network resources.
* Support native file and file system information levels in query and set operations.
* Support more extended information in response to share connection and file open operations.
* Security
* New authentication methods, including Kerberos and NT LAN Manager (NTLM).
* The Negotiate and Session Setup commands have been enhanced to carry opaque security tokens to support mechanisms compatible with the Generic Security Service (GSS).
* Message signing to guarantee packet integrity.

### Relationship to Other Protocols

Figure 1-2 illustrates the relationships between MS-SMB and other protocols.



*Figure 1-2 Relationships to other protocols*

* MS-SMB is an extension of MS-CIFS.
* The transports of SMB can be Direct TCP (Transmission Control Protocol), NetBIOS over IPX or NetBIOS Extended User Interface (NetBEUI).
* [MS-FSA] is algorithms, concepts and informative reference, it is not necessary to contain [MS-FSA] in the relationship diagram.

### Message

Message Syntax

* SMB message is the payload packet encapsulated in a transport packet.
  + Direct TCP:



*Figure 1‑3: SMB Packet in Direct TCP*

* + NetBIOS over IPX:

 *Figure 1‑4: SMB Packet in NetBIOS over IPX*

* + NetBIOS Extended User Interface (NetBEUI):

*Figure 1‑5: SMB Packet in NETBEUI*

* SMB message consists of two parts:
* SMB header, which has a fixed-length SMB protocol header
* SMB payload, which has a variable length depending on the SMB command type in header that the message represent.
* SMB Message Classification:

|  |  |
| --- | --- |
| SMB\_COM\_NEGOTIATE | Request and response messages used to handle dialect and capability negotiation between server and client. |
| SMB\_COM\_SESSION\_SETUP | Request and response messages used to session setup. They can be used to begin user authentication on an SMB connection and establish an SMB session. |
| SMB\_COM\_TREE\_CONNECT | Request and response messages used to create tree connection, which is a connection point to a share that exists on a server. |
| SMB\_COM\_NT\_CREATE | Request and response messages used to create or open a share on a server, including file, printer and IPC. |
| SMB\_COM\_OPEN | Request and response messages used to open a share on a server, including file, printer and IPC. Those messages are obsolete as SMB\_COM\_NT\_CREATE can do the same job. |
| SMB\_COM\_TRANSACTION | Request and response messages used to execute a named pipe or mailslot operation on a server. (The status of the two subcommands in this message is obsolescent.) |
| SMB\_COM\_TRANSACTION2 | Request and response messages used to execute a specific operation on the server. It includes file enumeration, query and set file attribute operations, and DFS referral retrieval. |
| SMB\_COM\_NT\_TRANSACTION | Request and response messages used to specify operations on the server. It includes file open, file create, device I/O control, file rename, notify directory change, and set and query security descriptors. |
| SMB\_COM\_READ\_ANDX | Request and response messages used to read from a file or named pipe on a server. |
| SMB\_COM\_WRITE\_ANDX | Request and response messages used to write data to a file or named pipe on a server. |

*Table 1‑1: SMB Message Classification*

Message Flow



*Figure 1‑6: Message Flow of SMB*

Notice:Every successful operation must follow the three steps to create the SMB connection: negotiate, session setup and tree connect. After that, shares on remote server can be accessed in different operations. As there are more than 30 kinds of operations in SMB, we just call these operations as resource request/response instead of listing the details in the diagram. All of these operations follow a request/response pattern in which the client initiates all requests except oplock breaks.

Here is a sample of SMB file open operation:

*Figure 1‑7: Sample of SMB File Open Operation*

As shown in the figure above, the typical message flow between the client and server is as follows:

1. For negotiating a dialect between server and client:
   * The client sends Negotiate Request to the server.
   * The server sends Negotiate Response to the client.
2. For setting up a SMB session:
   * The client sends SessionSetup Request to the server.
   * The server sends SessionSetup Response to the client.
3. For Creating a connection to a specific share:
   * The client sends TreeConnect Request to the server.
   * The server sends TreeConnect Response to the client.
4. For Opening a file:
   * The client sends Create Request to the server.
   * The Server sends Create Response to the client.

## Protocol Properties

Protocol properties:

* SMB is a Client/Server protocol.
* SMB is a block, synchronous protocol.
* All data exchanged on the network is not encrypted. Clients and server can choose to sign the message or not.
* The transports of SMB can be Direct TCP, NetBIOS over IPX or NetBIOS Extended User Interface (NetBEUI).
* MS-SMB is both an MCPP (Microsoft Communication Protocol Program) and a WSPP (Workgroup Group Protocol Program) program protocol modified based on Windows 7.

## Test Approach

Recommended Test Approach:

The test suite uses a hybrid testing approach including Model Based Testing (MBT) and Traditional Testing.

Reasons for MBT

Test Suite Developer (TSD) adopts Model Based Testing to verify most of the requirements:

* There is a similar protocol named MS-SMB which has succeeded in MBT.
* MS-SMB is a state-full protocol which contains 64 messages. MBT is suitable to be used as the testing method.
* Large amount of test cases are needed in the SMB test suite, and MBT can help us to generate them automatically.

Reasons for Traditional Testing:

TSD also adopt Traditional Testing to cover cases which are hard for model to cover as there may be some test cases about time event.

MS-SMB protocol of the current version has been rewritten comparing with the TD (Technical Document) v20090719. The test suite has been updated according to the changes in TD.

## Adapter Method

Adapter Overview

One Protocol Adapter, one Interactive Adapter and one SUT (System Under Control) Adapter are designed for the MS-SMB Test Suite.

Protocol Adapter

* The protocol adapter uses managed code.
* The protocol adapter sends 10 types of messages to the SUT, receives responses from the SUT, parses the messages and capture requirements.

Interactive Adapter

As on some platforms, the SUT Control Adapter can’t be configured automatically, an Interactive adapter is added for avoiding this issue.

The Interactive Adapter is implemented by PowerShell scripts, and the functionalities are the same as SUT Control Adapter functionalities, which are:

* The adapter provides interfaces through which test cases can perform configurations on the server.
* The adapter provides interfaces through which disk I/O errors can be raised.
* The adapter provides interfaces through which a generic server open failure can be raised.

SUT control adapter

It is implemented using PowerShell Script and it is to:

* Create/delete a remote queue.
* Send messages to a remote queue, or retrieve unique lookup identifier from a remote queue.
* Create a distributed atomic transaction for receiving or moving the messages.

Message Generation

MS-SMB test suite generates all messages synthetically.

The ServerAdapter will invoke the protocol SDK (Software Development Kit) to generate all the messages in this protocol, and then sends it to the server. Then the ServerAdapter will use the protocol SDK to receive and parse the messages. Later the ServerAdapter will send the parsed result to relevant test case for further validation.

# Test Suite Planning

## Assumptions, Scope and Constraints

### Assumptions

* The test suite assumes that all the shares (disk share, named pipe share, printer share and device share) already exist before running test suite.

WHY:All the shares can be configured according to deployment guide. So it is not necessary to configure by the test suite.

### Scope

* The test suite will verify 20 SMB operations out of 23 SMB operations. For the purpose of better analyzing the protocol, the 20 SMB operations are partitioned into 6 Facets, as follows:

|  |  |
| --- | --- |
| **Facets** | **SMB Operations** |
| Connection | SMB\_COM\_ NEGOTIATE |
| Session | SMB\_COM\_ SESSION\_SETUP\_ANDX |
| Tree | SMB\_COM\_ TREE\_CONNECT\_ANDX |
| SMB Common Operations  (4 operations) | SMB \_COM\_NT\_CREATE\_ANDX |
| SMB\_COM\_OPEN\_ANDX |
| SMB\_COM\_READ\_ANDX |
| SMB\_COM\_WRITE\_ANDX |
| SMB Transaction2 Operations  (8operations) | TRANS2\_FIND\_FIRST2:   * SMB\_FIND\_FILE\_BOTH\_DIRECTORY\_INFO * SMB\_FIND\_FILE\_ID\_FULL\_DIRECTORY\_INFO * SMB\_FIND\_FILE\_ID\_BOTH\_DIRECTORY\_INFO |
| TRANS2\_FIND\_NEXT2:   * SMB\_FIND\_FILE\_ID\_FULL\_DIRECTORY\_INFO * SMB\_FIND\_FILE\_ID\_BOTH\_DIRECTORY\_INFO |
| TRANS2\_QUERY\_FS\_INFORMTION |
| TRANS2\_SET\_FS\_INFORMATION |
| TRANS2\_QUERY\_PATH\_INFORMATION |
| TRANS2\_SET\_PATH\_INFORMATION |
| TRANS2\_QUERY\_FILE\_INFORMATION |
| TRANS2\_SET\_FILE\_INFORMATION |
| SMB NT Transaction Operations  (5 operations) | NT\_TRANSACT\_QUERY\_QUOTA |
| NT\_TRANSACT\_SET\_QUOTA |
| FSCTL\_SRV\_ENUMERATE\_SNAPSHOTS |
| FSCTL\_SRV\_REQUEST\_RESUME\_KEY |
| FSCTL\_SRV\_COPYCHUNK |

*Table 2‑1: Operations that will be tested in MS-SMB scope*

* The test suite is designed to cover all the requirements of MS-SMB on a generic file system.
* The test suite is going to test requirements of [MS-FSCC] (File System Control Codes) that are related to [MS-SMB].

Further explanation: [MS-FSCC] provides the information of structures and flags related to file system. The test suite will verify structures and flags of [MS-FSCC] that are documented explicitly in or referenced by [MS-SMB]. These requirements are contained in <MS-SMB_ServerRequirementSpec.xlsm>.

### Out of Scope

* The test suite will not verify the below operations in test suite of SMB:

WHY: The below operations are described only in CIFS instead of SMB, and there is no testable requirement in MS-SMB TD about them.

|  |  |
| --- | --- |
| **SMB Commands** | **Sub Commands** |
| SMB\_COM\_TRANSACTION2 | |  | | --- | | TRANS2\_OPEN2 | |
| |  | | --- | | TRANS2\_FSCTL | |
| |  | | --- | | TRANS2\_IOCTL2 | |
| |  | | --- | | TRANS2\_FIND\_NOTIFY\_FIRST | |
| |  | | --- | | TRANS2\_FIND\_NOTIFY\_NEXT | |
| |  | | --- | | TRANS2\_CREATE\_DIRECTORY | |
| |  | | --- | | TRANS2\_SESSION\_SETUP | |
| |  | | --- | | TRANS2\_REPORT\_DFS\_INCONSISTENCY | |
| SMB\_COM\_NT\_TRANSACTION | |  | | --- | | NT\_TRANSACT\_SET\_SECURITY\_DESC | |
| |  | | --- | | NT\_TRANSACT\_NOTIFY\_CHANGE | |
| |  | | --- | | NT\_TRANSACT\_QUERY\_SECURITY\_DESC | |

*Table 2‑2: Operations out of MS-SMB scope*

* The test suite will not verify SMB\_COM\_TRANSACTION on named pipe and mailslot.

WHY: The commands have been obsolescent.

### Constraints

* The test suite tests a single connection instead of multiple connections.

WHY:There is no behavior specified for multiple connections in TD.

## Test Suite Approach

### Test Suite Overview

Test Approach

The test suite uses a hybrid testing approach including MBT and Traditional Testing.

System Under Test (SUT)

From the third party’s point of view, the SUT is an implementation of MS-SMB.

On a Windows platform, the SUT is SMB server.



*Figure 2‑1: SMB Test Suite Architecture*

Further explanation to Figure 2-1:

* There are two kinds of test cases, one is generated by SMB model and the other one is traditional test case.
* The SUT control adapter or interactive adapter implements create file and share file on SUT.
* The protocol adapter calls Protocol SDK to generate and send the request messages, receive and parse the response messages.
* TCP works as the transport between the test suite and SUT.

### Model Plan

SMB model is separated into 3 C# files: Model, Transaction2Model and NTTransactionModel.

* Model: Covers the operations related to connections and creating files.
* Transaction2Model: Covers the operations related to SMB\_COM\_TRANSACTION2 Extensions.
* NTTransactionModel: Covers the Clarification and operations related to SMB\_COM\_NT\_TRANSACTION Extensions, except the operations included in [Section 2.1.3](#s213).

#### Model Abstraction

Action Abstraction:

A total of 81 actions are abstracted in the test suite.

* 75 actions are abstracted based on the 20 types of message exchange from the protocol.
* 18 types of operations have 3 corresponding actions: a request, a response and an error response.
* The operation SMB\_COM\_NEGOTIATE has 4 actions abstracted, which are NegotiateRequest, NegotiateResponse, NonExtendedNegotiateResponse and ErrorNegotiateResponse.
* The operation SMB\_COM\_SESSION\_SETUP\_ANDX has 5 actions abstracted, which are SessionSetupRequest, NonExtendedSessionSetupRequest, SessionSetupResponse, NonExtendedSessionSetupResponse and ErrorSessionSetupResponse.
* FsctlBadCommandRequest: This action is used to send invalid FSCTL command to server.
* ErrorFsctlBadCommandResponse: This action is the response to the FSCTLBadCommandRequest action.
* ErrorInvalidCommandResponse: This action is the response to the BadCommandRequest action.
* 3 actions related to TRANS2\_SET\_FS\_INFORMATION, which are Trans2SetFsInfoRequestAdditional,Trans2SetFsInfoResponseAdditional and ErrorTrans2SetFsInfoResponseAdditional
* 3 actions related to NT\_TRANSACT\_SET\_QUOTA, which are Trans2SetFsInfoRequestAdditional, Trans2SetFsInfoResponseAdditional and ErrorTrans2SetFsInfoResponseAdditional.
* 3 actions related to Session setup, which are SessionSetupRequestAdditional,SessionSetupResponseAdditional and ErrorSessionSetupResponseAdditional.
* 6 actions are defined by the test suite:
* SmbConnectionRequest: This action initializes the SMB underlying transport connection.
* SmbConnectionResponse: The response to the SMBConnectionRequest action.
* ServerSetup: This action sets up the server side environment, including message signing requirement and maximum read/write size.
* ServerSetupResponse: The response to the ServerSetup action.
* CreatePipeAndMailslot: This action creates a named pipe or mailslot at the server side, in order to enable client create its instances.
* CheckPreviousVersion: This action checks the previous version for the designated file identifier.

The 7 abstract data model classes are SmbConnection class, SmbFile class, SmbPipe class, SmbSession class, SmbShare class, SmbTree and SmbMailslot.

SMB Helper Methods Abstraction

There are 4 classes designed as helper methods.

* Checker is used to check all the request and response messages.
* Parameter is used to test the overall variable definitions for environment setup.
* Update is used to update local data according to different messages.
* ModelHelper is used to capture requirements.

SMB Data Abstraction

In SMB models, there are 7 classes abstracted to simulate the state of both client and server.

* SmbConnection: SmbConnection indicates the network connection between client and server while an SMB connection is active on the SMB transport
* SmbFile: SmbFile maintains necessary file information data according to all file operations. SMBFile serves as a file server image at the client side in order to verify all file operations.
* SmbPipe: SmbPipe maintains necessary named pipe data according to all named pipe operations. SMBPipe serves as a pipe server image at the client side in order to verify all named pipe operations.
* SmbMailslot: SmbMailslot maintains necessary file information data according to all mailslot operations. SMBMailslot serves as a mailslot server image at the client side in order to verify all mailslot operations.
* SmbSession: SmbSession represents the session established over a connection. Client can establish multiple sessions over one connection.
* SmbShare: SmbShare represents the share information at server side.
* SmbTree: SmbTree Connect represents the tree-connect information for a share. A client can establish multiple trees.

#### Model pattern

* Abstract Identifiers Pattern:
* The MS-SMB model explores a set of simple types (int, enum, string and bool) as placeholders for the concrete values that will appear during protocol testing. For example, the actual values of sessionId, treeId and fileId cannot be identified at modeling time, so these variables are replaced by abstract identifiers, and the real data is stored in the protocol adapter.
* Synchronous Protocols Pattern:
* The MS-SMB model uses the underlying transport TCP to make the RPC calls when performing the other functionalities except event notification. This model pattern offers an appearance of a local call/return mechanism to verify the logic of the protocol, including the parameters being passed in both directions and the return values. For example, client sends SMB Create Request to server and synchronously receives the SMB Create Response from server.
* Server Initialization Pattern:
* The MS-SMB model sets up the server, including create folder and share folder. For example, when setting up server environment, Action ServerSetup is called to configure enableSign which is used to set the server to support signature or not.
* Check OS Version Pattern
* The MS-SMB model checks the SUT OS version on which the requirements are to be verified and stores this information as a variable for further use.
* Check Return Value Pattern
* The MS-SMB model verifies protocol model actions’ return values to cover the related requirements. For example, the model will verify the enumeration MessageStatus in the responses.
* Helper File Pattern:
* The MS-SMB model adds a new file ModelHelper.cs in Model project, and designs the capture requirement method in ModelHelper.cs file.

#### States in Model

* Before establishing the session: Global variable “smbState” is used for binding the model logic and state transfer.
* After establishing the session, all the abstracted data defined in connection are used for state transfer.
* Two accepting conditions are defined in model program:
  + - The session is closed because of error response;
    - All the requests have received their associated responses after Tree Connection is established.

#### Distribution between Model and Adapters

The test suite will verify messages sent from the server, the model is responsible for driving adapters and verifying requirements relevant to logic, and the adapters verify the structures and values of packets.

* Model verifies if the received message is what the test suite expected.
* Adapters verify the specific structures and concrete values of messages.

For example, when the test suite receives Negotiate Response, the model will verify whether it is the expected value by comparing it with the expected values stored in the classes, and the protocol adapter will verify whether the ChallengeLength field is set to 0 when the CAP\_EXTENDED\_SECURITY bit is set.

Please refer to the following 2 diagrams for more information:



## Technical Feasibility of Message Generation/Adapter Approach

Technical Feasibility

No technical issues are encountered in MS-SMB test suite.

Adapter Approach

Two adapters are used in the test suite, one is SUT Control Adapter or Interactive Adapter that is used to configure server. The other is Protocol Adapter that is used to cover test cases. Protocol Adapter is separated into 3 C# files, which are ISMBAdapter, ISMBTransaction2Adapter and ISMBNTTransactionAdapter, they construct the whole Protocol Adapter.

* ISMBAdapter.cs covers the operations in Model.cs.
* ISMBTransaction2Adapter.cs covers the operations in Transaction2Model.cs.
* ISMBNTTransactionAdapter.cs covers the operations in NTTransactionModel.cs.

For the diagram of the adapter approach, please refer to [Figure 2-1](#f21).

Details about MS-SMB Adapter, please refer to:

<MS-SMB_ServerAdapter.chm>

Protocol Adapter

The protocol SDK will be used for message generation and parse for SMB test suite.

* Message Generation

Protocol Adapter calls the SMB SDK instead of PAC (Protocol Adapter Compiler) to generate SMB packets. Each formatted packet must satisfy all the specific requirements in SMB technical document and then sent by SendPacket function in SMB SDK.

* Message Consumption
* When protocol adapter finishes sending SMB packets, it will receive packet from the server by calling ReceivePacket function in SMB SDK. During the process of receiving the packets, SMB SDK will also parse SMB packets and then return them to protocol adapter. The protocol adapter will capture and verify the adapter requirements, while passing the test case requirements to the test cases to verify.
* Besides, FSCC structures are embedded in the SMB packets received as data buffer, the parse of FSCC structures are done by FSCC SDK.

Interactive Adapter

As on some platforms, the SUT Control Adapter can’t be configured automatically, an Interactive adapter is added for avoiding this issue.

The Interactive Adapter is implemented by PowerShell scripts, and the functionalities are the same as SUT Control Adapter functionalities, which are:

* The adapter provides interfaces through which test cases can perform configurations on the server.
* The adapter provides interfaces through which disk I/O errors can be raised.
* The adapter provides interfaces through which a generic server open failure can be raised.

SUT Control Adapter

SUT control is implemented using PowerShell Script and is responsible for

* Creating/deleting a remote queue.
* Sending messages to a remote queue, or retrieving unique lookup identifier from a remote queue.
* Creating a distributed atomic transaction for receiving or moving the messages.

## Considerations

### Encryption Considerations

The MS-SMB test suite has no encryption considerations.

### Other Considerations

The MS-SMB test suite is developed with hybrid test approach as mentioned in [section 1.3](#_Test_Approach).

## Scenarios

There are total of 4 scenarios in SMB.

Note: There are more than 300 machines, so only 2 machines are listed for each type of environment below.

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Name** | **Machine** | **Covered Operation** |
| 1 | Windows 7--Windows Server 2008 | … | … |
| NoSignSET\_QUERY\_PATHModelProgram\_Win7\_Win2K8 | SmbConnectionResponse  ServerSetup  NegotiateRequest  SessionSetupRequest  TreeConnectRequest  CreateRequest  Trans2SetPathInfoRequest  Trans2QueryPathInfoRequest |
| NoSignSET\_QUERY\_PATH\_Win7\_Win2K8 | SmbConnectionRequest  SmbConnectionResponse  ServerSetup  ServerSetupResponse  CreatePipeAndMailslot  NegotiateRequest  NegotiateResponse  SessionSetupRequest  SessionSetupResponse  TreeConnectRequest  TreeConnectResponse  CreateRequest  CreateResponse CheckPreviousVersion  Trans2SetPathInfoRequest  Trans2SetPathInfoResponse  Trans2QueryPathInfoRequest  Trans2QueryPathInfoResponse |
| … |  |
| 2 | Windows 7--Windows Server 2008 R2 | … | … |
| SET\_QUERY\_FILEModelProgram\_Win7\_Win2K8R2 | SmbConnectionResponse  ServerSetup  NegotiateRequest  SessionSetupRequest  TreeConnectRequest  CreateRequest  Trans2SetFileInfoRequest  Trans2QueryFileInfoRequest |
| SET\_QUERY\_FILE\_Win7\_Win2K8R2 | SmbConnectionRequest  SmbConnectionResponse  ServerSetup  ServerSetupResponse  CreatePipeAndMailslot  NegotiateRequest  NegotiateResponse  SessionSetupRequest  SessionSetupResponse  TreeConnectRequest  TreeConnectResponse  CreateRequest  CreateResponse Trans2SetFileInfoRequest Trans2SetFileInfoResponse Trans2QueryFileInfoRequest  Trans2QueryFileInfoResponse Trans2QueryFileInfoRequest  Trans2QueryFileInfoResponse |
| … | … |
| 3 | Windows Vista --Windows Server 2008 | … | … |
| NoSignSET\_QUERY\_FILEModelProgram\_WinVista\_Win2K8\_InvalidLevel | SmbConnectionResponse  ServerSetup  NegotiateRequest  SessionSetupRequest  TreeConnectRequest  CreateRequest  Trans2QueryFileInfoRequest |
| NoSignSET\_QUERY\_FILE\_WinVista\_Win2K8\_InvalidLevel | SmbConnectionRequest  SmbConnectionResponse  ServerSetup  ServerSetupResponse  CreatePipeAndMailslot  NegotiateRequest  NegotiateResponse  SessionSetupRequest  SessionSetupResponse  TreeConnectRequest  TreeConnectResponse  CreateRequest  CreateResponse Trans2QueryFileInfoRequest |
| … | … |
| 4 | Windows Vista --Windows Server 2008 R2 | … | … |
| SET\_QUERY\_FILEModelProgram\_WinVista\_Win2K8R2 | SmbConnectionResponse  ServerSetup  NegotiateRequest  SessionSetupRequest  TreeConnectRequest  CreateRequest  Trans2SetFileInfoRequest  Trans2QueryFileInfoRequest |
| SET\_QUERY\_FILE\_WinVista\_Win2K8R2 | SmbConnectionRequest  SmbConnectionResponse  ServerSetup  ServerSetupResponse  CreatePipeAndMailslot  NegotiateRequest  NegotiateResponse  SessionSetupRequest  SessionSetupResponse  TreeConnectRequest  TreeConnectResponse  CreateRequest  CreateResponse  Trans2SetFileInfoRequest  Trans2SetFileInfoResponse Trans2QueryFileInfoRequest  Trans2QueryFileInfoResponse  Trans2QueryFileInfoRequest  Trans2QueryFileInfoResponse |
| … | … |

*Table 2‑3: Scenarios of MS-SMB*

Common Steps:

* SmbConnectionRequest

The client sends an SMB connection request to the server.

* SmbConnectionResponse

The server sends an SMB connection response to the client.

* ServerSetup

The client sends an SMB connection setup request with the server.

* ServerSetupResponse

The server sends an SMB connection setup response to the client.

* CreatePipeAndMailslot

The server creates named pipe and mail slot.

* NegotiateRequest

The client sends negotiate request to the server.

* NegotiateResponse

The server sends negotiate response to the client.

* SessionSetupRequest

The client sends session setup request to the server.

* SessionSetupResponse

The server sends session setup response to the client.

* TreeConnectRequest

The client sends a tree connection request to the server.

* TreeConnectResponse

The server sends a tree connection response to the client.

* CreateRequest

The client sends a file creation request to the server.

* CreateResponse

The server sends a file creation response to the client.

**Typical machine**

Windows Vista --Windows Server 2008 R2: SET\_QUERY\_FILE\_WinVista\_Win2K8R2

Description: This machine is related to the environment Windows Vista --Windows Server 2008 R2. And it is used to cover the requirements in Trans2SetFileInfoRequest

Trans2SetFileInfoResponse, Trans2QueryFileInfoRequest, Trans2QueryFileInfoResponse, Trans2QueryFileInfoRequest, Trans2QueryFileInfoResponse.

Detailed Sequence:

1. Common steps.
2. The client sends a set file information request to the server.
3. The server sends a set file information response to the client.
4. The client sends a file information query request to the server.
5. The server sends a file information query response to the client.
6. The client sends a file information query request to the server.
7. The server sends a file information query response to the client.

Note: Other machines are similar with the typical machine.

# Test Suite Design

## Model or Test Design Description

### Model Design

Hybrid testing, that is MBT and traditional testing combined, is adopted as the test approach for the MS-SMB test suite. TSD (Test suite developer) designed one Model according to the protocol behaviors, and the scenarios designed will generate test cases to test the server behaviors. Model Design is based on Model Plan as specified in [section 2.2.2](#s22).

Model Class Diagram

The MS-SMB model consists of 12 classes: 1 model class, 7 abstract data model classes, and 4 helper classes.

The following figure shows the class diagram of the MS-SMB Model:



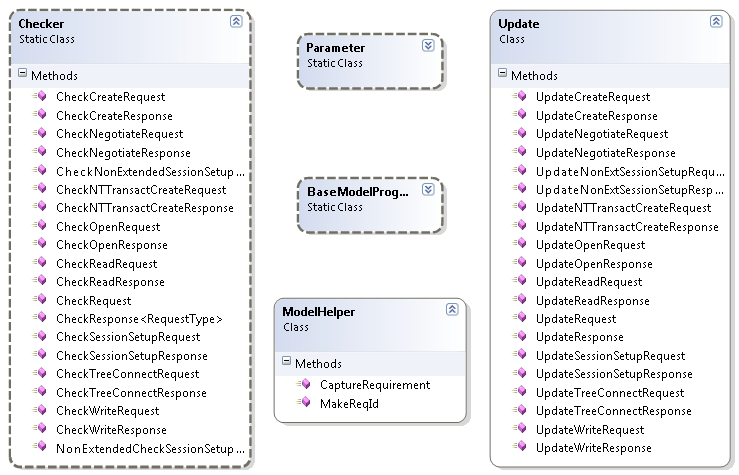
The model class ModelProgram is the main class in the MS-SMB model. It defines 81 model actions, as specified in [section 2.2.2](#s22).

* 18 types of operations have 3 corresponding actions: a request, a response and an error response.
* The operation SMB\_COM\_NEGOTIATE has 4 actions abstracted, which are NegotiateRequest, NegotiateResponse, NonExtendedNegotiateResponse and ErrorNegotiateResponse.
* The operation SMB\_COM\_SESSION\_SETUP\_ANDX has 5 actions abstracted, which are SessionSetupRequest, NonExtendedSessionSetupRequest, SessionSetupResponse, NonExtendedSessionSetupResponse and ErrorSessionSetupResponse.
* FSCTLBadCommandRequest: This action is used to send invalid FSCTL command to server.
* ErrorFsctlBadCommandResponse: This action is the response to the FsctlBadCommandRequest action.
* ErrorInvalidCommandResponse: This action is the response to the BadCommandRequest action.
* 3 actions related to TRANS2\_SET\_FS\_INFORMATION, which are Trans2SetFsInfoRequestAdditional,Trans2SetFsInfoResponseAdditional and ErrorTrans2SetFsInfoResponseAdditional
* 3 actions related to NT\_TRANSACT\_SET\_QUOTA, which are Trans2SetFsInfoRequestAdditional, Trans2SetFsInfoResponseAdditional and ErrorTrans2SetFsInfoResponseAdditional.
* 3 actions related to Session setup, which are SessionSetupRequestAdditional,SessionSetupResponseAdditional and ErrorSessionSetupResponseAdditional.

All the above actions are implemented in the same manner: The client sends one request to the server and the server will return two kinds of response, which are successful response or failed response. For example, there are 3 actions related to the operation “SMB\_COM\_WRITE\_ANDX”, which are WriteRequest, WriteResponse and ErrorWriteResponse. WriteRequest is used to send a request to server to write data to a file or a named pipe. If the operation succeeds, the WriteResponse will be called to receive the successful respone from server. If the operation is failed, the ErrorWriteResponse will be called to receive the failed respone from server.

* 6 actions are defined by the test suite:
* ServerSetup: This action sets up the server side environment, including message signing requirement and maximum read/write size.
* ServerSetupResponse: The response to the ServerSetup action.
* CreatePipeAndMailslot：This action creates a named pipe or mailslot at the server side, in order to enable client create its instances.
* CheckPreviousVersion: This action checks the previous version for the designated file identifier.
* The 7 abstract data model classes are SmbConnection class, SmbFile class, SmbPipe class, SmbSession class, SmbShare class, SmbTree and SmbMailslot as specified in [section 2.2.2](#s22).

There are 4 helper classes to the MS-SMB model. The relationship between these helper classes and the model class is illustrated in the following figure:

****

*Figure 3-1-1 Helper Class Diagram*

Further explanation of the helper classes:

* The class Parameters helps to set up the environment and the initial values of the model.
* Each time a request or a response is processed, model must validate the messages and update the model state. The 2 classes Checker and ModelHelper guarantee that the model follows the logics specified in the protocol.
* Checker: Each request has its own validation procedure with “Condition.isTrue()” statements. And there are common procedures which can be extracted as a uniform method. MessageChecks is used to do all common message validation. Responses are validated by method CheckReponse in this class.
* ModelHelper is used to capture code.
* The class Update is responsible for updating all local data structures after a request is sent or a response is received.

Global Variables

12 global variables are designed in the model.

* The enumeration variable “smbState” is used to indicate the initial state of SMB model.
* The class variable “connection” is used to indicate the network connection between the client and the server. As described before, there is only one connection in the Test Suite.
* The enumeration variable “smbRequest” is used to save subclass object.
* If the Boolean variable “isSendBufferSizeExceedMaxBufferSize” is set, it indicates the maximum server buffer size of bytes it writes can exceed the MaxBufferSize field.
* If the Boolean variable “isWriteBufferSizeExceedMaxBufferSize” is set, it indicates the maximum server buffer size of bytes it writes can exceed the MaxBufferSize field.
* The int variable “quotaUsed” is used to record quota information.
* The int variable “fIdUsed” is used to define fid used for setting quota.
* The Boolean variable “isQuotaSet” is used to record whether quota has been set by client
* The string variable “resumeKey” is used to define resumeKey.
* The int variable “copychunkSourceFid” is used to set copychunkSourceFid to 0.

Requirement Verification Strategy in the Model

Every action in the model will handle success and error operations. If the action returns success or an expected error code, the model will move on to the next state and continue with the next action if any. Otherwise, if any unexpected error is returned, the test case will stop and exit. The model verifies the requirements involving logics, such as the requirements concerning return values.

Details about MS-SMB Adapter, please refer to:

<MS-SMB_ServerAdapter.chm>

### Scenario Machine Design

The MS-SMB test suite contains about 320 machines in cord script to generate test cases that are utilized to cover the server requirements.

State Transitions

The following diagram shows the state transitions in MS-SMB test suite.



As shown in the figure above, the state transitions in MS-SMB test suite are implemented as follows:

1. State variable is initialized to “Init”
2. When “SmbConnectionRequest” is called, the state will be transited to “ConnectionRequest”.
3. When “SmbConnectionResponse” is returned, the state will be transited to ”ConnectionSuccess”.
4. When “ServerSetup” is called, the state will be transited to “SeverSetupRequest”.
5. When “ServerSetupResponse” is returned, the state will be transited to” ServerSetupSuccess”.
6. When “CreateNamePipeAndMailslot” is called, the state will be transited to “CreateNamePipeAndMailslotSuccess”.
7. When “NegotiateRequest” is called, the state will be transited to “NegotiateSent”.
8. If “NegotiateResponse” is returned, the state will be transited to “NegotiateSuccess”

If “ErrorResponse” is returned, the state will be transited to “Closed”.

1. When “SessionSetupRequest” is called, the state will be transited to “SessionSetup Success”.
2. If “SessionSetupResponse” is returned, the state will be transited to “SessionSetup Success”.

If “ErrorResponse” is returned, the state will be transited to “Closed”.

1. When “TreeConnectRequest” is called, the state will be transited to “TreeConnectSent”.
2. If “TreeConnectResponse” is returned, the state will be transited to “TreeConnectionSuccess”.

If “ErrorResponse” is returned, the state will be transited to “Closed”.

1. When the client sends requests for SMB operations, such as sending request for“SMB\_COM\_NT\_CREATE\_ANDX”, if the operation succeeds, the state will be returned to “TreeConnectionSuccess”, if the operation is failed, the state will be transited to “End”.

The Principle of Scenario Machine Design

The model actions are designed according to the 22 request types.

* Normally the MS-SMB Model sends one request and waits to receive the corresponding response.

For example: After sending out the ReadRequest, the model waits to receive a ReadResponse. Only after the model receives it, it will continue sending out the next request.

Sample Scenario

The following is the message sequence of scenario03: SMB\_S13\_Write:

machine NoSignInvalidTokenTraditional\_Model\_Win7\_Win2K8R2() : Main

{

(

SmbConnectionRequest;

SmbConnectionResponse;

ServerSetup;

ServerSetupResponse;

CreatePipeAndMailslot;

NegotiateRequest;

NegotiateResponse;

SessionSetupRequest;

SessionSetupResponse;

TreeConnectRequest;

TreeConnectResponse;

CreateRequest;

CreateResponse;

WriteRequest;

(WriteResponse|(ErrorWriteResponse;SessionClose\*));

)

The figure embedded below illustrates the state exploration diagram of this typical machine.



## Test Cases

1. Model-Based Testing

The MS-SMB model generates 3467 test cases to cover the requirements in the model and in the adapter. The generated test cases also cover the protocol examples mentioned in section 4 of the protocol.

Test Case Limiting

In order to limit the number of the test cases, the MS-SMB Test Suite uses the strategies of scenario slicing, parameter generation, binding and long test case.

* Bindings:

The test suite binds parameters to their possible values in order to reduce the amount of test cases. For example, the shareType parameter is bound to ShareType.DISK*.* Parameter bindings can avoid generating excessive test cases. The model only generates the test cases with parameter values that are necessary to cover necessary requirements. The requirement coverage will not be reduced as a result of this binding.

* Long test cases:

The test suite uses the long test cases strategy for all the SpecExplorermachines that are used for test case generation. During test case generation, the machines that adopt long test cases attempt to explore more states in a single test case in order to limit the amount of generated test cases.

* Requirement coverage:

The test suite uses requirement coverage rule for all the SpecExplorermachines that are used for test case generation. During test case generation, the machines that adopt enough test cases cover all the requirements and avoid superfluous cases.

There are 3610 test cases designed for the MS-SMB Test Suite. The SpecExplorermachines and the number of test cases for each scenario are shown below.

Note: There are more than 300 machines, so only 4 machines are listed for each type of environment below.

|  |  |  |
| --- | --- | --- |
| **Scenario**  **(Environment)** | **Spec Explorer (SE) Machine for Test Case Generation** | **Test Case Count** |
| Windows 7--Windows Server 2008 | AdditionalNonExtendRequirementCoverage\_QueryQuota\_Win7\_Win2K8  AdditionalNonExtendRequirementCoverage\_SET\_QUERY\_FS\_Win7\_Win2K8  AdditionalNoSignNonExtendRequirementCoverage\_QueryQuota\_Win7\_Win2K8  AdditionalNoSignNonExtendRequirementCoverage\_SET\_QUERY\_FS\_Win7\_Win2K8  … | 606 |
| Windows 7--Windows Server 2008 R2 | …  AdditionalNoSignRequirementCoverage\_QueryQuota\_Win7\_Win2K8R2  AdditionalNoSignRequirementCoverage\_SET\_QUERY\_FS\_Win7\_Win2K8R2  AdditionalRequirementCoverage\_QueryQuota\_Win7\_Win2K8R2  AdditionalRequirementCoverage\_SET\_QUERY\_FS\_Win7\_Win2K8R2  … | 1208 |
| Windows Vista --Windows Server 2008 | …  NonExtendedRequirementCoverage\_EnumerateSnapshot\_WinVista\_Win2K8  NonExtendedRequirementCoverage\_FIND\_WinVista\_Win2K8  NonExtendedRequirementCoverage\_SET\_QUERY\_FILE\_WinVista\_Win2K8  NonExtendRequirementCoverage\_CopyChunk\_WinVista\_Win2K8  … | 604 |
| Windows Vista --Windows Server 2008 R2 | …  NonExtendRequirementCoverage\_Create\_WinVista\_Win2K8R2  NonExtendRequirementCoverage\_Invalid\_FSCTL\_Command\_WinVista\_Win2K8R2  NonExtendRequirementCoverage\_Printer\_WinVista\_Win2K8R2  NonExtendRequirementCoverage\_QueryQuota\_WinVista\_Win2K8R2  … | 1183 |

Table 3-2-1 Scenario Count

1. Traditional Testing

There are 9 test cases designed for the traditional scenario. Details about the test cases are as follows.

Common steps of all the traditional test cases:

1. The client sends a SMB\_COM\_NEGOTIATE request to the server to initiate an SMB session between the client and the server.
2. The server replies a successful SMB\_COM\_NEGOTIATE response.
3. The client sends a SMB\_COM\_SESSION\_SETUP\_ANDX request to begin user authentication on an SMB connection and establish an SMB session.
4. The server replies a SMB\_COM\_SESSION\_SETUP\_ANDX response with STATUS\_MORE\_PROCESSING\_REQUIRED.
5. The client begins second round trips and send SMB\_COM\_SESSION\_SETUP\_ANDX request.
6. The server replies a successful SMB\_COM\_SESSION\_SETUP\_ANDX response.
7. The client sends a SMB\_COM\_TREE\_CONNECT\_ANDX request to establish a [tree connect](file:///C:\Users\v-lzhai\AppData\Local\Microsoft\Windows\Temporary%20Internet%20Files\Content.Outlook\1JRTS91O\%5bMS-CIFS%5d.pdf) to a share.
8. The server replies a successful SMB\_COM\_TREE\_CONNECT\_ANDX response.

|  |  |
| --- | --- |
| **Test ID** | Traditional\_TraditionalTestCase\_IgnoreFields\_Create\_01 |
| **Description** | The test case is to test the requirements that the client sets the unused or reserved fields of SMB\_COM\_NT\_CREATE\_ANDX and the server ignores and replies the same. |
| **Prerequisites** | There is a share in the sever side. |
| **Scenario** | SMB\_S08\_Ignore |
| **Test Execution Steps** | 1. Execute all common steps. 2. The client sends a SMB\_COM\_NT\_CREATE\_ANDX request to create a file:  * Reserved bits in ExtFileAttributes field set to 0. * CreateDisposition field set to FILE\_OPEN\_IF. * Unused bits of Flags field set to 0. * Other fields set according to TD.  1. The server returns a successful SMB\_COM\_NT\_CREATE\_ANDX response. 2. The client sends a SMB\_COM\_NT\_CREATE\_ANDX request to open the file in step 2.  * Reserved bits in ExtFileAttributes field set to 1. * CreateDisposition field set to FILE\_OPEN\_IF. * Unused bits of Flags field set to 1. * Other fields set according to TD and the same with step 2.  1. The server returns a successful SMB\_COM\_NT\_CREATE\_ANDX response. 2. Compare the responses in step 3 and step 5, if they are the same, verify R109027 and R109243. 3. Disconnect the tree, session and connection. |
| **Cleanup** | N/A |
| **Requirements covered** | MS-SMB\_109027, MS-SMB\_109243 |

Table 3-2-2 Traditional Test Case1 in MS-SMB Test Suite

|  |  |
| --- | --- |
| **Test ID** | TraditionalTestCase\_IgnoreFields\_FIND\_FIRST2\_02 |
| **Description** | The test case is to test the requirements that the client sets the unused or reserved fields of TRANS2\_FIND\_NEXT2 and the server ignores and replies the same. |
| **Prerequisites** | There is a share in the sever side. |
| **Scenario** | SMB\_S08\_Ignore |
| **Test Execution Steps** | 1. Execute all common steps. 2. The client sends a SMB\_COM\_NT\_CREATE\_ANDX request to open a directory:  * CreateDisposition field set to FILE\_OPEN. * Other fields set according to TD.  1. The server returns a successful SMB\_COM\_NT\_CREATE\_ANDX response. 2. The client sends a TRANS2\_FIND\_FIRST2 request to retrieve an enumeration of files.  * The bits unlisted in TD 2.2.1.2.2 of SearchAttributes field set to 0. * The Reserved bit in SearchAttributes field set to 0. * InformationLevelfield set to SMB\_INFO\_STANDARD. * Other fields set according to TD.  1. The server returns a successful TRANS2\_FIND\_FIRST2 response. 2. The client sends a TRANS2\_FIND\_NEXT2 request to continue the file enumeration.  * InformationLevel field set to SMB\_INFO\_STANDARD. * Other fields set according to TD.  1. The server returns a successful TRANS2\_FIND\_NEXT2 response. 2. The client sends a TRANS2\_FIND\_FIRST2 request to retrieve an enumeration of files again under the directory.  * The bits unlisted in TD 2.2.1.2.2 of SearchAttributes field set to 1. * The Reserved bit in SearchAttributes field set to 1. * InformationLevelfield set to SMB\_INFO\_STANDARD. * Other fields set according to TD and the same with step 4.  1. The server returns a successful TRANS2\_FIND\_FIRST2 response. 2. The client sends a TRANS2\_FIND\_NEXT2 request to continue the file enumeration.  * InformationLevel field set to SMB\_INFO\_STANDARD. * Other fields set according to TD.  1. The server returns a successful TRANS2\_FIND\_NEXT2 response. 2. Compare the responses in step 7 and step 11, if they are the same, verify R109033 and R109056. 3. Disconnect the tree, session and connection. |
| **Cleanup** | N/A |
| **Requirements covered** | MS-SMB\_109033, MS-SMB\_109056 |

Table 3-2-3 Traditional Test Case2 in MS-SMB Test Suite

|  |  |
| --- | --- |
| **Test ID** | TraditionalTestCase\_IgnoreFields\_Negotiate\_03 |
| **Description** | The test case is to test the requirements that the client sets the unused or reserved fields of SMB\_COM\_NEGOTIATE and the server ignores and replies the same. |
| **Prerequisites** | There is a share in the sever side. |
| **Scenario** | SMB\_S08\_Ignore |
| **Test Execution Steps** | 1. The client sends a SMB\_COM\_NEGOTIATE request to server to initiate an SMB session between the client and the server.  * Unused bits in Flags2 field set to 0. * SMB\_FLAGS2\_SMB\_SECURITY\_SIGNATURE\_REQUIRED bit in Flags2 field set to 0. * Other fields set according to TD.  1. The server replies a successful SMB\_COM\_NEGOTIATE response. 2. Disconnect. 3. The client sends a SMB\_COM\_NEGOTIATE request again to server to initiate an SMB session between the client and the server.  * Unused bits in Flags2 field set to 1. * SMB\_FLAGS2\_SMB\_SECURITY\_SIGNATURE\_REQUIRED bit in Flags2 field set to 1. * Other fields set according to TD and the same with step 1.  1. The server replies a successful SMB\_COM\_NEGOTIATE response. 2. Check the follows:  * Compare all the fields of the responses in step 2 and step 5 except for unused bits in Flags2, if they are the same, verify R5277 and R5298. * Compare the value in Unused bits in Flags2 of the client request and the server response, if they are the same, verify R108.  1. Disconnect |
| **Cleanup** | N/A |
| **Requirements covered** | MS-SMB\_5277, MS-SMB\_5298, MS-SMB\_108 |

Table 3-2-4 Traditional Test Case3 in MS-SMB Test Suite

|  |  |
| --- | --- |
| **Test ID** | Traditional\_TraditionalTestCase\_IgnoreFields\_Tree\_04 |
| **Description** | The test case is to test the requirements that the client sets the unused or reserved fields of successful SMB\_COM\_NT\_CREATE\_ANDX and the server ignores and replies the same. |
| **Prerequisites** | There is a share in the sever side. |
| **Scenario** | SMB\_S08\_Ignore |
| **Test Execution Steps** | 1. Execute Common Step 1~6. 2. The client sends a SMB\_COM\_NT\_CREATE\_ANDX request to create a file:  * The bits not listed in TD 2.2.4.7.1 of Flags field set to 0. * Other fields set according to TD.  1. The server returns a successful SMB\_COM\_NT\_CREATE\_ANDX response. 2. The client sends a SMB\_COM\_NT\_CREATE\_ANDX request to open the file in step 2.  * The bits not listed in TD 2.2.4.7.1 of Flags field set to 1. * Other fields set according to TD and the same with step 2.  1. The server returns a successful SMB\_COM\_NT\_CREATE\_ANDX response. 2. Compare the responses in step 3 and step 5, if they are the same, verify R5596. 3. Disconnect the tree, session and connection. |
| **Cleanup** | N/A |
| **Requirements covered** | MS-SMB\_5596 |

Table 3-2-5 Traditional Test Case4 in MS-SMB Test Suite

|  |  |
| --- | --- |
| **Test ID** | TraditionalTestCase\_IgnoreFields\_CopyChunk\_05 |
| **Description** | The test case is to test the requirements that the client sets the unused or reserved fields of successful SMB\_COM\_NT\_CREATE\_ANDX and the server ignores and replies the same. |
| **Prerequisites** | There is a share in the sever side. |
| **Scenario** | SMB\_S08\_Ignore |
| **Test Execution Steps** | 1. Execute Common Step 1~6. 2. The client sends a SMB\_COM\_NT\_CREATE\_ANDX request to create a file:  * The bits not listed in TD 2.2.4.7.1 of Flags field set to 0. * Other fields set according to TD.  1. The server returns a successful SMB\_COM\_NT\_CREATE\_ANDX response. 2. The client sends a SMB\_COM\_NT\_CREATE\_ANDX request to open the file in step 2.  * The bits not listed in TD 2.2.4.7.1 of Flags field set to 1. * Other fields set according to TD and the same with step 2.  1. The server returns a successful SMB\_COM\_NT\_CREATE\_ANDX response. 2. Compare the responses in step 3 and step 5, if they are the same, verify R5596. 3. Disconnect the tree, session and connection. |
| **Cleanup** | N/A |
| **Requirements covered** | MS-SMB\_5596 |

Table 3-2-6 Traditional Test Case5 in MS-SMB Test Suite

|  |  |
| --- | --- |
| **Test ID** | TraditionalTestCase\_IgnoreFields\_QueryQuota\_06 |
| **Description** | The test case is to test the requirements that the client sets the unused or reserved fields of successful NT\_TRANSACT\_QUERY\_QUOTA and the server ignores and replies the same. |
| **Prerequisites** | There is a share in the sever side. |
| **Scenario** | SMB\_S08\_Ignore |
| **Test Execution Steps** | 1. Execute all common steps. 2. The client sends a SMB\_COM\_NT\_CREATE\_ANDX request to open a file:  * CreateDisposition field set to FILE\_OPEN. * Other fields set according to TD.  1. The server returns a successful SMB\_COM\_NT\_CREATE\_ANDX response. 2. The client sends a NT\_TRANSACT\_QUERY\_QUOTA request to query quota information.  * StartSidLength field set to 0. * StartSidOffset field set to 0. * Other fields set according to TD.  1. The server returns a successful NT\_TRANSACT\_QUERY\_QUOTA response. 2. The client sends a NT\_TRANSACT\_QUERY\_QUOTA request to again query quota information.  * StartSidLength field set to 0. * StartSidOffset field set to 1. * Other fields set according to TD and the same with step 4.  1. The server returns a successful NT\_TRANSACT\_QUERY\_QUOTA response. 2. Compare the responses in step 5 and step 7, if they are the same, verify R109446. 3. Disconnect the tree, session and connection. |
| **Cleanup** | N/A |
| **Requirements covered** | MS-SMB\_109446 |

Table 3-2-7 Traditional Test Case6 in MS-SMB Test Suite

|  |  |
| --- | --- |
| **Test ID** | TraditionalTestCase\_IgnoreFields\_SET\_FILE\_09 |
| **Description** | The test case is to test the requirements that the client sets the unused or reserved fields of successful TRANS2\_SET\_FILE\_INFORMATION and the server ignores and replies the same. |
| **Prerequisites** | There is a share in the sever side. |
| **Scenario** | SMB\_S08\_Ignore |
| **Test Execution Steps** | 1. Execute all Common steps. 2. The client sends a SMB\_COM\_NT\_CREATE\_ANDX request to create a named pipe.  * CreateDisposition field set to FILE\_OPEN\_IF. * Other fields set according to TD.  1. The server returns a successful SMB\_COM\_NT\_CREATE\_ANDX response. 2. The client sends a TRANS2\_SET\_FILE\_INFORMATION request to open the named pipe.  * InformationLevel field set to FILE\_LINK\_INFORMATION. * Reserved field in FILE\_LINK\_INFORMATION structure set to 0 * Other fields set according to TD.  1. The server returns a successful TRANS2\_SET\_FILE\_INFORMATION response. 2. The client sends a TRANS2\_SET\_FILE\_INFORMATION request to open the named pipe.  * InformationLevel field set to FILE\_LINK\_INFORMATION. * Reserved field in FILE\_LINK\_INFORMATION structure set to 0 * Other fields set according to TD.  1. The server returns a successful TRANS2\_SET\_FILE\_INFORMATION response. 2. Compare the responses in step 5 and step 7, if they are the same, verify R109590. 3. Disconnect the tree, session and connection. |
| **Cleanup** | N/A |
| **Requirements covered** | MS-SMB\_109590 |

Table 3-2-8 Traditional Test Case7 in MS-SMB Test Suite

|  |  |
| --- | --- |
| **Test ID** | TraditionalTestCase\_LARGE\_Read\_Write\_10 |
| **Description** | The test case is to test the requirements that the client Sends the SMB\_WRITE\_ADNX request, client and server both support the CAP\_LARGE\_WRITEX capability, then server returns response with more than the maximum buffer size. |
| **Prerequisites** | There is a share in the sever side. |
| **Scenario** | N/A |
| **Test Execution Steps** | 1. Execute all common steps.   In SMB\_COM\_SESSION\_SETUP\_ANDX request, set the Capabilities to 0xC000.   1. The client sends a SMB\_COM\_NT\_CREATE\_ANDX request to create a named pipe.  * CreateDisposition field is set to FILE\_OPEN\_IF. * CreateOptions field is set to FILE\_OPEN\_REPARSE\_POINT, FILE\_SEQUENTIAL\_ONLY, and FILE\_NON\_DIRECTORY\_FILE. * Other fields are set according to the TD.  1. The server returns a successful SMB\_COM\_NT\_CREATE\_ANDX response. 2. The client sends a SMB\_COM\_WRITE\_ANDX request to write bytes to a file. 3. The server returns a successful SMB\_COM\_WRITE\_ANDX response. 4. Check the count in the SmbParameters of the response, to verify requirement 9207. 5. The client sends a SMB\_COM\_READ\_ANDX request to query quota information. 6. The server returns a successful SMB\_COM\_READ\_ANDX response. 7. Check the ByteCount in the SmbData of the response, to verify requirements R9957, R5402, R206932, and R106932. 8. Disconnect the tree, session and connection. |
| **Cleanup** | N/A |
| **Requirements covered** | MS-SMB\_8505 |

Table 3-2-9 Traditional Test Case8 in MS-SMB Test Suite

|  |  |
| --- | --- |
| **Test ID** | TraditionalTestCase\_Disconnect\_14\_Case |
| **Description** | The test case is to test the requirements that the client sets the TREE\_CONNECT\_ANDX\_DISCONNECT\_TID bit in the Flags field of SMB\_COM\_TREE\_CONNECT\_ANDX and the server disconnects. |
| **Prerequisites** | There is a tree connect with specified TID in the SMB header of the request. |
| **Scenario** | N/A |
| **Test Execution Steps** | 1. Execute all common steps. 2. The client sends a SMB\_COM\_NT\_CREATE\_ANDX request to create a named pipe.             CreateDisposition field is set to FILE\_OPEN\_IF.            FILE\_WRITE\_THROUGH, FILE\_SYNCHRONOUS\_IO\_ALERT bits of CreateOptions field is set to 0.            FILE\_SYNCHRONOUS\_IO\_NONALERT bit of CreateOptions field is set to 1.            Other fields are set according to the TD.   1. The server returns a successful SMB\_COM\_NT\_CREATE\_ANDX response. 2. The client sends a NT\_TRANSACT\_QUERY\_QUOTA request to query quota information from a server.              SidListLength filed is set to 0.             StartSidLength field is set to 0.             StartSidOffset field is set to 90.             Other fields are set according to the TD.   1. The server returns a successful NT\_TRANSACT\_QUERY\_QUOTA response. 2. Check the Flags field in the SMB header of NT\_TRANSACT\_QUERY\_QUOTA request, if it contains TREE\_CONNECT\_ANDX\_DISCONNECT\_TID bit, then verify the requirement 10357. 3. Disconnect the connection. |
| **Cleanup** | N/A |
| **Requirements covered** | MS-SMB\_10357 |

Table 3-2-10 Traditional Test Case9 in MS-SMB Test Suite

## Adapter Design

The MS-SMB Test Suite implements 2 adapters: 1 protocol adapter, 1 Interactive adapter and 1 SUT control adapter.

### Protocol Adapter Design

The Protocol Adapter has 43 methods and 35 events.

* MS-SMB Adapter uses its methods to send request to the SUT and to receive the response from the SUT.

For example: After sending out the ReadRequest, the model waits to receive a ReadResponse. Only after the model receives it, it will continue sending out the next request.

* MS-SMB Adapter uses its events to trigger bound methods to notify the test case that the expected message has been received.

For example: ReadResponse is triggered when receiving the ReadResponse message to invoke a bound method.

Protocol Adapter Class

The class diagram of MS-SMB Adapter is given below.

* The MS-SMB Adapter Interface:



* The MS-SMB Adapter Implementation:

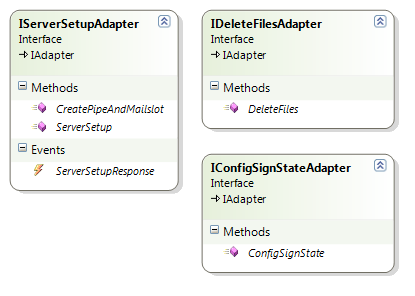


### SUT Control Adapter Design

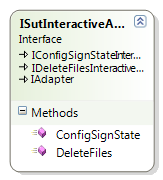
SUT Control Adapter or Interactive Adapter is used to create/delete a remote queue, send messages to a remote queue, or retrieve unique lookup identifier from a remote queue and create a distributed atomic transaction for receiving or moving the messages. Correspondingly, there are 2 methods designed. For details of the methods used in SUT control adapter or Interactive Adapter, please refer to [Section 2.3](#s23).

SUT Control Adapter Class

The diagram of MS-SMB Adapter Interface is given below.



*Figure 3-3-1: SUT Control Adapter Interface*



*Figure 3-3-2: Interactive Adapter Interface*

* The SUT Control Adapter or Interactive Adapter is implemented using Powershell script.

Details about MS-SMB Adapter, please refer to:

<MS-SMB_ServerAdapter.chm>

* + 1. **Sequence Diagram**

The figure below shows the interactions among the test cases, protocol adapter, SUT control adapter and the SUT.



*Figure 3-3-3: Inner Working of Adapters*

The diagram uses the message sequence of scenario S13: SMB\_S13\_Write to demonstrate the inner workings among adapters, test case and SMB server. The detailed message sequence is as follows:

1. The test case drives SUT control adapter or Interactive Adapter to initialize the configuration of SUT.
2. SUT control adapter or Interactive Adapter calls PowerShell scripts to configure SUT.
3. The test case drives the protocol adapter to send an SMB Negotiate Request.
4. The protocol adapter sends a SMB Negotiate Request to SUT.
5. SUT returns a SMB Negotiate Response to the protocol adapter which then passes the response to the test case.
6. The test case drives the protocol adapter to send a SMB SessionSetup Request.
7. The protocol adapter sends a SMB SessionSetup Request to SUT.
8. SUT returns a SMB SessionSetup Response to the protocol adapter which then passes the response to the test case.
9. The test case drives the protocol adapter to send a SMB TreeConnect Request.
10. The protocol adapter sends a SMB TreeConnect Request to SUT.
11. SUT returns a SMB TreeConnect Response to the protocol adapter which then passes the response to the test case.
12. The test case drives the protocol adapter to send a SMB Create Request.
13. The protocol adapter sends a SMB Create Request to SUT.
14. SUT returns a SMB Create Response to the protocol adapter which then passes the response to the test case.
15. The test case drives the protocol adapter to send a Write Request.
16. The protocol adapter sends a Write Request to SUT.
17. SUT returns a Write Response to the protocol adapter which then passes the response to the test case.

# References

## Related Links

* Link to Technical Document

[.\[MS-SMB].pdf](%5bMS-SMB%5d.pdf)

* Link to Requirement Spec

[.\MS-SMB\_ServerRequirementSpec.xlsm](MS-SMB_ServerRequirementSpec.xlsm)

* Link to Test Environment Deployment Guide

[.\MS-SMB\_ServerTestEnvironmentDeploymentGuide.docx](MS-SMB_ServerTestEnvironmentDeploymentGuide.docx)

* Link to Adapter CHM File

[.\MS-SMB\_ServerAdapter.chm](MS-SMB_ServerAdapter.chm)

* Link to Release Notes

[.\MS-SMB\_ReleaseNotes.docx](MS-SMB_ServerReleaseNotes.docx)