Technical Report

SMB 2.0 – Next Generation CIFS protocol in Data ONTAP®

Reena Gupta, NetApp
March 2009 | TR-3740

SMB 2.0 in DATA ONTAP® 7.3.1

Server Message Block (SMB) 2.0 is the next version of the Common Internet File System (CIFS)/SMB protocol. CIFS can also be considered as SMB 1.0. This document describes the SMB 2.0 features, configuration details, and its implementation in Data ONTAP® 7.3.1. This document also describes SMB 2.0 benefits over CIFS (SMB 1.0), deployment use cases, best practices, and tools to diagnose and capture SMB 2.0 information. Note that CIFS and SMB refer to the same protocol version, and these two terms are being used interchangeably as CIFS (SMB 1.0) in this document.

TABLE OF CONTENTS

1 INTRODUCTION

Server Message Block (SMB) is a remote file-sharing protocol used by Microsoft® Windows® clients and servers starting in the early 1980s. The existing CIFS or SMB 1.0 was designed and implemented to support file-serving solutions based on the assumptions existing then. For the past decade or so, there have been some ongoing minor changes and tweaks to the protocol to support some new functionality such as network resiliency, scalability, and so on. SMB 2.0 had the first major redesign considering the needs of the next generation of file servers and clients. These needs include a redesign for modern networking environments such as wide area networks (WANs), possible high-loss networks, time-outs, high latency, and so on.

Microsoft's continuous efforts to evolve SMB 2.0 have brought it as a next generation of the previous CIFS (SMB 1.0) protocol. It was first introduced in Windows Vista in 2007 and updated with the release of Windows Server 2008 and Windows Vista SP1 in 2008. Microsoft would support SMB 2.0 as the file system protocol of choice on all the future releases of Microsoft operating systems. NetApp released SMB 2.0 in coexistence with CIFS (SMB 1.0) starting in Data ONTAP 7.3.1. NetApp® Network-Attached Storage (NAS) storage platforms would now be able to serve Windows XP or other legacy clients and Windows Vista clients simultaneously.

2 AUDIENCES

This document is targeted for technical audiences such as system administrators, architects, system engineers, and application vendors who would like to explore and unleash SMB 2.0 in their environments. This document requires prior knowledge of file sharing in Microsoft Windows networks and familiarity with Microsoft terminology.

3 HISTORY OF CIFS (SMB)

When it was first introduced to the public, the remote file protocol was called Server Message Block (SMB). SMB was used by Microsoft LAN Manager in 1987 and by Windows for Workgroups in 1992. Later, a draft specification was submitted to the Internet Engineering Task Force (IETF) under the name Common Internet File System (CIFS). The CIFS specification is a description of the protocol as it was implemented in 1996 as part of Microsoft Windows NT® 4.0. A preliminary draft of the IETF CIFS 1.0 specification was published in 1997. Later, extensions were made to address other Microsoft features such as domains, Kerberos, shadow copy, server to server copy, and SMB signing. Windows 2000 (released in 2000) included those extensions. At that time, some people went back to calling the protocol SMB once again. CIFS (SMB 1.0) has also been implemented on UNIX®, Linux®, and many other operating systems (either as part of the operating system [OS] or as a server suite such as Samba[1]). A few times, those UNIX and Linux communities also extended the CIFS (SMB 1.0) protocol to address their own specific requirements.

CIFS (SMB 1.0) protocol had a few limitations:

- The protocol was not created with WAN or high-latency networks in mind. Specifically, CIFS (SMB 1.0) is "chatty." Chattiness is taking a series of roundtrips to accomplish many of the most common tasks, such as opening a file, reading data from same file etc.
- The field values in the SMB header were limited for the number of open files, shares, and users.
- There were large numbers of commands and subcommands (over 100) in the protocol design, making it difficult to extend, maintain, and secure.
- There were no considerations for temporary network connections loss.

4 BENEFITS

The SMB 2.0 protocol is much more resilient to network interruptions. It is designed to scale and perform better, as well as provide more security as compared to the CIFS (SMB 1.0) protocol.

The following table describes the benefits of the SMB 2.0 protocol over the CIFS (SMB 1.0) protocol.

[1] Samba is an open source/free software suite that provides file and print services to CIFS (SMB 1.0) clients.

Benefits	Features	What It Means to Customers
Enhanced performance	• Compounding operations • Larger buffer size • Crediting (QoS)	• Larger reads and writes in less round trips (64KB) • Improved WAN performance • Server can do some load balancing with credit granting
Increased server scalability	• Extended Session ID and TreeID fields • Extended UID and FID namespace	Up to 128K number of user sessions and tree connections per TCP connection
Network resiliency and increased reliability	• Asynchronous messages • Durable handles	• Less timeouts on the CIFS sessions • Avoids data loss on the client side
Enhanced security	SMB signing using SHA256	More robust “secured signing algorithm”

5 PROTOCOL OVERVIEW

The protocol overview consists of how it interacts with respect to the Internet Protocol (IP) layers, what command sets are available, and what’s new in the extended identifier fields with SMB 2.0.

Simplified Command Sets

SMB 2.0 reduces the complexity in the protocol by reducing the number of command sets. There are only 19 opcodes or commands as compared to the 100+ commands in the CIFS protocol, used in the message exchanges between the client and the server, grouped in three categories:

- Protocol negotiation, user authentication, and share access:
  - SMB2_OP_NEGPROT 0x00
  - SMB2_OP_SESSSETUP 0x01
  - SMB2_OP_LOGOFF 0x02
  - SMB2_OP_TCON 0x03
  - SMB2_OP_TDIS 0x04
- File, directory, and volume access:
  - SMB2_OP_CLOSE 0x06
  - SMB2_OP_FLUSH 0x07
  - SMB2_OP_READ 0x08
  - SMB2_OP_WRITE 0x09
  - SMB2_OP_LOCK 0x0a
  - SMB2_OP_IOCTL 0x0b
  - SMB2_OP_CANCEL 0x0c
  - SMB2_OP_NOTIFY 0x0f
  - SMB2_OP_GETINFO 0x10
  - SMB2_OP_CREATE 0x05
  - SMB2_OP_SETINFO 0x11
- Others:
  - SMB2_OP_FIND 0x0e
  - SMB2_OP_KEEPALIVE 0x0d
  - SMB2_OP_BREAK 0x12

CIFS (SMB 1.0) vs. SMB 2.0 Over-the-Wire Comparison

SMB 2.0 runs over port 445 only and uses TCP as its underlying transport protocol. The packet header formats in SMB 2.0 are different from CIFS (SMB 1.0), as shown in the following tables.

CIFS (SMB 1.0) Packet Header

Field	Size (Bytes)	Description
Protocol	4	Protocol identifier. The value must be 0xFF, 'SMB'.
Command	1	Command code, from 0x00 to 0xFF.
Status	4	32-bit error status code. A server returns error information to the client in the Status field.
Flags	2	Flags characterizing the CIFS request/response. If bit 7 (SMB_FLAGS_SERVER_TO_REDIR) is set, this packet is a server response.
Flags2	2	Flags2 - 16-bit flag field defining the capabilities of the client/server transaction.
TID	2	Tree identifier; a unique ID for a resource in use by client.
PID	2	Caller process ID.
UID	2	User identifier.
MID	2	Multiplex identifier; used to route requests inside a process.
WordCount	1	Count of parameter words defining the data portion of the packet.
ParameterWords [WordCount]	2	Parameter words defining the data portion of the packet.
ByteCount	2	Size of the data portion of the packet.
Buffer[ByteCoun]	Variable	Data portion of the packet. The format of the data portion depends on the command code. Fields in the data portion consist of an identifier byte followed by the data.

SMB2 Packet Header

Field	Size (Bytes)	Description
Protocol	4	The protocol identifier. The value MUST be (in network order) 0xFE, 'S', 'M', and 'B'.
StructureSize	2	MUST be set to 64, which is the size, in bytes, of the SMB2 header structure.
Epoch	2	Unused and MUST be treated as reserved. The sender MUST set this to 0, and the receiver MUST ignore it.
Status	4	The status code for a response. For a request, the client MUST set this field to 0, and the server MUST ignore it on receipt. For a response, this field can be set to any value.
Command	2	The command code. This field MUST contain one of the SMB2 valid command OpCodes.
CreditRequest/ Response	2	On a request, this field indicates the number of credits the client is requesting. On a response, it indicates the number of credits granted to the client. If a client does not want more credits, it MUST set this field to 1.
Flags	4	A flags field, which indicates how to process the operation.
NextCommand	4	For a compounded request, this field MUST be set to the offset, in bytes, from the beginning of this SMB2 header to the start of the subsequent 8-byte aligned SMB2 header. If this is not a compounded request, or this is the last header in a compounded request, this value MUST be 0.
MessageId	8	A value that identifies a message request and response uniquely across all messages that are sent on the same SMB 2 Protocol transport connection.
AsyncId	8	A unique identification number that is created by the server to handle operations asynchronously.
SessionId	8	Uniquely identifies the established session for the command.
Signature	16	The 16-byte signature of the message, if SMB2_FLAGS_SIGNED is set in the Flags field of the SMB2 header. If the message is not signed, this field MUST be 0.

8-Byte-Aligned Buffers

SMB 2.0 uses the data buffers and each request in a compounded chain to be aligned to an 8-byte aligned offset. This would be handled by the SMB redirector, and the file server would be transparent to the application or the client.

Extended ID Fields

Most of the ID fields have been extended to 64 bits (ULONG or UINT64) as compared to 16 bits in SMB, so there is less chance to wrap the information in the next SMB block. This helps in having more number of user sessions and tree connections per TCP connection, which means that a user can have more number of open files and open share connections for a given SMB 2.0 session on the server side.

- File IDs are 64 bit + 64 bit against 16 bit.
- Message IDs are 64 bit against 16 bit.
- Tree IDs are 32 bit against 16 bit.
- Security signature is 16 bytes against 8 bytes.

Completely Unicode and NTSTATUS Codes Used

All the text fields used in SMB 2.0 packets are in Unicode* format. Also, it uses only NTSTATUS error codes and does not use DOS error codes.

* Unicode is a comprehensive way of defining characters electronically for compatibility with most of the speaking languages in the world.

6 SMB PROTOCOL NEGOTIATION/COMPATIBILITY

The process to agree upon, or negotiate, a common level of SMBs each host can understand is referred to as SMB protocol negotiation. SMB protocol version used for file-sharing operation is determined during this negotiation. The following table explains how the SMB protocol version is negotiated between different operating systems. SMB 2.0 is enabled by default on Windows Vista or Windows 2008 systems, but it is disabled by default on Data ONTAP 7.3.1. Therefore a Vista client or a Windows 2008 server would communicate by default over SMB 2.0, whereas when any of these clients connects to a NetApp storage system running Data ONTAP 7.3.1, by default the communication would happen over SMB 1.0, unless you enable SMB 2.0 manually in Data ONTAP 7.3.1.

Default Protocol Used in Data ONTAP

Windows Client Type	Data ONTAP Version	Protocol Used
Vista clients/Windows 2008 as clients	NetApp system Data ONTAP 7.3.1	SMB 2.0/ SMB 1.0
Windows clients prior to Vista	NetApp system Data ONTAP 7.3.1	SMB 1.0
All Windows clients	NetApp system Data ONTAP 7.3 or prior	SMB 1.0

Note:

- SMB 2.0 and CIFS (SMB 1.0) can coexist in Data ONTAP 7.3.1.
- There is no need to upgrade all the clients to Windows Vista to work with Data ONTAP 7.3.1. Any legacy Windows clients such as XP, Windows 2000, and Windows 2003 can still work with Data ONTAP 7.3.1 over SMB 1.0.

7 CONFIGURATION

7.1 WINDOWS VISTA/2008

SMB 2.0 is enabled by default in the Windows Vista and Windows Server 2008 operating systems.

7.2 Data ONTAP

SMB 2.0 is disabled by default in Data ONTAP 7.3.1. It should be enabled for any Vista clients in order to connect to the NetApp systems over SMB 2.0; otherwise Vista clients continue to connect over SMB 1.0. Following are the different commands and options to manage SMB 2.0 in Data ONTAP.

- Options:
  - **`cifs.smb2.enable`**
    This option is to enable/disable SMB 2.0 in Data ONTAP. When this option is disabled, the NetApp system will not accept any new SMB 2.0 sessions, but the existing sessions will not be terminated [Default "off"].
  - **`cifs.smb2.signing.required`**
    This option is to enforce SMB signing on all SMB 2.0 sessions [Default "off"].
  - **`cifs.smb2.client.enable`**
    This option is to enable/disable SMB 2.0 client capability on the storage system. When this option is enabled, NetApp system initiated connections to Windows domain controllers will attempt to use the SMB 2.0 protocol. In case the Windows domain controller does not support the SMB 2.0 protocol, then the NetApp system will fall back to using SMB. If a session had been established over SMB 2.0 and later this option is disabled, existing sessions will not be terminated; the NetApp system will continue to use SMB 2.0 for the existing sessions, but no new sessions will attempt to use SMB 2.0 [Default "off"].
  - **`cifs.smb2.durable_handle.enable`**
    This option is to enable/disable the durable handle functionality for SMB 2.0 clients. If this option is enabled, the open files from a client are preserved when the client gets disconnected from the NetApp system. These open files can be reclaimed when the client reconnects to the NetApp system [Default "on"].
  - **`cifs.smb2.durable_handle.timeout`**
    This option is to configure the duration in seconds for which the NetApp system will preserve the durable handle after a temporary network failure. This timer has a default value of 16 minutes, but its value could be changed by system policy to any range between 5 seconds and 2147483647 seconds. It can also be configured for infinite value as '-1.'
- Command:
  - **`cifs sessions –p [smb|smb2]`**
    This command has a new option, '-p.' This option filters the sessions on the basis of protocol version used. When the -p option is used with 'smb' as the argument, only SMB 1.0 sessions are displayed. When the -p option is used with 'smb2' as the argument, only SMB 2.0 sessions are displayed. When the -p option is not used, both SMB 1.0 and SMB 2.0 sessions are displayed. The -p option can be used along with -c and -s options.

8 SMB 2.0 FEATURES IN DATA ONTAP

8.1 COMPOUNDED OPERATIONS

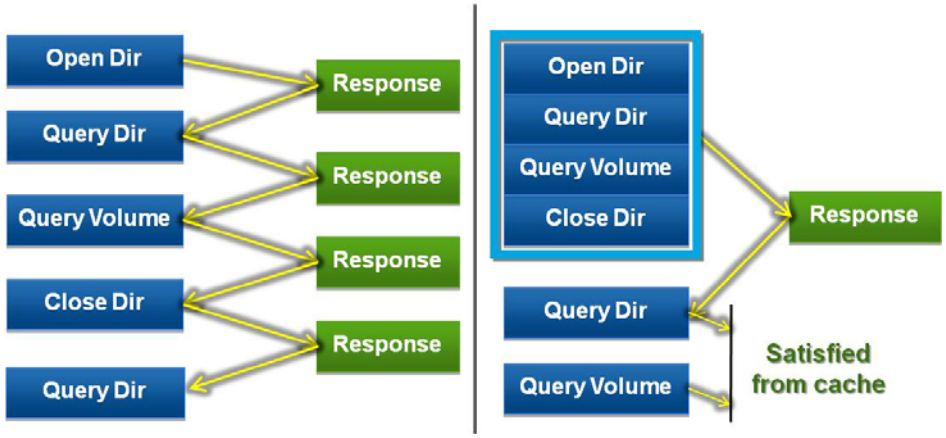
SMB 2.0 provides a method of combining multiple SMB 2.0 messages/commands into a single transmission request for submission to the underlying transport. This compounding reduces the round trips between the client and the server, thus reduces the CIFS protocol "chattiness" and improves the protocol performance. A common example of compounding the commands is putting 'OpenDir,' 'QueryDir,' 'QueryVolume,' and 'CloseDir' together in a single SMB packet for requesting a directory browsing operation.

There are two types of compounded messages.

- **Related compounded messages**

  For a related compounded request, the NetApp system processes the compounded requests in a sequential manner. The processing continues until all the requests in the compounded request are processed, even if one of the requests fails. If a subsequent command can succeed, even if a previous command failed, it would be marked succeeded. Respective responses are compounded together and sent back to the client.

Note: If one of these compounded requests becomes asynchronous type of request (as defined in section 6.4), all subsequent ones go async as well and have an interim response from the server for each of the async requests.



- **Unrelated compounded messages**

  For an unrelated compounded request, all the requests are processed independently irrespective of the result of processing other requests, and the responses are sent independently.

The NetApp system will support both related and unrelated compounded requests. It will compound the response for related compounded requests. For unrelated compounded requests, the NetApp system will not compound the responses.

8.2 DURABLE HANDLES

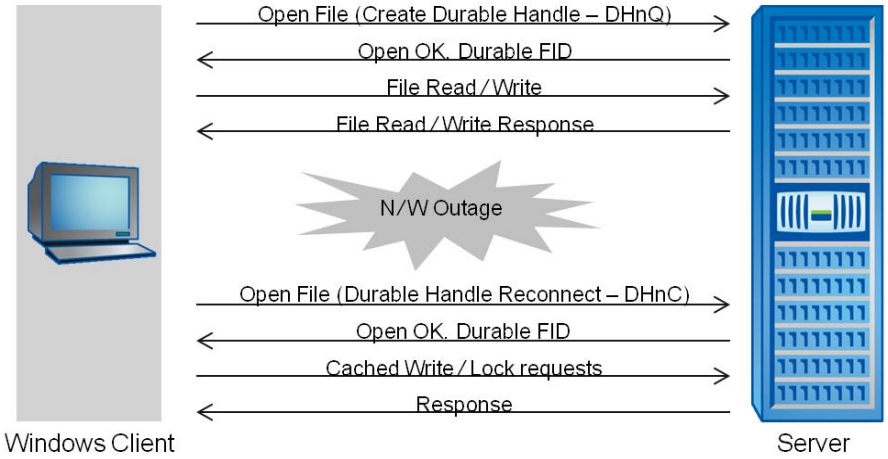
Durable handles are the file handles that persist across SMB 2.0 sessions. They are designed to prevent data loss caused by short network outages by absorbing writes cached on the client on a different SMB 2.0 session. When a client opens a file, it specifies if it needs the file handle to be durable. If the current connection goes away, the client would try to use the durable handle on a new connection if it is still valid on the server. The server on its part issues a durable handle only if it supports the functionality. The server keeps track of open files even after the connection drops. Upon session disconnection, the server makes the handles available for reclaim by the same authenticated user context on a different connection. Any pending cached writes during the disconnection can be flushed to the disk upon the reconnection. It brings resiliency to the network by avoiding data loss despite connections being dropped, especially on WANs.

The applications make use of durable handles through the redirector only, not by any client APIs. The redirector silently requests (and uses) durable handles on every file it opens with a batch opportunistic lock (oplock) without requiring the application to do anything differently.

The durable handles FID field is 128 bits long and has two parts:

- Persistent (64 bits long): stays valid as long as the file is open on the storage system. That means it can be valid across multiple client sessions. A persistent ID doesn't identify a file uniquely on the server, so if a file is closed and opened again, there is no guarantee that the persistent ID would be the same.
- Volatile (64 bits long): stays valid for one SMB session.

During a network outage, if some other client requests for the same file, it has to send an oplock break request for this disconnected durable handle. In this situation, the durable handle will be cleaned up to prevent any sort of access denied messages, and the client would not be able to claim the file handle made durable.



Durable handles are not designed to survive a server or client reboot or cluster failover, as the durable file handle structure is maintained in the server's memory, not on the disk. For example, in a clustered system, during an OS upgrade of the primary server, the partner server doesn't automatically inherit the durable handle from the primary server. Clients would have to make a fresh connection to the server; hence, it is not a transparent reconnection. If the Windows client panics, the durable handles information on the client is lost. The retry mechanism used by the client is limited to less than five times and primarily forced by the application above it using the handle rather than being driven by a durable handle timer.

There are two commands in Data ONTAP to manage the durable handles, as described in section 7.2:

- `options cifs.smb2.durable_handle.enable`
- `options cifs.smb2.durable_handle.timeout`

8.3 CREDIT SYSTEM

SMB 2.0 has a mechanism for the clients to send a number of outstanding requests to a server. This allows the client to build a pipeline of requests instead of waiting for a response before sending the next request. This is especially relevant when using a high-latency network. It also makes sure that a client cannot overload the server with a large number of pending requests, thus providing quality of service (QoS).

SMB 2.0 uses credit-based flow control, which allows a server to control the number of outstanding requests on a given SMB 2.0 session. The client is given a certain small number of credits from the server. For every credit it has, it may send one message to the server. As the client sends messages, it continually requests more credits to continue sending traffic. The server can therefore control the amount of traffic (in outstanding SMB operations on a session) by granting more or less credits to the client in question; thus some basic QoS is provided by avoiding excess use of resources by one client. This allows the server to throttle back connections when it becomes overburdened, as well as grant certain clients a higher "priority" than other clients. With this feature, the protocol can keep more data in flight and better utilize the available bandwidth. This is a key to make a large transfer take much less time in a high-bandwidth, high-latency network.

A Windows Vista client requests 128 credits by default in each message sent to the server and also uses one credit for each message. A NetApp storage server would grant 50 credits by default to a client, and it can load balance dynamically by adjusting the number of credits on the fly for each request. There might be multiple requests on a single SMB session, so the credits are requested/granted per request.

Note: Windows Vista and Windows Server 2008 do not grant credits on interim responses. An interim response for an asynchronously processed SMB2 CHANGE_NOTIFY request will grant credits to keep the transaction from stalling in case the client is out of credits.

8.4 ASYNCHRONOUS OPERATIONS

Certain SMB commands from the clients could take a longer time to process on the server, so then the server sends an interim asynchronous response to the client. Examples for these commands are Oplock Break, Change-notify, and Named-pipe operations on blocking pipes; byte range lock requests that might wait for lock availability; and a Create that triggers an oplock-break. These asynchronous responses are sent before the final response to the client and are in the form of async headers.

A client cannot request an async header; the server decides based on the type of request whether to send an async response. Async operations don't actively consume credits, but the responses may be used to grant credits. Async processing provides the ability for the server to modify the crediting behavior for clients performing significant async operations.

8.5 LARGER BUFFER SIZE

SMB 2.0 now has a much larger read and write buffer size; it's been increased to 64k default as compared to the 32k default in CIFS/SMB. Larger reads and writes make better use of faster networks, even with high latency. Any applications that can make use of the larger reads and writes would be able to see performance improvements.

8.6 INCREASED SCALABILITY

SMB 2.0 increases the restrictive constants within the protocol design to allow for scalability for file sharing. Number of users, open shares, and open files per TCP connection for a server are greatly increased. The following table differentiates the field sizes and limits increased in SMB 2.0 over CIFS/SMB.

Protocol	Type of Identifier	Field Size	Limits	Comments
CIFS/SMB	UID	16 bits	64k	Number of sessions
	TID	16 bits	64k	Open share connections
	FID	16 bits	64k	Open file connections
SMB 2.0	SessionID (UID)	64 bits	128K	Number of sessions
	TreeID (TID)	32 bits	128K	Open share connections
	FID (FID)	128 bits	128K	Open file connections

8.7 SMB SIGNING

SMB signing is a feature through which all communications using the SMB protocol can be digitally signed at the packet level. Digitally signing the packets enables the recipient of the packets to confirm their point of origination and their authenticity, thus avoiding man-in-the-middle attacks.

The SMB 2.0 signing feature uses the much more secured HMAC-SHA256 algorithm instead of MD5 (in CIFS/SMB) for generating the digital signature. SMB 2.0 signing is never disabled as seen in CIFS/SMB; the possible configurations are either "required" or "not required."

SMB 2.0 Signing	Server Required	Server Not Required
Client Required	Signed	Signed
Client Not Required	Signed	Not signed

If signing is negotiated for an SMB 2.0 session during session setup, the signatures of all the SMB 2.0 messages received on that session are verified, and the request is rejected if the message is not signed or if the signature is not valid.

SMB signing with other features:

- Signing with compounded messages: each individual request is signed.
- Signing with asynchronous responses: interim responses are also signed.

Note:

- Windows servers do not sign the interim responses and oplock breaks.
- SMB signing has a performance impact.

9 SUPPORTABILITY

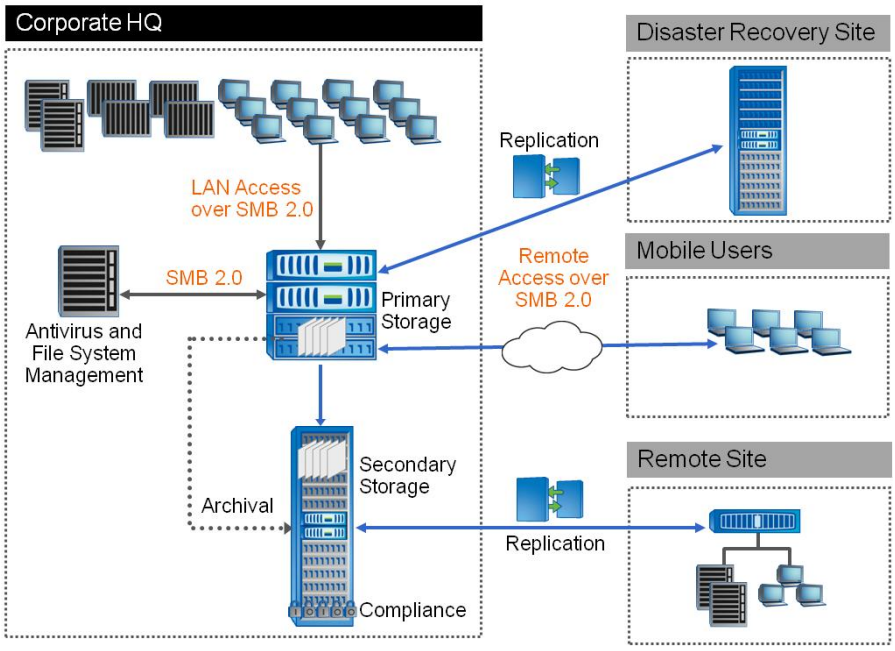
SMB 2.0 is supported on NetApp NAS storage platforms beginning with Data ONTAP 7.3.1. Microsoft supports SMB 2.0 beginning with Windows Vista and Windows Server 2008. Microsoft is continuing supporting SMB 2.0 on all future releases of Microsoft operating systems; for example, Windows 7 is going to have SMB 2.1 as a new dialect.

Data ONTAP 7.3.1 and higher would support both CIFS (SMB 1.0) and SMB 2.0; therefore customers would still be able to connect to a NetApp storage system from any existing Windows XP clients or Windows Vista clients. There is no change in the license requirement; the same CIFS license is applicable for SMB 2.0 as well. Please note that SMB 2.0 is not available in the workgroup mode, as extended security is not available in workgroup mode.

10 DEPLOYMENT USE CASES

A typical Windows file-sharing deployment can be illustrated from the image given below. SMB 2.0 can be used in all these typical Windows file-sharing deployments:

- LAN access among all the supported Windows servers and clients to the NetApp storage systems
- LAN access from any virus scanning and the file screening servers
- WAN access from the supported remote Windows systems in the remote offices
- Virtual private network (VPN) access from any mobile users



11 IMPACT ON APPLICATIONS

There are opportunities for the new applications to use the underlying protocol as SMB 2.0; for example, Petrel (an oil and gas application) is being tested to run over SMB 2.0. Applications that use the CIFS protocol will benefit further from SMB 2.0 in terms of the reliability, scalability, and other features it offers.

CIFS features such as auditing, group policy objects, and access-based enumeration have been identified to see no impact from using SMB 2.0. Virus scanning would depend on the AV scan server, where the AV scan engine is running. If the AV scan server is a Windows 2008 server, then the communication between the NetApp system and the AV scan server would happen over SMB 2.0 (if it's enabled on the NetApp system). There won't be any impact on any cross-protocol file access between SMB 2.0 and NFSv3/v4, as SMB 2.0 continues to use the same NTFS ACL structure as in CIFS/SMB.

12 PERFORMANCE

SMB 2.0 from a performance perspective is better suited in certain cases than CIFS/SMB. For example, NAS access over the WAN where the SMB 2.0 protocol's ability to do compounded operations, larger buffer sizes, and durable handles can help boost performance. The concept of credits can also help implement QoS for clients, especially in a high-utilization environment.

As of now there is no industry performance benchmark for SMB 2.0. The performance improvement over the CIFS (SMB 1.0) protocol is heavily dependent on the customer's environment, including workload and network infrastructure, where workload could mean sequential/random, op mix, and number of clients. Some of our customers using their own test methodology observed higher performance using Data ONTAP 7.3.1 and Windows Vista compared to their previous environment using Windows XP.

13 BEST PRACTICES

Windows Vista clients are mostly autotuned for getting the maximum benefits of the SMB 2.0 protocol. Certain best practices guidelines can be followed in order to achieve the maximum performance benefit for the SMB 2.0 protocol.

- Use Windows Vista SP1 or later as a client. Windows Vista SP1 has the full implementation of the SMB 2.0 protocol; Vista RTM had partial implementations of some SMB 2.0 features.
- If you are using any applications over SMB 2.0, then wherever applicable, tune your applications to:
  - Leverage the larger block size to send data
  - Send requests for more concurrent blocks
- Use Gigabit Ethernet or better network for high bandwidths,
- Make use of the best hardware on the client and server side, as a powerful configuration would be able to yield more.
- Turn SMB signing off if not needed.

14 DIAGNOSING TOOLS

Packet Trace Analyzer

Microsoft NetMon v3.1 and v3.2 as well as Wireshark can capture and decode SMB 2.0 packets. The protocol identifier is 0xFE 'S' 'M' 'B', although NetMon is more complete and better at decoding the SMB 2.0 packets.

Data ONTAP Tools

- **`cifs sessions -p [smb|smb2]`**
  This command filters the sessions on the basis of protocol version used to find out which clients are connected to NetApp system using SMB 2.0 or which clients are connected using SMB 1.0.
- **`cifs stat`**
  The command shows the CIFS statistics and also displays the specific SMB 2.0 stats that are being generated.

15 REFERENCES

NetApp Streamlines Data Management for Petrotechnical Applications

http://media.netapp.com/documents/netapp-zeus-technology-reprint.pdf

SMB 2.0 Protocol Specification

http://msdn.microsoft.com/en-us/library/cc212614.aspx

Network Monitor 3.2 tracing tool

www.microsoft.com/downloads/details.aspx?FamilyID=f4db40af-1e08-4a21-a26b-ec2f4dc4190d&displaylang=en

Microsoft's SMB 2.0 Performance white paper by Tolly Group

www.microsoft.com/downloads/details.aspx?FamilyID=04cad8b9-9f9f-453a-893a-458d22dbb3c5&DisplayLang=en

16 CONCLUSION

SMB 2.0 is a next-generation NAS protocol for Windows. The protocol has been redesigned to accommodate next-generation NAS servers' requirements, especially for wireless networks and remote office deployments. It offers enhanced performance flexibility, reliability, scalability, and security. There are opportunities for application vendors to use their applications to get maximum benefits from SMB 2.0 features.

17 GLOSSARY

ACL – Access Control List

API – Application Programming Interface

FID – Filehandle Identifier

LAN – Local Area Network

NAS – Network Attached Storage

NFS – Network File System

QoS – Quality of Service

VPN – Virtual Private Network

18 REVISIONS

Date	Name	Description
March 2009	Reena Gupta	Creation