

EMC VNXe Series: Introduction to SMB 3.0 Support

Abstract

This white paper introduces the Server Message Block (SMB) 3.0 support available in the EMC® VNXe™ series and the advantages gained over the previous SMB versions. A VNXe system can be implemented in a Microsoft® Windows® environment (with Microsoft Windows 8 and Windows Server 2012) to leverage the enhancements in the SMB 3.0 protocol.

January 2013

Copyright © 2013 EMC Corporation. All Rights Reserved.

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

The information in this publication is provided “as is.” EMC Corporation makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

For the most up-to-date listing of EMC product names, see EMC Corporation Trademarks on EMC.com.

VMware is a registered trademark of VMware, Inc. in the United States and/or other jurisdictions. All other trademarks used herein are the property of their respective owners.

Part Number h11383

Table of Contents

Executive summary	4
Audience.....	5
Terminology	5
Concepts	6
SMB Version and Negotiation.....	6
SMB Signing.....	6
File Locking	7
Continuous Availability	7
Server Failover.....	7
Client Failover.....	10
Enabling Continuous Availability	11
Multipath I/O	12
Directory Leasing	13
Offload Copy	15
SMB Encryption	17
Encryption Settings	17
Encryption keys.....	19
Enabling Protocol Encryption.....	20
Remote Volume Shadow Copy Service	21
Conclusion	23
References	23

Executive summary

EMC VNXe series has incorporated the Common Internet File System (CIFS) protocol as an open standard for network file service. CIFS is a file access protocol designed for the Internet and is based on the Server Message Block (SMB) protocol that the Microsoft Windows operating system uses for distributing file sharing, printing, and communication services from a server over a network. SMB 3.0 protocol support is available with Microsoft Windows 8 and Microsoft Windows Server 2012 systems and has significant improvements over the previous SMB versions.

As of VNXe Operating Environment (OE) version 2.4, the SMB 3.0 protocol is enabled by default. With this, the VNXe series supports SMB 1.0, SMB 2.x, and SMB 3.0 protocol communications between client and server. The VNXe series supports SMB 3.0 enhancements that include:

- **Continuous Availability (CA)** – Continuous Availability enables applications to be less impacted during a Shared Folder server failure or recovery of application. In the situation where a storage processor is faulted or placed in service mode, storage resources are failed over to the peer storage processor. With CA, application access to all open files that were present prior to the failover is re-established and the failover is transparent to the end users. In addition, CA increases performance by making synchronous writes to the backend storage.
- **Multipath I/O (MPIO)** - Multiple TCP connections can be associated with the same SMB3.0 session; a client application can use several connections to transfer I/O on a Shared Folder server share. This optimizes bandwidth and enables failover and load balancing with multiple NICs.
- **Directory leasing** - By setting a lease on a directory, the client is automatically aware of changes made in an open directory. The client can manage a cache of directory content and avoid making several calls to the server through the network.
- **Offload Copy** - Copying data between shares by a Windows client is performed within the back-end storage itself, reducing the workload on the client, server, and network. If two SMB 3.0 servers share the same back-end storage, the client can do the copy across different servers.
- **SMB Encryption** - Provides secure access to data on SMB file shares, protecting data on untrusted networks and providing end-to-end encryption of data “in-flight.”
- **Remote Volume Shadow Copy Service (RVSS)** – With RVSS, point-in-time snapshots of the Shared Folder storage resources can be taken across multiple shares, providing improved performance in backup and restore.

Organizations with geographically-dispersed sites must uphold data availability without sacrificing security. Furthermore, performance implications need to be considered, especially when operations are distributed among remote sites.

Administrators responsible for managing a Windows environment in their company can leverage the VNXe series' support for SMB 3.0 in their IT infrastructure implementations to alleviate these concerns.

Audience

This white paper is intended for EMC customers, partners, and employees who want to understand the SMB 3.0 protocol support available with the VNXe series. Readers of this document should be familiar with VNXe CIFS shared folder storage and the Windows Operating System.

For additional information about VNXe CIFS Shared Folders, refer to *EMC VNXe Series Using a VNXe System with CIFS Shared Folders* on EMC Online Support (<https://support.emc.com>) › VNXe Product Page.

Terminology

- **Common Internet File System (CIFS)** – An access protocol that allows you to access files and folders from Windows hosts located on a network.
- **Shadow Copy** – A snapshot of a volume at one well-defined instant in time. VSS identifies each shadow copy by a 16-byte GUID (Globally Unique Identifier).
- **Share** – A named, mountable instance of shared-folder storage, accessible through a shared folder. Each share is accessible through the protocol (NFS or CIFS) defined for the shared folder where the share resides.
- **Shared folder** – A VNXe storage resource that provides access to individual file systems for sharing files and folders. Shared folders contain Windows shares (which transfer data according to the CIFS protocol) or NFS shares (which transfer data according to the NFS protocol). NFS shares are sometimes referred to as NFS exports.
- **Shared folder server** – A VNXe server that uses the CIFS or NFS protocol to catalog, organize, and transfer files within designated shares. A shared folder server is required to create shared folders that contain CIFS or NFS shares, or NFS VMware datastores.
- **Storage processor (SP)** – A hardware component that provides the processing resources for performing storage operations, such as creating, managing, and monitoring storage resources.
- **Unisphere** – A web-based management environment for creating storage resources, configuring and scheduling protection for stored data, and managing and monitoring other storage operations.

- **Volume Shadow Copy Service (VSS)** – Windows service and architecture that coordinate various components to create consistent point-in-time copies of data called shadow copies.

Concepts

SMB Version and Negotiation

The SMB protocol follows the client-server model; the protocol level is negotiated by the client request and server response when establishing a new SMB connection.

- **SMB 1.0** – Version used in Microsoft Windows 2000, Windows XP, Windows Server 2003, and Windows Server 2003 R2.
- **SMB 2.0** – Version used in Windows Vista (SP1 or later) and Windows Server 2008.
- **SMB 2.1** – Version used in Windows 7 and Windows Server 2008 R2.
- **SMB 3.0** – Version used in Windows 8 and Windows Server 2012.

Before establishing a session between the client and server, a common SMB dialect must be negotiated. As shown in Table 1, the common dialect used depends on the SMB version supported by both the client and server.

Table 1: SMB dialect used between client and server

Client / Server	SMB3.0	SMB2.1	SMB2.0
SMB3.0	SMB3.0	SMB2.1	SMB2.0
SMB2.1	SMB2.1	SMB2.1	SMB2.0
SMB2.0	SMB2.0	SMB2.0	SMB2.0
SMB1.0	SMB1.0	SMB1.0	SMB1.0

It is important to note that SMB 3.0 leverages the SMB 2 protocol; the same SMB 2 packet formats are utilized by SMB 3.0.

For more information on SMB 2.x versions and negotiations, refer to the Microsoft TechNet Website at <http://technet.microsoft.com>.

SMB Signing

SMB signing is a mechanism in the SMB protocol that is used to ensure that packets have not been intercepted, changed, or replayed. Signing adds a signature to every packet and guarantees that a third party has not changed the packets. The client and VNXe Shared Folder Servers use this signature to verify the integrity of the packet. It is

important to note that the VNXe does not require use of signing; this is dependent on the client's configuration.

For SMB signing to work, the client and the server in a transaction must have SMB signing enabled. SMB signing is always enabled on the VNXe Shared Folder Servers, but is not required. As a result, if SMB signing is enabled on the client, signing is used, and if SMB signing is disabled on the client, no signing is used.

The SMB signing policy can be changed through group policy objects (GPO) settings or in the registry.

When signed, the SMB 2.x messages contain a 16-byte signature using HMAC SHA-256 in the SMB2_HEADER buffer that guarantees the integrity of the message. If SMB 3.0 is negotiated, the sender must compute a 16-byte hash using AES-CMAC-128 over the entire message, beginning with the SMB 2.x Header and using the signing key.

File Locking

File locking ensures file integrity when multiple users attempt to access the same file at the same time. File locks manage attempts to read, write, or lock a file that is in use by another user.

CIFS uses opportunistic locks (oplocks) and deny modes. The CIFS protocol enforces strict file locking and unlocked access to files. The CIFS locks are mandatory. When a CIFS process locks a file, other users are denied certain types of access to the file, depending on the type of lock imposed.

A CIFS client can lock a file by denying read/write access on the whole file or a lock range on a portion of the file.

An oplock leasing model was introduced in SMB 2.1. Leasing limits the amount of data needed to be transferred between the client and the server. This allows for improved file and metadata caching by the SMB client and better response time by the client's applications.

Continuous Availability

In an event of a client or Shared Folder Server failure, the Continuous Availability (CA) feature in the VNXe series allows Windows-based clients to persistently access shared folder storage without the loss of the session state. You can leverage this feature when planning with storage availability in mind for business-critical applications such as Microsoft SQL Server, IIS, or Hyper-V.

It is important to note that, with the VNXe, CA working in a SP failover scenario is dependent on the failover timeout of the application.

Server Failover

In order for a server using SMB 3.0 to recover client content in case of a server failover, persistent handles have been introduced in SMB 3.0. Persistent handles enable a server to save on disk specific metadata associated to an open handle. If a

failover occurs, it does not impact the applications accessing the open files or their content.

VNXe series mitigates failure at the SP-level by failing over to the peer SP. If one SP fails, a Shared Folder Server (and associated storage resources) on that SP would still be accessible to retain the content after a failover (as opposed to the previous implementation where an application had to restart its entire content).

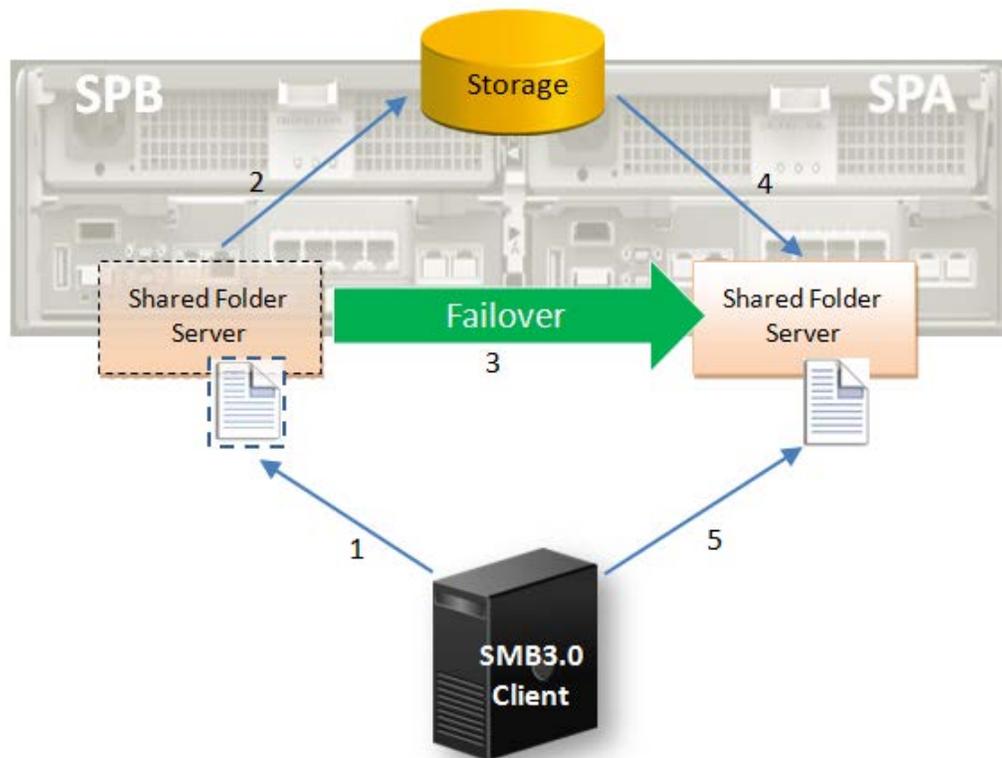


Figure 1 Failover transparent to the client

The series of steps taken when the Windows client supporting SMB 3.0 accesses data from a VNXe Shared Folder Server and SP failover occurs are highlighted below (see Figure 1 for reference):

1. The Windows client requests a persistent handle by opening a file (with associated leases and locks) on a Shared Folder share residing on SPB.
2. The Shared Folder Server saves the open state and persistent handle on the disk.
3. If SPB fails, is restarted, or placed in service mode, the Shared Folder Server fails over to SPA.
4. The Shared Folder Server (that has failed over) reads and restores the persistent open state from the disk before starting the service.

5. Using the previous persistent handle, the Windows client re-establishes connection to the same Shared Folder Server on SPA, and recovers the same context (for example, leases and locks) associated with the open file as before the failover occurred.

CA allows SMB clients to transparently reconnect to the peer SP without affecting clients accessing files on the shares.

A persistent open requires that certain file system changes be committed to the back-end storage before the request is completed, which includes:

- All file data.
- All namespaces changes.
- Security descriptor.
- File size.
- File dos attributes.
- File timestamps on explicit setInfo or create.

If a failover of an SP occurs, the server will preserve the state of the persistent handle. This means that any new open state conflicting with the share mode or new range lock requests will be denied. Furthermore, any new open state involving a lease break will be suspended.

Before enabling the CA feature, keep in mind that:

- With CA enabled, you can achieve a transparent server failover for implementations where the failover time is no longer than the application timeout. In such implementations, hosts can continue to access a CIFS resource without the loss of a CIFS session state, following a failover event.
- CA was designed for applications, such as Microsoft Hyper-V or SQL Server, where the file open/close operations are limited. It is not recommended to use in scenarios where the impact of storing the open states is not negligible (for example, Home Directory implementations).

Client Failover

A common approach to mitigate data unavailability at the client level is to configure Windows clients in a Windows cluster.

In the previous SMB 2.x implementation, the problem with this scenario was that the server kept the handles of the clients with associated locks. Since the peer node (in which the application had failed over to) is not the same as the original node, a conflict occurred when the application tried to re-open the file and set the same locks. The server did not realize that this was the same application trying to re-open its file.

To address this issue, SMB 3.0 introduces the concept of an application ID. Using this, the application will not conflict with the previously established session prior to the failover because it is now identified by the application ID. This ID stays persistent with the failover.

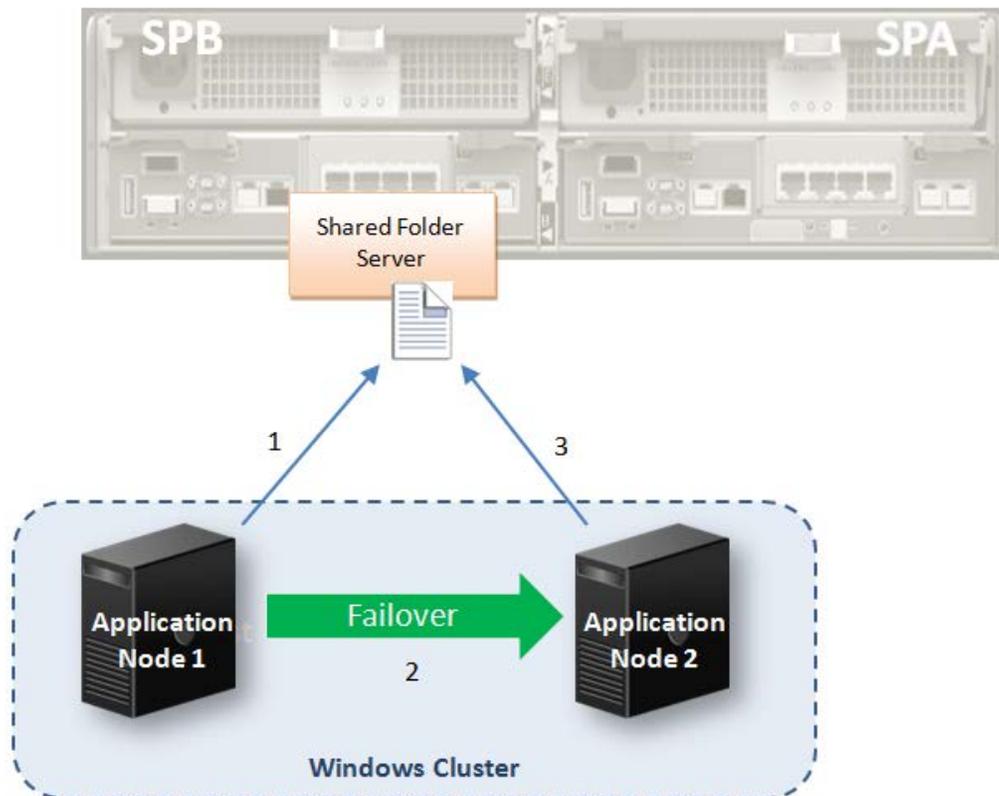


Figure 2 Application node in Windows cluster fails over

The following steps describe the basic sequence when a node in a Windows cluster fails over (see Figure 2 for reference):

1. The application opens a file or set locks on the Shared Folder share associated with storage on SPB.
2. If a failure occurs on node 1, the Windows cluster moves the application from node 1 to node 2.

The application re-opens its content on the Shared Folder share from node 2. Applications accessing data stored in Shared Folder shares are not interrupted.

Enabling Continuous Availability

The Continuous Availability (CA) feature is disabled by default. However, you can set it at the share-level when creating Shared Folder storage in Unisphere (shown in Figure 3).

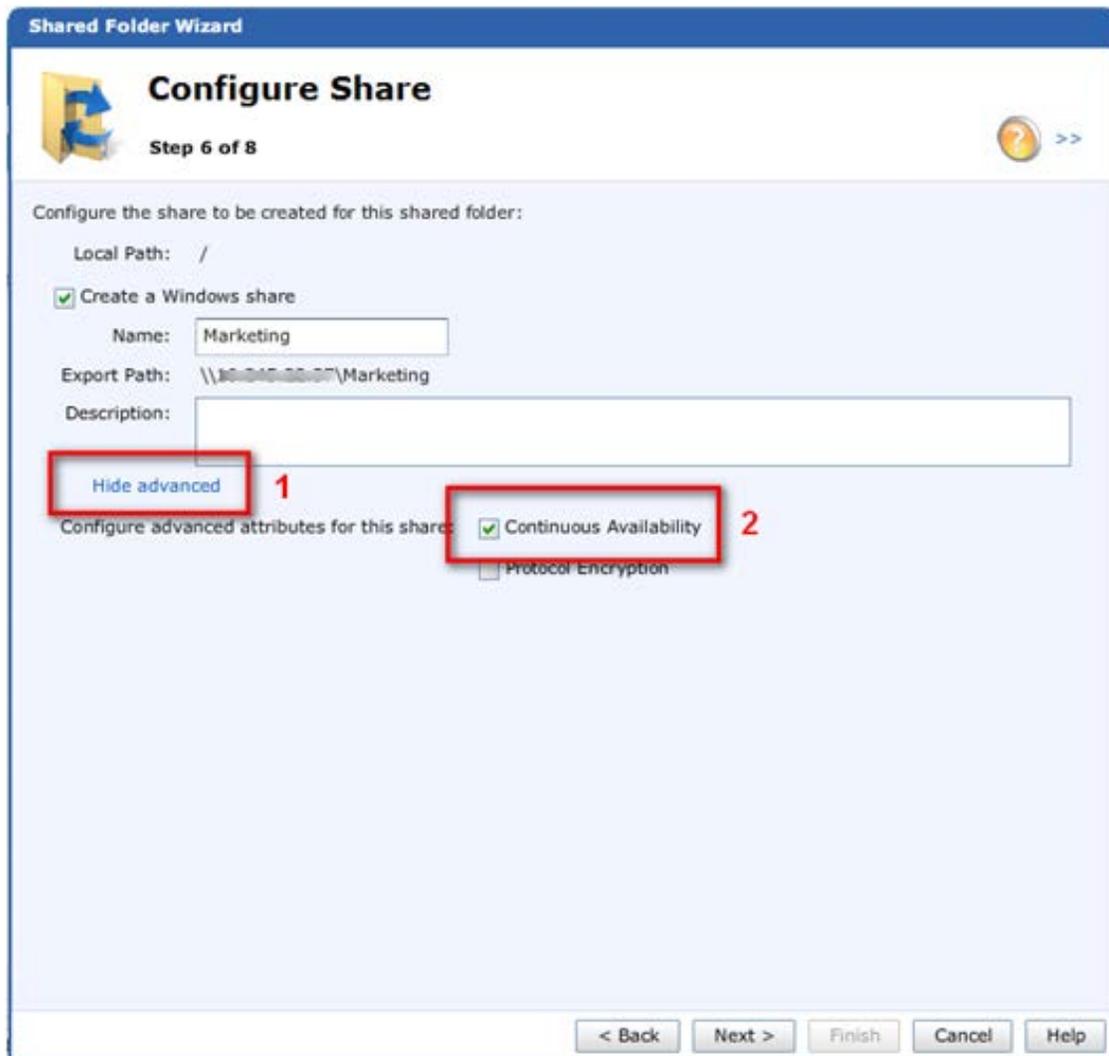


Figure 3 Enabling CA when creating a new share

Alternatively, you can also enable CA for any existing Shared Folder storage by navigating to the **Details** page of the Shared Folder storage resource > **Shares** tab > **Details** > **Show advanced** link (as shown in Figure 4).

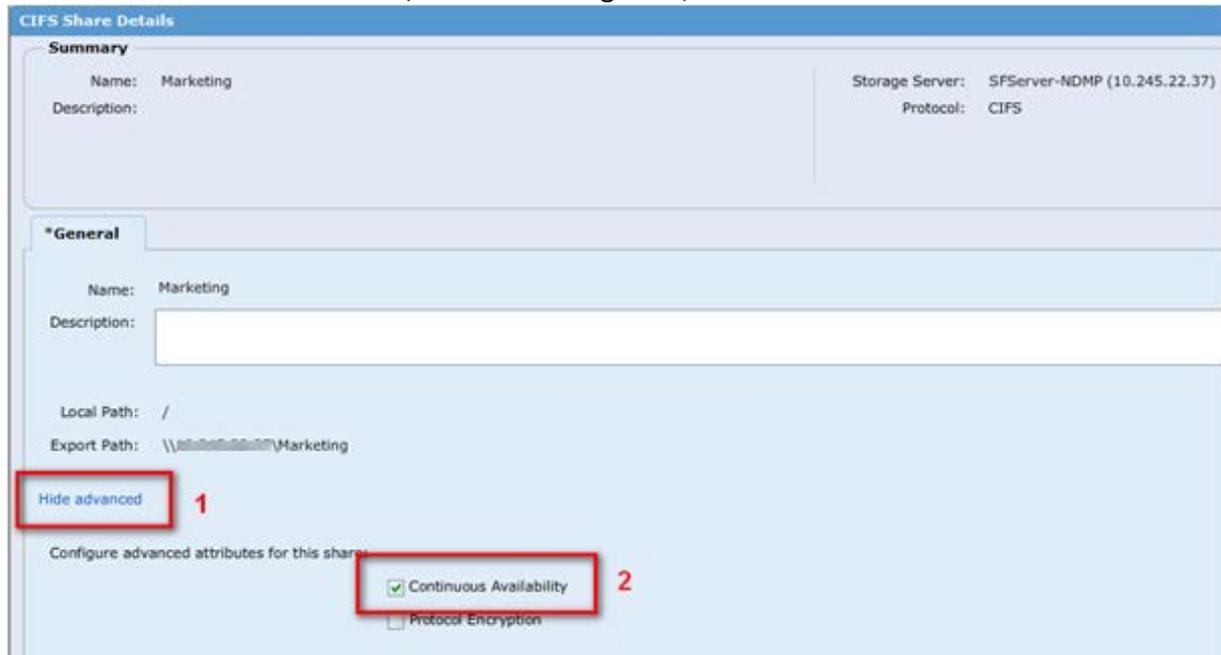


Figure 4 Enabling CA for an existing share

Multipath I/O

SMB 3.0 introduces Multipath I/O (MPIO), where multiple TCP connections can be associated with a given SMB session. A restriction in the previous SMB version was that only one TCP connection could be established per session. Now, if one TCP connection is broken, the user session can still continue using the remaining, active TCP connections. The session remains established until the last TCP connection is terminated.

MPIO provides several benefits, including:

- Increase of bandwidth associated with the SMB 2.x session a client application session opened on the VNXe Shared Folder server.
- Enables transparent network interface failover.
- Load balancing per session.

When establishing the session, the Shared Folder Server must send a response on the same TCP connection the request arrived on. As shown in Figure 5, there are two TCP connections associated with the given SMB session. With VNXe series, you can create up to 4 network interfaces per Shared Folder Server.

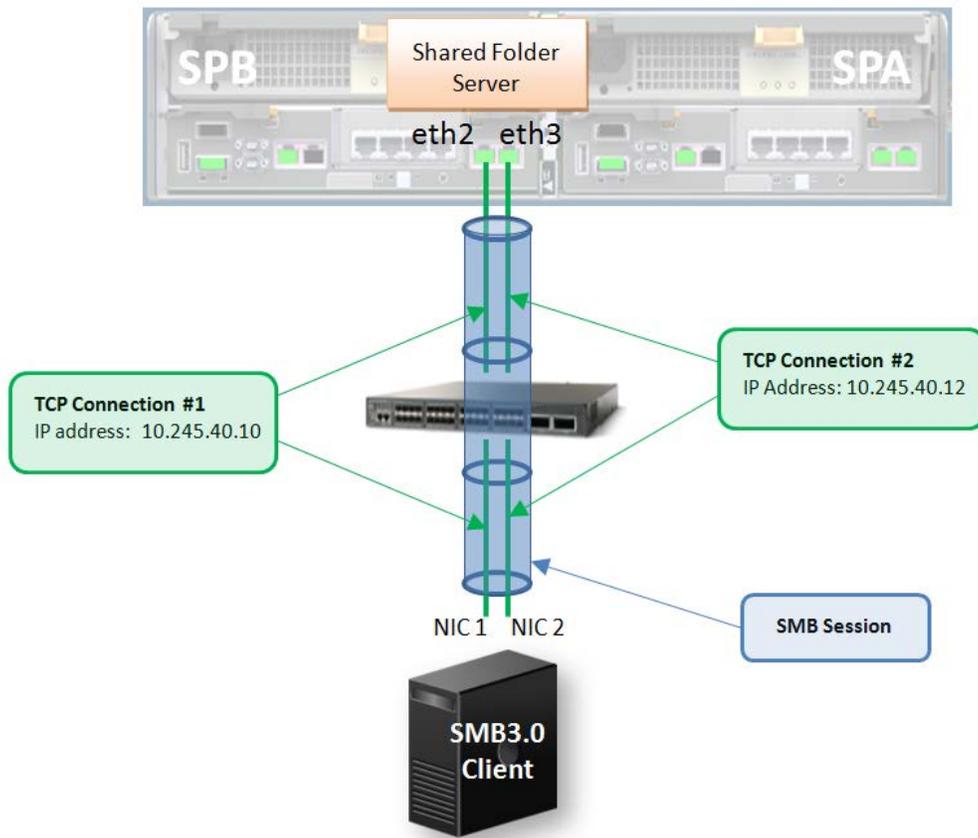


Figure 5 Multiple TCP connections per session

When connecting to the Shared Folder server, the client should have as many physical interfaces configured as there are TCP connections needed.

Directory Leasing

In SMB 2.0, Microsoft developed a directory content cache residing on the SMB 2.0 clients. This increased performance by reducing the amount of directory list requests transmitted over the network. However, a disadvantage of this feature was when multiple clients access the same files or folders at the same time; subsequent clients would not be aware of the directory content change even when the directory content was refreshed by the user.

SMB 3.0 improves on the directory content cache with the concept of a directory lease. By setting a lease on a directory, the Windows client is automatically aware of a content change made in an open directory. If content is created, modified, or deleted in the directory, then the directory lease set by the Windows client is broken.

There are three types of leases:

- Read-caching lease (R) allows a client to cache reads and can be granted to multiple clients.
- Write-caching lease (W) allows a client to cache writes.
- A handle-caching lease (H) allows a client to cache open handles and can be granted to multiple clients.

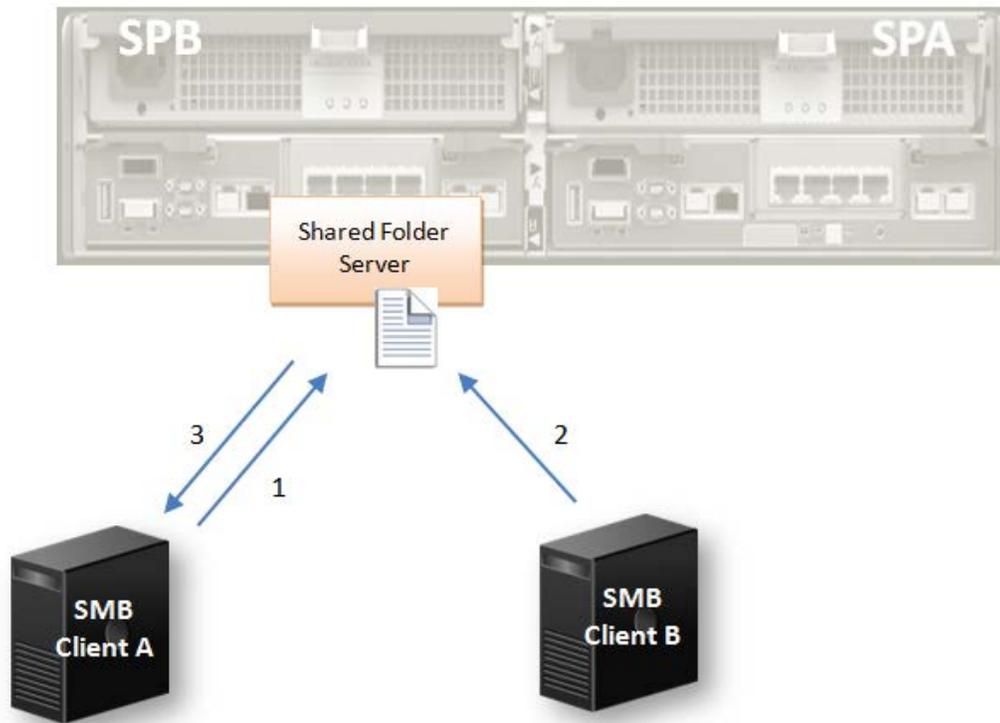


Figure 7 Directory content is automatically updated

The following steps describe the sequence when a directory lease is broken (see Figure 7 for reference):

1. Client A opens a directory on the Shared Folder server and sets R and H lease on it.
2. Client B changes a file or folder in the same directory. By making this change, the directory lease is broken.
3. Client A is notified automatically that its cached directory content is no longer up to date and sends a query to update that directory.

Offload Copy

With SMB 3.0, Offload Copy can be leveraged when copying between shares associated with the same Shared Folder server or across different servers if these servers both reside on the same SP.

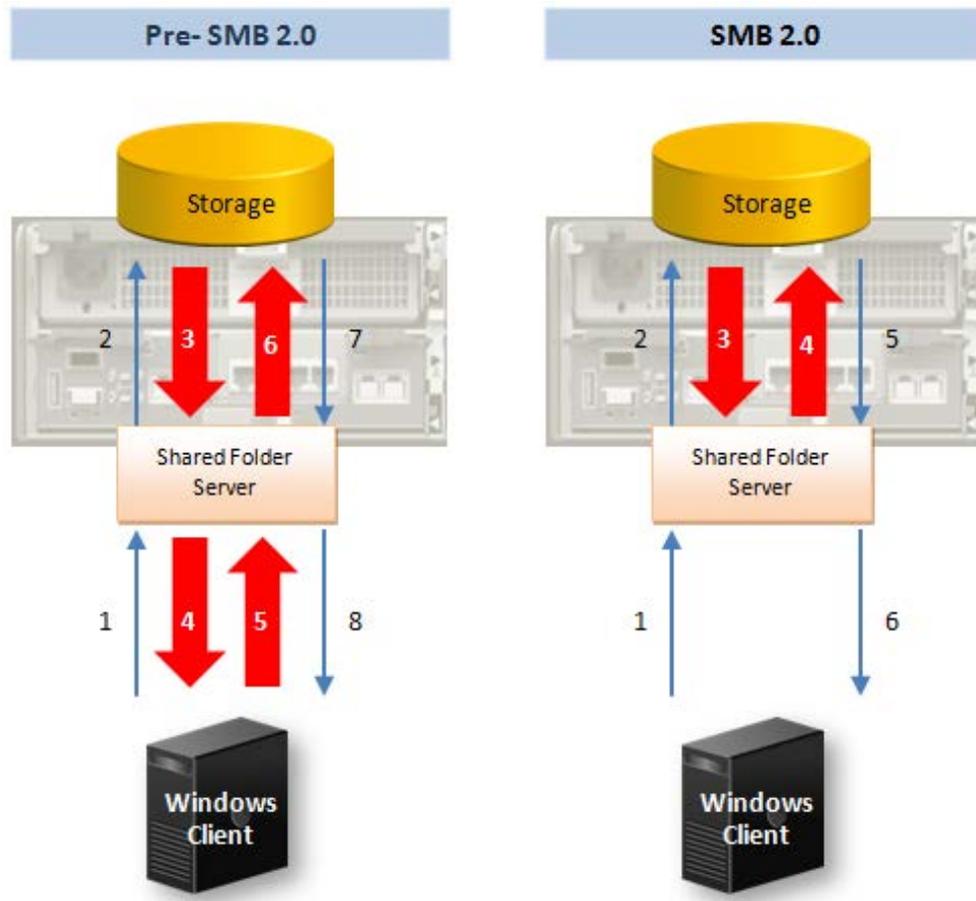


Figure 8 Copying data prior to SMB3.0

Prior to the SMB 2.0 protocol, a request from the Windows client to copy data from one share to another in the VNXe system required the following interaction between the Windows client, Shared Folder Server, and back-end storage (see Figure 8 for reference):

1. The client requests data to read.
2. The Shared Folder Server requests data from the back-end storage.
3. The back-end storage sends data to the Shared Folder Server.
4. The Shared Folder Server sends data to the client over the network.
5. The client sends back the same data to the Shared Folder Server over the network.
6. The Shared Folder Server requests data from the back-end storage.

6. The Shared Folder Server sends back the same data to the storage in the copy-to location.
7. The back-end storage acknowledges the Shared Folder Server.
8. The Shared Folder Server acknowledges the client.

Transferring data between the client, server, and back-end storage resulted in an overhead in performance and resources. In SMB 2.0, enhancements allowed the client to directly send requests to the server to perform the copy itself. Having data moved between the back-end storage and the SMB 2.0 server lessened the stress on the network.

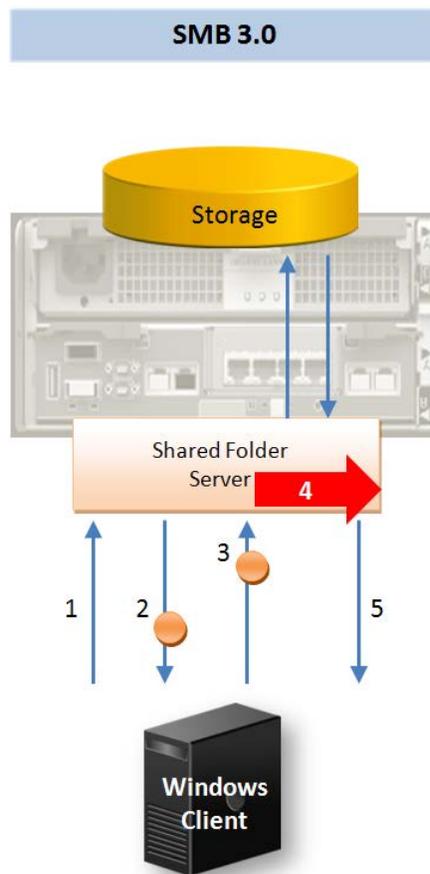


Figure 9 Offload Copy transfers data with storage

With the introduction of the Offload Copy feature, the data exchange needed between the back-end storage and the Shared Folder Server for a copy operation has been reduced.¹ The following steps describe the sequence of Offload Copy (see Figure 9 for reference):

¹ Files larger than 256KB are copied within the back-end storage itself.

1. The client requests a token representing a range of bytes of an open file to copy.
2. The Shared Folder Server responds to the client with a token.
3. The client requests for the server to copy the data to the destination location using the token that was previously returned (The destination location can be associated another Shared Folder server).
4. The Shared Folder Server performs the copy between the source and destination location.
5. The Shared Folder Server acknowledges the copy to the client.

By lessening the need for the client to read and write across the network, the CPU usage and network bandwidth required of the client is reduced.

SMB Encryption

SMB encryption provides secure access to data over untrusted networks by providing end-to-end encryption between the client and the VNXe system. It does not require specialized hardware, IPSec, or WAN accelerators. Scenarios where SMB encryption can be utilized include Remote Office Branch Office (ROBO) over WAN networks and application workload over unsecured networks.

Encryption at the share-level is enabled on the particular share, and encryption is enforced when that share is accessed. Optionally, encryption can be enforced at the system-level (where encryption is set in the registry of the Shared Folder server), and all share access would require encryption. There is no configuration needed at the client-level.

Encryption Settings

To support the SMB 3.0 protocol, VNXe series has two new values added to the registry of the Shared Folder Server; *EncryptData* and *RejectUnencryptedAccess*.

Setting the *EncryptData* value enforces that Shared Folder Server to advertise the encryption capability in the negotiate response. By default, the *EncryptData* value is disabled. Setting the *RejectUnencryptedAccess* value prevents clients that do not support encryption from establishing a session to the share. Instead, the client receives an ACCESS_DENIED message after the failed attempt. By default, the *RejectUnencryptedAccess* value is enabled.

Table 2 SMB Encryption registry values

Value	Type	Default Value	Description
EncryptData	DWORD	0	If set, all sessions established from SMB3.0 clients to the Shared Folder server will be encrypted.
RejectUnencryptedAccess	DWORD	1	If set, the Shared Folder server rejects session requests from clients that do not support encryption.

In general, these parameters can be left at their default values. If you need to meet specific needs in your IT environment, these parameters can be configured in the registry of the Shared Folder server:

1. Navigate to the registry of the local computer.
2. Click on **File > Connect Network Registry**.
3. Enter the hostname or IP Address of the Shared Folder server and click the **Check Names** button. When the server is recognized, click the **Ok** button to close the window.
4. Navigate to the path (as shown in Figure 10):

HKLM\System\CurrentControlSet\Services\LanmanServer\Parameters

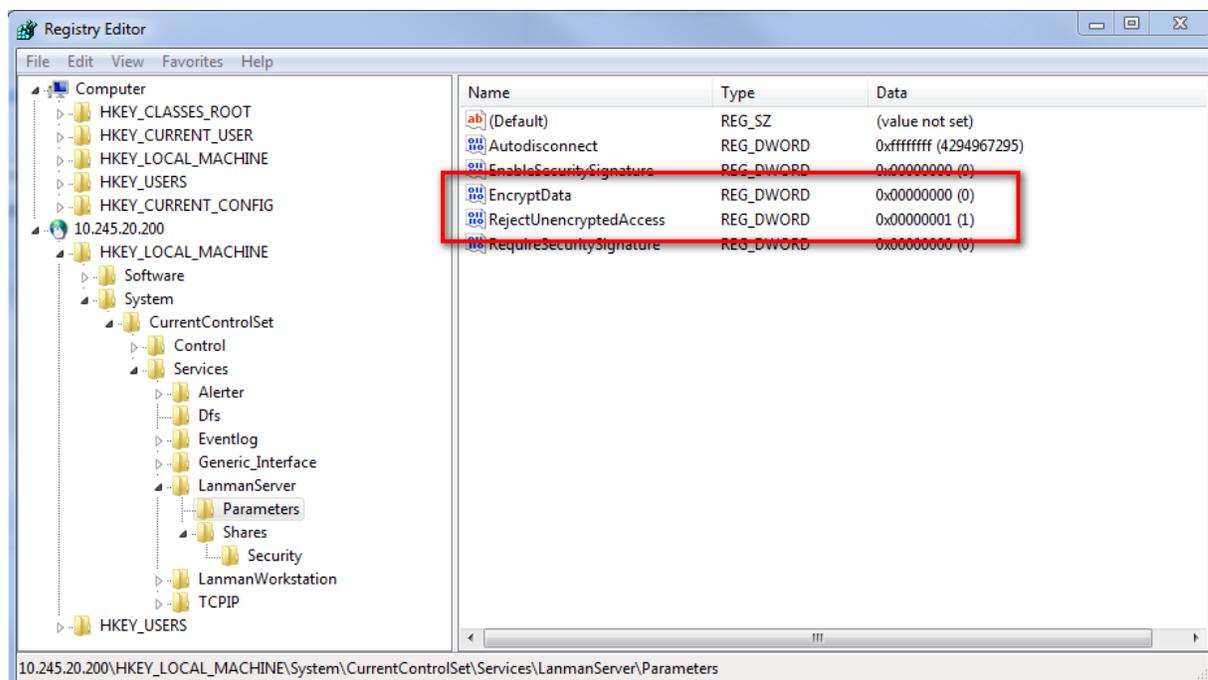


Figure 10 Location of SMB encryption values in the Registry

The way the *EncryptData* and *RejectUnencryptedAccess* values are set determines whether or not a client that does not support encryption will be granted access to the Shared Folder share (see Table 6).

Table 3 Establishing an encrypted session and access shares

EncryptData	RejectUnencryptedAccess	Pre-SMB3.0 client
0	0	Able to establish session with server and access encrypted shares.
0	1	Able to establish session with server and access unencrypted shares.
1	0	Able to establish session with server and access encrypted shares.
1	1	Unable to establish session with server and access shares.

It is important to note that disabling the *RejectUnencryptedAccess* value gives pre-SMB 3.0 clients the ability to access both encrypted shares and unencrypted shares.

In the VNXe series, encryption can be enabled at the share-level via the Protocol Encryption option. Once enabled, the SMB payload is encrypted only if an encrypted share is accessed.

In you want to allow pre-SMB 3.0 clients to access unencrypted shares, but deny access to encrypted share, you can use the *RejectUnencryptedAccess* value in conjunction with the Protocol Encryption option (see Table 7).

Table 4 Accessing encrypted shares

Protocol Encryption	RejectUnencryptedAccess	Pre-SMB3.0
0	0	Able to access share.
0	1	Unable to access share.
1	0	Able to access share.
1	1	Unable to access share.

If the *RejectUnencryptedAccess* value is enabled, the Shared Folder Server will reject access for pre-SMB 3.0 clients attempting to access the encrypted share.

Encryption keys

Incoming and outgoing traffic are encrypted using two different secret keys. Both are computed once the user is authenticated successfully. The encryption and decryption 16-bytes keys are generated using the KDF algorithm in Counter Mode.

SMB messages on the network are encrypted between the client and server using the AES128-CCM cryptographic algorithm, as described in RFC4309 and RFC3610.

The format of the encrypted messages consists of an SMB 2 TRANSFORM_HEADER header followed by the payload. Any SMB 2 message can be encrypted, except SMB2_NEGOTIATE and SMB2_SESSION_SETUP.

Enabling Protocol Encryption

You can enable protocol encryption when creating the Shared Folder storage resource (see Figure 11).

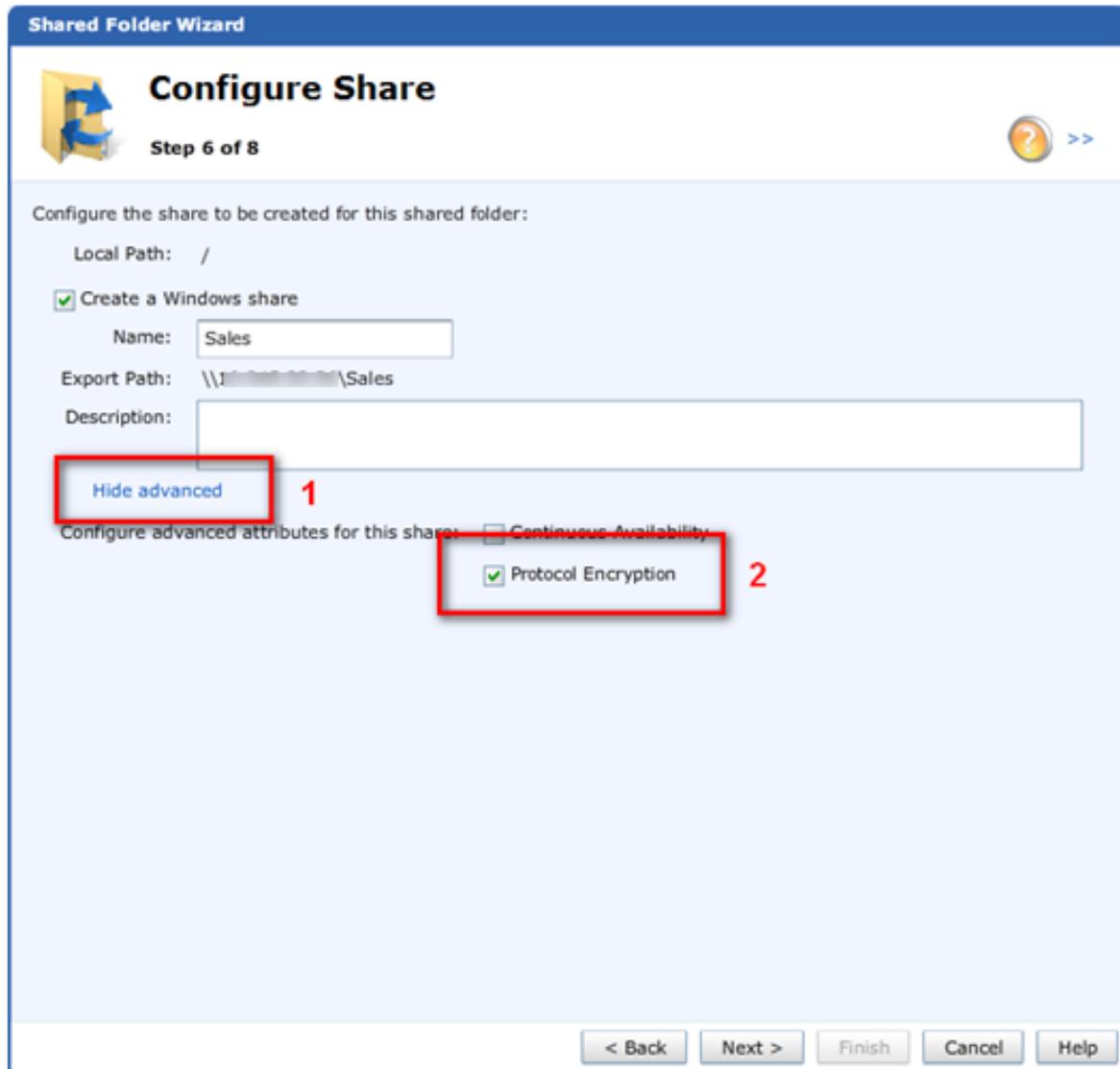


Figure 11: Enabling Protocol Encryption creating a new share

Alternatively, you can set the protocol encryption on an existing Shared Folder share by navigating to the **Details** page of the Shared Folder storage resource > **Shares** tab > **Details** > **Show advanced** link (as shown in Figure 12).

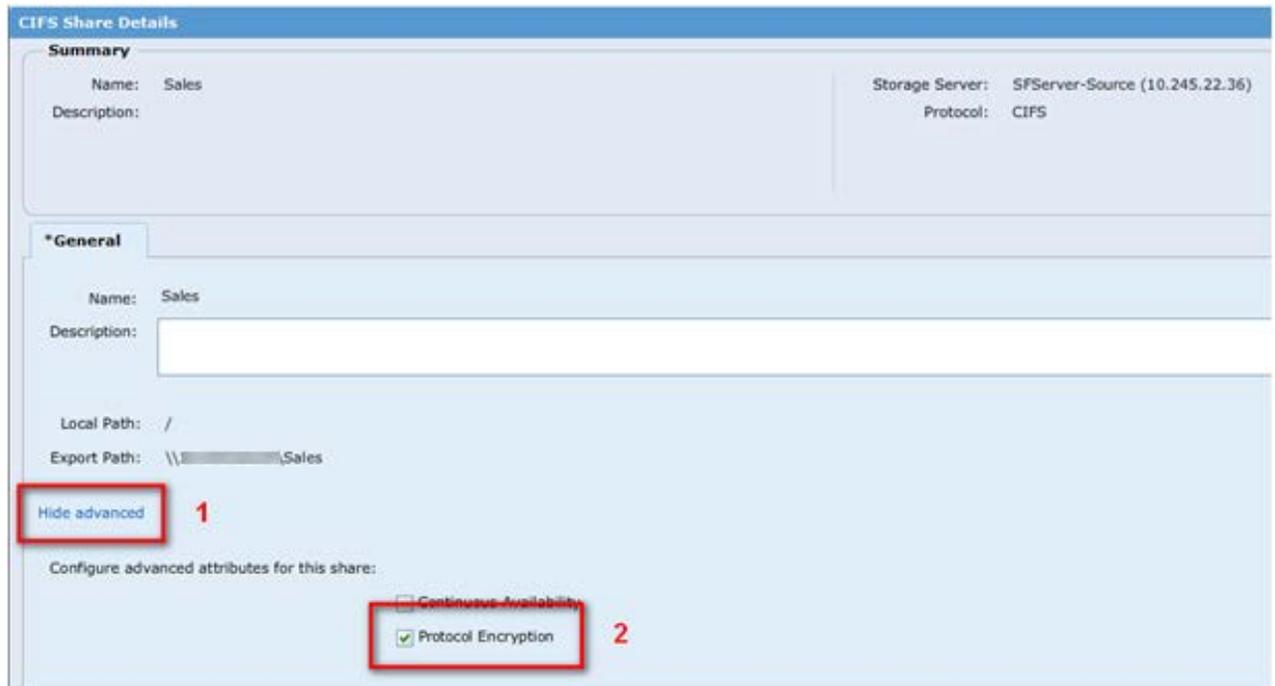


Figure 12: Enabling Protocol Encryption for an existing share

It is important to note that there is a performance impact associated with enabling Protocol Encryption. Data is encrypted at the source system before it gets send across the network and unencrypted at the destination as it is received. CPU resources are required on both client- and server-side for these operations to occur.

Remote Volume Shadow Copy Service

Volume Shadow Copy Service (VSS) infrastructure allows file systems to be backed up while applications continue to write to these volumes and enables creation of snapshots (called shadow copies for Microsoft VSS). Prior to SMB 3.0, VSS could only create and manage snapshots of data stored on local volumes. With Windows 2012, Microsoft has extended this feature with Remote Volume Shadow Copy Service (RVSS).

RVSS supports application backup across multiple file servers and shares. Backup applications that are VSS-aware can now perform snapshots of server applications that store data on the VNXe Shared Folder shares. Hyper-V now has the ability to store virtual machine files (for example, virtual hard disk, configuration, and so on) on Shared Folder shares; RVSS can be leveraged in taking point-in time copies of the share content.

For instance, let's take a look at a simple deployment of a single Hyper-V server utilizing the VNXe system to store virtual machines and user data (see Figure 13).

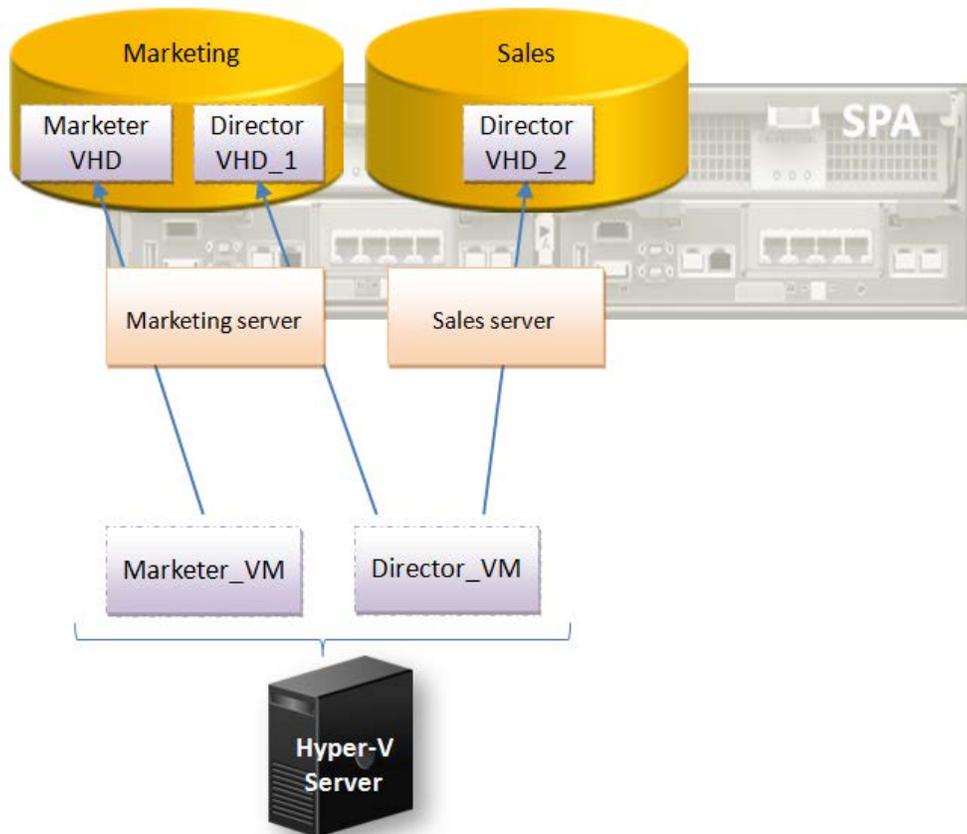


Figure 13: Backing up Hyper-V VMs on file shares

Two Hyper-V storage resources and associated servers have been provisioned on the VNXe system; one for the Marketing team and the other for the Sales team. Each member of these teams has an assigned virtual machine and accesses files via their group's respective shares.

Marketer_VM, a marketer's virtual machine, has virtual machine files stored on the Marketing share whereas Director_VM, the director's (of both the Marketing and Sales departments) virtual machine, has virtual machine files stored on both the Marketing and Sales shares. When the backup application performs a snapshot of Marketer_VM, the VNXe system will take a snapshot of the Marketing volume. A snapshot of the Sales volume will not be taken since only the Marketing share was reported by the Hyper-V application. After the snapshot has been taken, a share containing the snapshot will be available for the backup application to perform a backup of the data. When a snapshot of Director_VM is taken, both the Marketing and Sales volumes will be included in the snapshot.

Conclusion

Data availability, performance, and security are factors you must consider in their organization's IT infrastructure. This is especially the case when these organizations have remote offices across different parts of the world. You can utilize the VNXe series' support of the SMB 3.0 protocol, along with Microsoft Windows 8 clients and Windows Server 2012, to address some of the common concerns.

Continuous Availability and Multipath I/O are features that maintain data availability in situations of a failure at the client, server, or network level. Offload Copy allows for efficient data consolidation or migration of content stored in Shared Folder storage resources.

In terms of security, SMB Encryption enforces protection when data is exchanged between Windows clients and a VNXe system over untrusted networks and allows administrators to decide which Windows clients are granted access to this data.

Leveraging RVSS, point-in-time snapshots of the Shared Folder storage resources can be taken across multiple shares, provide improved performance in backup and restore.

References

- EMC VNXe Storage Systems – A Detailed Review
- EMC VNXe High Availability – Overview
- EMC VNXe Series Using a VNXe System with CIFS Shared Folders
- Microsoft TechNet website