Integrating Mac® OS and NetApp Storage

Integration, Configuration and Performance in Mixed File Sharing Environments

Geert van Teylingen, Kevin Scott Karafa, Network Appliance, Inc.

March 2006, TR- 3472

Abstract

The inclusion of both the NFS and CIFS network file sharing protocols in Mac OS X dramatically simplifies the integration of Apple® systems into multiprotocol file sharing environments that include Windows®, UNIX®, and other systems. To assess the network file sharing functionality and performance of OS X, Network Appliance undertook extensive integration testing of the OS X native NFS, native CIFS, and DAVE™ and ADmitMac™, a third-party implementation suite of CIFS from Thursby Software Systems. Although native NFS and CIFS were found to have file size, file naming, and cross accessibility limitations that might be significant in some environments, they are adequate for many common applications.

Under the conditions tested, DAVE and ADmitMac were found to offer the best compatibility and performance in a mixed file sharing Windows/Mac OS environment and may be the preferred file sharing choice for the most demanding environments. This paper explores OS X integration, functionality and performance and offers guidelines for the implementation of OS X systems with NetApp storage systems in various mixed platform environments.

Table of Contents

1. Introduction

The inclusion of both the NFS and CIFS network file sharing protocols in Mac OS X dramatically simplifies the integration of Apple systems into multiprotocol file sharing environments that include Windows, UNIX, and other systems.

To assess the network file sharing functionality of Mac OS X, Network Appliance undertook integration testing of the Mac OS X native NFS, native CIFS, and DAVE and ADmitMac, an alternative third-party CIFS implementation suite from Thursby Software Systems (www.thursby.com).

To assess the network file sharing performance of OS X, Network Appliance and Ball State University in the past have performed extensive performance testing of OS X's native NFS, native CIFS, and DAVE. These performance tests and the results are also presented in this document.

This paper explores Mac OS X integration, functionality and performance and offers guidelines for the implementation of Mac OS X systems with NetApp storage systems in various client environments. Furthermore it describes various scenarios for the migration of existing Mac OS X storage environments to NetApp storage infrastructure to fully leverage the NetApp value add.

2. Background on Apple Networking

This chapter provides some background on Apple Macintosh networking and file sharing. The chapter will cover traditional Mac OS network connectivity, file structures, and file naming conventions.

2.1 Network connectivity

The traditional native Mac OS file sharing protocol is AFP (Apple Filing Protocol), known as AppleShare, which is the equivalent to SMB/CIFS in the Windows world and NFS in the UNIX world.

The AppleShare protocol is a communications protocol from Apple Computer that allows client applications on an Apple computer to exchange files with and request services from server programs in a computer network. AppleShare can be used over the network on top of the TCP/IP protocol (AppleShareIP) or on top of other network protocols such as AppleTalk (AppleShare). Using the AppleShare protocol, users can access files, applications, printers, and other resources on a remote server. AppleShare can communicate with any server program that is set up to receive an AppleShare client request.

All Macintosh systems include client and server AppleShare protocol support. Microsoft® provides an add-on service to the Windows 2000/2003 server family, called File Services for Macintosh (SFM). Other third-party AppleShare client/server support is also available for Windows and UNIX systems.

Note: NetApp storage systems only provide support for CIFS and NFS and cannot be extended with an AppleShare (AFP) add-on. When integrating NetApp storage systems into a Mac OS environment this has to be taken into account.

2.2 File structures (HFS+, Data and Resource Forks)

Under the Macintosh file system (called HFS+, for Hierarchical File System), files are not monolithic and do not consist of one single segment. They may be composed of two pieces, called Forks, i.e., a Data Fork and a Resource Fork. Any one of the two forks may be empty. The creators of the Macintosh file system chose to put apart what is stored in a unique data stream on most computers.

- **Data Fork**
  The Data Fork contains what we generally call a file on a PC, that is, the text created by a word processor, the pixels building a picture, etc. When transferring data files between a Macintosh and

a PC, the data fork is generally the only part of interest (the major exceptions are font files and program executables; see below).

- **Resource Fork**
  The Resource Fork may contain the code of a program, the commands of a font file, etc. Some programs can store a preview picture in the Resource Fork. The Resource Fork is organized as a database, which allows for fast access to all components. This is a big advantage for some applications. It is also possible to change and replace some elements without disturbing the rest of the fork.

Note: Unlike HFS+, other platforms do not use a "dual file fork" approach. When Mac files need to be stored on a platform other than a Mac platform, this has to be taken into account. For instance, program code is stored in the Resource Fork so program files often have an empty Data Fork. This explains why a program that has been transferred through channels ignoring file forks and has lost its Resource Fork can no longer be executed (Apple error -39, meaning unexpected end of file).

There are a couple of possible ways of storing files on a "foreign" file system using various formats, of which the *AppleSingle* and *AppleDouble* formats are most well known:

- **AppleSingle**
  In the AppleSingle format, a file's contents and attributes are stored in a single file in the foreign file system. For example, both forks of a Macintosh file, the Finder™ information, and an associated comment are arranged in a single file with a simple structure.
  An AppleSingle file consists of a header followed by one or more data entries. The header consists of several fixed fields and a list of entry descriptors, each pointing to a data entry. Each entry is optional and may or may not appear in the file.

- **AppleDouble**
  The AppleDouble format uses two files to store data, resources, and attributes. The AppleDouble data file contains the Data Fork and the AppleDouble header file contains the Resource Fork.

2.3 File naming conventions

HFS+ uses Mac ANSI, which supports the use of a lot of different characters in file names (with the exception of the colon [:], which is used as a directory separator). This means that file names can consist of any character including / \ * ? < > etc. (typically not allowed in other environments such as Windows and UNIX), and even a NULL character.

In addition, AFP (both AppleShare and AppleShareIP) natively supports the transfer of files with names containing these characters and the transfer of both the Data Fork and Resource Fork between Mac clients (and servers).

Note: Normally the "special" characters (such as / \ * ? etc.) that can be used on the native Mac platform are not supported on many other platforms. This has to be taken into account and taken care of when Mac files with file names consisting of one or more of those characters must be stored on another than the native platform.

3. Windows File Services For Macintosh

This chapter gives some background on Microsoft File Services for Macintosh (SFM) and how it provides support for file serving in a Macintosh client environment.

3.1 Network connectivity

The Microsoft File Services for Macintosh enables an AFP service on Microsoft Windows servers (on Windows NT initially over [inefficient] AppleTalk®; since Windows 2000 over TCP/IP). Now Mac clients can connect to Windows servers using their native AFP stack. Data is shared by sharing volumes on local NTFS folders, similar to CIFS shares. In fact, the same NTFS folder can be shared with both Mac OS clients (exporting an AppleShare volume) and Windows clients (exporting CIFS shares) simultaneously.

3.2 File structures (Data and Resource Forks)

Microsoft SFM uses an approach that seems like a combination of AppleSingle and AppleDouble when storing Mac files on NTFS. SFM writes the Data and Resource Forks to a single file—storing the Resource Fork to this file using an NTFS-only feature called *Alternate Data Streams* (ADS; see www.heysoft.de/nt/ntfs-ads.htm). The result is that the Data Fork is stored in an ordinary file and the Resource Fork is stored in a "hidden" *stream within the same file*.

Note: Most command line copy programs (such as COPY and XCOPY) do not handle ADS properly. Any additional stream will be lost when copying those files with programs such as COPY and XCOPY. However, Windows Explorer is able to copy additional streams properly (of course, if the destination file system supports ADS)!

3.3 File naming conventions

Microsoft SFM internally uses Unicode for storing characters on NTFS for files with "invalid" chars (see *support.microsoft.com/kb/q117258*). The Apple computer will be able to see those file names properly (SFM and the AFP protocol take care of the translation) but on the Windows server or clients connecting through CIFS these characters will appear mangled. Explorer may display strange-looking characters and "dir" in the command prompt will show question marks (?) for each of these chars. Copying those directories with Windows Explorer is possible; however, using Command Line—for scripting purposes—can be difficult.

See the figures below for an example.

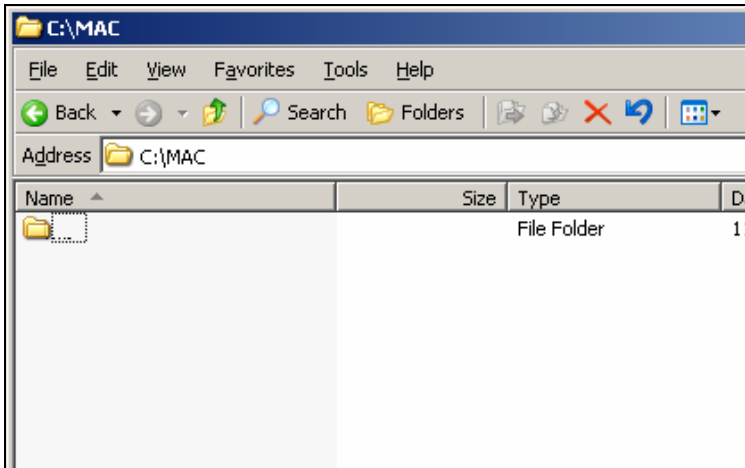**Figure 1) A folder name with "invalid" characters in an SFM AFP volume on a Mac client…**

Figure 2) …gets mangled into dots in Windows Explorer on the SFM server…
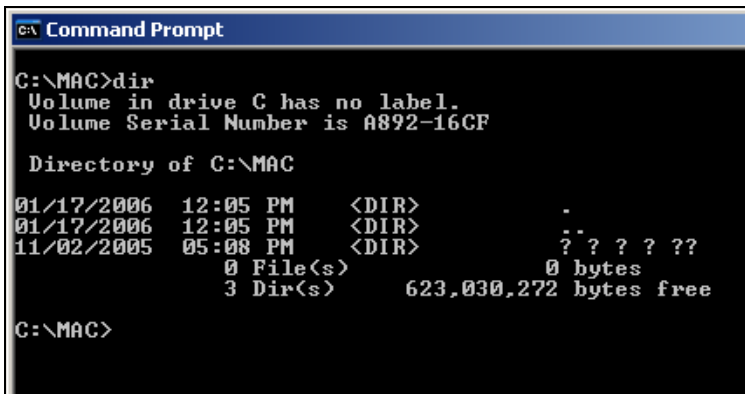

Figure 3) …and into question marks in the command prompt.

4. Mac OS X File Sharing Enhancements

With the introduction of Mac OS X, Apple has taken steps to facilitate the integration of Apple systems into heterogeneous file sharing environments. In addition to AppleShare, Mac OS X now includes native support for both NFS, for file sharing with UNIX and many other operating systems, and SMB/CIFS for integration with Windows environments. A single Mac OS X system can use AppleShare, NFS, and SMB/CIFS simultaneously, allowing file access in multiprotocol environments.

While this has major advantages in heterogeneous file serving environments and enables the use of NetApp storage systems for Mac OS X platforms—by using NFS or SMB/CIFS—there are consequences when using a protocol other than AppleShare with regarding file structures and file naming. This is covered in the following paragraphs.

4.1 Network connectivity

Instead of using AppleShare, now the Mac OS X native SMB/CIFS and NFS protocol stacks can be used to connect to file shares on the network. This enables Mac OS X clients to connect to regular Windows CIFS shares as well as to NetApp storage systems CIFS shares or NFS exports without the need for additional server components on either platform.

4.2 File structures (Data and Resource Forks)

When copying a Mac file to a CIFS share or NFS export using the Mac OS X native CIFS/SMB or NFS protocols, Mac OS X uses the AppleDouble approach.

This means files are split into two files: the Data Fork in a regular file and the Resource Fork in a separate file with the same name preceded by "._". So "Picture 1.png" on the Mac becomes "Picture 1.png" and "._Picture 1.png" on the CIFS shared NTFS or NFS exported UNIX file system. The Mac CIFS/NFS client takes care of handling the AppleDouble formatted files. The behavior is the same for Windows and NetApp CIFS shares as well as for UNIX NFS and NetApp NFS exports. See the figures below for an example.
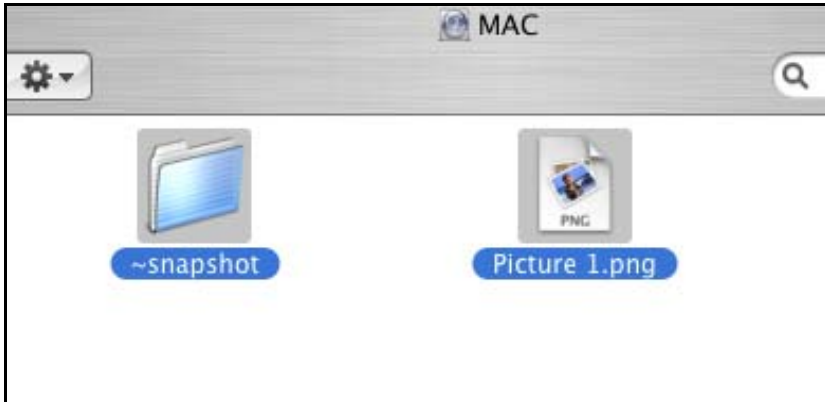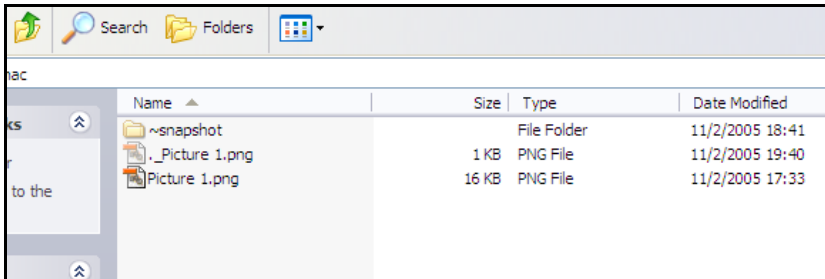

Figure 4) A single file on a Mac client…

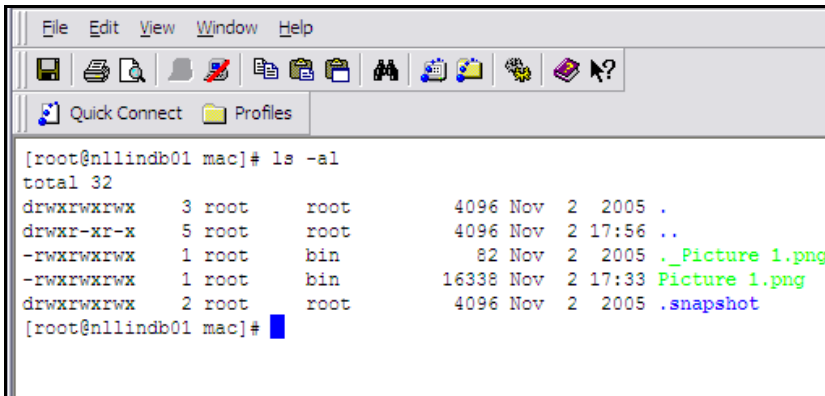**Figure 5) … becomes two files on a Windows client…]**

Figure 6) … as well as on a UNIX client.

4.3 File naming conventions

Depending on the protocol used (CIFS or NFS), various characters cannot be used on the target file system. Therefore the Mac OS X protocol will use some Unicode or other translation to store those characters on the target. The method to be used will be negotiated with the target and depends on the protocol..

4.3.1 CIFS shares Copying a Mac file with a Windows "illegal" character in the file name to a CIFS share using the Mac OS X native CIFS/SMB protocol is not supported by the Apple CIFS implementation. *Attempting to copy these files will result in an error message on the client.* See the figure below. This behavior is identical for Windows and NetApp CIFS shares.
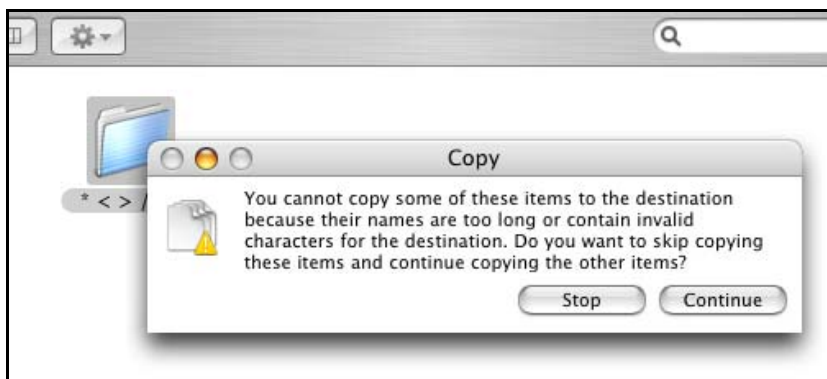

Figure 7) Copying files to CIFS share generates an error on the Mac OS X client.

4.3.2 NFS exports Most characters can be used in file names when storing on NFS exports. There are, however, some special cases, namely, files and folder names beginning with a period (.). These file names are not allowed in Mac OS X as they are considered hidden files/directories. A major drawback of this fact is that the .snapshot folder will not be visible in the Mac OS X Finder. Restoring data from NetApp Snapshot™ copies on NFS-mounted volumes will be covered in paragraph 6.5.2.

While a lot of characters are allowed on NFS-mounted volumes it is important to consider the destination's OS file naming conventions and the use of certain characters (such as %, $, |, <, >, etc.) because they are used to indicate variables or redirections in shell scripts in UNIX environments. This can cause conflicts when trying to access those files in shell scripts on other UNIX systems mounting the same export over NFS or when accessing data from the Terminal. See the figures below.
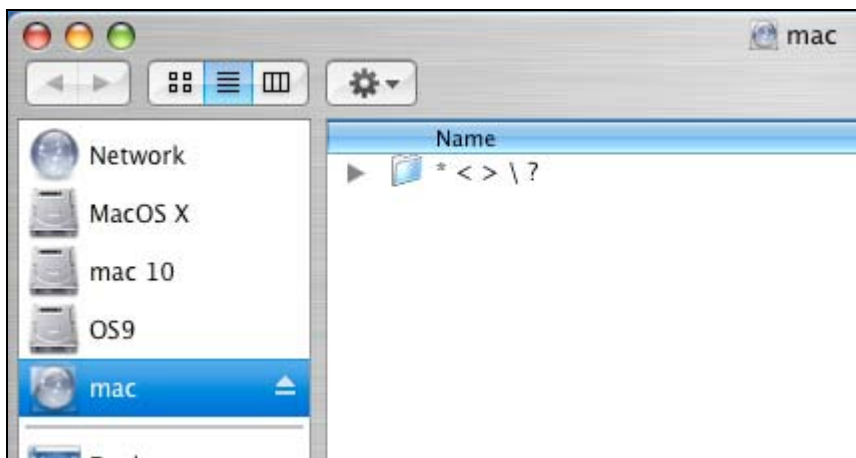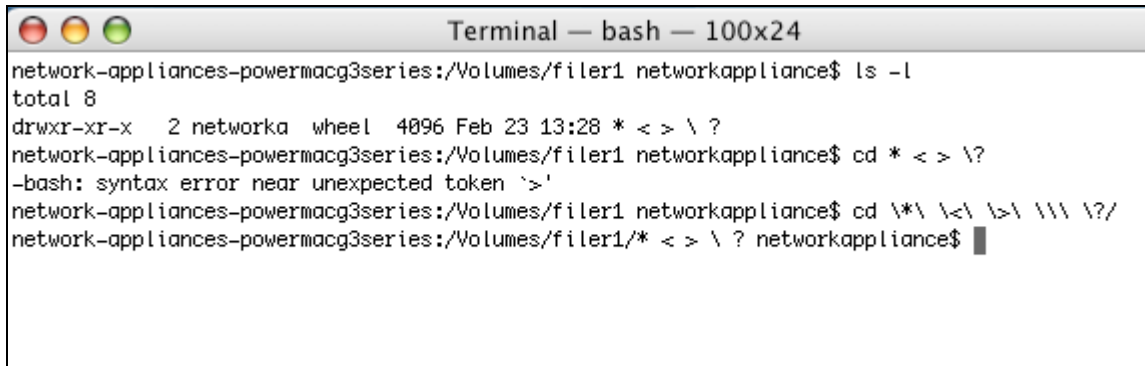

Figure 8) Folders/files can be created with special characters in the Finder…

```
network-appliances-powermacg3series:/Volumes/filer1 networkappliance$ ls -l
total 8
drwxr-xr-x   2 networka  wheel  4096 Feb 23 13:28 * < > \ ?
network-appliances-powermacg3series:/Volumes/filer1 networkappliance$ cd * < > \?
-bash: syntax error near unexpected token `>'
network-appliances-powermacg3series:/Volumes/filer1 networkappliance$ cd \*\ \<\ \>\ \\\ \?/
network-appliances-powermacg3series:/Volumes/filer1/* < > \ ? networkappliance$
```

Figure 9) …but can be difficult to access using a shell prompt.

5. Thursby Dave/ADmitMac Enhancements

5.1 Network connectivity

Thursby ADmitMac and DAVE replace the native Mac OS X SMB/CIFS client with its own (more complete) CIFS implementation (see www.thursby.com/products/dave-vs-tiger.html). Thursby DAVE is also supported on older Mac OS versions such as 8.6-9.2.x.

5.2 File structures (Data and Resource Forks)

ADmitMac and DAVE utilize the identical file structure as Microsoft SFM by storing Resource Forks in Alternate Data Streams in exactly the same way (see paragraph 3.2). The file formats are compatible and interchangeable (see www.thursby.com/products/dave-vs-tiger.html).

5.3 File naming conventions

ADmitMac and DAVE utilize the same file naming conventions as SFM, which utilizes Unicode for Windows "invalid" characters. Files with these characters can now be stored on CIFS shares. See paragraph 3.3.

6. Mac OS-Netapp Storage Connectivity

6.1 Network connectivity

NetApp storage systems do not natively support AFP and cannot be extended with an AFP add-on.

With Mac OS X, clients can connect to NetApp storage systems using the Mac OS X native NFS or SMB/CIFS protocols. Older Mac OS clients, however, neither have native SMB/CIFS nor NFS support, preventing connectivity. However, when using ADmitMac/DAVE, both Mac OS X and older Mac OS versions can connect to NetApp storage systems using the CIFS protocol.

6.2 File structures (Data and Resource Forks)

NetApp storage systems support ADS on CIFS shares, so the AppleSingle format can be used. When using DAVE/ADmitMac, data from Mac clients can be stored on NetApp storage systems without loss of Resource Forks and in an AppleSingle-like file format that is compatible with DAVE/ADmitMac and (possibly existing) SFM data files.

When connecting to NetApp storage systems through the Mac OS X native SMB/CIFS or NFS protocol, Data and Resource Forks will be split into two separate files, as described in paragraph 4.2.

6.3 File naming conventions

NetApp storage systems support Unicode. When using ADmitMac/DAVE, files containing "invalid" characters can be stored on NetApp storage systems CIFS shares in a format that is compatible with SFM file naming conventions.

This means, when using DAVE or ADmitMac, that data can be copied (using Windows Explorer) directly from an existing Windows SFM volume (residing on a local NTFS disk) to a NetApp CIFS share without loss of data and without the need to copy those files using a Mac client (avoiding excessive network traffic). This can be very useful when consolidating Windows servers and migrating data to NetApp storage systems.

In addition, DAVE (since version 6) and ADmitMac support DFS (Distributed File System), which means users can leverage global namespace and transparent failover functionality—provided by the NetApp Virtual File Manager™ (VFM™; see www.netapp.com/products/software/vfm.html)—in a Mac environment. Note that this is beyond the scope of this document.

When using the Mac OS X native SMB/CIFS client, data must be migrated from Windows servers to NetApp storage systems using a Mac as the intermediate. Also, the VFM global namespace cannot be leveraged because the OS X native CIFS/SMB implementation does not support DFS.

Furthermore, depending on cross-platform data access requirements, users can choose NFS connectivity. NetApp storage systems fully support the NFS standards, so they will behave as described in paragraph 4.3.2.

6.4 Cross-platform access

6.4.1 Mixed Windows/Mac environments When DAVE/ADmitMac is deployed, both Mac and Windows clients can access the same data on NetApp storage systems CIFS shares. Normally Windows clients do not use the additional streams when reading files. However, depending on the application an existing ADS may get lost when a Mac file is (re)written by a Windows application. This behavior is equivalent to Windows applications accessing data stored on an SFM volume, and is therefore no exception to current real-world cross-platform environments on Windows file servers.

On Windows servers and clients, "invalid" chars in Mac file names will appear mangled. However, copying these files with the appropriate tools (e.g., Windows Explorer or VFM) to NetApp storage CIFS shares is still possible, and Data, Resource Forks and the ability to reference these files from Mac clients will remain intact. This behavior is equivalent to Windows applications accessing data stored on an SFM volume, and is therefore no exception to current real-world cross-platform environments on Windows file servers.

Data that was previously stored on an SFM (AFP) volume on a Windows "local" NTFS disk can be accessed using DAVE/ADmitMac over CIFS, without the need for migration. So any legacy data on a Windows server can be accessed through DAVE/ADmitMac on the Mac client and can then be moved to NetApp storage systems when ready. This allows for a more seamless transition from AFP on Windows—to CIFS on Windows—to CIFS on NetApp storage systems. VFM also supports the migration of files, including ADS, so this can be used to provide user transparent data migration.

6.4.2 Mixed UNIX/Mac OS X environments If there is a mixed UNIX/Mac OS X environment, it may make most sense to deploy NFS for NetApp storage connectivity as both UNIX and Mac OS X clients natively support NFS. Doing this requires only an NFS license at the NetApp storage side.

The only thing to consider, then, is the fact that UNIX clients will see two files for each file created or accessed by a Mac OS X client, as these files will be broken up into separate Data and Resource Forks (see 4.2). Also, certain file naming behavior must be taken into account when accessing files created on Mac OS X clients that are accessed by UNIX clients over NFS (see 4.3.2).

6.5 NetApp Snapshot technology with Mac OS clients

One of the best known value-add features of NetApp storage systems is the ability to use zero performance impacting Snapshot copies for backup and online data recovery (which is covered extensively at www.netapp.com/library/tr/3001.pdf and www.netapp.com/library/tr/3002.pdf).

Users can directly access point-in-time "representations" of the file system for easy file recovery. Depending on the protocol, access to Snapshot copies on NetApp storage systems is provided in a certain way.

6.5.1 CIFS access to Snapshot copies When using CIFS file sharing a (hidden) folder to Snapshot copies can be provided (this is configurable) to the user. This folder, with the name ~snapshot, will be shown at the root of each share for which it is configured. For Windows clients this looks like:
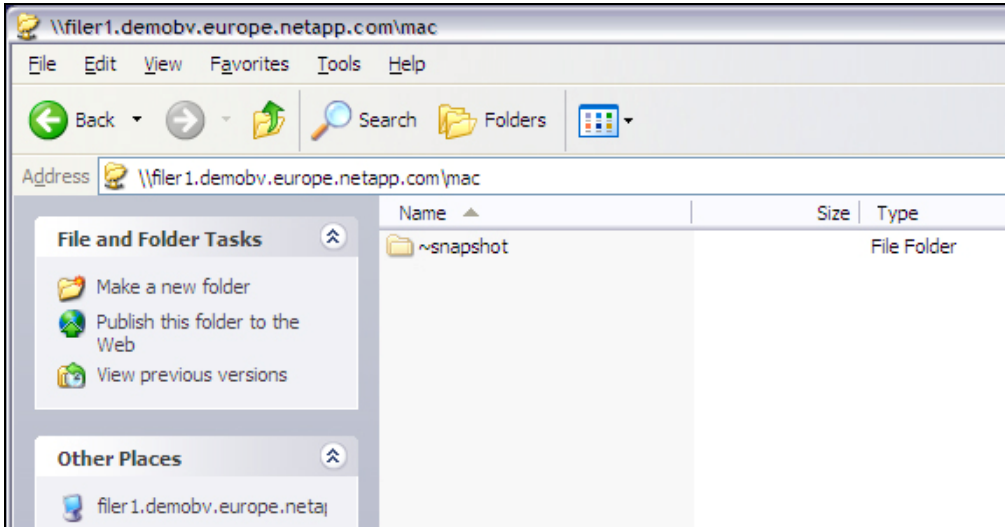

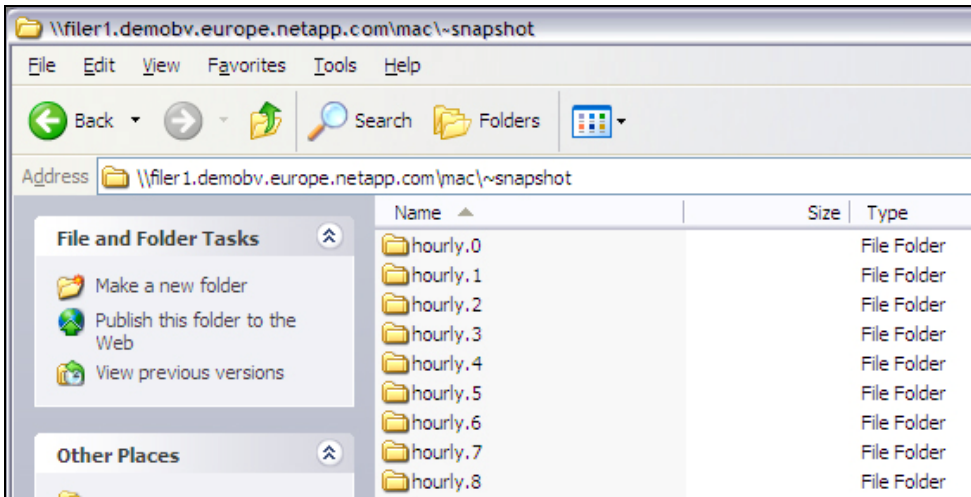**Figure 10) Access to ~snapshot folder from Windows CIFS client….**

Figure 11) …provides access to Snapshot copies created with a predefined schedule.

Access to those Snapshot copy folders from a Mac connecting to a NetApp storage system through (native or third-party) CIFS is exactly the same.
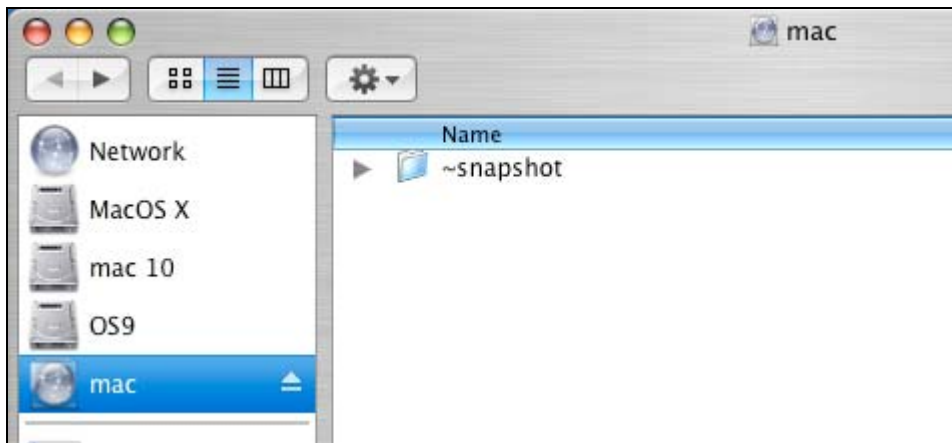

Figure 12) Access on a Mac OS X client to a ~snapshot folder on a NetApp CIFS volume...
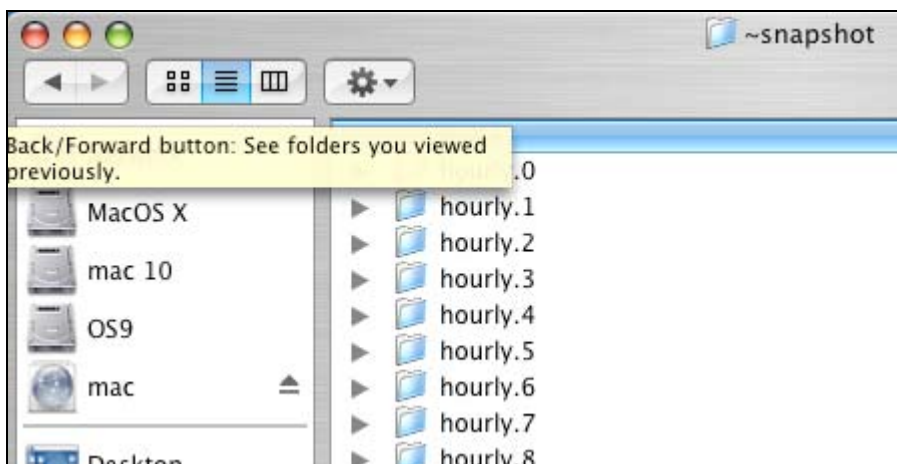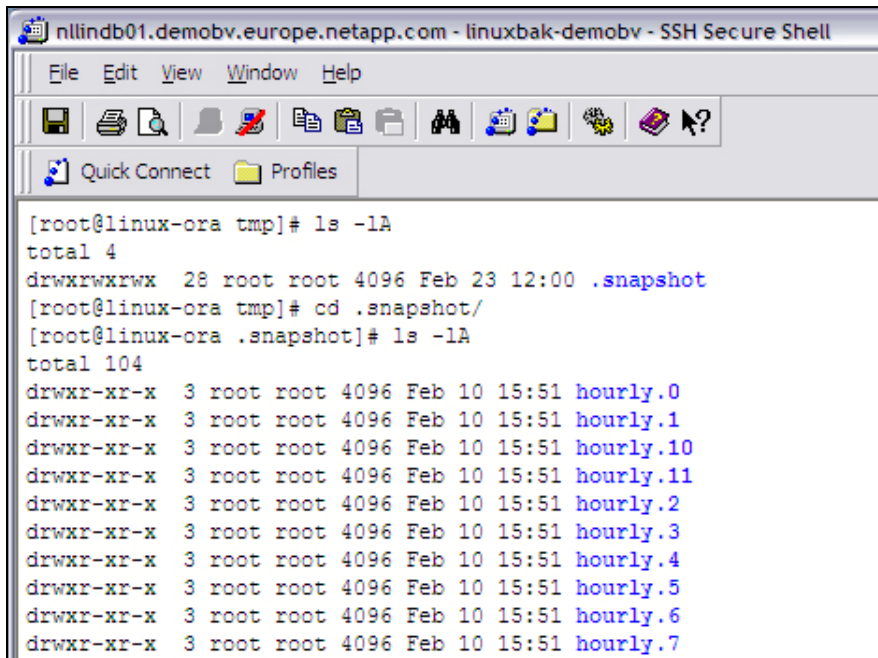

Figure 13) …provides access to the Snapshot copies in the same way as Windows CIFS clients.

6.5.2 NFS access to Snapshot copies When using NFS file sharing a (hidden) folder to Snapshot copies can (this is also configurable) be provided to the user. This folder, with the name .snapshot, will be shown at the top level of a mounted NFS file system.

An example for a Linux® client looks like this:

Figure 14) Access to a .snapshot folder on an NFS-mounted file system.

However, as discussed in paragraph 4.3.2, while most characters can be used in file names when storing on NFS exports, there are some restrictions; files starting with a "." is the most important. These file names are not allowed in Mac OS X as they are considered hidden files/directories. A major drawback of this fact is that the .snapshot folder will not be visible in the Mac OS X Finder:

Figure 15) The .snapshot folder is not shown in the Finder on NFS-mounted volumes…

```
Terminal — bash — 80x24

network-appliances-powermacg3series:/Volumes/filer1 networkappliance$ ls -Al
total 8
drwxrwxrwx   28 root   wheel   4096 Feb 23 12:00 .snapshot
network-appliances-powermacg3series:/Volumes/filer1 networkappliance$
```

Figure 16) …but is shown on the command line in the Mac OS Terminal.

This means restoring data from NetApp Snapshot copies on NFS-mounted volumes on Mac clients will have to be performed differently. One method is to use the shell command line "cp" program and manually copy files from the .snapshot folder structure back to the active file system. This might become a cumbersome operation when a large number of files is involved.

Note: In order to perform a correct file restore from the .snapshot folder using the "cp" program both Data and Resource Fork files (filename and ._filename) must be restored. This can become complicated when multiple files need to be restored.

However, data can be restored using the Finder by using the "Go to Folder" menu option:

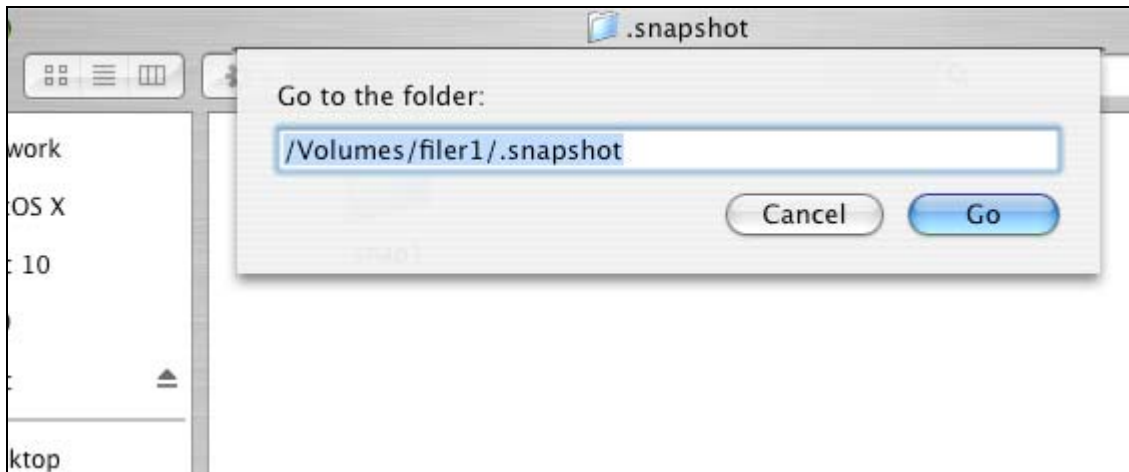**Figure 17) To restore from a Snapshot copy using Finder on NFS mounts, use "Go to Folder"…**

15

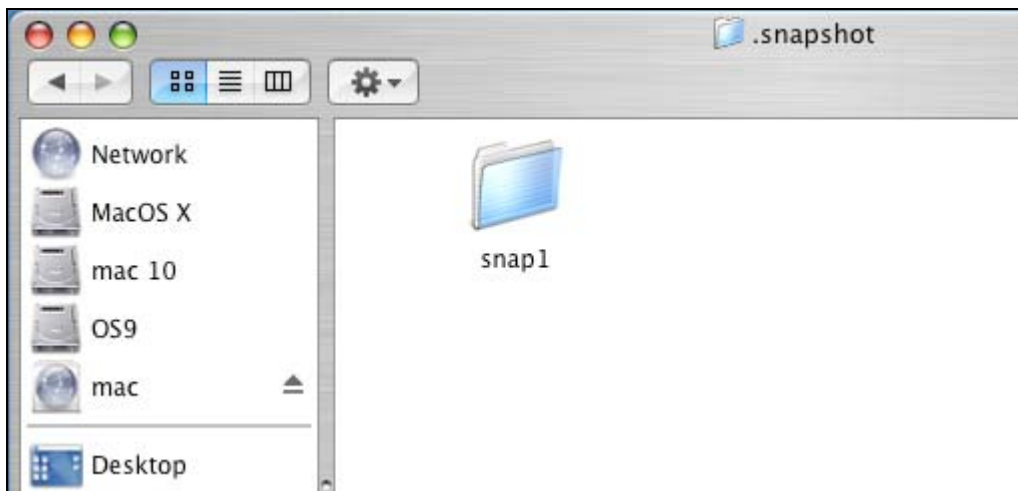Figure 18) …and enter the full .snapshot path for the mounted volume…

Figure 19) …and then Snapshot copies can be accessed and data restored as normal.

7. Mac OS X File Sharing Performance

7.1 Background on Performance Testing

Because of the previously mentioned capabilities, many sites have become interested in combining the file storage and serving capabilities of NetApp storage systems with the multimedia power of Apple systems running Mac OS X. To ensure proper performance of Mac OS X with NetApp storage systems, Network Appliance in the past has carried out a comprehensive set of performance tests in conjunction with Ball State University's Center for Media Design—a state-of-the-art facility dedicated to digital media research, development, and education. The capabilities of native NFS and CIFS were tested along with CIFS performance using DAVE.

According to Philip Repp, associate vice president of information technology at Ball State University, "The Center for Media Design (CMD) is part of Ball State's iCommunication Initiative, funded by the Lilly Endowment, Inc. CMD's charter is to test and expand the real-world applications of digital media content. To

achieve this goal we are creating collaborative working environments that facilitate cross-platform sharing and simultaneous use of multiple applications. The testing that CMD and the Advanced Graphics group have performed with OS X in collaboration with Network Appliance is an excellent step in this direction, proving that we can harness the multimedia capabilities of our Macintosh systems in heterogeneous environments. A student working in one facility on a PC can now go to the Apple lab and have immediate access to the same files. This will allow us to move from initial planning into the deployment of real-world applications."

This chapter describes the results of this testing and offers some guidelines for optimizing operations using the various protocols.

7.2 Configuring OS X for Optimal File Sharing Performance

The testing performed at Ball State provides a sound basis for configuration decisions when deploying OS X systems with NetApp filers in multiprotocol environments. The overall goal of the testing process was to assess any protocol limitations of Ball State's Power Mac G4 systems running OS X, to help Ball State determine the best protocol choice for its needs.

Because of Ball State's requirement to test with the very large file sizes that result from advanced digital media, some significant limitations immediately became apparent. Using NFS, OS X cannot read or write files larger than 2GB. With native CIFS, OS X can read large files but cannot write files larger than 4GB[1] These limitations may not be significant in all environments, but it is important to keep them in mind when planning an OS X installation. It is hoped that these limitations will be eliminated in future OS X releases. Clients running the DAVE software experienced no similar file size limitations.

For environments such as Ball State where performance is paramount or where the Mac OS X file size limitations (2GB for NFS, 4GB for CIFS) are an issue, DAVE is the preferred choice. Detailed performance results are presented in a later section and can serve as a baseline for deciding which protocol is most appropriate for a given application. Naturally, the information can only be used as a guideline since performance will vary depending on the specifics of a particular environment, including filer configuration, network load, clients, applications, and usage patterns.

For environments where Mac clients are configured with 10Base-T or 100Base-T network connections and file sizes are less than 2GB, the choice of protocol will depend more on the file sharing needs of other clients in the environment since any of the protocols can perform above or near the performance limits of a single 100Base-T connection (100Mb/sec or approximately 12MB/sec maximum).

There are a few additional considerations to bear in mind to achieve optimal performance when deploying Mac OS X systems with NetApp storage systems:.

- Use the latest version of Data ONTAP™ and Mac OS X. The versions used for the testing are shown in table 1. These versions or higher should be used to achieve comparable or better performance. (Earlier versions of OS X had substantially lower performance.)

- At the time of testing the protocols in OS X were relatively new, and Apple has been making frequent changes. Make sure the latest Apple fixes are installed. (Updates can be installed automatically by turning on Updater from the Finder menu, or see www.apple.com/support.) Network Appliance has been actively working with Apple and others to resolve the file size limitations in native NFS and CIFS.

- If you are using Gigabit Ethernet (GbE), keep in mind that Apple Power Mac G4 systems only support Cat 5 copper connections (at the time of testing).

- Always perform mount operations using the OS X GUI. Based on testing, it appears that mounts performed from a Terminal session result in significantly lower performance. (An almost 30% reduction was observed.)

[1] Network Appliance reported the CIFS and NFS file size limitations to Apple, and Apple has eliminated the limitations in Mac OS X 10.2.4. Network Appliance has verified that this release fixes the problems, but has not repeated full performance testing after initial publication of the results in related Technical Reports.

7.2.1 NFS Configuration Guidelines

- Native NFS on older versions of Mac OS X could not be used to read or write a file larger than 1.99GB. (Mac OS X crashes on attempts to access beyond 1.99GB.) Whenever file sizes are expected that exceed this size, make sure correct versions of Mac OS X are deployed.

- All tests were performed using NFS version 3, 32K block sizes, and TCP (the Mac OS X defaults). Varying these parameters resulted in no significant increase in performance.

- Instructions for mounting an NFS filesystem with OS X can be found at:
  http://docs.info.apple.com/article.html?artnum=25344

7.2.2 CIFS Configuration Guidelines

- Native CIFS on older versions of Mac OS X could not write a file larger than 4GB. Whenever file sizes are expected that exceed this size, make sure correct versions of Mac OS X are deployed.

- Instructions for mounting a CIFS filesystem with OS X can be found at:
  http://docs.info.apple.com/article.html?artnum=106471

7.2.3 DAVE/ADmitMac Configuration Guidelines

- The version of DAVE used in testing was 4.0.1 (at the time of testing). This version or later (at the time of writing v6.1) or ADmitMac (at the time of writing v3.1) should be used.

- Neither DAVE nor ADmitMac have limitations with regard to file size.

- In addition to CIFS, DAVE and ADmitMac offer other features to facilitate interaction with Windows, such as various flavors of Windows authentication.

- More information on DAVE and ADmitMac can be found at:
  http://www.thursby.com

- More information on configuring DAVE and ADmitMac can be found at:
  http://download.thursby.com/DAVE_Quick_Start_Guide.pdf
  http://download.thursby.com/admitmac-configuration.pdf
  http://download.thursby.com/administration-guide.pdf

7.3 Assessing OS X Network File Performance

The goal of performance testing at Ball State was to determine which protocol (NFS, native CIFS, or DAVE) was most appropriate to integrate with existing systems while delivering the required performance for a very demanding mix of applications. This section summarizes the results of over 1,100 individual test runs. Enough information is provided to duplicate the tests that were performed if desired.

7.3.1 File Sizes Because of Ball State's intense focus on digital video and other advanced digital media, all tests in this study were performed with files ranging in size from 500MB to 10GB. A significant advantage of this is that the files were too large to be conveniently cached in memory either on the NetApp storage system or the Apple clients. While caching is an important part of network file sharing, tests that do not take steps to eliminate the effects of caching can yield artificially high numbers that may not be a good measure of performance for real-world applications.

7.3.2 NetApp storage and PowerMac Configuration All performance tests were carried out using 10 identical Power Mac G4 systems connected to a NetApp FAS960 filer. Configuration information for these systems is provided in Table 1.

	NETAPP FAS960	APPLE POWER MAC G4
Operating System	Data ONTAP 6.3.1	Mac OS X 10.2.1
Processor Configuration	2.2 GHz Pentium 4	Dual 1 GHz PowerPC G4
Main Memory	6GB	1.5GB
Non-volatile Memory	256MB	NA
Network	4 GbE (copper)	1 GbE (copper)
Number of Disks	28	2
Type of Disks	72GB	80GB
RAID Configuration	RAID4	NA
Additional Software	None	DAVE version 4.0.1

The FAS960 used the default (traditional volume) configuration for its 28 drives. A volume consisting of two drives was used for root. Three RAID groups of 8 drives (1 parity plus 7 data drives) utilized an additional 24 drives, while 2 drives were reserved as hot spares. Various other drive configurations were tested, but none yielded a significant improvement in performance over the default configuration[2]. During testing with 10 Mac clients, disk utilization on the FAS960 was about 65%, indicating that additional performance is possible under different test conditions.

The Power Mac systems were also used in their default configuration. During the course of testing, both Power Mac CPUs and local memory were observed to be running at 100% utilization, indicating that the clients were being pushed to their limits.

To eliminate the possible effects of disk fragmentation on Power Mac performance, the operating system was erased and reinstalled for each protocol tested to ensure that each series of tests had the same starting point. Since DAVE was not installed until it was tested, this ensured that native CIFS and NFS did not benefit from any DAVE-enhanced software modules that the protocols may share.

[2] With the introduction of FlexVol™ volumes in Data ONTAP 7G significant performance gains can be expected when deploying flexible volumes instead of the tradition volumes used in this configuration. FlexVol volumes provide higher performance by leveraging increased disk spindles by grouping them into larger aggregates. See also http://www.netapp.com/library/tr/3356.pdf.

7.3.3 Network Configuration The network was designed to eliminate network bottlenecks and ensure that the Mac OS X systems could be pushed to their maximum possible performance for each protocol under the conditions tested. All four GbE interfaces on the FAS960 were trunked together using the filer's built-in virtual interface (VIF) capability, providing automatic load balancing across all interfaces for optimal

performance. All systems were connected to a single Cisco Catalyst 6506 switch. The switch was also configured to load balance across the four GbE connections to the filer. During testing, backplane load on the switch never exceeded 15%. The testing configuration is illustrated in the figure below.
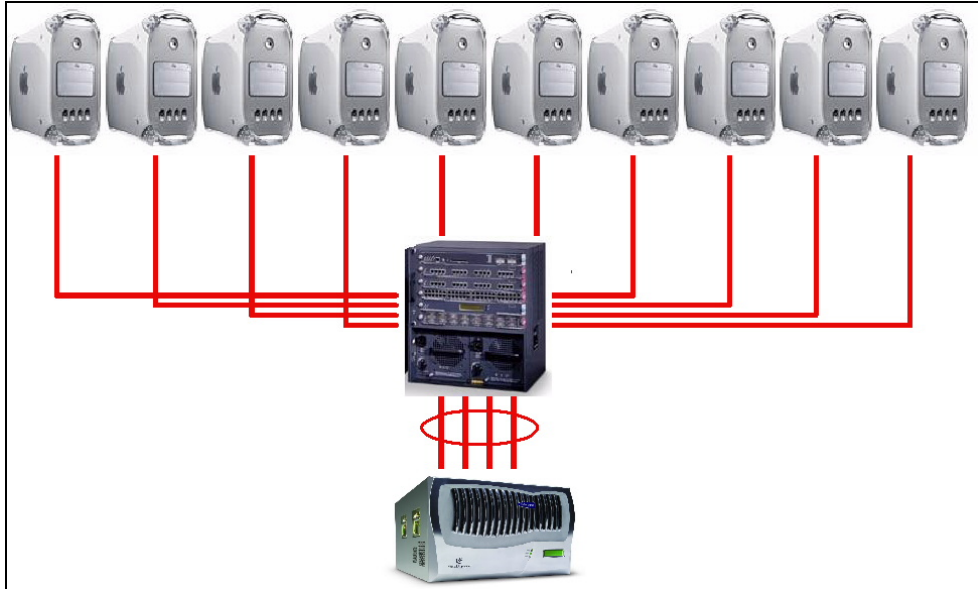

Figure 20) Network configuration for performance testing.

7.3.4 Test Procedures All tests were performed using 1, 5, or 10 clients. Single client tests allow the maximum performance of a single system to be determined.

Read and write performance were measured by manually copying files of the prescribed size to and from the filer. Each client had its own directory on the NetApp FAS960 storage system and its own set of files to help mitigate the effects of data cached in memory. (If all clients were sharing the same set of files, the FAS960 might be able to satisfy multiple read requests after a single read from disk, skewing the final results.)

Each test was performed a minimum of three times, and the results were averaged. Initially, tests were performed from an Mac OS X Terminal session so that the operating system could be used to time each test to facilitate throughput calculation. However, this was discovered to be significantly slower than copies performed using Finder, so ultimately all tests were performed manually using drag-and-drop[3]. This manual process required considerable assistance from students at Ball State to ensure that tests were started simultaneously. This is consistent with Ball State's goal of simulating 10 students entering a lab and uploading their project folders to the local desktop.

[3] Because drag-and-drop was used to achieve optimal performance, all timing had to be performed with a precision chronograph. Tests were timed to the nearest 100th of a second and averaged. Final results were rounded to the nearest 10th of a second to help correct for the inaccuracies introduced by manual processes.

7.3.5 Read and Write Test Results Read tests were conducted using file sizes of 500MB, 1GB, 5GB, and 10GB. (Because of the file size limitation discovered with NFS, file sizes for those tests were limited to 500MB, 1GB, and 1.99 GB.) Files of the appropriate size were manually copied from the filer to disk storage on each client system. Results of these read tests are summarized in the figure below.
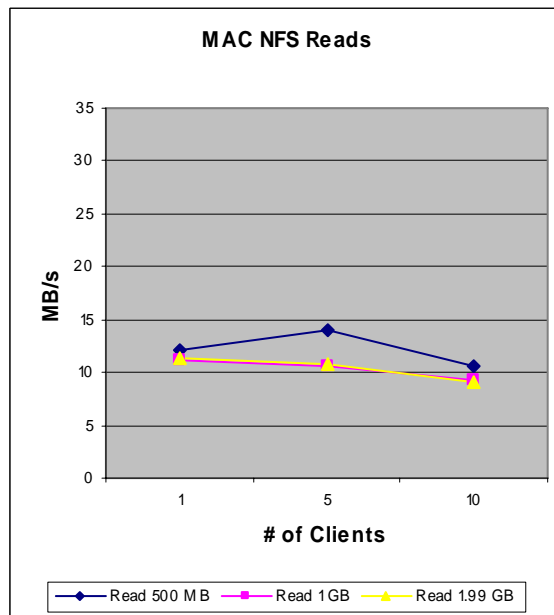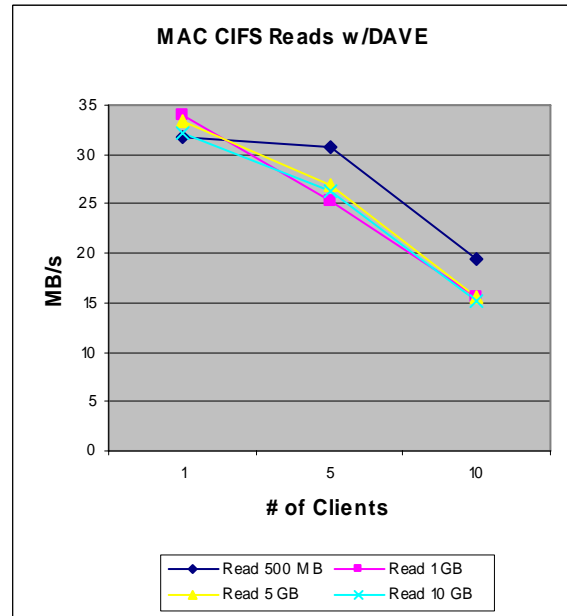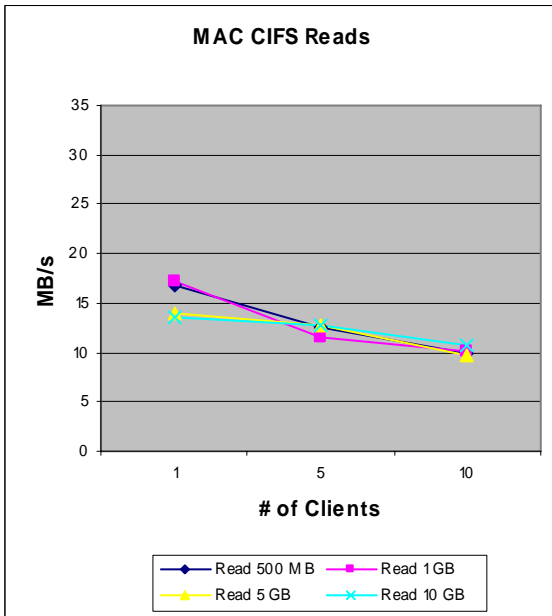
Figure 21) Results of read tests for each protocol with varied file size using 1, 5, or 10 clients.

As the figure demonstrates, maximum single-client and multi-client read performance are achieved using DAVE software. A single DAVE client is capable of reading data at 31 to 34MB/sec depending on file size, while the native CIFS implementation is capable of 13 to 17MB/sec, and NFS achieves 11 to 12MB/sec.

Write tests were conducted using the same file sizes as for reads. (Because the native CIFS implementation cannot write files larger than 4GB, file sizes of 500MB, 1GB, and 4GB were used for that testing.)
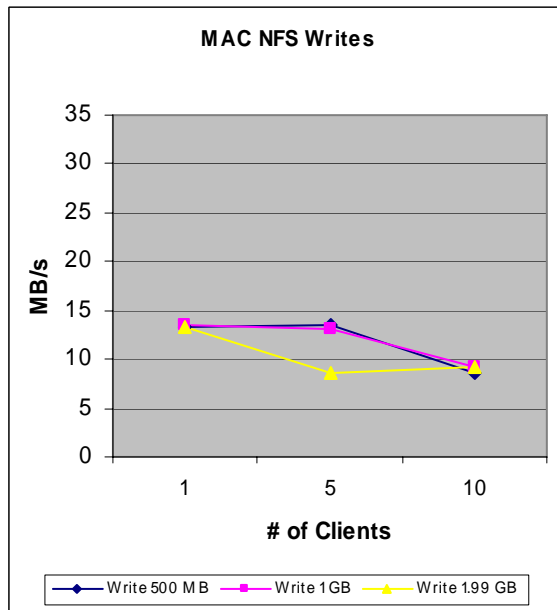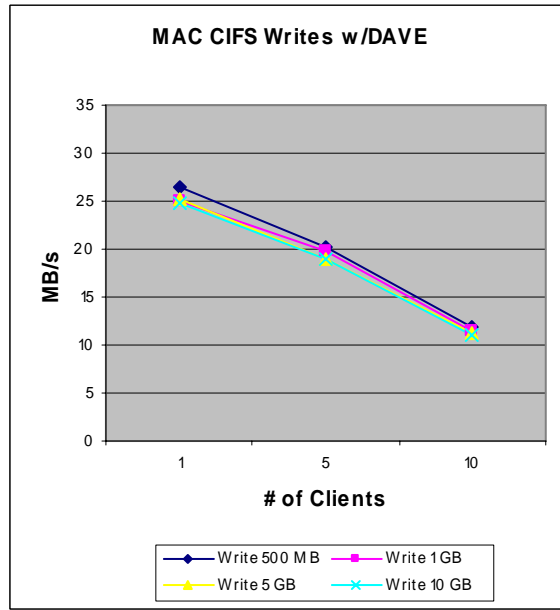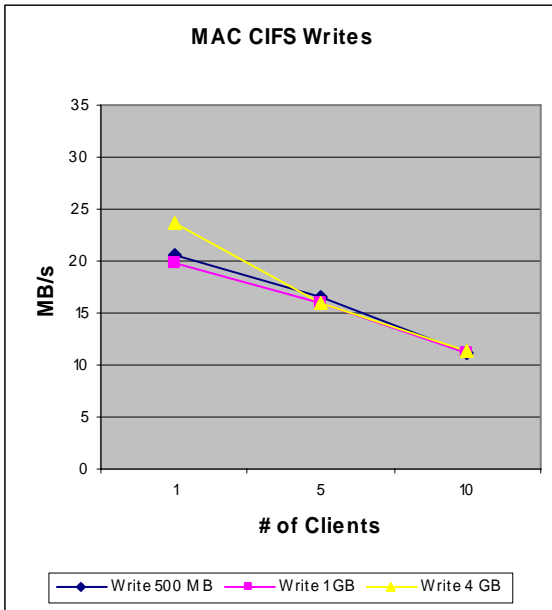
Figure 22) Results of write tests for each protocol with varied file size using 1, 5, or 10 clients.

As with read performance, DAVE software demonstrated the best write performance. Single client performance ranged from 25 to 26.5MB/sec for DAVE, 20 to 24MB/sec for native CIFS, while NFS achieved only 13.4MB/sec.

Conclusion

The inclusion of both the NFS and CIFS network file sharing protocols in Mac OS X dramatically simplifies the integration of Apple systems into multiprotocol file sharing environments that include Windows, UNIX, and other systems.

To assess the network file sharing functionality and performance of Mac OS X, Network Appliance undertook integration testing of the Mac OS X native NFS, native CIFS, and DAVE and ADmitMac, an alternative third-party CIFS implementation suite from Thursby Software Systems (www.thursby.com). Although older native Mac OS X NFS and CIFS were found to have file size, file naming, and cross accessibility limitations that might be significant in some environments, they are adequate for many common applications.

However, under the conditions tested, DAVE and ADmitMac were found to offer better compatibility in a mixed Windows/Mac OS environment and may be the preferred file sharing choice for the most demanding environments. DAVE and ADmitMac enable customers to more easily leverage NetApp specific value adds such as Snapshot copies, global namespace (DFS), and simplified failover and migration support through VFM and use of legacy data without the need for complex data conversions.

Testing of the native CIFS and NFS implementations found in Apple Mac OS X demonstrates that both are capable of performing adequately under demanding conditions (subject to the limitations discussed above), thereby making it possible to use these protocols to integrate systems running Mac OS X into heterogeneous multiprotocol file sharing environments.

For situations where performance is paramount, DAVE and ADmitMac are the preferred choices. Ball State University ultimately chose to implement DAVE over the native protocols for these reasons.