Using Mac OS X Clients with Isilon OneFS 7.x

Resources for Integration and Configuration

September 23, 2014

EMC²

First published in the USA. December 18, 2013

EMC²

About this Guide

Intended Audience

This guide describes scenarios specific to using OS X clients with an Isilon cluster. It is written for Isilon administrators, OS X administrators, and OS X users including creative professionals, video production professionals, and the general user.

This guide is for users of OneFS 7.x. For clusters running OneFS 6.5.x, please see Using Mac OS X Clients with OneFS 6.5 on the EMC support site.

Assumptions

This guide assumes that the reader has:

1. A general understanding of the OneFS operating system and how to configure it.
2. An understanding of the OS X operating system in general, and the Finder, and OS X's resource forks, which are used for storing structured data, in particular.
3. Knowledge of how to use the Finder to access network storage.
4. The ability to use the OS X Terminal application to run commands from the command line.
5. The ability to use the OS X Console application to read log files.

OneFS Support for OS X

OneFS supports OS X 10.4 and later. Clients running OS X can connect to an Isilon cluster using the NFS or SMB protocol. OS X 10.4 over SMB is not compatible with OneFS 6.5 or later.

For more information about OneFS and OS X compatibility, see the Isilon Supportability and Compatibility Guide.

Configuration Checklist

Use the following checklist as you perform the steps to configure your cluster and OS X clients.

Note: To fully understand the reasoning behind and implications of implementing each step of this checklist, review the appropriate section referenced in parentheses.

- ☐ Cluster is configured for low-latency, and the file system data layout is configured appropriately (see "Performance")
- ☐ Set the TCP Send and Receive buffer optimizations (see "Send and Receive Windows")
- ☐ Set TCP delayed acknowledgement to compatibility mode (see "TCP Delayed Acknowledgment")
- ☐ Disable TCP inflight calculation on the cluster (see "TCP Inflight Bandwidth Calculation")
- ☐ Tune TCP slow start parameters (see "TCP Slow Start")
- ☐ Configure on-disk identity mapping based on your environment requirements (see "On-Disk User and Group Identity")
- ☐ Configure OS X clients to use List or Icon view in the Finder (see "Metadata Retrieval Speed")
- ☐ Select which protocol is best for your environment: SMB or NFS (see "Workflow and Protocol Considerations")

  If you chose SMB...

  - ☐ Enable Alternate Data Streams (see "Alternate Data Streams")
  - ☐ Configure shares to use an inheritable ACL (see "Suggested SMB Share and File System Permissions Configuration")
  - ☐ Disable SMB packet signing (see "SMB Packet Signing")

  If you chose NFS...

  - ☐ Configure the desired client mount options (see "Cluster Configuration for NFS")
  - ☐ Specify the number of items that OneFS prefetches for folder listings (see "READDIRPLUS")
  - ☐ Configure exports appropriately for your environment (see "Suggested NFS Export and File System Permissions Configurations")
  - ☐ Set the correct permissions on exports, using inheritable ACLs when appropriate for your environment (see "Suggested NFS Export and File System Permissions Configurations")

1. Working with OS X and Network Storage

Protocols

OS X clients communicate with Isilon nodes using the SMB protocol or the NFS protocol (OS X 10.4 must use the NFS protocol). Both protocols provide access to network storage according to each protocol's feature set.

As you read through this guide consider your environment's performance requirements when selecting a protocol to use; OS X supports Server Message Block (SMB) version 1 (also known as the Common Internet File System or CIFS), SMB version 2, and Network File System (NFS) versions 2, 3, and 4 protocols. The following table lists SMB and NFS protocol support by OS X version.

	SMB/CIFS	NFS
10.5 (Leopard)	Version 1	Version 2, 3, 4 * †
10.6 (Snow Leopard)	Version 1	Version 2, 3, 4 * †
10.7 (Lion)	Version 1	Version 2, 3, 4 †
10.8 (Mountain Lion)	Version 1	Version 2, 3, 4 †
10.9 (Mavericks)	Version 1, 2.0, 2.1	Version 2, 3, 4 †

* The NFS version 4 client in these releases was considered "Alpha" quality by Apple, and is documented as such in the mount_nfs man page.
† Isilon does not recommend the use of NFS version 4 at this time because of known issues with the OS X ID mapper component. Instead, use NFS version 3. This document will be updated when NFS version 4 is recommended.

Note: Isilon OneFS uses open network protocols and does not support OS X clients connecting to an Isilon cluster via Apple File Protocol (AFP), a proprietary protocol developed by Apple.

The following table provides a brief comparison of the SMB/CIFS and NFS Protocols.

	SMB/CIFS	NFS
Performance	Not as fast as NFS	Faster performance than SMB
Authentication	Credentials-based (username & password, Kerberos) authentication	UID and GID, or Kerberos, Authentication
Resource Forks	Stores resource fork in an alternate data stream, or in an AppleDouble dotbar file	Stores resource fork in an AppleDouble dotbar file
Active Directory	Fully supported	Fully supported

	SMB/CIFS	NFS
Open Directory	Not supported	Fully supported

Note: The terms "SMB" and "CIFS" can be used interchangeably. For the remainder of this guide, only "SMB" will be used.

SMB environments can take advantage of authentication features in the protocol not directly supported by NFS, but SMB 1 is not considered a high-performance protocol. SMB 1 supports only 64KB read sizes and requires many individual network calls to enumerate items in a folder. Because of these limitations, OS X clients connected to gigabit Ethernet should expect at most FireWire 400-level speeds on sustained transfers when using SMB 1. It is possible to get slightly better performance than that, but unlikely given the limitations of the protocol.

SMB 2 support was introduced in OS X 10.9 (Mavericks). OS X 10.9 supports SMB 2.0 and 2.1. While OS X 10.9 clients can use SMB 2.0 with OneFS 6.5, performance will be about the same as SMB 1.0. For maximum performance, OS X clients should connect to Isilon OneFS 7.0.1 or later.

Resource Forks

OS X clients use an OS X-specific data type called a resource fork to store metadata for each file. A resource fork can include information such as the application that created the file and the type of the file (creator and type codes), custom icon information, and any Finder color labels or tags that might have been applied to the file. More information about resource forks can be found from Wikipedia and other sources on the Web.

Data stored in resource forks can be stored on file servers and other systems that do not natively support them. One way is to store the file data in a normal file (for example, photo.jpg), and then store the resource fork data in a separate file (for example, ._photo.jpg). This way of storing resource fork data is referred to as AppleDouble and the files whose filename begins with "._" are referred to as "dotbar" files.

AppleDouble is the only way OS X clients can store resource fork data on NFS exports. For SMB shares, OneFS provides support for alternate data streams, as described in the Alternate Data Streams section below. In addition, for SMB shares, OS X can support AppleDouble, which allows resource forks to be shared between Mac SMB and NFS clients. For more information about how to control the use of AppleDouble, see Apple KB HT4017.

OS X does not allow you to disable the creation of AppleDouble files because this would render resource forks unavailable for OS X clients. That could leave users with no known way to open files, and no identifying metadata about the file type. Some older Mac applications, such as QuarkXpress, create files without a three-letter file extension and rely on resource fork data to identify the type of file and how to open it. Without this metadata, the file shows up in the Finder as a blank icon, with no apparent use.

Shared Protocol Considerations

For metadata, NFS and SMB OS X clients should use AppleDouble. This gives OS X clients a consistent view of all resource fork data because they are configured to look for and use dotbar files on the file server. To force all OS X SMB clients to use AppleDouble, see the Apple Knowledge Base article HT4017 and refer to the instructions for disabling named streams on the OS X client by editing the */etc/nsmb.conf* file.

Metadata Retrieval Speed

The OS X Finder requires consistent and complete views of metadata before displaying files. For folders with hundreds or thousands of objects, an OS X client takes much longer than a comparable Windows computer to display a folder over SMB, or a *NIX computer over NFS. This is because the Finder typically does not display objects until it has retrieved all metadata for all files in the folder. This leaves a user looking at an empty Finder window with a spinning progress indicator until all objects in the folder have been retrieved.

Slow Finder listings can be exaggerated by high network latency plus the filesharing protocol being used, and can make the difference between a Finder experience that seems quick and usable and one that seems slow. For example, when you load a folder that contains 5,000 objects in the Finder, the OS X must first obtain a list of file names, and then must retrieve metadata for each file in the folder. If it takes 2 ms for a packet to travel roundtrip from your client to the Isilon cluster, you can experience a delay of 30 seconds or more before the Finder displays a full list of objects. For environments that routinely deal with folders that contain thousands of files, SMB is not usually recommended because SMB requires many network calls to gather all data for Finder listings.

OS X Mavericks (10.9) includes SMB 2.1 support; however, its speed in retrieving metadata is similar to older SMB 1-based OS X clients (10.8 and earlier).

Network latency is only one consideration. Resource fork data in OneFS is stored as a separate file linked to its parent file, when using alternate data streams, or as a separate dotbar file when using AppleDouble. Disk-seek-latencies with the drives used in Isilon clusters can degrade Finder responsiveness on file listings. In the 5,000 objects scenario, if every file has a resource fork, retrieval of file data and resource fork data could double the number of files that have to be retrieved from OneFS. Network latency plus disk-seek-latency can increase the time it takes for the Finder to generate a complete list of a folder's contents.

In some situations, Solid State Disk (SSD)-based storage can provide a significant benefit. Given the low-latency metadata retrieval capabilities of Isilon's SSD-based nodes, the latency for metadata retrieval is reduced when OS X clients connect to these types of nodes. In informal testing, Isilon has demonstrated that Finder lists folder contents of large folders 5 - 7 times faster when accessing SSD nodes in the cluster, particularly for NFS-based workflows. Note, however, that resource fork data stored in alternate data streams and AppleDouble files are actually stored as regular files in OneFS. SSDs will accelerate the retrieval of metadata associated with those files, such as inodes, and permissions, but not the data contained in them. The data contained in those alternate data streams and AppleDouble files is stored on spinning disks.

To improve slow folder listings in Finder, Isilon recommends using List or Icon view rather than Column view. The reason for this is that Finder caches a folder's content only for a certain amount

of time before it must re-retrieve object names and metadata for that folder. With Column View, the contents of multiple folders might be displayed on-screen at the same time. The last folder displayed in the hierarchy (that is, the right-most folder in the window) will not fully display until OS X has finished retrieving or refreshing any folder contents for intermediate folders displayed on screen. Retrieving contents for multiple folders in a hierarchy can take some time to complete, depending on network and storage latency and can lead to a less than optimal user experience.

Another Finder configuration we recommend is disabling the view option called “Calculate All Sizes”. In order to calculate all sizes for folders within an NFS export or SMB share, the Finder has to perform a full treewalk through the folder structure. This is time consuming and creates both significant amounts of network traffic and additional load on the Isilon cluster. Users can still calculate sizes for individual folders by using the Get Info option for any item within a viewed folder. Files viewed within a folder will display their size, bundled folders, like an application bundle, will not.

For additional metadata considerations specific to accessing an Isilon cluster using an OS X client connected via the NFS protocol, see the "READDIRPLUS" and "File Locking" sections below.

Spotlight Indexing

Since version 10.4, OS X includes a desktop search feature called Spotlight that indexes various types of content across the entire client machine. Spotlight indexing is not supported on Isilon-provided SMB shares or NFS exports. Spotlight indexing on SMB shares requires a server-side component that would generate an index for the clients to consult over the network, and this is not supported. Spotlight indexing for NFS exports is unsupported by both the OS X client and Isilon server.

Using the Spotlight functionality in OS X with NFS or SMB is limited to simple name-based searches, and performance may vary depending on the size and depth of your NFS export or SMB share. As documented previously, the speed of name-based searches is dependent on how quickly the OS X client can retrieve filenames and walk a folder hierarchy. A cluster with metadata on SSD will help, as will a low latency network.

2. Migrating Workflows and Data to Isilon

This section discusses migration concerns, including workflow, protocol selection, maintaining user identity, and then migrating data to Isilon from other storage types.

Workflow and Protocol Considerations

You should be able to answer the following questions before choosing how OS X clients will access the Isilon cluster:

- Is your network an all-OS X environment, or a mixed OS X and Windows environment?
    - If the client types are mixed, do the Windows clients need to access the same data as the OS X clients?

- What are your throughput and metadata performance requirements?
    - Are the clients going to be using applications that need high bandwidth and lower latency, such as real-time video editing applications?
- Does your organization have strict permissions or authentication requirements?
- Will you need to authenticate with a username and password to access data, or will relatively open access be needed?
- Will users work with wide folders that include thousands of objects?

The following environment and workflow examples might help you answer these questions and choose an appropriate protocol for your OS X clients' workflows.

A/V, Post-production

In audio, video, and post-production environments, workflow is typically dictated by how quickly large media files can be accessed. This workflow requires high bandwidth and low latency to support applications such as Final Cut Pro, Avid, Adobe Premiere Pro, and other audio/video editing and post-production software.

In many environments, NFS is the protocol of choice because it supports multi-gigabit access over 10-gigabit Ethernet connections (10 GbE). Additionally, many A/V and post-production environments do not have strict requirements for permissions or authentication.

If Windows clients need shared access to files, and if the performance requirement for OS X clients is less than 50 megabytes/sec of sustained throughput, SMB may be a valid option. SMB can also be useful in multi-protocol environments for OS X clients that connect to an Isilon cluster because of how OS X clients store Apple file metadata. For more information on how OS X clients store metadata, see the "Resource Forks" section.

Environments that have workflows where thousands of files are stored in one folder will see benefits by using NFS, and fewer delays when browsing those folders through the Finder because of how NFS retrieves data from the Isilon server. This alone may be one reason to choose NFS over SMB. Bear in mind if NFS is used in a multi-protocol environment, Windows clients will see AppleDouble files throughout the file system.

For more information about how to use Final Cut Pro with an Isilon cluster, see *Best Practices for Configuring Isilon IQ for Final Cut Pro* on the EMC Online Support website.

Creative and Graphic Arts

Creative and graphics arts environments typically require a protocol that provides good performance, seamless metadata storage and retrieval, ease of file access with or without authentication, and sometimes multi-client compatibility.

OS X supports the concept of file name labeling, in which a color can be applied to a filename in a list. Users in a creative workflow may use this metadata to determine what actions to take on a file, whether to move it to the next stage in processing, or to leave it alone because another user is editing it.

SMB can be a good choice in these environments because it stores metadata in a way that integrates with Windows clients, and because it provides a mechanism for notifying clients when file metadata changes. This occurs without user intervention through the SMB protocol's "change notification" feature.

For creative environments that use both Mac and Windows computers, SMB is often the logical choice for Mac-centric or mixed-platform workflows because standardizing on one protocol makes administering that environment more efficient and manageable. SMB supports stricter share permissions than accessing shares using the NFS protocol. Its storage of metadata can also function more seamlessly with Windows clients via the use of alternate data streams.

Environments that have workflows where thousands of files are stored in one folder will see performance benefits by using NFS, and fewer delays when browsing files and folders in Finder because of how NFS retrieves data from the Isilon server. This alone may be one reason to choose NFS over SMB. Bear in mind if NFS is used in a multi-protocol environment, Windows clients will see AppleDouble files throughout the file system.

You should also consider asset management in this workflow. Because of the limitations in Spotlight indexing with both NFS and SMB, some environments may find it useful to use a digital asset manager to provide access to creative professionals. For an example of one supported solution, see the document *Design and Publishing Workflows with EMC Isilon Scale-Out NAS and Elvis DAM*.

General Use as a Storage Repository

For general data access, consider whether clients accessing the storage environment are OS X-based, Windows-based, a combination of OS X and Windows clients, or something else.

If the clients are primarily OS X clients, is change notification necessary, or is high performance necessary? If change notification is necessary, SMB is the correct choice. If high performance is necessary, NFS is the correct choice.

If the clients are primarily Windows clients, SMB might be the better choice. When using NFS, OS X clients sometimes create additional files for metadata storage, and these files (AppleDouble files) might confuse Windows users. SMB does not create extra metadata files, but presents a trade-off of less throughput for large file transfers.

Environments that have workflows where thousands of files are stored in one folder will see performance benefits by using NFS, and fewer delays when browsing files and folders in Finder because of how NFS retrieves data from the Isilon server. This alone may be one reason to choose NFS over SMB. Bear in mind if NFS is used in a multi-protocol environment, Windows clients will see AppleDouble files throughout the file system.

In environments where clients use different operating systems to access the same data via different protocols, the challenge becomes how to create data on the share that is readable and writable for everyone, regardless of the protocol. This can be accomplished by configuring permissions, a topic that is discussed later in this document. For more information, see the white paper called *EMC Isilon Multiprotocol Data Access with a Unified Security Model for SMB and NFS* on the EMC Online Support website.

User Identity and Authentication

OS X is a UNIX-based operating system. OS X uses a POSIX-based ownership and permissions model. Every user created on an OS X system is allocated a user ID (UID) and groups are allocated a group ID (GID). OS X supports being joined to multiple authentication repositories, but for this document, two of the more popular options are considered: Microsoft's Active Directory and Apple's Open Directory (OD). OneFS, too, can be joined to those repositories, and contains an identity management system that stores UNIX UIDs/GIDs and Windows SIDs. The challenge for administrators is how to ensure coherency between OS X and OneFS for known user and group information, that is, "user identity."

Maintaining user identity with SMB users is easy when Active Directory is used as the backend authentication store. OS X clients joined to Active Directory will attempt to authenticate using Kerberos authentication if the client making the SMB connection logs in using an Active Directory account. OS X clients that have not been joined to Active Directory will display a user and password dialog, and users can enter their Active Directory credentials. Note: SMB authentication with a backend LDAP server requires creating and importing NTLM hashes in the backend LDAP server, and this is beyond the scope of this document. SMB authentication with a backend NIS environment is not possible with the OS X.

Authentication with NFS is different than with SMB. By default, when a user logs in to the cluster, OS X sends the user's UID, primary group ID (GID), and up to 16 supplemental group IDs to the NFS server. This happens because OS X, by default, uses AUTH_SYS, or UNIX authentication, for NFS connections. OneFS NFS exports are, by default, configured to support this type of authentication. All file operations performed against the cluster are done with the UIDs and GIDs sent by the OS X client.

Mapping users on an NFS export transforms incoming user UIDs to a specific user account on the Isilon. This is useful in situations where the environment does not need to provide strict security for users. For example, if the user logging in to the cluster was UID 501, OneFS maps the user's file operations to a specific user account on the cluster. New files are owned by the mapped username that is configured on the NFS export, and the mapped username is checked against any permissions in the file system. This topic is discussed later in this document.

By default, when joining Active Directory, OS X clients automatically generate UIDs and GIDs based on a hash of the object name in Active Directory. You can confirm this on the OS X client by running the id command when logged in as an Active Directory user; an Active Directory user's UID should not be in the OS X standard 500-range of UIDs, or 1000-range for Open Directory accounts. The UIDs and GIDs generated by OS X are not present, by default, on an Active Directory domain that has Services For UNIX or RFC 2307 support enabled. The administrator can import those UIDs and GIDs into the Active Directory domain. EMC Isilon's professional services organization can assist you with (1) gathering a list of Active Directory-based UIDs and GIDs from OS X clients, (2) importing that list of UIDs and GIDs into Active Directory, and (3) creating an operational process that you can use to continue importing new UIDs and GIDs into Active Directory as users are added to the directory.

Alternatively, the OS X client can be configured to use UID and GID information in Active Directory instead of automatically generating UIDs and GIDs. This option requires additional configuration, and is covered later in this document.

With the Isilon cluster configured to use RFC2307 on its Active Directory provider, anything created by the OS X client will be created with UID/GID information that is known by the cluster. UID and GID coherency in this scheme allows for multi-protocol access from other clients that use SMB on the same set of data.

The same is true for Open Directory. OS X clients can join Open Directory domains similar to how they join Active Directory. Isilon OneFS can pull UIDs and GIDs for users out of Open Directory via OneFS's LDAP provider. Examples of how to do this are provided later in this document.

Migrating data to an Isilon Cluster

Migrating OS X users' data from an existing storage system to an Isilon cluster involves making sure that when you migrate the data you:

- Have renamed files whose names include unsupported characters
- Preserve data contents and resource fork data contents
- Preserve ownership and permissions information for migrated data

Unsupported Characters

Isilon recommends searching your datasets for file and folder names that include the following characters:

? / \ < > " | * :

These characters are allowed by SMB and NFS; however, these characters can cause cross-protocol and cross-platform (OS X and Windows) problems if used.

These characters may cause difficulties when migrating data. In some cases, the system performing the data migration might not write files or folders named with these characters to the Isilon cluster.

If files named with these characters are managed by a digital asset manager such as Adobe Drive, Elvis DAM, Aperture, iTunes, or Extensis Portfolio, they should be renamed first in those systems so that the on-disk filename reflects the change. Failure to rename the file in the digital asset manager could prevent access to the file after migration.

Preserving Data and Resource Fork Contents

Once you have selected the protocol to use for OS X clients, you can begin to migrate the data to the Isilon cluster.

For administrators who decide to do the data migration on their own, here are some best practices to follow. To migrate data from your original storage system, you should connect a high-powered OS X client with a fast network connection between the original storage system and the Isilon cluster. This Mac will be responsible for copying data from the old storage system to the Isilon cluster, maintaining file and metadata along the way. The OS X client performing the migration should mount the Isilon cluster using the protocol that you chose to use for your workflows.

You can migrate data with the rsync and robocopy tools. Additionally, the environment that you are migrating from may support other tools like isi_vol_copy or EMCopy. To make the most of the data

migration process for your environment, you should consult your account team and EMC Isilon professional services for assistance.

Rsync is a command-line tool that synchronizes the contents between different folders, and offers the ability to continue synchronizing those folders incrementally. Rsync 2.6.9 is included with OS X as of OS X 10.9.4. The version of rsync supplied by Apple supports most environments, but if your environment needs to preserve file and directory creation times, you should consider using the rsync version supplied by the MacPorts system (http://www.macports.org). Take note of certain rsync behaviors below, as well.

Use should use the following command-line options with rsync:

- -r: recurse into folders
- -l: copy symlinks as symlinks
- -p: preserve permissions
- -t: preserve modification times
- -g: preserve group
- -o: preserve owner (you must be root in order to use this option)
- -E: preserve extended attributes
- -S: handle sparse files efficiently
- -H: preserve hard links
- -x: don't cross filesystem boundaries

The rsync command should look like this:

```
sudo rsync –rlptgoXSHNx /source_directory /Volumes/<Isilon-mountpoint>
```

When using rsync with an Isilon NFS mount, you should make certain that you are mapping root users to root. Sudo is used in the above command to ensure that ownership and POSIX permissions are preserved. If you need to preserve file and directory creation time by using the MacPorts version of rsync, you will need to use a protocol other than NFSv3; NFSv3 does not support create time. Using rsync with the SMB protocol preserves only file and resource fork data; files will be owned by the user account that is used to connect to the Isilon SMB share.

Successive runs of rsync with the example command will synchronize changes in the source folder with the Isilon cluster. This will significantly reduce the time it takes to do a final sync of data before transferring production users to the Isilon cluster.

Although rsync efficiently evaluates changed data, you should divide your source dataset into folders so that you can run multiple rsync jobs together. This speeds up the synchronization process by utilizing the available CPU and network resources on the OS X client, so that multiple files are transferred simultaneously.

Preserving Ownership and Permissions for Migrated Data

Once you have selected the protocol to use for OS X clients, you should consult with your account team and Isilon professional services to ensure a successful data migration. Complicated environments may need more testing and special configurations to ensure a successful data

migration. Depending on the complexity of your environment, the following suggestions may not be appropriate.

The rsync example provided above is a simple way to ensure that file and resource fork contents are synchronized between the source to destination locations, and that the migrated data conforms to the way that the protocol functions (AppleSingle for SMB or AppleDouble for NFS). Additionally, rsync preserves POSIX ownership and permissions when used in NFS environments where rsync is run with root privileges. OS X, however, can only perform certain permissions operations over SMB and NFS. Third-party utilities or operating systems may be needed to complete this phase of the migration. The following table details the limitations of OS X by protocol:

	SMB/CIFS	NFS 3
ACLs	Can read, but does not display natively, and cannot set.	Cannot read or set.
Ownership	Displays current logged-in user as owner when viewed in Finder. Files created on Isilon are owned by the user who established the SMB connection.	Reads UID and GID as stored on Isilon. Can change user or group ownership (with appropriate privileges).

If your original storage environment used ACLs for permissions, you should use a tool to copy just the ACLs to the Isilon cluster. Robocopy can copy ACLs from the source location to the target location, preserving the data you already migrated using rsync. Robocopy requires a mount of your original server and the Isilon cluster with the SMB protocol. Robocopy requires a Windows machine to run, as Robocopy does not run natively on OS X.

For more information about using Robocopy to preserve ACLs on an existing dataset, see the Isilon knowledge base article *OneFS: How to configure an Isilon cluster to use Robocopy*.

3. Network Setup, Tuning, and Usage

Serial Access

You can connect an OS X client to the cluster over a serial connection to perform initial setup tasks and to troubleshoot hardware or network configuration issues. For more information, see the Isilon knowledge base article *Isilon: How to connect to the management port of a node*.

Performance

Data Layout, Throughput, and Latency

The overall load of the Isilon cluster can affect the performance of a client computer. The busier the cluster, the higher the latency for protocol operations: file listings might take longer to complete,

read and write throughput might be reduced, and overall user experience might be diminished. When troubleshooting a performance problem, start by ruling out any job operations, like MultiScan, SmartPools, FSAnalyze, MediaScan, and so on. All these tasks consume disk I/O and throughput, which can affect user experience. To determine the status of cluster operations, log in to the OneFS web administration interface and click **Cluster › Cluster Management › Operations › Operations Summary**. For details about this status page, see the *OneFS User Guide*.

You should ensure that the cluster's file system is configured appropriately with the correct data layout, as this configuration can greatly affect client performance, especially for environments that require high bandwidth file transfers, such as video production environments. For more information about specific optimizations for these types of workflows, see *Best Practices for Configuring Isilon IQ for Final Cut Pro* on the EMC Online Support website.

A low-latency network combined with a cluster experiencing only a light load typically equates to a good user experience. Some environments, such as media and entertainment environments, are extremely sensitive to latency. Final Cut Pro, for instance, requires no more than 2-3 milliseconds of roundtrip delay between client and server to sustain adequate throughput for video playback. If the latency is greater than that, users may experience alerts for dropped frames or poor performance. Network latency also influences the responsiveness of the Finder during operations performed on many files, for example, folder listings or file copies.

To monitor cluster performance, including protocol, disk, and network latency, you can use InsightIQ.

Client Network Tuning

For network-attached storage (NAS)-based access to data, the best network environment for an OS X client is one that provides low latency and high throughput. An OS X client's reliance on metadata means that it can be very network-intensive when initially enumerating a folder. To retrieve all the metadata the Finder needs to display a folder, OS X clients make multiple network calls to a server, oftentimes sending multiple calls just to get all the data for just one file. The lower the latency on the network and disks storing the data, the quicker the response from the server and the quicker OS X clients can view the objects on the server. Throughput is not very important for metadata, unlike latency. Both latency and throughput are important, however, for high-bandwidth applications such as Final Cut Pro.

The following sections describe how to configure OS X clients for improved network response in 1 GbE networks. These sections include commands that must be run from the OS X Terminal application. If you are uncertain how to run these commands, contact EMC Isilon Technical Support.

Send and Receive Windows

By multiplying the network bandwidth by the roundtrip delay, you can determine how many bytes an OS X client should use for its send and receive windows. This ensures that there is always enough data being transmitted between the client and server to keep the TCP window as full as possible, thereby avoiding connection slowdowns. OS X sets the default to 64 KB using two system controls (sysctls):

net.inet.tcp.sendspace – TCP send buffer
net.inet.tcp.recvspace – TCP receive buffer

64 KB is an acceptable value, but you can calculate an appropriate value for your network using this formula:

Total network bandwidth (in bytes/second) x roundtrip delay (in seconds) = approximate send/receive buffer (in bytes)

For example, if you have a 1 Gbps network with 1 ms of end-to-end delay, an appropriate buffer value would be:

125,000,000 x 0.001 = 125,000 bytes

If the result of this calculation is fewer than 65,536 bytes, you should keep the OS X defaults. If not, you can change the buffer values by running the following commands from the Terminal application:

Note: We recommend using multiples of 512 when setting this value.

```
sudo sysctl –w net.inet.tcp.sendspace=131072
sudo sysctl –w net.inet.tcp.recvspace=131072
```

To make these values permanent for the OS X client so that they survive a reboot, run these commands:

```
sudo bash –c “echo ‘net.inet.tcp.sendspace=131072’ >>  /etc/sysctl.conf”
sudo bash –c “echo ‘net.inet.tcp.recvspace=131072’ >>  /etc/sysctl.conf”
```

TCP Delayed Acknowledgment

By default, TCP delayed acknowledgment is enabled in OS X, and OS X will automatically detect when to use TCP delayed acknowledgment. Occasionally, this setting causes issues with some protocols and network environments. To avoid this, you sould run delayed acknowledgment in "compatibility mode" for better performance. You can do this by setting the following sysctl from the Terminal application:

```
sudo sysctl –w net.inet.tcp.delayed_ack=2
```

To make these values permanent for the OS X client so that they survive a reboot, run these commands:

```
sudo sysctl –w net.inet.tcp.delayed_ack=2
sudo bash -c "echo 'net.inet.tcp.delayed_ack=2' >> /etc/sysctl.conf"
```

Maximum raw datagram size

By default, OS X has a maximum raw datagram size of 8,192 bytes. If you are using jumbo frames in your network and you need to use ping to test whether you can send 9,000-byte frames, ping will

fail with a “message too large” error. To avoid this error, you can increase the maximum datagram size by setting the following sysctl from the Terminal application:

```
sudo sysctl –w net.inet.raw.maxdgram=16384
```

In addition, you should increase the amount of data the OS X client will receive in raw datagrams from the default of 8,192 bytes to 16,384 bytes. This can be done by increasing the maximum receive space size by setting the following sysctl from the Terminal application:

- `sudo sysctl –w net.inet.raw.recvspace=16384`

To make these values permanent for the OS X client so that they survive a reboot, run these commands:

```
sudo sysctl –w net.inet.raw.maxdgram=16384
sudo sysctl –w net.inet.raw.recvspace=16384
sudo bash -c "echo 'net.inet.raw.maxdgram=16384' >> /etc/sysctl.conf"
sudo bash -c "echo 'net.inet.raw.recvspace=16384' >> /etc/sysctl.conf"
```

Third-Party 10 GbE Network Cards

Some 10GbE network interface card manufacturers might recommend different values to use for the TCP tuning sysctls described above. See the manufacturer's documentation for the appropriate configuration settings.

Cluster Network Tuning

The following sections include the recommended steps to tune network settings for the Isilon cluster.

TCP Inflight Bandwidth Calculation

To improve the network throughput for read operations, disable TCP inflight calculation on the Isilon cluster. This way, the cluster will not calculate the bandwidth delay between the cluster and the client, a calculation that occasionally limits overall throughput to the client.

To disable the TCP inflight calculation:

1. Open a secure shell (SSH) connection on any node in the cluster and log in using the root account.
2. Run the following command:

   ```
   sysctl net.inet.tcp.inflight.enable
   ```

3. If the value reported is 0, you do not need to continue. If the value is 1, continue with step 3.If the value is 1, run the following command:

   ```
   isi_sysctl_cluster net.inet.tcp.inflight.enable=0
   ```

After a short delay, this setting propagates to all nodes in the cluster.

TCP Slow Start

Isilon clusters employ a TCP congestion control strategy called slow-start. On some networks, tuning TCP slow-start parameters allows the network connection to achieve full speed more quickly. To set these settings permanently on the cluster, run the commands:

1. Open a secure shell (SSH) connection on any node in the cluster and log in using the root account.
2. Run the following command:

   ```
   isi_sysctl_cluster net.inet.tcp.slowstart_flightsize=6
   ```

3. Then run the following command:

   ```
   isi_sysctl_cluster net.inet.tcp.local_slowstart_flightsize=10
   ```

After a short delay, these settings propagate to all nodes in the cluster.

On-Disk User and Group Identity

Preferably, you should configure the cluster for native on-disk identity mapping. Clusters that have been upgraded from OneFS 6.0 and earlier may be configured for UNIX on-disk identity. For information on how to change this setting and convert the on-disk permissions to this new format, see the Isilon knowledge base article *Setting the cluster on-disk identity to Native mode in OneFS 6.5*.

Accessing Snapshot Data

OneFS includes a application called SnapshotIQ, which captures point-in-time images of data stored on an Isilon cluster. Snapshots of data can be useful to have if a user needs an older version of an existing file or needs to recover a file that was deleted. If the data that the user needs is included in a snapshot, the user might be able to access the data via a hidden folder in the Isilon NFS or SMB mount point called *.snapshot*. Because the folder is hidden, the user might not realize that it is there. For more information about accessing the *.snapshot* folder from an OS X client, see the Isilon knowledge base articles *When browsing shares, Mac OS X clients do not display files and folders that start with a period* and *Mac OS X clients cannot move files from a .snapshot directory to another directory on the cluster* on the EMC Online Support website. For more information on SnapshotIQ, see the *OneFS User Guide*.

4. Best Practices for SMB Protocol Configuration

OS X SMB clients can be joined to Active Directory domains, enabling them to perform Kerberos-based single sign-on with Isilon clusters that are joined to an Active Directory domain. OS X SMB client also support alternate data streams.

Prior to OS X 10.7, the OS X SMB client and server were based on Samba 3.0. In OS X 10.7, Apple introduced a new SMB client and server, not based on Samba. In OS X 10.9, Apple included SMB version 2.1 support.

The Apple Knowledge Base article HT1568 explains how to connect an OS X client to SMB file shares.

Cluster Configuration for SMB

Isilon does not recommend naming any SMB share "home". This can cause issues with Microsoft Office applications. For more information, see the Isilon knowledge base article *Mac OS X clients cannot open Microsoft Office files saved in a shared /home directory on the cluster*.

SmartConnect pools configured for SMB connections should be configured with an IP allocation method of **Static**. SMB typically will not tolerate dynamic IP address allocation because it is a stateful protocol.

Client Configuration for SMB

OS X's built-in SMB client is not as configurable as its NFS client, but you can configure it to control which authentication methods are used, and in OS X Mavericks (10.9) you can configure whether the OS X client will use SMB 2.1. OS X also provides support for alternate data streams (named streams), mount options, logging, and a few other options. The defaults are acceptable in most use cases, but you should run the following command from the Terminal application on the OS X client to learn more about the settings that you can change:

```
man nsmb.conf
```

Automounts

As a UNIX-based operation system, OS X supports automounts by using the autofs command and by editing automount configuration files in the */etc* folder. These options, however, do not show the mounted volume directly in the Finder. The following procedure explains how to have automounted volumes on the OS X client:

1. In the Finder, mount the share the way you normally would.
2. Open **System Preferences** and go to **Accounts** (OS X 10.5 - 10.6) or **Users & Groups** (OS X 10.7 and later).
3. Select your user account and then click **Login Items**.
4. Drag the icon of the volume you mounted from the desktop to the list of login items.

   Note: If your Mac is configured to not display connected servers on the desktop, open the */Volumes* folder in the Finder and drag the icon for the network share to the list of login items list in **System Preferences**.
5. Close **System Preferences**.

The next time you log in to the OS X client, it will immediately attempt to connect to that network volume.

Note: For SMB shares, an authentication dialog might appear when implementing the solution above. This can be avoided by configuring single sign-on. For more information, see the "Active Directory and Single Sign-On (SSO)" section in this document.

Automount Alternative

Some workflows need to support a specific mount method, such as files or projects on an Isilon cluster that require specific mount paths. For example, when data is migrated from an OS X client SAN to the Isilon cluster, the mount path at the */Volumes* level might not match using the Automount method presented above. To correct this, use a symbolic link on the OS X client to point the original, pre-Isilon mount path in */Volumes* where Isilon shares are mounted.

For example, assume that the original mount point was */nas* on your OS X client and now the mount point is */Volumes/nas* when you mount through the Finder. To link the two paths together, run these commands from Terminal application on the OS X client:

```
rm -rf /nas
ln –s /Volumes/nas /nas
```

Note: The rm command above assumes that you have created a folder called */nas* where you mounted your original storage on the OS X client—you must delete that original mount point folder to create the symbolic link. To confirm the path of the symbolic link, run the mount command from the Terminal application after mounting the Isilon server from the OS X Finder. The output of this command returns the server address and share on which the folder is mounted.

Alternate Data Streams

The SMB protocol supports a feature in Microsoft's NT File System (NTFS) called alternate data streams. Alternate data streams are similar to resource forks in that they are separate constructs where additional file data is stored. Isilon may refer to alternate data streams as "named streams." The SMB protocol can store all OS X-specific resource fork data in an alternate data stream. Alternate data streams are not supported by NFS clients; NFS clients default to using AppleDouble with dotbar files.

By default, some OS X clients do not natively support alternate data stream and therefore create dotbar files whenever new files are created. Alternate data stream is enabled by default in OS X 10.7 and later. To enable alternate data stream support for those clients, you must create the following file at the root of every SMB share:

.com.apple.smb.streams.on

Note: The leading period on the filename prevents the file from appearing in the OS X Finder.

To create the *.com.apple.smb.streams.on* file:

1. Open a secure shell (SSH) connection on any node in the cluster and log in using the root account.
2. Change the folder to the one referenced by the SMB share. For example:

```
cd /ifs/data/IT-share
```

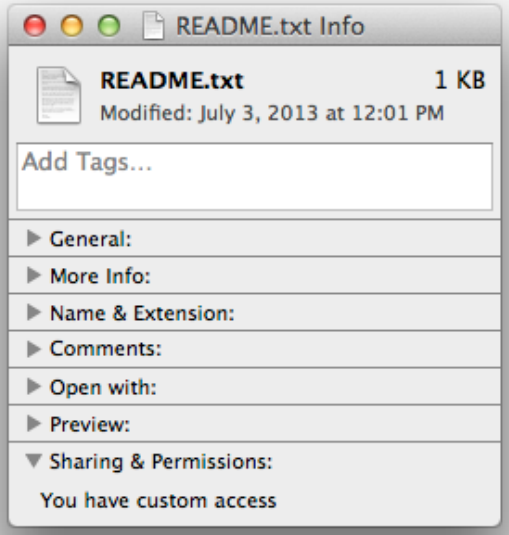
3. Create the file to enable alternate data stream support :

```
touch .com.apple.smb.streams.on
```

4. Remount the OS X SMB clients to the cluster.

If your OS X clients created dotbar files on the Isilon cluster prior to this change, use the OS X-command-line utility called dot_clean to merge the dotbar file back into the Alternate data stream, and then delete the dotbar file. This preserves the resource fork data while eliminating the dotbar file clutter on your file system. For information on how to use the dot_clean utility, run the man dot_clean command from the OS X Terminal application.

Permissions



On a Windows system, when you want to view the permissions for a file or folder on a SMB share, you view the item's properties. From the Security tab, you can see the entire access control list (ACL) for that object. You can then explore each individual access control entry (ACE) to view the permissions for each user or group on that object. The Windows client can modify these values if appropriate permissions are granted.

OS X clients, on the other hand, do not display this same information. If you use the Get Info command on a file or folder, for example, the Sharing & Permissions section indicates that you have custom access. The screenshot shows the **Get Info** window for an OS X 10.9 client accessing a file on an Isilon cluster running OneFS 7.0.2.1. The same **Get Info** window on an OS X client OS X 10.6 client always displays that a user has read and write privileges on an object, even if those are not the permissions that have been set within the file system of the Isilon cluster.

When you compare what the Finder displays to the output from the command line, the OS X client shows the local logged-in user (macuser) as the owner of the file with full read, write, and execute permissions. For example, if you run the following command from the Terminal application:

```
ls -l README.txt
```

The following output is displayed:

```
-rwx------ 1 macuser staff 1031 Jul 3 12:01 README.txt
```

If you view the file's permissions from the cluster's command line by running the following command:

```
ls -le README.txt
```

The following output is displayed:

```
-rwxrwxrwx + 1 DOMAIN\user DOMAIN\domain 1031 Jul 3 12:01 README.txt
 OWNER: user:DOMAIN\user
 GROUP: group:DOMAIN\domain users
 0: group:Administrators allow inherited file_gen_all,inherited_ace
 1: everyone allow inherited file_gen_all,inherited_ace
```

This shows that the file is owned by `DOMAIN\user` with a group of `DOMAIN\domain users`, and that it has an ACL with two ACEs: one for the Administrators group and one for the Everyone group.

This shows that, when it comes to SMB permissions and the OS X client, the OS X client does not natively display Windows ACLs from the SMB server. OS X supports POSIX ACLs, but for the purposes of SMB shares, the lack of Windows ACL support means that the OS X must adapt the permissions it gets from the SMB server into something that works within the OS X permissions model.

Share Permissions versus File System ACLs

SMB shares provide two layers of control to data access: the share permission, and the file system permission. When you first connect to a share, the server evaluates your username and group membership and validates the membership against the list of allowed users and groups on the share. If you are a member of a group that is allowed access, or if your specific username is allowed access, access is granted to the share with the privileges granted in that share permission, typically read access, write access, both, or full control.

Additionally, file system permissions control access to data after connecting to the share. If there are no file system permissions that would give you access to the file or folder, access is denied. File system permissions are determined from either the POSIX permissions on an object (assuming there is no ACL on the file) or from the ACL applied to the object. If the POSIX permissions or ACL do not grant the user access, the user will not be able to access the object. For more information about how OneFS handles permissions, see the *EMC Isilon Multiprotocol Data Access With a Unified Security Model* white paper.

As stated previously, the OS X client cannot manipulate Windows ACLs directly over an SMB share. For this reason, Isilon recommends using inheritable ACLs for file system permissions because OneFS applies those permissions when objects are created and changed. For more information, see the "Suggested Share and File System Permissions Configuration" section in this document.

Samba UNIX Extensions

Older versions of OneFS (6.0 and earlier) that included the Samba-based SMB/CIFS server supported a Samba-specific functionality called UNIX Extensions. This functionality supported the use of POSIX permissions (not ACLs) over an SMB-based connection. For clients that supported this functionality, these clients could see the POSIX permissions on the server and manipulate them over the SMB connection. This functionality had several limitations and integration issues with the OS X Finder and is not supported in OneFS 6.5 and later.

Suggested SMB Share and File System Permissions Configuration

When configuring permissions for folders that will be accessed by OS X clients over SMB, it is important to remember the following:

- The OS X client cannot view or modify Windows ACLs over SMB.
- Windows clients might be able to access data that you intend to restrict with the OS X client permissions configurations.

For these reasons, you should be prepared to use an inheritable ACL on the folders that you expect will be accessed so that when the OS X client creates new objects OneFS will apply the inheritable ACL to the objects at the time of creation and modification. This prevents users from losing access to data because the inheritable ACL ultimately provides access to that data. The examples below illustrate how to use a group-based ACL on the folder.

Note: Although Windows computers honor inheritable ACLs, you should prevent Windows clients from removing ACLs from files and folders because doing so could remove access to that data for the OS X clients. The permissions examples below attempt to take this into account.

For the suggested configuration, assume the following about the environment:

- Share name: *macshare*
- Share location: */ifs/data/macshare*
- Cluster has an Access Zone with a configured Active Directory provider
- The SMB share is available in the same configured Access Zone
- Access to the share is on a per-group basis and the groups are contained in Active Directory
- The Domain Users group has access to the share to read and write data

When the share is created, you have the option to create the share with Windows default ACLs. Clear this option because the default ACLs will be replaced with a specific group ACL later.

The OS X client share permissions (available from the OneFS web administration interface under **Protocols › Windows Sharing (SMB) › SMB Shares › <sharename> › View Details**) are configured so that the Domain Users group has **Read-Write** access. **Full Control** permission is unnecessary in this configuration, given that OS X clients cannot manipulate ACLs.

To configure the appropriate inheritable file system ACL, run the following command from the OneFS command line:

```
chmod +a group domain\\domain\ users allow
dir_gen_read,dir_gen_write,dir_gen_execute,std_delete,delete_child,
object_inherit,container_inherit macshare
```

This command adds an ACL to the folder, which sets the following access privileges for the "domain users" group within the "domain" domain. Replace "domain\\domain\ users" with your organization's Active Directory short domain name and applicable group within that domain.

Implemented	Not Implemented
Traverse Folder / Execute File	Full Control

Implemented	Not Implemented
List Folder / Read Data	Change Permissions
Read Attributes	Take Ownership
Read Extended Attributes	
Create Files / Write Data	
Create Folders / Append Data	
Write Attributes	
Write Extended Attributes	
Delete Subfolders and Files	
Delete	
Read Permissions	

With the share permissions and ACL configured in this manner, multiple users on multiple OS X clients can read and write data, rename objects, and create new objects.

Multi-Protocol Access Considerations

In environments where NFS clients access the same data as OS X SMB clients, setting inheritable ACLs on folders is recommended because doing so helps solve many access problems.

In most environments, the "Balanced" environment setting for ACL policies (set in the OneFS web administration interface by choosing **Protocols › ACLs › ACL Policies**) is sufficient for shared NFS and SMB clients. The NFS section of this document describes a special ACL policy that can be used to ignore the `chmod` command from NFS clients and return global read/write permissions to the NFS clients.

Authentication

Active Directory and Single Sign-On (SSO)

With built-in Active Directory support, OS X clients that are joined to an Active Directory domain are capable of Kerberos-based single sign-on (SSO) to SMB shares on Isilon clusters. When users log in to their accounts on an OS X client using Active Directory credentials and then connect to an Isilon cluster's SMB server that is joined to an Active Directory domain, they see no authentication prompt. The authentication between client and the Isilon cluster is handled using Kerberos tickets. In some versions of OS X, when a Kerberos ticket cannot be obtained, the OS X client will fall-back to NTLM authentication, using the logged-in user's previously entered credentials.

To implement single sign-on for SMB shares, the Isilon cluster and OS X client must be a member of the same Active Directory domain. OS X clients must connect to the cluster by DNS name, and the domain name in that DNS name must match the domain name used in the Active Directory domain. For example, if the Active Directory domain is *corp.isilon.com* then the cluster's fully qualified SmartConnect zone name would need to be *cluster.corp.isilon.com*. The OS X client uses DNS resolution to determine if it can use Kerberos for single sign-on authentication. In this example, both of the domains in these hostnames match, confirming to the OS X client that the Isilon cluster is within its own Kerberos realm.

For more information on implementing Kerberos-based SSO to Isilon SMB shares, see article the Isilon knowledge base article *How to enable Mac OS X single sign-on (SSO) to Active Directory-enabled CIFS shares in OneFS* on the EMC Online Support website.

Note: It is not recommended that you use a SmartConnect zone name that ends with .local, for example, smartconnect.zone.local. Using a .local domain for SmartConnect and your Active Directory environment might create a conflict with the Mac's Bonjour zero-configuration networking service, and could cause connection delays for your SMB users. See the Isilon knowledge base article *Mac OS X: Long Delay in SMB Connection When Using .local Domain* for more information.

SSO on the OS X client is not limited to Kerberos or NLTM for Active Directory-based SMB shares. When users connect to an SMB share for the first time and the OS X client cannot use single sign-on, users are given the option to save their username and password in their login keychain. If this option is selected, the OS X client enables SSO for subsequent authentication attempts. An SMB share's saved password can be cleared by editing your login keychain using the OS X Keychain application that is located in the */Applications/Utilities/Keychain Access* folder.

Open Directory

Open Directory is Apple's implementation of OpenLDAP and is included in OS X server. OneFS can use Open Directory to obtain user and group information, such as user IDs (UIDs) and group IDs (GIDs) for users, for Active Directory coherency in environments where RFC 2307 support is not used.

Note: The Isilon knowledge base article *CIFS authentication against Mac OS X server running Open Directory as an LDAP server* explains how to enable CIFS authentication against Open Directory through the use of unencrypted passwords. This procedure no longer functions with OneFS 7.x because the SMB server in OneFS 7.x does not support unencrypted passwords.

OneFS has been designed and tested to work best with Active Directory, and that configuration is what is recommended by EMC Isilon Technical Support.

Open Directory environments configured in a Golden or Magic Triangle (see the "Golden Triangle and Other Authentication Environments" section) with Active Directory function normally with OneFS. In this configuration, Active Directory is used to store user and group information for the OS X clients that are joined both to Active Directory and to Open Directory. Open Directory is used to distribute system-wide preferences to OS X clients.

Performance

To achieve optimal performance for SMB-connected OS X clients, you should do the following:

- Disable SMB packet signing
- Configure Change Notification for “Norecurse”
- Configure the OS X client to use an optimized TCP delayed acknowledgment setting. For details, see the "TCP Delayed Acknowledgment" section.

SMB Packet Signing

SMB packet signing can degrade performance of SMB operations and may cause other performance issues with some clients. This is due to the overhead of adding a signature to every SMB packet sent by a client. Unless your organization’s security policy requires SMB packet signing, you should disable this feature on the Isilon cluster. SMB packet signing is disabled by default in OneFS 7.0.x.

OS X allows SMB packet signing by default, but does not require it. This setting is controlled on the OS X client with the `dsconfigad –packetsign` command.

Change Notification

OS X uses an SMB feature called change notification to monitor a folder for changes, such as file updates, additions, or deletions. When a change occurs, the SMB server sends a notification to the SMB client, which causes the OS X Finder to refresh the view of a folder.

Workflows that include folders that have change frequently, or that have large numbers of files in a deep tree, may send many change notifications to Mac SMB clients, which can affect Finder refresh performance. For these workflows, you should set the change notification setting to “Norecurse” for maximum performance. To do this, perform these steps:

1. Log in the OneFS web administration interface.
2. Click **Protocols › Windows Sharing (SMB) › SMB shares**.
3. Click **View details** for the SMB share your Mac users connect to.
4. Click **Advanced SMB Share Settings**.
5. Click **Performance Settings**.
6. In the **Change Notify** list, select **Use custom**.
7. In the Confirm Default Override dialog, click **Continue**.
8. In the **Setting Value** list, select **Norecurse**.
9. Click **Save**.

5. Best Practices for NFS Protocol Configuration

Cluster Configuration for NFS

SmartConnect pools configured used for NFS 3 connections can be configured with an IP allocation method of **Dynamic**. NFS 3 is a stateless protocol and can tolerate IP addresses moving between nodes or between network interfaces. OS X, by default, hard mounts NFS exports, meaning that file operations will stop responding until the server comes back or until the NFS mount is forcibly unmounted. Dynamic IP address allocation helps with this behavior because if a node or interface

goes offline, that IP address moves to another interface. The client can retry its operation against that same IP address, causing only a momentary interruption in user access.

NFS 3, using the default UNIX authentication type, can send only one primary group and 16 supplemental groups; and by default, OneFS evaluates only the groups sent to it by the NFS client. In authentication environments where a user is a member of more than 16 groups, users could be denied access to file system objects because a group on the object was not in the list of supplemental groups sent by the client. By using an NFS export option called "Map Lookup UID," OneFS generates a full authentication token that includes all the groups that OneFS can retrieve for a user, not just the groups it receives from the NFS client. This overcomes the 16-group limitation in the NFS 3. Map Lookup UID can be enabled per-export, or globally through the **NFS Settings** tab in OneFS web administration interface, and can be found in the **Export Behavior Settings** section of the **NFS Export Settings** heading.

Client Configuration and Mount Suggestions

Note: This section includes example client configurations for different types of NFS mounts, as well as examples configurations for NFS mount and authentication.

Isilon recommends running OS X 10.8.4 or later for NFS access because of an issue with Finder refresh for NFS clients in earlier versions of OS X. For more information, please see the Isilon knowledge base *Mac OS X NFS clients cannot open files in the Finder if the files are renameda*.

OS X clients can be configured with default mount options that will be used every time the client mounts an NFS export. On the OS X client, these mount options go into the */etc/nfs.conf* file.

OS X describes all available mount options in the `mount_nfs` man page. There is a separate man page (`nfs.conf`) for the additional options that you can add to the */etc/nfs.conf* file.

Note: To implement NFS automounts, see the "Automounts" section.

Private Mount

"Private" means that the OS X client will mount a specific export on the Isilon cluster and will not compete with other clients for access to data in that export. This configuration is useful in environments when OS X clients work in separate folders, and will not create conflicts when updating files and folders.

Note: When file locking is disabled, multiple clients might attempt to write to the same file at the same time, which could corrupt the file.

To configure the OS X clients as a private mount, add the following line to the */etc/nfs.conf* file using a text editor:

```
nfs.client.mount.options=tcp,rw,nolock,rdirplus,nfc,rwsize=32768
```

These options configure the OS X client NFS client for the following:

- NFS 3 over TCP
- Read/write access
- No file locking

- Enhanced metadata retrieval using READDIRPLUS calls to the cluster
- Unicode Normalization Form C, for better Multilanguage support
- Data reads and writes to the NFS mount in 32 KB chunks

Shared Mount

"Shared" means that the OS X client will mount a specific export on the Isilon cluster, and might contend with other clients for access to data in that export.

This configuration is useful in environments where multiple OS X clients share access to data and make changes to files within the same directory. NFS file locking prevents clients from making conflicting file changes.

To configure the OS X clients as a shared mount, add the following line to the */etc/nfs.conf* file using a text editor:

```
nfs.client.mount.options=tcp,rw,rdirplus,nfc,rwsize=32768
```

These options configure the NFS client for the following:

- NFS 3 over TCP
- Read/write access
- Enhanced metadata retrieval using READDIRPLUS calls to the cluster
- Unicode Normalization Form C, for better Multilanguage support
- Data reads and writes to the NFS mount in 32KB chunks

Higher Performance Private Mount

This configuration provides higher performance than the standard private mount configuration and is acceptable in high-bandwidth environments, particularly environments with 10 Gigabit Ethernet.

To configure the OS X clients as a higher performance private mount, add the following lines to the */etc/nfs.conf* file using a text editor:

```
nfs.client.mount.options=tcp,rw,async,nolock,nfc,rdirplus,rwsize=65536
nfs.client.allow_async=1
```

These options configure the OS X client NFS client for the following:

- NFS 3 over TCP
- Read/write access
- Writes to the server can be sent without explicit acknowledgment from the server that the data has been flushed to disk
- No file locking

- Enhanced metadata retrieval using READDIRPLUS calls to the cluster
- Unicode Normalization Form C, for better Multilanguage support
- Data reads and writes to the NFS mount in 64KB chunks

Note: The async mount option can introduce a situation where a file becomes corrupt if a server crashes in the middle of a write operation.

Higher performance Shared Mount

This configuration provides higher performance than the standard shared mount configuration and is acceptable in high-bandwidth environments, particularly environments with 10 Gigabit Ethernet.

To configure the OS X clients as a higher performance shared mount, add the following lines to the */etc/nfs.conf* file using a text editor:

```
nfs.client.mount.options=tcp,rw,async,rdirplus,nfc,rwsize=65536
nfs.client.allow_async=1
```

These options configure the OS X client NFS client for the following:

- NFS 3 over TCP
- Read/write access
- Writes to the server can be sent without explicit acknowledgment from the server that the data has been flushed to disk
- Enhanced metadata retrieval using READDIRPLUS calls to the cluster
- Data reads and writes to the NFS mount in 64KB chunks

Note: The async mount option might introduce a situation where a file is corrupted if a server crashes in the middle of a write operation.

READDIRPLUS

The NFS mount options described above include an option called rdirplus, which enables the NFS READDIRPLUS call. This call combines several calls (LOOKUP, ACCESS, GETATTR, and READDIR) into a single call. When you enable this option, the Isilon NFS server returns all requested data to the client in fewer network calls than if the client had to make each call and receive a response separately. This significantly reduces the time it takes to populate the Finder window with the requested items.

Furthermore, the Isilon NFS server attempts to speed up this data retrieval by prefetching READDIRPLUS data from the file system. If you have folders with thousands of items, you can improve folder listing speeds by specifying how many items OneFS prefetches (by default, OneFS prefetches 10 items at a time).

Note: Increasing this value is useful only if you do not have SSD nodes or Global Namespace Acceleration (GNA) in your cluster, and if you have thousands of items in single folders.

To specify the number of items that OneFS prefetches on a specific NFS export:

1. Log in to the OneFS web administration interface.
2. Click **Protocols › UNIX Sharing (NFS) › NFS Export**.
3. Click **View Details** on the line of the NFS export you wish to edit.
4. Click **Advanced NFS Export Settings**.
5. Click **Performance Settings**.
6. In the **Readdirplus Prefetch** list, select **Use custom**.
7. In the **Confirm Default Override** dialog, click **Continue**.
8. In the **Readdirplus prefetch** box, increase the default value from 10.

   Note: The most optimal values range from 10 - 35. Exceeding 35 does not provide improvement in folder listing performance.
9. Click **Save**.

Additionally, if you wish to set this option globally, you can set it from the **NFS Settings** tab under **Performance Settings**, following the same steps as above. Setting this option globally may consume additional disk operations and throughput throughout the cluster. This could negatively affect performance for clients and applications not accessing metadata.

File Locking

When the OS X Finder lists the contents of a folder, it obtains the initial list and then attempts to get icon previews for the objects in the folder (if configured to do so). As the Finder does this, it tries to take an exclusive lock on the object, read enough of the object to generate the preview, release the lock, and then continue to the next object in the folder.

In rare situations, a deadlock in the Finder can occur if other clients are manipulating files in the same folder and have exclusive locks. The Finder attempts to get its own exclusive lock and Isilon's NFS server denies that attempt because another user already has a lock on that file. The Finder stops retrieving data in the folder until the original holder of the exclusive lock releases its lock.

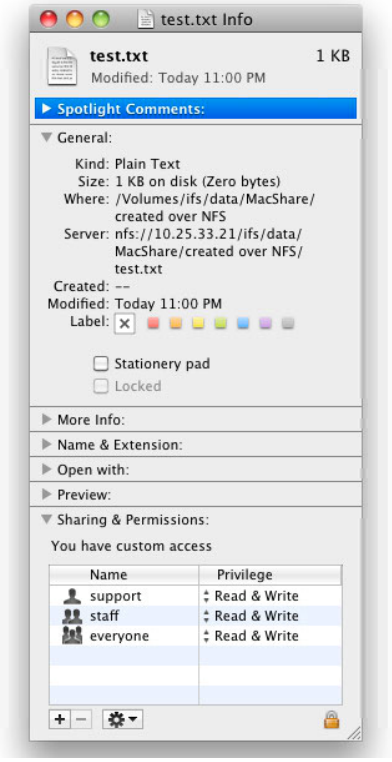
For this reason, it might make sense in some environments to disable file locking. In addition to avoiding this deadlock situation, disabling file locking reduces the amount of network calls the OS X client sends to the Isilon NFS server, which, in turn, helps the Finder list the folder contents faster.

Authentication and Permissions

Permissions issues, on any platform, can be challenging to understand, plan for, and troubleshoot. This section describes what administrators should know, how they should plan their permissions

structure for their OS X clients, and how the OS X client interprets permissions from the Isilon cluster.

Displaying and Modifying Permissions



The NFS 3 client in OS X does not support the display or modification of ACLs. OS X does, however, support POSIX-based permissions, and standard UNIX tools like chmod and chown function from the OS X client if the NFS export is not mapping users and groups (user/group squash, in other words). You can use the Finder to change permissions on files using the Get Info dialog.

For example, the screenshot shows the Get Info dialog for the *test.txt* file and indicates that the file is owned by the support user in the staff group. When you view the same information from the command line, the permissions are the same. For example, if you run the following command from the Terminal application:

```
ls -l test.txt
```

The following output is displayed:

```
-rwxrwxrwx 1 support staff 0 Nov 22 23:00 test.txt
```

This is an example of a file created by a user whose user ID (UID) and group ID (GID) are known to the OS X client, because the OS X client can map the UID and GID to usernames. There are times, however, when the UID and GID might not be known by the OS X client, and will display differently. The Finder typically reports the username and group as "_unknown" and the output for the `ls -l` command displays numerical values in the user and group columns. This difference is known as UID and GID coherency and is discussed in the next section.

Types of Authentication Environments and UID/GID Coherency

The OS X client can function in several authentication environments, the most common of which include:

- A member computer of an Active Directory domain.
    - It can be configured to pull UIDs and GIDs out of Active Directory, or generate its own UID and GID mappings.
- A member computer of an Open Directory (OD) domain.
- A standalone client, not joined to any authentication repository.

As detailed earlier in the "User Identity and Authentication," with the default type of NFS authentication, called AUTH_SYS, also known as UNIX authentication, the NFS client reads and writes all data to the NFS server using the UID and GID of the user who is logged in locally. For example, if the user's UID and GID are 501 and 20 respectively, as would typically be the case for a

standalone client, any files created on the server by that user are owned by that UID and GID, and that UID and GID are used for access permission checks on existing files and folders.

Given this NFS authentication type and the different types of authentication environments, one of the biggest challenges for the OS X NFS client is maintaining UID and GID coherency. UID and GID coherency means that no matter what computer you use to access a shared storage system, your UID and GID are always the same and that your UID and GID are known by the backend authentication store and by the storage system. UID and GID coherency allows NFS clients to create files within OneFS with consistent ownership, making permissions easier to understand and apply. Isilon ensures functional UID and GID coherency by adopting RFC 2307 support, which is already supported by Active Directory and the OS X client. With RFC 2307, Active Directory stores additional schema information for each user that includes specific UNIX attributes that are used to map an Active Directory account and group membership to a UID and GIDs.

When joined to an Active Directory domain, the OS X client can be configured using the Directory Utility program to map UIDs, user GIDs, and group GIDs to specific attributes pulled from Active Directory. By default, these options are disabled so the OS X client generates its own mappings. The OS X client-generated mappings are numerical strings that do not follow a sequential pattern, but they are unique and are generated similarly between multiple OS X clients. If you have an Active Directory environment where your Active Directory-joined OS X clients are generating their own UIDs and GIDs from what they obtain from Active Directory, you can take those UIDs and GIDs and manually add them to Active Directory for use by the Isilon cluster. The Isilon cluster would then be able to map the UID/GID it gets from Active Directory against the on-disk permissions within */ifs*. An administrator who runs the `ls -l` command would see the actual user and group names as retrieved from Active Directory instead of seeing long numerical strings that had been set by the OS X clients.

You can use command line utilities in the OS X client to get a list of Active Directory usernames and groups, and the OS X client-generated UIDs and GIDs. Using that information, you could then populate the UNIX Attributes fields of the user's Active Directory account. For more information, see the Isilon knowledge base article How to use the Mac OS X dscl utility to find the UIDs and GIDs of Active Directory users on the EMC Online Support website. Isilon professional services can help you import the UIDs and GIDs into your Active Directory environment using this data, and can help you develop an operational process to continue to import UIDs and GIDs into Active Directory for new users and groups.

Open Directory (OD) works similarly to Active Directory and stores UID and GID information, and users that log into their OS X clients using Open Directory accounts automatically get UIDs and GIDs from Open Directory, eliminating any need for complicated UID and GID imports. OneFS can be configured to query OD over LDAP, and OS X clients can join OD for authentication purposes. OneFS retrieves UIDs and GIDs from OD using LDAP.

The caveat to UID and GID coherency on the OS X client is that it is best configured when no or few files have been written to the cluster. Changing or enabling UID and GID coherency after many files been written to the cluster can be a large administrative task to accomplish. This requires manually changing the ownership of many files in the file system (by UID and GID), so that file and folder ownership matches up with the UIDs and GIDs that exist in the authentication repository.

Suggested NFS Export and File System Permissions Configurations

For all the following configuration examples, assume the following:

- The cluster is running OneFS 7.0.1 or later
- The NFS export is located at */ifs/data/macexport*
- The administrator has set up an NFS export using the OneFS web administration interface

Wide Open, OS X Clients Only

In this environment, the primary clients accessing the cluster are OS X clients. There is no expectation of permissions control. Essentially, the cluster and NFS exports will be configured with wide-open access. The cluster could be joined to an Active Directory domain, or it could be configured for anonymous access (but authentication is not a concern in this environment). This sort of environment is commonly seen in creative and collaborative environments like those that use Final Cut Pro or Photoshop.

Map all users and groups on the NFS export to nobody.

- In the OneFS web administration interface, click **Protocols** › **UNIX Sharing (NFS)** › **NFS Export**.
- Find your NFS export and click **View details**.
- In the **User/Group mapping** section, click **Edit** and then select **Use custom**
- By default, **Map these users** will be set to **All Users**. Leave this setting unchanged.
- By default, **to this username** will be set to **Username = nobody**. Leave this setting unchanged.
- Click **Save**.

Configure an inheritable ACL at the root of the share giving everybody access.

1. Open a secure shell (SSH) connection on any node in the cluster and log in using the root account.
2. Change to the NFS export's folder by running the following command:

   ```
   cd /ifs/data/macexport
   ```

3. Remove any existing ACL from the folder and set permissions to 777:

   ```
   chmod –b 777 .
   ```

4. Add the inheritable ACL:

   ```
   chmod +a group everyone allow
   generic_all,object_inherit,container_inherit .
   ```

Set the ACL policy for the cluster to a balanced environment.

1. In the OneFS web administration interface, click **Protocols** › **ACLs.**

2. In the **Environment** section, select the **Balanced** option.
3. Click **Submit**.

With this configuration, the cluster always informs clients that permissions are wide open. In other words, the cluster always reports 777 POSIX permissions to the clients, as long as an ACL exists on the object in the file system.

Configure the cluster to ignore any attempts by clients to change permission on objects (in other words, ignore chmod requests that are made from the client) and also report POSIX permissions as 777 if an ACL exists.

For OneFS 7.0.2.3 or later

1. In the OneFS web administration interface, click **Protocols › ACLs.**
2. In the **Permissions Policies** section, in the **chmod on files with existing ACLs** configiruation list, select **Ignore operation if file has an existing ACL.**
3. In the **Advanced Settings** section, in the **Displayed mode bits configuration list**, select **Always display 777 if ACL exists**.
4. Click Submit

For OneFS 7.0.2.2 and earlier

1. Open a secure shell (SSH) connection on any node in the cluster and log in using the root account.
2. Run the following command to get the current global ACL policies value:

   ```
   sysctl efs.bam.acl_policies
   ```
   The value returned will be used to calculate a new value with that contains the additional options.

3. Using a Perl command, combine the original value with the new policy values:

   ```
   perl -e 'printf "%d\n", <value> & ~0xc27 | 0x800 | 0x40000000 | 0x2;'
   ```

   Where <value> is the value that was returned in step 2.

4. Set the new ACL policy cluster-wide:

   ```
   isi_sysctl_cluster efs.bam.acl_policies=<new value>
   ```

   Where <new value> is the value returned from the Perl command in step 3.

Any future changes to the ACL policy, on OneFS 7.0.1 and earlier, require an administrator to perform the four steps above to reset the correct ACL policy value.

Multiple Protocol Access to Data, No UID and GID Coherency

Note: OS X NFS clients connected to a cluster with this configuration create AppleDouble files (files whose names begin with "._"). Windows users will see these files in the SMB share. For more information about AppleDouble files, see the "Resource Forks" section.

In this environment, OS X clients and Windows machines need to access the same set of data. While either or both of the clients may be joined to an external authentication repository such as Active Directory, RFC 2307 is not being used. Windows clients using SMB need to be able to read and write to the same data that the OS X clients access. Permissions need to be open to accomplish this, and to accommodate the lack of UID and GID coherency, the NFS export maps all users and the SMB share will impersonate the same NFS mapped user.

The first four stages of this configuration are the same as the previous "Wide Open, OS X clients Only" section. Complete the *map all users and groups, configure an inheritable ACL, set the ACL policy, and ignore chmod request* steps as described above, then configure the SMB share as described below.

Configure the SMB share that contains the same path as the NFS export to impersonate the "nobody" user.

1. In the OneFS web administration interface, click **Protocols** › **Windows Sharing (SMB)** › **SMB Shares**.
2. Create a new share if one has not been created, or edit the share that contains the path in the NFS export. (*/ifs/data/macexport* in this example).
    a. If you create a new share, select **Do not change existing permissions** for **Directory ACLs**.
    b. In the **Users/Group Accounts** section, the users and groups listed are given a permission level of read-write.
3. Click **Security Settings**.
4. In the **Impersonate user** list, select **Use custom**.
5. Click **Continue** if a warning dialog appears.
6. Type **nobody** in the **Setting Value** box.
7. Click **Save**.

UID and GID Coherency with Active Directory

In this environment, the OS X client and Isilon cluster are joined to the same Active Directory domain. Users log in to the OS X clients using Active Directory credentials and the Active Directory domain supports RFC 2307, by way of Services for UNIX. Active Directory domain accounts have been populated with UNIX attributes such as UID and GIDs. In this environment, Isilon is configured to get those UIDs and GIDs from Active Directory.

The OS X client can be configured to pull the same UID and GID information from Active Directory. As discussed earlier in this document, if you have not already configured your OS X clients to pull

specific UID and GID information out of Active Directory, then the OS X client will automatically generate UIDs and GIDs for Active Directory accounts. To retrieve those OS X-generated IDs follow the instructions in the Isilon knowledge base article How to use the Mac OS X dscl utility to find the UIDs and GIDs of Active Directory users, and then import those IDs into Active Directory.

For environments where the OS X clients should pull UIDs and GIDs from Active Directory, instead of dynamically generating them, reconfigure the OS X clients using these instructions:

1. On the OS X client, open the /System/Library/CoreServices/Directory Utility.
2. Click the lock to make changes. Enter administrator credentials when prompted.
3. Select the **Active Directory** service, and click the **Edit** button in the lower-left corner of the **Services** pane.
4. Click the disclosure triangle next to **Show Advanced Options**.
5. Click the **Mappings** tab.
6. Select the **Map UID to** attribute, **Map user GID to** attribute, and **Map group GID to** attribute options and enter the following values:
    a. For **Map UID to** attribute, change the value to **uidNumber**.
    b. For **Map user GID to** attribute, change the value to **gidNumber**.
    c. For **Map group GID to** attribute, change the value to **gidNumber**.
7. On the **User Experience** tab, select the appropriate options for the OS X client for whether to **Force local home directory on startup disk** and/or **Use UNC path from Active Directory to derive network home location**. If you are uncertain, leave both selected.

    Note: If there is no home folder defined in Active Directory, the home folder will be created on the startup disk of the OS X client.
8. Click **OK** to save the configuration.
9. If you have not logged in to the OS X client with the Active Directory account that you want to use, log in now. Your home folder will be created and owned by the UID that is retrieved from Active Directory.

    Note: If you have already logged in to the OS X client with the Active Directory account that you want to use, you may need to manually change ownership on the user's home folder to the same UID and GID stored in Active Directory.

Configure the Isilon cluster's Active Directory provider to support Services for UNIX (SFU) or RFC 2307.

1. In the OneFS web administration interface, click **Cluster Management** › **Access Management** › **Active Directory**.
2. Click **View details** for the appropriate Active Directory domain.
3. Click **Advanced Active Directory Settings**.
4. In the **Services for UNIX** row, click **Edit**.
5. Select **rfc2307** in the list.
6. Click **Save**.
7. Flush the users and groups cache by running the following command:

   ```
   isi_for_array 'isi auth users flush; isi auth groups flush'
   ```

Configure the NFS export to use POSIX permissions, not ACLs.

1. Open a secure shell (SSH) connection on any node in the cluster and log in using the root account.
2. Clear any existing ACL on the NFS export folder and start with 777 permissions by running the following command:

   ```
   chmod -b 777 /ifs/data/macexport
   ```

3. Change the ownership of the export folder to the proper user and group based on your security policy by using the chown command. For example:

   ```
   chown user:group /ifs/data/macexport
   ```

4. Change the permissions on the export to an appropriate value for your organization by using the chmod command. For example:

   ```
   chmod 755 /ifs/data/macexport
   ```

Note: Because there is no ACL on the folder, group owner inheritance (as defined in the **ACL Policies** › **Advanced Settings** configuration) is determined by the parent folder's group. This is the default setting for OneFS and can be modified.

The default umask for the OS X client is 0022, which means that permissions are set to 644 for all new files, or to 755 for folders and executables. In shared environments that require group access, it might make sense to either change the default umask or use an ACL on the NFS export folder instead of just POSIX permissions. Depending on the complexity of your environment, changing the umask might require administrative overhead and could potentially interfere with applications that expect the umask to stay at its default setting. For more information, see the Apple Knowledge Base article HT2202. If configuring an ACL, you should use an inheritable group ACL that gives users access to those objects in the export.

Configure the NFS export not to map users and groups other than root.

1. In the OneFS web administration interface, click **Protocols** › **UNIX Sharing (NFS)** › **NFS Export**.
2. Find your NFS export and click **View details**.
3. In the **User/Group Mappings** list, select **Use default**.
4. Click **Save**.

Set the ACL policy for the cluster to a balanced environment.

1. In the OneFS web administration interface, click **Protocols** › **ACLs**.
2. In the **Environment** section, select the **Balanced** option.
3. Click **Submit**.

UID and GID Coherency with Open Directory

In this type of environment, OS X clients are joined to Open Directory (OD), Apple's own directory service that runs on OS X Server. OD is LDAP-based and the cluster can retrieve UIDs and GIDs from OD via LDAP.

Configure the cluster to use Open Directory as a LDAP server.

1. In the OneFS web administration interface, click **Cluster Management** › **Access Management** › **LDAP**.
2. Click **Add an LDAP provider.**
3. In the **LDAP Provider Name** box, enter the name to use for the LDAP provider.
4. In the **Server URIs** box, enter the address of your LDAP server in the format *ldap://serveraddress*. For example: *ldap://macserver.macdomain.corp.isilon.com*
5. In the **Base distinguished name** box, type the base domain name (DN) for your Open Directory environment.

   Note: You can look up this information in the OS X Server Admin utility (OS X 10.5 through 10.7) on the OS X Server in the Open Directory service configuration page. Look for **LDAP Search Base** entry similar to the following:

   `dc=macserver,dc=macdomain,dc=corp,dc=isilon,dc=com`

   For OS X 10.8 and later, you need to derive the distinguished name from the **Server** application under **Open Directory** › **Servers**. If the server's address is *macserver.macdomain.corp.isilon.com*, then your distinguished name would be:

   `dc=macserver,dc=macdomain,dc=corp,dc=isilon,dc=com`

6. In the **Bind to** box, type the full DN for the account that will have access to bind to the directory to do user lookups.

   Note: The diradmin account works for this purpose, unless you have another account you prefer to use. For example:

   ```
   uid=diradmin,cn=users,dc=macserver,dc=macdomain,dc=corp,dc=isilon,dc=com
   ```

7. In the **password** box, enter the password for the account you used in step 6.
8. Click **Add LDAP Provider** to save the configuration
9. To confirm that the cluster can contact the Open Directory server, run the following command from the OneFS command-line interface, replacing `<LDAP provider name>` with the name of your LDAP provider from the cluster's configuration. Remember to use quotes:

   ```
   isi auth users list –-provider "lsa-ldap-provider:<LDAP provider name>"
   ```

   The output displays a list of all users in your Open Directory domain.

Configure the NFS export to use POSIX permissions, not ACLs.

1. Open a secure shell (SSH) connection on any node in the cluster and log in using the root account.
2. Clear any existing ACL on the NFS export folder and start with 777 permissions by running the following command:

   ```
   chmod -b 777 /ifs/data/macexport
   ```

3. Change the ownership of the export folder to the proper user and group based on your security policy by using the chown command. For example:

   ```
   chown user:group /ifs/data/macexport
   ```

4. Change the permissions on the export to an appropriate value for your organization by using the chmod command. For example:

   ```
   chmod 755 /ifs/data/macexport
   ```

Note: Because there is no ACL on the folder, group owner inheritance (as defined in the **Protocols › ACLs › Advanced Settings** configuration) is determined by the parent folder's group. This is the default setting for OneFS and can be modified.

The default umask for the OS X client is 0022, which means that permissions are set to 644 for all new files, or to 755 for folders and executables. In shared environments that require group access, it might make sense to either change the default umask or use an ACL on the NFS export folder instead of just POSIX permissions. Depending on the complexity of your environment, changing the umask might require administrative overhead and could potentially interfere with applications that expect the umask to stay at its default setting. For more information, see the Apple Knowledge

Base article HT2202. If configuring an ACL, you should use an inheritable group ACL that gives users access to those objects in the export.

Configure the NFS export not to map users and groups other than root.

1. In The OneFS web administration interface, click **Protocols** › **UNIX Sharing (NFS)** › **NFS Export**.
2. Find your NFS export and click **View details**.
3. In the **User/Group Mappings** section, select **Use default**, if not already selected.
4. Click **Save**.

Set the ACL policy for the cluster to a balanced environment.

1. In the OneFS web administration interface, click **Protocols** › **ACLs**.
2. In the **Environment** section, select the **Balanced** option.
3. Click **Submit**.

Multi-Protocol and Client Access Considerations

Consider the following questions when SMB and NFS clients access the same data:

- Will the AppleDouble metadata files (dotbar files) that are written to the share by Mac NFS clients make the data less usable for the Windows clients?
    - If the answer is yes, then configuring the OS X clients for SMB with alternate data streams support may be the best solution, but keep in mind the performance differences between SMB 1 and NFS, if your OS X clients are not running OS X 10.9, which supports SMB 2. Also, be aware of the metadata performance differences between SMB 1/SMB 2 and NFS.
    - If the answer is no, then administrators can enable an SMB-share level option called "Hide dot files", which sets the hidden attribute on any file that begins with a period (including all AppleDouble files). Setting this option has the effect of hiding files that begin with a period on Windows clients that are not configured to show all files, including hidden files. You can modify this option from the OneFS command-line interface:
        1. Open a secure shell (SSH) connection on any node in the cluster and log in using the root account.
        2. Run the following command to modify your share, replacing <share> with the actual name of your SMB share:

```
isi smb shares modify --share=<share> --hide-dot-files=true
```

Note: Windows clients that are configured to show hidden files will still display AppleDouble (dotbar) files.

- Will there be UID and GID coherency between Active Directory-joined SMB clients and the OS X clients? If the OS X clients are not joined to the Active Directory domain, then you should configure the NFS export to map all users, as was described in the "Multiple Protocol Access to Data, No UID and GID Coherency" section.
- Will there be other non-Mac NFS clients accessing the same data, such as Linux or other UNIX clients? Is there UID and GID coherency between all these clients? If the answer is no, then configuring the NFS export to map all users is the easiest way to ensure that every user has access to the data. This ensures that all clients access data as the same user, as described in the "Wide Open, OS X Clients Only" section.

Golden Triangle and Other Authentication Environments

The Golden Triangle or Magic Triangle is a configuration where an OS X client is joined to Active Directory and Open Directory (OD) at the same time. OD, in this configuration, is primarily used for system-level preferences, or possible home folder storage for users. OS X clients, in this configuration, authenticate against Active Directory, which means UID and GID coherency would be dependent on Active Directory.

Another possible configuration is one where Active Directory is used for a set of Active Directory-joined computers and Active Directory users, but OS X clients are not joined to Active Directory. Instead, the OS X clients are joined to OD. The Isilon cluster could be joined to both Active Directory and separately query OD via LDAP for additional user information for those OS X NFS clients. In the case of Active Directory accounts with usernames that are identical to what is in OD, the Isilon cluster automatically maps those Active Directory accounts and OD accounts together into one authentication persona, or token. Otherwise, for standalone OD accounts, the behavior is much like UID and GID coherency with OD alone: the Isilon cluster uses LDAP to retrieve the information from OD and OS X clients access the cluster with the UIDs and GIDs that they obtain from OD.

There are no specific Isilon configurations required to support Golden Triangle environments, other than configuring an Active Directory provider, configuring an LDAP provider that contacts your Open Directory server, and putting both providers into the same Access Zone.