## CONTENTS

### Preface

### Chapter 1  CDMS Overview

- Introduction ................................................................. 17
- Supported capabilities .................................................. 18
- Scope .............................................................................. 21
- Assumptions ................................................................. 22
- Configuration ................................................................. 24
- System requirements .................................................... 26
- Restrictions and limitations .......................................... 29
- Migration tools ............................................................. 31
- CDMS functionality ...................................................... 34
  - Options ........................................................................ 35
  - Updating data ............................................................ 35
  - Data migration example .............................................. 35
- E-Lab Interoperability Navigator .................................. 39
- Issue tracker ................................................................. 39

### Chapter 2  Site Preparation

- Introduction .................................................................... 42
- Checklist ......................................................................... 42
  - NFS ............................................................................ 42
  - CIFS .......................................................................... 44
- Installing migration tools ............................................. 45
  - Establishing a Control Station interface ...................... 45
  - Procedure ............................................................... 45
  - Using the server_cdms command ................................. 46
- User interfaces choices ................................................. 47
  - GUI supported functionality ....................................... 47
  - Running the GUI ....................................................... 48
  - Accessing the GUI online help facility ......................... 49
- Determining file system size ......................................... 49
  - Running the diskUsage.pl script ................................. 49
  - Extending the MGFS .................................................. 49
- Preliminary setup items ................................................ 50
- International character sets .......................................... 51
Contents

International character set support (NFS) ........................................ 51
I18N support .................................................................................. 52
Unicode support (CIFS) ............................................................... 53
Usermapper methods ..................................................................... 54
Adding Win32 API to Perl script (CIFS) ........................................ 55
Backup considerations ................................................................. 59

Chapter 3  Migration Process Phases
Migration process phases.............................................................. 62
Personnel qualifications ............................................................... 62

Chapter 4  Planning and Design
Operational issues and frequently asked questions ....................... 66
Planning and design..................................................................... 69
Assessment .................................................................................. 70
Data Migration Analysis and detailed planning............................ 74
Optional files ............................................................................... 76

Chapter 5  NFS
Introduction .................................................................................. 80
Summary ...................................................................................... 80
One-to-one migration ................................................................... 81
Many-to-one migration ............................................................... 111
  Parallelization or serialization .................................................. 115
  Verification ............................................................................... 116
  Conversion ............................................................................... 117
  Changing the file system type ................................................... 117
Correcting GIDs (optional) ............................................................ 117
Postmigration testing ................................................................. 118

Chapter 6  CIFS
Introduction .................................................................................. 120
Administrative share methodology ............................................... 121
  Local groups .......................................................................... 121
  Shares .................................................................................. 121
  Data .................................................................................... 121
  Strategies ............................................................................ 122
One-to-one migration .................................................................. 123
  Source file server assumptions ............................................... 123
  VNX assumptions .................................................................. 124
Conventions ................................................................. 124
Summary ......................................................................... 125
Implementation ............................................................. 126
Many-to-one migration ................................................... 176
Benefits .......................................................................... 176
Merge strategies ............................................................ 176
Summary ......................................................................... 177
Implementation ............................................................. 179
Retained server name merge strategy ............................... 181
New server name merge strategy .................................... 207
Postmigration testing ...................................................... 212

Chapter 7 Troubleshooting CDMS

Introduction ...................................................................... 214
Problems and solutions .................................................... 214
NFS ................................................................................. 215
CIFS ............................................................................... 216
Using the server_cdms -info command ............................ 218
Examples .......................................................................... 218
Migration suspension (hang) conditions (NFS) ................. 218
Causes ............................................................................. 218
Network problems between Data Mover and source file server 219
Permission-related hangs ................................................. 219
Stale NFS handles .......................................................... 222
Connection command failures (CIFS) ............................... 223
Summary .......................................................................... 223
Explanations .................................................................... 224
Managing a failed verification or conversion ..................... 229
Analyzing the error ......................................................... 232
Removing an unwanted connection ................................. 232
Using the specialCmd utility .......................................... 233
Using the server_cdms command ................................ ...... 233
Error codes ...................................................................... 234
CDMS server log error messages .................................... 237

Index
FIGURES

<table>
<thead>
<tr>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 NFS configuration</td>
<td>34</td>
</tr>
<tr>
<td>2 CIFS configuration</td>
<td>35</td>
</tr>
<tr>
<td>3 Source file server directory structure</td>
<td>36</td>
</tr>
<tr>
<td>4 Application program read request</td>
<td>37</td>
</tr>
<tr>
<td>5 Transferring /Mail/Users directory attributes</td>
<td>38</td>
</tr>
<tr>
<td>6 Transferring /Mail/Users/Joe directory attributes</td>
<td>38</td>
</tr>
<tr>
<td>7 Migration of an 8 KB file</td>
<td>39</td>
</tr>
<tr>
<td>8 One-to-one migration strategy</td>
<td>123</td>
</tr>
<tr>
<td>9 CIFS source file server to Data Mover migration</td>
<td>126</td>
</tr>
<tr>
<td>10 User account on Windows 2000 or Windows Server 2003 system: One-to-one migration</td>
<td>128</td>
</tr>
<tr>
<td>11 User account on a Windows NT 4.0 system: One-to-one migration</td>
<td>129</td>
</tr>
<tr>
<td>12 File Conversion dialog box</td>
<td>132</td>
</tr>
<tr>
<td>14 User Rights Policy: Windows NT 4.0 system</td>
<td>148</td>
</tr>
<tr>
<td>15 Restricting access on source file server: Windows 2000 or Windows Server 2003 system</td>
<td>150</td>
</tr>
<tr>
<td>16 Restricting access on source file server: Windows NT 4.0 system</td>
<td>151</td>
</tr>
<tr>
<td>17 User Rights Policy: Windows NT 4.0 system</td>
<td>156</td>
</tr>
<tr>
<td>18 Many-to-one migration strategy</td>
<td>177</td>
</tr>
<tr>
<td>19 Retained server name merge</td>
<td>180</td>
</tr>
<tr>
<td>20 New server name merge</td>
<td>180</td>
</tr>
<tr>
<td>21 Local security settings: Windows 2000 or Windows Server 2003 system</td>
<td>187</td>
</tr>
<tr>
<td>22 User Manager for domains: Windows NT 4.0 system</td>
<td>188</td>
</tr>
<tr>
<td>23 Local Security Settings: Windows 2000 or Windows Server 2003 system</td>
<td>194</td>
</tr>
<tr>
<td>24 User Rights Policy: Windows NT 4.0 system</td>
<td>195</td>
</tr>
<tr>
<td>25 The sharedup.exe utility output file (before editing)</td>
<td>203</td>
</tr>
<tr>
<td>26 The sharedup.exe utility output file (after editing)</td>
<td>205</td>
</tr>
<tr>
<td>28 User Rights Policy: Windows NT 4.0 system</td>
<td>210</td>
</tr>
</tbody>
</table>
# TABLES

<table>
<thead>
<tr>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIFS migration source file servers</td>
<td>20</td>
</tr>
<tr>
<td>NFS system requirements</td>
<td>27</td>
</tr>
<tr>
<td>CIFS system requirements</td>
<td>28</td>
</tr>
<tr>
<td>Tool locations</td>
<td>45</td>
</tr>
<tr>
<td>Adding Win32 API to Perl script</td>
<td>57</td>
</tr>
<tr>
<td>One-to-one migration summary</td>
<td>80</td>
</tr>
<tr>
<td>Many-to-one migration summary</td>
<td>111</td>
</tr>
<tr>
<td>One-to-one migration summary</td>
<td>125</td>
</tr>
<tr>
<td>Many-to-one retained server name merge migration summary</td>
<td>177</td>
</tr>
<tr>
<td>Many-to-one new server name merge migration summary</td>
<td>178</td>
</tr>
<tr>
<td>NFS problems and solutions</td>
<td>215</td>
</tr>
<tr>
<td>CIFS problems and solutions</td>
<td>217</td>
</tr>
<tr>
<td>CIFS connection command error indications</td>
<td>223</td>
</tr>
<tr>
<td>Migration error codes</td>
<td>234</td>
</tr>
<tr>
<td>Migration tool descriptions</td>
<td>241</td>
</tr>
<tr>
<td>Window output format descriptions</td>
<td>258</td>
</tr>
<tr>
<td>Disk Usage Info table entry definitions</td>
<td>280</td>
</tr>
<tr>
<td>The lgdup.exe utility options</td>
<td>283</td>
</tr>
<tr>
<td>The shareup.exe utility options</td>
<td>284</td>
</tr>
<tr>
<td>Migration transfer rates</td>
<td>289</td>
</tr>
<tr>
<td>Calculating migration times</td>
<td>292</td>
</tr>
</tbody>
</table>
PREFACE

As part of an effort to improve and enhance the performance and capabilities of its product line, EMC periodically releases revisions of its hardware and software. Therefore, some functions described in this guide might not be supported by all revisions of the software or hardware currently in use. For the most up-to-date information on product features, refer to your product release notes.

If a product does not function properly or does not function as described in this guide, please contact your EMC representative.

Audience

This guide is part of the VNX documentation set, and is intended for use by EMC Customer Support Representatives, end-user customers, and third-party consulting vendors to prepare for and execute data migrations by using the CLI.

Readers of this guide are expected to be familiar with the following topics:

◆ VNX, networking concepts, and UNIX or Windows operating systems.

◆ UNIX, Windows 2000, Windows Server 2003, or Windows NT 4.0 operating-system environments. One to two years experience is preferred.

◆ Configuration and implementation of VNX software and hardware for NFS and CIFS data migration.
Related documents include:

- VNX Glossary
- Problem Resolution Roadmap for VNX
- EMC VNX Command Line Interface Reference for File
- VNX 1.0 Release Notes
- Configuring Events and Notifications on VNX for File
- Configuring NDMP Backups on VNX
- Configuring NDMP Backups to Disk on VNX
- Configuring and Managing CIFS on VNX
- Configuring VNX User Mapping
- Configuring and Managing Networking on VNX
- Managing a Multiprotocol Environment on VNX
- Managing Volumes and File Systems for VNX Manually
- Managing Volumes and File Systems with VNX Automatic Volume Management
- Using VNX Event Enabler
- Using EMC Utilities for the CIFS Environment
- Using FTP and TFTP on VNX
- Using International Character Sets on VNX for File
- Using Windows Administrative Tools on VNX

EMC uses the following conventions for special notices.

Note: A note presents information that is important, but not hazard-related.

CAUTION

A caution contains information essential to avoid data loss or damage to the system or equipment.
A warning contains information essential to avoid a hazard that can cause severe personal injury, death, or substantial property damage if you ignore the warning.

Typographical conventions

EMC uses the following type style conventions in this document:

**Normal**

- Used in running (nonprocedural) text for:
  - Names of interface elements (such as names of windows, dialog boxes, buttons, fields, and menus)
  - Names of resources, attributes, pools, Boolean expressions, buttons, DQL statements, keywords, clauses, environment variables, functions, utilities
  - URLs, pathnames, filenames, directory names, computer names, filenames, links, groups, service keys, file systems, notifications

**Bold**

- Used in running (nonprocedural) text for:
  - Names of commands, daemons, options, programs, processes, services, applications, utilities, kernels, notifications, system calls, man pages
- Used in procedures for:
  - Names of interface elements (such as names of windows, dialog boxes, buttons, fields, and menus)
  - What user specifically selects, clicks, presses, or types

**Italic**

- Used in all text (including procedures) for:
  - Full titles of publications referenced in text
  - Emphasis (for example a new term)
  - Variables

**Courier**

- Used for:
  - System output, such as an error message or script
  - URLs, complete paths, filenames, prompts, and syntax when shown outside of running text

**Courier bold**

- Used for:
  - Specific user input (such as commands)

**Courier italic**

- Used in procedures for:
  - Variables on command line
  - User input variables

< >

Angle brackets enclose parameter or variable values supplied by the user

[]

Square brackets enclose optional values
Vertical bar indicates alternate selections - the bar means “or”

Braces indicate content that you must specify (that is, x or y or z)

Ellipses indicate nonessential information omitted from the example
EMC support, product, and licensing information can be obtained as follows.

**Product information** — For documentation, release notes, software updates, or for information about EMC products, licensing, and service, go to the EMC Online Support website (registration required) at:

http://Support.EMC.com

**Technical support** — For technical support, go to EMC Customer Service on the EMC Online Support website. After logging in, locate the applicable Support by Product page, and choose either Live Chat or Create a service request. To open a service request through EMC Online Support, you must have a valid support agreement. Contact your EMC Customer Support Representative for details about obtaining a valid support agreement or to answer any questions about your account.

Your suggestions will help us continue to improve the accuracy, organization, and overall quality of the user publications. Please send your opinion of this guide to:

techpubcomments@EMC.com
Preface
CHAPTER 1
CDMS Overview

The following topics introduce EMC VNX File System Migration for network file system and Common Internet File System migrations to a VNX:

- Introduction ........................................................................................ 17
- Supported capabilities ....................................................................... 18
- Scope .................................................................................................... 21
- Assumptions ....................................................................................... 22
- Configuration...................................................................................... 24
- System requirements .......................................................................... 26
- Restrictions and limitations .............................................................. 29
- Migration tools ................................................................................... 31
- CDMS functionality ........................................................................... 34
- E-Lab Interoperability Navigator..................................................... 39
- Issue tracker ........................................................................................ 39

Introduction

This section contains EMC® VNX® File System Migration (also known as CDMS) overviews for the network file system (NFS) and Common Internet File System (CIFS) protocols. It contains information on document structure, relevant terminology, NFS and CIFS restrictions, migration tools, supported capabilities, document assumptions, migration requirements, and configuration guidelines.

NFS

CDMS allows you to seamlessly migrate existing NFS file systems from source file servers to the VNX while providing full read/write access to the affected data. It preserves file attribute-related information such as permissions, atime/mtime, and UID/GIDs. NFS is a file-sharing protocol that enables you to share a file system over a TCP/IP network. The file systems might be migrated from a general-purpose NFS source file server, VNX, or an appliance product such as NetApp. During data migration, there is little or no disruption of data availability and access to the UNIX workstation client.
During data migration, it is a best practice to monitor the VNX and EMC Symmetrix® system resources. It is recommended that you use the tools that are provided to do this monitoring such as the EMC Unisphere® GUI. If a further level of monitoring is needed, use Workload Analyzer as this tool allows you to monitor the system’s resources.

**CIFS**

CDMS allows you to seamlessly migrate existing data by using the CIFS protocol from source file servers to the VNX with limited interruption to normal business operations. The CIFS protocol enables Microsoft Windows clients to map share file systems on the VNX as network drives. Table 1 on page 20 provides a list of supported CIFS source file servers.

Each Data Mover can be configured as one or more virtual CIFS servers. Each virtual CIFS server can have its own shares and can belong to a different Windows domain. The VNX supports access control lists (ACLs) for shares, directories, and files accessed with the CIFS protocol.

**Supported capabilities**

This section describes currently supported NFS and CIFS migration capabilities. It describes capabilities common to protocols as well as NFS-specific and CIFS-specific capabilities.

**Common**

The following supported functions are common to NFS and CIFS data migrations:

- Simultaneous migration of multiple servers and file systems.
- Use of international characters in file system migration, I18N for file/directory names for NFS. For CIFS, Unicode is used to format the character set. Using International Character Sets on VNX for File provides more information.
- CDMS on all models of Data Movers. At a minimum, 507 Data Movers are recommended for best performance.
- Consolidation of multiple, existing, file systems into a single, standard, Data Mover UxFS file system.
The following supported functions are unique to NFS data migrations:

- Migration of a single file system in parts to several UxFS file systems by using one or more Data Movers.

- Unchanged migration of the following file inode information:
  - Size
  - Access time
  - Modification time
  - User ID (UID)
  - Group ID, privilege, and mode bits
  - Link count

- Migration of file systems from NFS general-purpose hosts.

---

**Note:** All source file servers use NFS V2/NFS V3 and UDP/TCP protocols. The target VNX must be at version 4.2 or later, with a Data Mover 507 or later with a minimum of 512 MB of RAM.

The E-Lab™ Interoperability Navigator is a searchable, web-based application that provides access to EMC interoperability support matrices. It is available on the EMC Online Support website at [http://Support.EMC.com](http://Support.EMC.com). After logging in to the EMC Online Support website, locate the applicable Support by Product page, find **Tools**, and click **E-Lab Interoperability Navigator**.

- Migrated file systems can use international character sets for directory and filenames. However, the VNX must be configured to support international character sets prior to beginning the migration. *Using International Character Sets on VNX for File* provides information on how to configure and enable the VNX to support a variety of international character sets.

- NFS version 2 and NFS version 3 protocols.

- User Datagram Protocol (UDP) and Transmission Control Protocol (TCP).

- Export aliasing to permit consolidated file systems to be reexported under their original names.

For aliasing, read “Step 5: Connecting the file systems” on page 90.
The following supported functions are unique to CIFS data migrations:

- A native Windows client on the Data Mover.
- Migration of shares from Windows source file servers by using the CIFS protocol, as shown in Table 1 on page 20.

Table 1  CIFS migration source file servers

<table>
<thead>
<tr>
<th>Source CIFS platform</th>
<th>OS versions</th>
<th>Target VNX NAS</th>
</tr>
</thead>
</table>
| Windows NT           | Windows NT  | In 4.x code stream, 4.2.10 or later  
In 5.x code stream, 5.0.11 or later  
Data Mover 507 or later with minimum 512 MB RAM |
| Windows 2000         | Windows 2000| In 4.x code stream, 4.2.10 or later  
In 5.x code stream, 5.0.11 or later  
Data Mover 507 or later with minimum 512 MB RAM |
| Windows Server 2003  | Windows Server 2003| In 4.x code stream, 4.2.10 or later  
In 5.x code stream, 5.0.11 or later  
Data Mover 507 or later with minimum 512 MB RAM |
| CIFS Servers         | CIFS servers from NAS 2.2 onwards | In 4.x code stream, 4.2.10 or later  
In 5.x code stream, 5.0.11 or later  
Data Mover 507 or later with minimum 512 MB RAM |
| All other CIFS Servers | N/A via RPQ only | In 4.x code stream, 4.2.10 or later  
In 5.x code stream, 5.0.11 or later  
Data Mover 507 or later with minimum 512 MB RAM |

The E-Lab Interoperability Navigator, which is available at EMC Online Support, provides the latest information on supported Windows hosts and VNX Data Movers. “Restrictions and limitations” on page 29 provides more information about CIFS source file servers.

- Windows NT 4.0, Windows 2000, Windows Server 2003, and Windows XP are the network clients.
- CIFS protocols.
- Data migration between a source file server and target server within a common Windows NT 4.0 domain, or migration between a source file server and target server within a common Windows 2000 or Windows Server 2003 domain.
- Data migration of the following CIFS attributes:
  - ACLs
  - DOS attributes
  - Audit bits
- When local user accounts are migrated to the destination Windows server by using LGDUP or a similar tool, CDMS supports the migration of local user security IDs (SIDs) in ACLs on files.

Scope

This section contains information on the scope and boundaries of NFS and CIFS migration processes. It describes capabilities common to protocols as well as NFS- and CIFS-specific information.

Common

The following scope and boundaries are common to NFS and CIFS data migrations:

- The Control Station, a dedicated management computer that monitors processes and sends commands to the Data Mover, must be accessible from the network.
- Data migration is accomplished by using the features of CDMS and industry “best practices.”
- Data migration can include one or more servers. Simultaneous migration generally employs multiple exports or shares. In this context, an export or share is a file system, directory, or subdirectory that has been made available to users on the network.
- The NFS and CIFS storage and the NFS and CIFS server consolidation can be accommodated through the merging of exports or shares from single or multiple source file servers.
- In addition to the facilities provided by the UNIX, Windows Server 2003, Windows 2000, and Windows NT 4.0 operating systems on the migration client, you need to use a secure, encrypted, remote login application to the Control Station.
- The migration might require a formal network analysis if a private network or virtual local area network (VLAN) cannot be provided to support the migration effort.
CDMS Overview

**NFS specific**

CDMS does not support NFSv4. There are no scope and boundaries unique to NFS data migrations.

**CIFS specific**

The following scope and boundaries are unique to CIFS data migrations:

- Only Windows Server 2003, Windows 2000, and Windows NT 4.0 servers are supported as CIFS source file servers.

  **Note:** Windows XP systems are not supported as source file servers.

- Usermapper must be configured for all domains so the VNX can resolve all ACLs associated with users and groups during data migration from the source file server. “Usermapper methods” on page 54 provides more details.

  *Configuring VNX User Mapping* provides more information.

- The source server must be a Unicode CIFS server. ASCII CIFS is not supported.

**Assumptions**

This section contains document assumptions for NFS and CIFS migrations. It describes assumptions common to protocols as well as NFS- and CIFS-specific assumptions.

**Common**

The following assumptions with regard to the migration environment are common to NFS and CIFS data migrations:

- Network connections have been installed, adequate bandwidth is available, and the network has no congestion during the period that would affect migration times.

- There are no known issues on the source file server that could affect the migration such as faulty network connection, or a corrupted file system on the source file server, and so forth.

- All servers involved in the migration effort are part of a campus network environment.

- All network connections to the VNX are configured for a minimum of 100 Mb/s. Migrations performed with CDMS use a significant amount of bandwidth. Any network below 100 Mb/s might be unable to support this type of migration effort,
and might incur long delays of frequent timeouts. In addition, the ability to support full duplex operations is highly recommended between the source file server and the Data Mover. This configuration option further enhances migration performance.

◆ Installing cabling, network equipment (fiber or Ethernet cables, interface cards, and so forth), and extensions to the physical network might be required to complete physical connections. Installation might include provisions for multiple network connections to the Data Movers, adding additional network interface cards (NICs) to specific file servers, or even replacing lower-speed components with higher-speed ones. NICs in the Data Movers determine the speed and cable types supported in this environment.

◆ Although it is possible to configure a migration effort by using common or public interconnections of an intranet, it is highly recommended that all migration efforts be configured to occur on a dedicated connection between the source file server and target Data Mover. This design affords the highest possible bandwidth availability to the migration.

◆ For small migrations, the level of planning required to migrate data from source file servers to a VNX is likely modest, but still a necessary part of the process. At a minimum, the following tasks are performed:
  - Analyzing CIFS shares and NFS exports
  - Calculating the amount of data to be migrated
  - Configuring VNX storage
  - Estimating migration times
  - Optionally, excluding certain files/directories from migration
  - Optionally, identifying large/highly used files
  - Mapping source and destination shares and exports
  - Reviewing the basic network for the migration process
  - Scheduling the migration
  - Understanding the source file server’s data structure

However, migrations of more than 10 servers require a significant effort in planning and designing the migration. For large migrations, a detailed project plan might be required as a roadmap.
CDMS Overview

**NFS specific**

The following environmental assumptions are unique to NFS data migrations:

- The UNIX environment, including the source file server and Data Mover, is configured correctly for appropriate access, including Domain Name System (DNS) and Network Information Service (NIS) access, if required.

- For Perl migration planning scripts only, CDMS has a UNIX workstation capable of supporting an NFS connection. In addition, ensure that the Perl package is 64-bit compatible.

**CIFS specific**

The following environmental assumptions are unique to CIFS data migrations:

- The CIFS environment, including the Windows source file server and Data Mover, is configured within the same domain. Usermapper is strongly recommended for CDMS migrations that use the CIFS protocol. Usermapper should be configured for all domains that have access to files and shares on the source file server. 

  *Configuring VNX User Mapping* provides more information.

- For Perl migration planning scripts only, CDMS has a Windows client with Windows NT 4.0, Windows 2000, or Windows Server 2003 software within the Windows domain where migration is being performed. If the Perl script is used, the Perl package should be 5.6 or 5.8; but in the case of 5.8, the procedure for incorporating the Roth pragmas needs to also be updated to reflect changes in Perl Package Manager (PPM) commands. This requirement is valid for the diskUsage.pl, dirprivW.pl scripts, and so forth.

**Configuration**

This section contains configuration guidelines and helpful tips you should consider before setting up an NFS or CIFS migration. It describes configuration information common to protocols as well as NFS- and CIFS-specific information.

**Common**

The following configuration guidelines are common to NFS and CIFS data migrations:

- The source file server must be read-only to prevent users from having write or modify access from the client side.

- CDMS migrations can only be targeted to Migration File System (MGFS) file systems on the Data Mover.

- The Perl package is 64-bit compatible.
The following configuration guidelines are common to NFS data migrations:

- If consolidating from multiple NFS source file servers that are not covered by a common NIS, you should verify that GID numbers and names are consistent on each source file server. Otherwise, the ch_group.pl script might be needed after the migration to change GID numbers, if they match another GID number in a directory or subdirectory tree.
- Either the owner of each file must have read privileges on every file within the file system, and execute privileges on all directories to be migrated, or the Data Mover must have been granted root access on the source file server export.

The following configuration guidelines are common to CIFS data migrations:

- Data Mover security should be set to Windows NT mode for CDMS migrations.
- The CIFS source file server must be set to read-only after share migration has begun. The file system should be read-only for all accounts except the account performing the migration. You can remove the Access from Network privilege from the source file server to prevent users from accessing the server.
- The user account used for migration should have enough privileges to access a share, meaning the user is able to read all entries within a share. The minimum requirements are the user rights (under Local Security Policies) to back up and restore files. These rights are frequently granted by becoming a member of the Backup Operators local group.
- Domain account privileges with backup operator, generate security audits, and manage auditing and security log rights.
- CDMS provides a method to migrate data of the file system objects (files, directories, short cuts, and so forth) and their attributes. Other Windows security-related information is not migrated by CDMS, such as:
  - CIFS local groups
  - All CIFS shares

However, EMC tools are provided to migrate the following items:

- Local group migration must be done by using the `lgdup.exe` utility. Run this tool before you set up a CDMS connection because it preserves privileges to the groups/accounts for that system.
• Share definition migration must be done using the `sharedup.exe` utility. Run this tool after you set up the connection because the Windows client needs a valid pathname when the share is created. The `sharedup.exe` utility handles the ACL on the share.

◆ If you are migrating data from:
  • A Windows NT 4.0 server, the Network Basic Input Output System (NetBIOS) name syntax must be used for server domain names.
  • A Windows 2000 server or Windows Server 2003, the fully qualified domain name (FQDN) must be used for server domain names.

The account used for migration should always follow the migrated server’s format, either in:

• `domain\user` (Windows NT 4.0)  
  or

• `FQDN\user` (Windows 2000 or Windows Server 2003)

This procedure eliminates confusion about migration in a Windows 2000 or Windows Server 2003 domain with a Windows NT 4.0 system as the source file server.

◆ Evaluate and document all domains assigned privileges on the source file server. In a multidomain environment, this task is important for proper Windows user and group authentication on the VNX. This is particularly true when using the Usermapper `usrmap.cfg` file in which all domains must be identified and given user ID (UID) and group ID (GID) ranges.

**System requirements**

This section lists system requirements for NFS and CIFS migrations, including hardware, software, network, and storage considerations. This section describes minimum system requirements necessary to run CDMS version 2.0 successfully.
Table 2 on page 27 provides requirement details for hardware, software, network, and storage items.

### Table 2  NFS system requirements

<table>
<thead>
<tr>
<th>Item</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware</td>
<td>A minimum of one UNIX workstation must be available to the data migration specialist to act as a client.</td>
</tr>
</tbody>
</table>
| Software    | • Perl script version 5.6 or later must be installed at the UNIX workstation. UNIX Perl script versions began dropping leading zeros on the version number (although some still retain them). Perl script binaries for most UNIX workstations can be found at the following URL: [http://perl.com/CPAN/ports/index.html](http://perl.com/CPAN/ports/index.html)  
• Version 5.6 or later of the software must be installed (latest revisions are encouraged) on the VNX. The [VNX 1.0 Release Notes](#) provide version-specific information. |
| Network     | • All servers involved in the migration should be part of a campus network environment. Migrations using wide area networks (WANs) are not supported.  
• A network analysis is highly recommended before any migration effort begins. The viability of the network to support the migration must be evaluated. A thorough understanding of the network is vital to a successful CDMS migration effort.  
• All network connections to the VNX must be configured for a minimum of 100 Mb/s.  
• Full-duplex operation support is highly recommended between the source file server and the VNX Data Mover.                                                      |
| Storage     | • Allocating adequate VNX storage is key to an effective network-attached storage (VNX for file) migration. Because the block size on the VNX is 8 KB, migration of very small files (1 KB or less) can cause an eight-fold increase in the amount of storage required to complete the migration. Therefore, ensure to run the diskUsage.pl script before the migration begins to determine your needs.  
“[diskUsage.pl script](#)” on page 275 provides more information about this tool. |
Table 3 on page 28 provides requirement details for hardware, software, network, and storage items.

### Table 3  CIFS system requirements

<table>
<thead>
<tr>
<th>Item</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hardware</strong></td>
<td>A minimum of one Windows Server 2003, Windows 2000, or Windows NT 4.0 system must be available to the data migration specialist to act as a client.</td>
</tr>
</tbody>
</table>
| **Software** | • Perl script version 5.6 or 5.8 must be installed at the Windows client. In addition, the Win32 API extensions must also be installed on the Windows client.  
  "Adding Win32 API to Perl script (CIFS)" on page 55 provides more information on the installation.  
  Perl script version 5.6 and the Win32 API for Windows Server 2003, Windows 2000, or Windows NT 4.0 might be obtained from the ActiveState website URL: http://activestate.com/  
  • Version 5.6 or later of the software must be installed (latest revisions are encouraged) on the VNX.  
  The VNX 1.0 Release Notes provide version-specific information.  
  • Microsoft’s Word and Excel 2000 or later are used with the EMC scripts and utilities. Therefore, they must be installed on the Windows client where the premigration scripts are run.  
  • Usermapper must be configured for all domain users and groups assigned on the source file servers. |
| **Network**  | • All servers involved in the migration should be part of a campus network environment. Migrations using WANs are not supported.  
  • A network analysis is highly recommended before any migration effort begins. The viability of the network to support the migration must be evaluated. A thorough understanding of the network is vital to a successful CDMS migration effort.  
  • All network connections to the VNX must be configured for a minimum of 100 Mb/s.  
  • Full-duplex operation support is highly recommended between the source file server and the Data Mover. |
| **Storage**  | • Allocating adequate VNX storage is key to an effective VNX migration. Because the block size on the VNX is 8 KB, migration of very small files (1 KB or less) can cause an eight-fold increase in the amount of storage required to complete the migration. Therefore, ensure to run the diskUsage.pl script before the migration begins to determine your needs.  
  "diskUsage.pl script" on page 275 provides more information about this tool. |
Restrictions and limitations

This section lists and briefly describes restrictions and limitations for NFS and CIFS migrations. It describes restrictions and limitations common to protocols as well as NFS- and CIFS-specific information.

**Common**

Common NFS and CIFS restrictions and limitations include the following:

- EMC VNX Replicator™, the EMC SnapSure™ feature, the EMC TimeFinder®/FS feature, and business continuance volumes (BCVs) are not supported on volumes containing MGFS file systems. They can be enabled, however, after the migration completes and the migration volume is remounted as a UxFS file system.
- EMC's VNX HighRoad® multiplex file system (MPFS) is not supported during the migration. It can be enabled, however, after file system conversion completes.
- Data Mover-based Network Data Management Protocol (NDMP) is not supported. However, LAN-based NDMP is supported.
- File system quota data is not migrated to the VNX. MGFS does not allow quotas to be turned on to the file system itself during migration. They can be enabled after migration completes, however.
- No extended attributes are supported beyond those supported by the VNX (none for Network Appliance's Qtrees, none for IBM's OS/2, none for media access control (MAC), none for Novell, and so forth).
- CDMS migrations can only be targeted to MGFS file systems.
- Virus checker has a performance impact for CDMS migration and the impact could be significant depending on the settings.
- Users should not start multiprotocol directory (MPD) translation during a migration.
- Verify that the source file server does not have any cron jobs that remove entries. These jobs cause migration verify and convert failures.

**NFS specific**

NFS restrictions and limitations include the following:

- Files that cannot be opened by the Data Mover by using the NFS protocol (for example, device files in use by the source operating system, named pipes, and so forth) cannot be migrated by CDMS. They can be copied to the destination Data Mover by other methods, however.
CDMS Overview

◆ Timestamps associated with attribute changes are not migrated.
◆ An NFS/CIFS file system shared by UNIX workstations and Windows clients cannot be migrated to an MGFS volume on a VNX. To migrate it, you must choose to migrate either CIFS or NFS attributes, therefore losing the other.

**CIFS specific**

CIFS restrictions and limitations include the following:

◆ Data encrypted by the Microsoft encrypt attribute feature must not be migrated to the VNX. If you need to migrate this data, you must decrypt it before the migration begins. The data is skipped during migration. The dirprivW.pl script can identify the Microsoft-encrypted data.

Refer to [http://www.microsoft.com](http://www.microsoft.com), and search on “decrypting” for information on decrypting any encrypted files.

◆ Filename alternate streams are not migrated in CIFS.

◆ A CIFS 8.3 DOS name might not be preserved. Since the mangling mechanism is different between Windows NT 4.0 and VNX, the M8.3 name might be different before and after migration.

◆ Compressed files based on NTFS are migrated as noncompressed files.

◆ Cross-domain migration (for example, Windows NT 4.0 to Windows 2000 or Windows Server 2003) is not supported. The source file server, Windows client, and Data Mover must all be within the same domain.

◆ Data migration is not supported from domain controllers such as DCs, PDCs, or BDCs.

◆ Domain local groups used in Windows 2000 or Windows Server 2003 can be migrated to a Data Mover. Domain local groups used in Windows NT 4.0 can only be copied by LGDUP and become ordinary local groups on the VNX.

◆ Share names that are connection points are not supported.

◆ Microsoft guarantees a maximum pathname length of 256 characters. If the current path is close to 256 characters and the new path for CDMS access (including the new NetBIOS name on the Data Mover) is greater than 256 characters, the Windows client loses access to the file due to this software restriction.
Migration tools

This section lists and briefly describes NFS and CIFS migration tools. These and other tools are available on the Celerra Network Server Applications and Tools CD shipped to the customer site.

Appendix A, “Using CDMS Migration Tools,” provides more information about the listed scripts and executables.

CDMS requires several EMC scripts and utilities to accomplish a complete NFS and CIFS migration by using the CLI.
The following tools are common to NFS and CIFS data migrations:

- **dircount.pl**
  This Perl script gathers information that allows you to observe what a directory tree structure looks like, and determine the number of files at each level.

- **diskUsage.pl**
  This Perl script allows you to estimate the amount of storage space you need on the target VNX when you migrate a specified directory or file. It also provides a list of the amount of data in various filesizes that you can use to estimate migration times.

- **specialCmd**
  This utility, run on a Control Station, allows you to bring an offline directory online, or disconnect a current connection.

The primary migration tool, the **server_cdms** command, run from the Control Station, is described in “Using the server_cdms command” on page 46 and in the *EMC VNX Command Line Interface Reference for File*. 
NFS specific

The following tools are unique to NFS data migrations:

- **ch_group.pl**
  This Perl script changes a GID number if it matches another GID number in a directory or subdirectory tree. All other files remain unchanged.

- **dirprivU.pl**
  This Perl script should be used before a migration starts to check that all files can be read and thus migrated successfully.

CIFS specific

The following tools are unique to CIFS data migrations:

- **backupWrapper.exe**
  This executable file provides the necessary backup operator privilege for the running process so a script such as dircount.pl can read files accessed exclusively by owners.

- **connBuilder.pl**
  This script generates the connection commands for top-level shares, and creates possible exclude files. It is intended to build the connection commands to minimize typing errors. However, you must still manually type text in Microsoft Word files, command files, and the optional exclude file.

- **dirprivW.pl**
  This Perl script should be used before a migration starts to check that all files can be read and thus migrated successfully.

- **lgdup.exe**
  This utility duplicates the local groups and the privileges database from a Windows source file server to the VNX.

- **sharedup.exe**
  This utility duplicates the share names from one server to another, preserving the permission and comments on the shares. It requires that you be a member of the administrator group. In addition, you can use this utility to identify all shares on a CIFS source file server.
CDMS functionality

To start migrating data, you must establish network connectivity from a VNX Data Mover to the existing source file servers containing the NFS or CIFS protocol data. When adding the Data Mover to the network, it can be assigned a new IP address, or it can assume the IP address of the original file server. Prior to beginning the migration, UNIX workstation or Windows client access must be shifted to the VNX from the source file server. This task might require you to rename the original source file server, and set the original source file server name on the VNX. Then, you must mount the source file server as a read-only file system (in other words, no allowed user updates or deletes). Now the Data Mover assumes total I/O activity. Figure 1 on page 34 shows the NFS logical (not physical) relationship among the VNX, a source file server, and the network.

**Figure 1** NFS configuration

On read requests, the VNX acts as a server to the network and as a client to the source file server. It only accesses the source file server if the information being requested is not already on the VNX. On write requests, the VNX accepts the new information that must be stored, instead of the source file server. The Data Mover uses the NFS or CIFS protocol to retrieve files and directories from the source file server in the background as they are requested from clients.
Figure 2 on page 35 shows the CIFS logical (not physical) relationship among the VNX, a source file server, and the network.

![CIFS configuration diagram]

**Figure 2  CIFS configuration**

**Options**

Data is migrated to the VNX when:

- Any file is accessed on the source file server by a UNIX workstation or Windows client user. This action initiates file migration to the VNX. For large files, only the requested data might be moved, while for small files, all data is moved.

- The `server_cdms -start` command is run from the Control Station. This command starts an internal code thread that reads all parts of all files, and forces untouched data to be migrated to the Data Mover. The command ensures that all source file server data is migrated to the VNX.

**Updating data**

After starting the migration of data, modified and new files are only stored on the VNX.

**Data migration example**

The following example shows how data is migrated to the VNX. You should assume the following directory structure exists on the source file server, as shown in Figure 3 on page 36.
In this example, migration begins when a client’s application program issues a request to read the first 8 KB of the mail.txt file through the VNX. This request triggers the opening and migration of each component in the path from the root of the file system to the mail.txt file, as shown in Figure 4 on page 37. The steps are as follows:

1. The /Mail directory is opened and migrated to the VNX.
   
   This migration includes attributes for the /Mail/Options and the /Mail/Users directories, but not their contents.

   A “placeholder” is created for the /Options directory.
**Note:** In all instances, a gray box indicates a file or directory's attributes are migrated, but not its data. The white boxes indicate the directory structure that must be opened to read the file.

2. The /Mail/Users directory is opened.

In this step, the /Mail/Users directory is migrated by transferring the attributes for the /Mail/Users/Alex, /Mail/Users/Ted, and /Mail/Users/Joe directories, as shown in Figure 5 on page 38.

A “placeholder” is created for the /Alex and /Ted directories.
3. The /Mail/Users/Joe directory is opened.

In this step, the /Mail/Users/Joe directory is migrated by transferring the attributes for the /Mail/Users/Joe/mail.txt file, as shown in Figure 6 on page 38.
4. The mail.txt file is read, as shown in Figure 7 on page 39.

The first 8 KB of the mail.txt file is migrated to satisfy the client’s application program request. The remainder of the file is migrated when the last byte of the file is read. If no application program reads the last byte of the file, it is read by the internal migration command later.

![Figure 7 Migration of an 8 KB file](image)

**E-Lab Interoperability Navigator**

The E-Lab Interoperability Navigator is a searchable, web-based application that provides access to EMC interoperability support matrices. It is available at [http://Support.EMC.com](http://Support.EMC.com). After logging in to the EMC Online Support website, locate the applicable Support by Product page, find **Tools**, and click **E-Lab Interoperability Navigator**.

**Issue tracker**

The issue tracker contains a dynamic list of bugs, similar to release notes, for selected EMC products. This tool allows to search on keyword fields to find particular bugs, and identify the software release in which the bugs were identified or fixed. You can use
CDMS Overview

this tool to check if a bug was fixed in a particular revision. You can also use it if you are upgrading and want to know about the revision to which you are going. This is not, however, a problem or status tracking system. To access this tool, go to EMC Online Support, EMC’s secure extranet site.
CHAPTER 2
Site Preparation

The following topics provide information that you need prior to implementing a CDMS migration:

- Introduction ........................................................................................................ 42
- Checklist ............................................................................................................ 42
- Installing migration tools .................................................................................. 45
- User interfaces choices ..................................................................................... 47
- Determining file system size ............................................................................ 49
- Preliminary setup items .................................................................................... 50
- International character sets ............................................................................. 51
- Usermapper methods ......................................................................................... 54
- Adding Win32 API to Perl script (CIFS) .......................................................... 55
- Backup considerations ....................................................................................... 59
Introduction

You must obtain specific information and complete certain procedures prior to starting any CDMS migration to a VNX.

This chapter provides:

◆ Process checklists for NFS and CIFS migrations
◆ Instructions for installing migration tools on the client
◆ User interface functionality through the Unisphere GUI
◆ Instructions on how to determine file size and accessibility
◆ Preliminary migration setup items
◆ Configuration information for international character support
◆ Instructions for installing a Win32 API module to the Perl script

It also includes several considerations for information backup, a list of items that are required for CDMS migration, configuration information for Unicode, and user translation methods.

Checklist

This section contains a checklist for NFS and CIFS migrations, including preliminary assessment, planning and design, premigration setup, migration of data, and postmigration considerations.

NFS

☐ Assessment
  ☐ Summarize installed hardware.
  ☐ Evaluate migration issues.

☐ Planning and design
  ☐ Define limits of migration.
  ☐ Determine the VNX storage configuration.
  ☐ Evaluate and define file system merging and export aliasing requirements.
  ☐ Define source file server-to-Data Mover mapping.
Site Preparation

- Check the access privilege to the data on the source file server with the dirprivU.pl script.
- Estimate migration times from diskUsage.pl script and dircount.pl script outputs.
- Review network evaluation, and identify network enhancements.
- Determine schedule and priorities.
- Define acceptance criteria and sign-off procedures.
- Develop a migration plan and implementation checklist.

Premigration setup
- Configure the VNX.
- Prepare the UNIX workstation client.
- Configure international character set support (if necessary).
- Set up migration logging.
- Back up the source file server.

Migrate data
- Add the Data Mover to the network.
- Prepare file systems for migration.
- Prepare source file servers for migration.
- Connect source file servers to the VNX.
- Export the file systems so clients can access them on the VNX.
- Run the server_cdms -start command to complete data migration.

Postmigration
- Ensure that all remaining source data is migrated to the VNX.
- Verify that migration has completed.
- Convert the migrated file system to a UxFS.
- Document the migration.
- Verify completion to customer acceptance criteria.
- Obtain customer acceptance.
- Close the EMC Customer Service call.
Site Preparation

CIFS

- Planning
  - Summarize installed hardware.
  - Evaluate migration issues.
  - Perform analysis and detailed planning of the data migration.

- Premigration setup
  - Copy the EMC premigration utilities onto the Windows client.
  - Evaluate the source file server directory structure for files and directories that might be excluded from the migration.
  - Identify any high-priority files on all drives from the source file server, and create the optional include file.
  - Back up the source file server.
  - Configure the Data Mover for the migration environment and for CIFS.
  - Create an MGFS on the VNX.
  - Prepare the source file server for migration.
  - Grant the required rights/privileges on the source file server and Windows client.

- Migrate data
  - Migrate the local groups.
  - Create CDMS connections for the shares specified in the planning and design phase.
  - Execute the `sharedup.exe` utility against each drive from the source file server.
  - Run the `server_cdms -start` command.

- Postmigration
  - Ensure that all remaining data is migrated to the VNX.
  - Verify that migration has completed.
  - Convert the migrated file system to a UxFS.
  - Document the migration.
  - Verify completion to customer acceptance criteria.
  - Obtain customer acceptance.
Close the EMC Customer Service call.

Installing migration tools

This section tells where the premigration planning tools are located, and how to install scripts and executables needed to perform an NFS or CIFS data migration.

Establishing a Control Station interface

You need to establish a secure, encrypted, remote login application interface to the Control Station so you can run utilities and commands from that unit.

Procedure

Premigration tools you use to migrate data from a source file server to a VNX are located in the Control Station /nas/tools/cdms directory. To install these tools, perform the following:

1. Download a Perl script to the UNIX workstation or Windows client from this URL:

   http://perl.org/CPAN/ports/index.html

   If performing a CIFS migration, download Perl script version 5.6 for the Windows client. If performing an NFS migration, the only requirement is that the Perl script be 64-bit compatible.

2. Install the Perl script on the UNIX workstation and/or Windows client.

3. Copy the appropriate tools from the CD to the UNIX workstation and/or Windows client.

   All EMC utilities and scripts that are used to automate premigration tasks should be placed in the same directory on the UNIX workstation or Windows client.

   Standard VNX tools, present on the Control Station, are run from that location. Table 4 on page 45 supplies the tool installation locations.

Table 4  Tool locations (page 1 of 2)

<table>
<thead>
<tr>
<th>Tool</th>
<th>Migration application</th>
<th>Installation location</th>
</tr>
</thead>
<tbody>
<tr>
<td>backupWrapper.exe</td>
<td>Utility</td>
<td>CIFS</td>
</tr>
<tr>
<td>ch_group.pl</td>
<td>Script</td>
<td>NFS</td>
</tr>
<tr>
<td>connBuilder.pl</td>
<td>Script</td>
<td>CIFS</td>
</tr>
</tbody>
</table>

   From CD to Windows client
   From CD to UNIX workstation
   From CD to Windows client
Appendix A, “Using CDMS Migration Tools,” provides more information about the listed scripts and executables.

For CIFS migrations, you must install the Win32 API module and replace the library on the Windows client, as described in “Adding Win32 API to Perl script (CIFS)” on page 55.

Using the server_cdms command

The server_cdms command, run from the Control Station, establishes and removes connections to remote systems, which allows users to start on-access migration. It handles all CDMS management, including supervision of new migration threads, and formatting of status information for the user.

The server_cdms command can create auto-migration threads on the Data Mover to ensure that all data is migrated from the remote system.

The server_cdms command also checks the MGFS state as well as all auto-migration processes, and the connection or the entire MGFS file system, reporting if all data has been migrated successfully (but still allows future connections on the MGFS).

Note that the MGFS identifies whether the file or directory is an offline or online (in other words, a normal system) inode. The UxFS has been modified to extend the inode structure to hold offline inodes.
Offline inodes represent file system objects that are in the process of being migrated, or have not yet started migration. Each of these offline inodes stores up-to-date attributes for the object it represents. UxFS inodes create temporary migration inodes. Migration inodes hold the actual data for an object during migration. When an object (or file system) is fully migrated, and verified or converted, the migration inode overwrites the offline inode, and the migration inode is destroyed.

The *EMC VNX Command Line Interface Reference for File* provides information about the `server_cdms` command.

**User interfaces choices**

CDMS can be executed from either a command line interface (CLI) or graphical user interface (GUI). This document describes how to perform *data migration* for the NFS and CIFS protocols by using the CLI.

---

**Note:** Web management is only offered through the use of the Unisphere Advanced Edition.

---

**GUI supported functionality**

The GUI provides a subset of the functionality supplied by the CLI. You can use the GUI to perform the following operational tasks:

- View a list of migrations.
- View the properties of a specific migration.
- View Usermapper on the Data Mover.
- View NFS source file server exports.
- View the properties of a specific MGFS file system.
- View a list of MGFS file systems.
- View migrations of an MGFS file system.
- Add a new migration.
- Add a new MGFS file system.
- Add a new CIFS source file server.
- Add a new CIFS source file server share.
- Add a new NFS source file server export.
Site Preparation

- Delete a connection.
- Delete an MGFS file system.
- Start an internal migration thread (limit is one thread per connection).
- Stop a migration.
- Modify a CIFS source file server configuration.
- Modify an NFS source file server export.
- Extend the capacity of an MGFS file system.
- Convert a file system from type MGFS to UxFS.
- Download migration files, error logs, include files, and/or exclude files to your desktop.

For other tasks that are not provided by GUI functionality, use the instructions found in this guide.

**Running the GUI**

To run Data Migration from the GUI:

1. Create a network interface with an IP address identical to the source file server.
   
   “Step 1: Adding the Data Mover” on page 81 provides more information about the following procedure:
   
   a. From the **Network Interface** page, click **New**.
   b. From the **Network Route** page, create a network route.
   c. From the **Network Services** page, set up the DNS service.
   d. From the **CIFS server** page, set up CIFS if it is a CIFS migration, or NFS if it is an NFS migration.

2. Click **New** on the **New Migration File System** page to create a new MGFS file system.

3. Click **New** on the **New Migration** page to add a new migration.

4. Type the share or export path.

5. Click **Start** on the **Start Migration** page to begin the migration to the VNX.

6. Repeat steps 2 through 5 if it is a many-to-one merge migration.

7. From the **File Systems** page, select the MGFS to be converted to type UxFS.
8. Click Convert.

Accessing the GUI online help facility

The GUI’s "How to" Help facility of CDMS provides more information about these features. In addition, the Page Help facility gives you information about each tab or page element.

Determining file system size

An important preliminary step during site preparation is to identify the amount of data that you want to migrate, and determine file sizes to enable you to allocate enough disk space on the VNX’s backend storage system for these file systems.

Running the diskUsage.pl script

The diskUsage.pl script, as described in “diskUsage.pl script” on page 275, takes the VNX block size into account when calculating the overall space to be used on the backend disk storage system.

Note: When planning disk usage, note that the VNX uses 8 KB blocks. A migrated file occupies a full block even if it is smaller than 8 KB. If, for example, a file of 12 KB is migrated, it requires two 8 KB blocks or 16 KB on the VNX.

Extending the MGFS

If the migration file system was configured without enough space, it can be extended while file activity and migration are taking place, with little or no interruption to the process. Extending the file system eliminates the need to restart the server_cdms command from the beginning.

Note: You can increase the allotted storage for a selected MGFS file system by using the Data Migration feature of the Unisphere GUI, or the VNX nas_fs -xtend <fs_name> CLI command.

Site Preparation

Preliminary setup items

This section lists the preliminary setup items for NFS and CIFS data migrations to a Data Mover MGFS file system:

- An NFS or CIFS source file server containing the data to be migrated to the VNX.
  
  It can be any source file server that supports the NFS version 2 or 3 protocol. It can be a Windows NT 4.0, Windows 2000, or Windows Server 2003 file server, or even a VNX, as listed in Table 1 on page 20.

- A target (destination) VNX to which the data is to be migrated from the source file server.
  
  The VNX must contain one or more Data Movers that are running version 5.6 or later that can access a sufficient amount of backend disk storage to accommodate the file systems to be migrated.

- For certain assessment and associated premigration tools, a validated server acting as a UNIX workstation or Windows client.
  
  EMC recommends the UNIX workstation and Windows client be on the same LAN subnet as the migration Data Mover. You must ensure that the client is not run over a WAN connection to the VNX Data Mover.

  The Windows NT 4.0, Windows 2000, or Windows Server 2003 client should optimally match the source file server type, and must be in the same domain. As a best practice, the VNX, domain controllers, DNS service, and when necessary, a Windows Internet Naming Service (WINS) server should also be within the same campus network.

- Perl script version 5.6 or later installed on the UNIX workstation and/or Windows client.

- If desired, logging set up, as described in Appendix D, “Logging.”

- If migrating NFS protocol data from a Network Appliance source file server that makes use of the Network Appliance Snapshot feature, installing additional software on the Control Station, and performing further steps during the migration.

  Appendix E, “Network Appliance Considerations,” provides details about this procedure.

- If the source file server uses an international character set, the VNX set up, as described in “International character set support (NFS)” on page 51.
Note: Do not migrate multiple encodings into one file system. For example, SJIS and EUC encodings should not be migrated to one MGFS.

- Preliminary migration planning tools such as the lgdup.exe, sharedup.exe, and backupWrapper.exe utilities, and the dirprivW.pl, dirprivU.pl, diskUsage.pl, and/or connBuilder.pl scripts residing on the same client.

- The ch_group.pl script copied to the UNIX workstation. Ensure that this script is configured correctly, as described in Appendix C, “NFS Mount and Export Options.”

Win32 API must be installed on the Windows client.

“Adding Win32 API to Perl script (CIFS)” on page 55 provides more information about this procedure. Appendix A, “Using CDMS Migration Tools,” provides more information about migration scripts and utilities.

However, if you want to support migrations from multiple Windows servers, then you should use a separate client for premigration efforts. Microsoft’s Word and Excel 2000 must be installed on this client as well. The connBuilder.pl script uses Excel spreadsheet output as an intermediary tool in the process to create optional include and/or exclude files.

International character sets

This section contains instructions on how to configure CDMS for international character set support (NFS) and Unicode support (CIFS), and describes I18N functionality.

Using International Character Sets on VNX for File provides information about setting up and configuring the VNX for file for international characters.

Enabling support

To enable international character support for NFS migration, you must make a small modification to the xlt.cfg file.

When you edit the xlt.cfg file, you must include a line specifying the encoding scheme of the source file server.

Example

For example, if your source file server’s IP address is 128.221.252.103 and the source file server uses big5 encoding, you would add the following line:
**Site Preparation**

This source server uses big5 encoding

**Note:** You must configure the VNX for file for international characters before migration begins. If you determine international character support is required after migration starts, you must wait until the migration completes, convert the file system from type MGFS to UxFS, and then perform the tasks in *Using International Character Sets on VNX for File.*

---

**I18N support**

Internationalization (sometimes shortened to I18N, meaning "I - eighteen letters -N") is the process of planning and implementing products and services so they can easily be adapted to specific local languages and cultures, a process called localization. The internationalization process is sometimes called translation or localization *enablement.*

Enablement can include:

- Allowing space in user interfaces (for example, hardware labels, help pages, and online menus) for translation into languages requiring more characters.
- Developing with products (such as Web editors or authoring tools) that can support international character sets.
- Creating print or website graphic images so their text labels can be translated inexpensively.
- Using written examples that have global meaning.
- For software, ensuring data space so messages can be translated from languages with single-byte character codes (such as English) into languages requiring multiple-byte character codes (such as Japanese Kanji).

**Setup process**

For CDMS migration, you must specify the source file server's encoding scheme in the same way as you set up the I18N clients. Create all source file server file systems before you turn on I18N.

For example:

```
IP address::encoding scheme in /.etc_common/xlt.cfg
```

You might also need to let the source file server understand that you are indicating big5 by setting up the file server correctly.
CDMS is set up while I18N is on as a normal migration process and when the aforementioned change is made to the xlt.cfg file. You must ensure that both sides (source and target) turn on I18N, and that the clients can detect big5.

**Conversion process**

CDMS does not work with I18N conversion. You cannot turn on the I18N in the middle of a CDMS migration. You either wait for the migration to complete (after the CDMS conversion process), or turn on the I18N first, and then create the migration file system:

- If I18N is turned on after a CDMS file system conversion, the destination file system is UxFS, a standard conversion.
- If I18N is turned on before the CDMS file system creation, CDMS does the conversion while migrating data.

**Examples**

**Condition:** I18N is off on both sides, the source file server is big5, and expected destination is big5.

**Result:** I18N is not involved in this case; disk names are big5.

**Condition:** I18N is on both sides, the source file server is Unicode, and the expected destination is Unicode.

**Result:** I18N is not involved because CDMS does nothing for UTF8.

**Condition:** I18N is off on the source file server, and the source ID is big5. On the target I18N is on, and the destination is Unicode.

**Result:** In this case, the source is not specified as big5 in the xlt.cfg file. CDMS stores big5 filenames to the disk. Then, CDMS performs an I18N conversion to encode it to UTF8.

**Unicode support (CIFS)**

The Windows NT 4.0, Windows 2000, and Windows Server 2003 environments support Unicode standard for character set translation. To ensure that the Data Mover and file systems are consistent before and after migration, the Data Mover must be set to use Unicode before the migration starts.

*Using International Character Sets on VNX for File* provides more details about how to configure and enable international characters sets on a VNX.
Usermapper methods

As part of the Data Mover configuration for the CIFS environment, Usermapper is highly recommended for CDMS.

Purpose

Usermapper is a service that helps the VNX automatically map distinct Windows users and groups to distinct UNIX-style user IDs (UIDs) and group IDs (GIDs). Because the VNX uses UIDs and GIDs to identify users, Windows clients must be assigned UIDs and GIDs to enforce CIFS quotas.

Background

Usermapper uses UIDs and GIDs to map the users and groups for that Windows domain. Any Windows user not specified in the Usermapper databases cannot access files through the Data Mover unless the user is otherwise known to the Data Mover from other user mapping resources. When a user logs in to a Windows domain and requests access to a Data Mover’s resources, the following sequence of events occurs:

1. When logging in to a Windows NT domain or when accessing a Data Mover that was declared as a pre-Windows 2000 computer, the user is authenticated by using NTLM (NT LAN Manager). If the Data Mover is using a computer name and is joined to a Windows 2000 or Windows Server 2003 domain, the user is authenticated through Kerberos or NTLMSSP (NT LAN Manager secure-socket provider).

2. The user's identification is forwarded to the Data Mover.

3. The Data Mover searches the following sources for an existing mapping of the user's SID to a UID/GID:
   a. The Data Mover first checks its local resources (its local cache and then its local passwd and group files) for an existing SID to UID/GID mapping.
   b. If no mapping is found, and NIS is configured, the Windows domain controller is queried for the user or group name associated with the SID, and then NIS is queried for a UID/GID to associate with the name.
   c. If no mapping is found, and queries to the Active Directory are configured (in Windows 2000 and Windows Server 2003 environments), the Data Mover queries the Active Directory for a SID to UID/GID mapping.
   d. If no mapping is found, the Data Mover queries Usermapper for a SID to UID/GID mapping.
Configuring VNX User Mapping provides more details, including how to implement and manage Usermapper.

Adding Win32 API to Perl script (CIFS)

The Win32 API Perl script module adds the necessary functions to provide Windows backup privileges when using the `server_cdms` command.
Site Preparation

The steps listed in Table 5 on page 57 apply only to CDMS migrations.
Table 5  Adding Win32 API to Perl script (page 1 of 2)

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
</table>
| 1    | For Windows migration and preparation, the 5.6 Perl package should be installed together with the special `prototype.pm`. Install the special library: `Win32::API::Prototype`. The URL, [http://www.roth.net](http://www.roth.net), provides information about supported ActivePerl versions under Win32-API-Prototype. You can install the software and installation guide from this URL: [http://www.roth.net/perl/packages/](http://www.roth.net/perl/packages/). For example, using ActivePerl 5.6:  
   ```
c:\>ppm  
PPM interactive shell (2.1.5) - type `help' for available commands.  
PPM>set repository RothConsulting http://www.roth.net/perl/packages
PPM>set
```
| Note: The syntax could change per version of ActivePerl. |
| Note: Perl script version 5.8.0.805 does not support the extension from URL: [http://www.roth.net](http://www.roth.net) |

The following actions occur:  
- Commands are confirmed  
- Temporary files are deleted  
- Download status is updated every 16384 bytes  
- Case-insensitive searches are performed  
- Package installations continue if a dependency cannot be installed  
- Tracing information is not written  
- Windows pauses after 24 lines  
- Query/search results are verbose  

Current PPD Repository paths:  
- ActiveState Package Repository:  
- RothConsulting: [http://www.roth.net/perl/packages](http://www.roth.net/perl/packages)  

---

**Adding Win32 API to Perl script (CIFS)**  
57
### Site Preparation

2. Packages are built under: c:\temp
   
   PPM>install Win32-API-Prototype
   
   Install package 'Win32-API-Prototype?' (y/n): y
   
   Installing package 'Win32-API-Prototype'...
   Bytes transferred: 12461
   Installing c:\Perl\site\lib\auto\Win32\API\API.bs
   Installing c:\Perl\site\lib\auto\Win32\API\API.dll
   Installing c:\Perl\site\lib\auto\Win32\API\API.exp
   Installing c:\Perl\site\lib\auto\Win32\API\API.lib
   Installing c:\Perl\site\lib\Win32\API\API.html
   Installing c:\Perl\site\lib\Win32\API.pm
   Writing c:\Perl\site\lib\auto\Win32\API\Prototype.PM
   Writing c:\Perl\site\lib\auto\Win32\API\Prototype\packlist
   Bytes transferred: 2615
   Installing c:\Perl\site\lib\Win32\API\Prototype.pm
   Writing c:\Perl\site\lib\auto\Win32\API\Prototype\packlist
   PPM>quit
   Quit!
   c:\>

3. The standard prototype.pm file that comes with the special installation does not cover certain data types used by Win32 APIs and the server_cdms command. In other words, the command uses some standard Win32 APIs that have data types undefined in the standard prototype.pm file, which can cause an error. Therefore, you must replace it with the one that comes with the CD.

   Copy the prototype.pm file, located in the CDMS\cifs folder on the Applications Tools CD, to the following location:

   C:\Perl\site\lib\Win32\API\

4. Keep the backupWrapper.exe utility and all the Perl scripts in the same directory.

5. The backupWrapper.exe utility, run as part of three Perl scripts, adds backup operator privileges. Add a fully qualified path to the backupWrapper.exe utility.

   When you are instructed to execute script commands while completing the migration process steps later in this document, execute the scripts by using the following syntax:

   - backupWrapper dircount.pl <directory>
   - backupWrapper dirprivW.pl <directory>
   - backupWrapper diskUsage.pl -m

---

**Table 5** Adding Win32 API to Perl script (page 2 of 2)

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
</table>
| 2    | Packages are built under: c:\temp  
   
   PPM>install Win32-API-Prototype  
   
   Install package 'Win32-API-Prototype?' (y/n): y  
   
   Installing package 'Win32-API-Prototype'...  
   Bytes transferred: 12461  
   Installing c:\Perl\site\lib\auto\Win32\API\API.bs  
   Installing c:\Perl\site\lib\auto\Win32\API\API.dll  
   Installing c:\Perl\site\lib\auto\Win32\API\API.exp  
   Installing c:\Perl\site\lib\auto\Win32\API\API.lib  
   Installing c:\Perl\site\lib\Win32\API\API.html  
   Installing c:\Perl\site\lib\Win32\API.pm  
   Writing c:\Perl\site\lib\auto\Win32\API\Prototype.PM  
   Writing c:\Perl\site\lib\auto\Win32\API\Prototype\packlist  
   Bytes transferred: 2615  
   Installing c:\Perl\site\lib\Win32\API\Prototype.pm  
   Writing c:\Perl\site\lib\auto\Win32\API\Prototype\packlist  
   PPM>quit  
   Quit!  
   c:\> |
| 3    | The standard prototype.pm file that comes with the special installation does not cover certain data types used by Win32 APIs and the server_cdms command. In other words, the command uses some standard Win32 APIs that have data types undefined in the standard prototype.pm file, which can cause an error. Therefore, you must replace it with the one that comes with the CD.  
   
   Copy the prototype.pm file, located in the CDMS\cifs folder on the Applications Tools CD, to the following location:  
   
   C:\Perl\site\lib\Win32\API\ |
| 4    | Keep the backupWrapper.exe utility and all the Perl scripts in the same directory. |
| 5    | The backupWrapper.exe utility, run as part of three Perl scripts, adds backup operator privileges. Add a fully qualified path to the backupWrapper.exe utility.  
   
   When you are instructed to execute script commands while completing the migration process steps later in this document, execute the scripts by using the following syntax:  
   
   - backupWrapper dircount.pl <directory>  
   - backupWrapper dirprivW.pl <directory>  
   - backupWrapper diskUsage.pl -m |
Backup considerations

Since the migration process might take a considerable amount of time, and user data might be updated on the VNX during this process, implementing a proven backup strategy is essential for this time period.

Configuring NDMP Backups on VNX and Configuring NDMP Backups to Disk on VNX provide information about VNX-qualified backup options and procedures.

During migration, all changes accumulate on the VNX. This process makes incremental backups crucial because if the migration fails or if the source file server crashes, migration must be started again.

Small data migration
If a small amount of migrated data is changed during migration and you expect migration to take less than a day, you might not need to make a full backup. Since data ultimately resides in two places (on the source file server and the VNX), you might consider the source file server as your backup location.

Large data migration
If you expect a lengthy migration process, consider a more robust backup strategy. For environments migrating a significant amount of data, EMC recommends the following backup steps:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>After you switch to restricting network access to the source file server, begin making a full backup of the server. After the connection is established between the MGFS and the source file server, updates are only done on the VNX. While the full backup is underway, migration can continue.</td>
</tr>
<tr>
<td>2</td>
<td>After the full backup of the source file server completes, you can begin taking incremental backups on the VNX.</td>
</tr>
</tbody>
</table>

Note: The first incremental backup might take longer than subsequent backups because it impacts every directory, causing each directory to be migrated.

Before beginning the procedure, be sure you have implemented a backup strategy. This process might take a considerable amount of time to complete, and data updates are made during the migration.
Site Preparation
CHAPTER 3
Migration Process Phases

The following topics provide information about the three phases of a CDMS migration as well as required job qualifications of migration personnel:

- Migration process phases .......................................................................................... 62
- Personnel qualifications ............................................................................................ 62
Migration process phases

There are three phases to most large migration efforts involving CDMS:

1. Planning and Design — Details the migration effort and identifies potential network issues associated with the migration.
   
   Chapter 4, “Planning and Design,” provides details.

2. Implementation — Executes the plan developed during the planning and design phase for one-to-one or many-to-one migrations.
   
   Chapter 5, “NFS,” and Chapter 6, “CIFS,” provide details.

3. Postmigration Testing — Verifies the migration went according to plan. Included in the respective Implementation chapters.

All three phases might employ the skills of one or more individuals. The roles and tasks might be combined or divided, as required by the complexity of the environment and the availability of resources.

Personnel qualifications

This section describes the job qualifications needed for migration specialist and network analysis personnel.

Migration specialist

Particular attention should be paid to the design and implementation skills of the migration personnel. In addition to a thorough understanding of the VNX hardware/software environment, these individuals should have one to two years of experience with UNIX workstations, Windows clients, and some knowledge of the Linux operating system running on the Control Station. The greater the level of experience with these operating environments, the more likely the success of the migration project.

Since the process of qualifying, planning, designing, and migrating file systems to the VNX requires the talents of a number of people, the skills of the network consultant and the VNX design analyst are key to the migration.
Secondarily to the migration personnel are those individuals involved in the analysis of the network environment and associated tasks. These individuals are key to any VNX data migration. The network analysis tasks demand quality results. Therefore, this staff should not be pooled under the auspices of an individual performing other tasks within the project.

The project might require the skills of a network administrator and a system administrator. You might also find it necessary to have application specialists available to consult on any unusual network interface needs of client-based applications.
Migration Process Phases
The following topics supply answers to tactical and frequently asked questions, discuss the two main elements in the planning and design phase of a migration, and describe the include and exclude files:

- Operational issues and frequently asked questions....................... 66
- Planning and design........................................................................... 69
- Optional files..................................................................................... 76
Operational issues and frequently asked questions

This section contains CDMS operational issue information and answers to frequently asked questions you should understand prior to beginning an NFS or CIFS data migration.

Will UNIX workstations be able to access the migrated data transparently without configuration changes?

CDMS minimizes the impact to clients. This assumes the IP address can be retrieved from the NFS source file server and a private (or dedicated) network is being used for migration. If this is not the case, there is a strong probability of disrupting the client interaction with the file serving environment. Disruptions can include file system mount point changes, fstab or vfstab modifications, application file pointers, network connectivity, IP address changes, and perhaps many other subtle issues could appear during and after the migration. You must be aware of these types of disruptions, which without preplanning, numerous failures can be expected of the environment. You should investigate the ramifications of migrating to a new file server.

Will Windows clients be able to access the migrated data transparently without configuration changes?

Depending on the chosen migration strategy, the CIFS migration has a moderate to significant impact on clients. In a one-to-one strategy, the impact to Windows clients is a limited outage while the CIFS source file server is renamed and a VNX assumes its identity. In a many-to-one merge strategy, it might be necessary to remap all share connections from a client. This task might be automated through the Sharedup utility, or might involve changing login scripts, desktop icons, and potentially some URLs. Ensure that you are fully aware of these interruptions and necessary client modifications.

What type of information should be provided by a network evaluation?

A network evaluation provides point-to-point migration path information concerning available bandwidth, routing complexities, and authentication and resolution services (in other words, NIS and DNS servers and WINS, DNS service, and PDC/BDC/DC servers). Evaluation information provided by this assessment is a prerequisite to VNX migrations that use any methodology. Many of the decisions concerning how the migration is planned, designed, and implemented are predicated upon network capabilities. For small migrations, a comprehensive network analysis might not be necessary. However, you must review the network as VNX migrations require suitable networks and might require significant network bandwidth.
Can parallel migration streams be initiated?

Starting multiple, simultaneous, migration processes per Data Mover can significantly reduce migration time. However, the number of parallel migrations that can occur over a single 100 Mb/s network segment should be limited to three.

In addition, in CIFS migrations with CDMS, it is very likely there will be multiple connections for each server. Each connection represents another migration process (or thread) and thus produces parallel migration streams and an increased load to the network.

Is there a priority to the migration of exports and shares in this effort?

User application or operational requirements might dictate that certain exports or shares be migrated prior to others. The gathered information from the analysis of the source file server-to-Data Mover mapping helps to track this sequence of importance. As part of the migration process, you also need to identify high-priority files that need to be migrated early in the migration cycle (in other words, when access to the servers is offline), or very large files that need a significant amount of migration time.

Generally, files that are larger than 100 MB should be considered high priority. In addition, large Outlook.pst files and access databases (.mdb files) should be evaluated for potential high-priority status.

Are there periods of inactivity available for migration? What is the production schedule?

To accomplish a complete and orderly migration, it might be necessary to perform migration activities during nonpeak periods of the processing day. CDMS requires some downtime on each server involved in the migration engagement. You should understand production schedule requirements. These needs should be balanced with the overall migration timeline.

How will network connections be configured between the source file server and the target Data Mover?

This effort is one of the more difficult challenges in the configuration of CDMS. The easiest configuration would be one in which the Data Mover takes over the IP address of the source file server and in essence clones its personality. All client activity would be directed to the Data Mover. This is only possible when all source file servers are being migrated, and the source file server no longer retains its current functionality. At least for some of the time, this is not the case. Therefore, alternative plans might need to be considered. Remember that network changes of any kind might require restart of the source file server.
Are there time constraints that need to be managed? Can a staged migration be implemented?

Depending on the size and number of servers, exports, and shares that need to be migrated, CDMS usage can be a very lengthy process. In a large, complex environment involving hundreds of servers and thousands of files, a complete migration effort might take several weeks, or more of virtually round-the-clock transfers. Therefore, it is better to assume things will take longer than to expect a quick migration effort. You should allow extra time for unexpected challenges.

Is VNX HighRoad (MPFS) a consideration for a VNX implementation?

VNX HighRoad should be configured after all migration has been completed.

Should a backup solution be implemented on the VNX?

A backup solution should be implemented on the VNX prior to data migration to avoid any delta data loss.

Can nightly backups be run while data is being migrated?

You should run a full backup of the source file server before beginning the migration process. After migration begins, you can perform incremental (or nightly) backups on the VNX according to the regular backup schedule. The first incremental backup might take a little extra time because it impacts every directory, thus causing it to be migrated. Backups trigger the migration of all data. The backup process can be slow if all data is not migrated, however.

Does running the migration command cause slower response times?

Yes, until more data is migrated. The first time data is accessed (causing it to be migrated), response time is slower. After the data is migrated, response time improves significantly.

How can you check where you are in the process?

You can use the `server_cdms <movename> -info <mgfs>` command to check progress. The command shows connection and thread status.

Do files get corrupted if the source file server or the network goes down during the migration?

User applications receive an I/O error, but the data is not corrupted. When you rerun the command, the migration begins where it was when the network or file server went down.
Note: If a source file server is corrupted badly, it might introduce a migration problem, preventing certain files and/or directories from being read and keeping them offline. In this case, you might have to bring those files and/or directories online manually by using the `specialCmd getOnline` command, and then restore the data by using another method such as from a tape backup. Alternatively, you can identify the problematic files and/or directories by using the procedure described in “Managing a failed verification or conversion” on page 229, and then remove these files and/or directories.

What do I do if the conversion command returns an error?

Investigate why the error occurred, fix the problem, and rerun the `server_cdms <movername> -Convert <mgfs>` command again, if necessary. “Managing a failed verification or conversion” on page 229 provides more information if the conversion does not complete successfully. If a second conversion does not run error free, you should follow the normal EMC Customer Service procedures to escalate the issue.

What happens to my symbolic links and hardlinks during migration?

Symbolic and hardlinks are all migrated and preserved. No validation checking is performed on these links, however, so that absolute (as opposed to relative) links might not work afterwards if the mount points are changed by the migration.

Relative links within the file system always work after migration.

Can I dial in from a remote location to check the status of the migration?

Use the `server_cdms <movername> -info <mgfs>` command to display the status of a migration. “Step 8: Verifying migration completion” on page 105 provides an example of the data that should appear.

Is it necessary to perform the migration effort offline?

Although CDMS is generally considered an online migration tool, it can be used in a totally offline capacity. This might be one of the options the migration specialist uses in designing the total migration effort.

Planning and design

This section contains NFS and CIFS planning and design recommendations for CDMS data migrations.
The planning and design phase is a very important step in any successful migration project. Detailed analysis and process planning provide you with the foundation for a timely implementation.

There are two major elements to planning and design:
- Assessment
- Analysis and detailed planning

**Assessment**

This section requests that you:
- Summarize the installed hardware
- Evaluate any migration issues

**Summarize installed hardware**

To understand the state of the currently installed hardware, perform the following:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Document the VNX disk storage systems involved in the migration.</td>
</tr>
</tbody>
</table>
| 2    | Review:
  - Current server storage configuration
  - Data layout
  - Content of source file servers, exported file systems, and/or shares to be migrated to the VNXData Mover.
  Obtain a full report on file systems and shares to be migrated. |
| 3    | Review user application access to targeted disk volumes on the NFS or CIFS source file servers, and determine any specific tasks required for migration. There might be extensive issues associated with application use of the data. This might involve migrating specific files and directories first in the migration cycle. Ensure that you understand the details so accommodations can be made accordingly. |
### Server Readiness:

Identify the NFS and/or CIFS source file servers and network paths to be evaluated before you begin the VNX data migration. Gather the necessary information:

- Identify the names, locations, and so forth of the original servers from which data will be migrated.
- Document the server names, location, and operating system of the original servers/filers from which data will be migrated, including details of their network connections. Ensure to include each IP address, speed, duplex mode, and so on.
- Determine if there is a priority to the migration effort for these server/filers.
- Determine if there is a priority for particular files within any file system or share. For example, since the file system is accessible during migration, if one of the files within it was an index to other file's contents, it might provide better performance if that file is migrated first, especially if it was a large file.
- Identify the NFS and MOUNT versions supported on these servers, if appropriate. If necessary, use the `rpcinfo` command to obtain this information.
- Use the `dirprivU.pl` or `dirprivW.pl` script to check each file system, and determine if any files cannot be read and/or might not be readable during the data migration. Also identify the solution to remedy any file problems. Some possible choices include changing the source file server’s export parameters, changing the MGFS mount parameters, or individual `chmod` or `chown` commands of the files. Appendix A, “Using CDMS Migration Tools,” provides more information about migration scripts and utilities.

**Note:** If there are any `Client cannot` error messages reported by the `dirprivU.pl` or `dirprivW.pl` script, the results of the `diskUsage.pl` script are incorrect by the size of those files or the contents of those directories, unless those errors are fixed first.

- Determine the destination size for each designated migration file system by using the `diskUsage.pl` script with hardlink size excluded. Record the file size distribution in each file system (at the end of the `diskUsage.pl` script output) to use to estimate migration times. Appendix B, “Estimating NFS Data Migration Times,” and Appendix E, “Network Appliance Considerations,” provide details.
- Check each file system’s structure by using the `dircount.pl` script, and determine if it is possible or desirable to move several parts in parallel.
- Verify if hard- or soft-link files are relative or absolute. If some are absolute, and you plan to change the file system or server name as part of the migration, determine what to do about the absolute links.
- Obtain the original export parameters from the source file server for each file system. You need to duplicate the access control from them on the `server_export` command.
- Determine if the original NFS or CIFS source file servers can be backed up and restored before data migration.
- Understand the test to prove the servers are restored correctly.
- Determine whether the NFS or CIFS source file servers contain files streaming data, or need to be accessed in a latency-jitter-free manner during the migration (a migration effort might affect the network streaming capability).

*Server Readiness:* continues on the following page.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
</table>
| 4    | **Server Readiness:** Identify the NFS and/or CIFS source file servers and network paths to be evaluated before you begin the VNX data migration. Gather the necessary information:  
- Identify the names, locations, and so forth of the original servers from which data will be migrated.  
- Document the server names, location, and operating system of the original servers/filers from which data will be migrated, including details of their network connections. Ensure to include each IP address, speed, duplex mode, and so on.  
- Determine if there is a priority to the migration effort for these server/filers.  
- Determine if there is a priority for particular files within any file system or share. For example, since the file system is accessible during migration, if one of the files within it was an index to other file's contents, it might provide better performance if that file is migrated first, especially if it was a large file.  
- Identify the NFS and MOUNT versions supported on these servers, if appropriate. If necessary, use the `rpcinfo` command to obtain this information.  
- Use the `dirprivU.pl` or `dirprivW.pl` script to check each file system, and determine if any files cannot be read and/or might not be readable during the data migration. Also identify the solution to remedy any file problems. Some possible choices include changing the source file server’s export parameters, changing the MGFS mount parameters, or individual `chmod` or `chown` commands of the files. Appendix A, “Using CDMS Migration Tools,” provides more information about migration scripts and utilities.  
  **Note:** If there are any `Client cannot` error messages reported by the `dirprivU.pl` or `dirprivW.pl` script, the results of the `diskUsage.pl` script are incorrect by the size of those files or the contents of those directories, unless those errors are fixed first.  
- Determine the destination size for each designated migration file system by using the `diskUsage.pl` script with hardlink size excluded. Record the file size distribution in each file system (at the end of the `diskUsage.pl` script output) to use to estimate migration times. Appendix B, “Estimating NFS Data Migration Times,” and Appendix E, “Network Appliance Considerations,” provide details.  
- Check each file system’s structure by using the `dircount.pl` script, and determine if it is possible or desirable to move several parts in parallel.  
- Verify if hard- or soft-link files are relative or absolute. If some are absolute, and you plan to change the file system or server name as part of the migration, determine what to do about the absolute links.  
- Obtain the original export parameters from the source file server for each file system. You need to duplicate the access control from them on the `server_export` command.  
- Determine if the original NFS or CIFS source file servers can be backed up and restored before data migration.  
- Understand the test to prove the servers are restored correctly.  
- Determine whether the NFS or CIFS source file servers contain files streaming data, or need to be accessed in a latency-jitter-free manner during the migration (a migration effort might affect the network streaming capability).  
  *Server Readiness:* continues on the following page. |
### Server Readiness: continued:
- Decide whether you plan to migrate the contents of several NFS or CIFS source file servers to one VNX volume/Data Mover combination. Understand whether you plan to merge one or more file systems into a single file system as these decisions affect traffic estimates.
- Identify the protocols currently used to share data on the original source file server, and the protocols that cannot be used on the migrated data on the VNX.

A VNX supports NFS, CIFS, and FTP over TCP/IP. IPX (even if encapsulated in IP) cannot be used on the VNX. HTTP and Java script might access data on the original servers. The VNX does not use either of these protocols.
- Inform the UNIX workstation or Windows client support staff of deficiencies in server readiness, or recommended modifications to the current system configuration.

### Network Evaluation:
For small migration engagements, the level of planning required to migrate data from NFS and CIFS source file servers to a VNX is likely modest, but still a necessary part of the process. At a minimum, the following tasks must be accomplished:
- Analyze disk usage
- Analyze file systems or shares
- Configure VNX storage
- Create a schedule for the migration
- Estimate the migration time
- Map source and destination file systems
- Review the basic network

However, on migrations of more than 20 file systems, a significant effort is required in planning and designing the migration. For large engagements, a detailed project plan might be required as a roadmap.

The network must be able to support the migration effort effectively. The success or failure of the project depends specifically on this functionality. The results of the network analysis might dictate that temporary or permanent changes to the network are required to accomplish the migration. Be prepared to adjust the migration schedule to accommodate the time necessary to make these changes.
Evaluate migration issues

To outline areas to consider with regard to migration issues, perform the following:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Document future plans for the VNX storage configuration. The migration analyst must understand growth potential regarding VNX requirements. At a minimum, obtain growth patterns over the last several months on each of the identified file systems. Take that information and increase the value by a minimum of 20%.</td>
</tr>
<tr>
<td>2</td>
<td>If appropriate, review and document the Windows domain model for trust relationships from the CIFS source file server’s domain to other domains in the environment. For every domain with a trust to the source file server, ensure that there is a domain entry in Usermapper.</td>
</tr>
<tr>
<td>3</td>
<td>Identify any additional hardware required to achieve the migration. Additional hardware could be limited to a UNIX workstation or Windows client that acts as an NFS or domain client in the network, or might include adding NICs to each server for the purposes of providing a private network. Additional switches and other network hardware might be necessary to accomplish the migration in a timely manner.</td>
</tr>
<tr>
<td>4</td>
<td>Consider that CDMS requires that a Data Mover mount an NFS or CIFS source file server to permit the copying of files to the MGFS. Although the source file servers must be shared or exported to permit this, they might be shared with permissions limited to allow only the destination Data Mover to access the source, and deny any other means of access which could modify the source during the migration.</td>
</tr>
<tr>
<td>5</td>
<td>Determine whether any of the NFS source file servers are NetApp servers. If NetApp servers are present, migration from the server requires additional migration steps. Appendix E, “Network Appliance Considerations,” provides further information about the steps required for this particular environment. This release of CDMS does not support migration of CIFS data from NetApp servers.</td>
</tr>
<tr>
<td>6</td>
<td>Identify any additional software or software upgrades required to achieve the migration. If the target VNX is not running version 5.6 or later, you must install a software upgrade.</td>
</tr>
<tr>
<td>7</td>
<td>Define any other constraints pertinent to the migration effort.</td>
</tr>
</tbody>
</table>

**Note:** Even if the Data Mover is configured for Virus Checker, the data migration of CDMS with the `server_cdms` command script does not trigger Virus Scan as long as scan on read is turned off on the Data Mover. If scan on read is turned on, the CDMS data migration does trigger Virus Scan. In this case, the performance impact on the Virus Checker software and the Data Mover must be considered.
To outline actions for analysis and planning of the data migration, perform the following:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
</table>
| 1    | Determine the required configuration for VNX disk storage. Base these calculations on:  
• File system  
• Sizing information  
• Performance requirements  
• Expected storage growth patterns as defined by the client  
Information about how to arrive at these estimates is in “Determining file system size” on page 49. |
| 2    | Map the NFS or CIFS source file server to the destination Data Mover and its associated migration file system.  
This setup is critical to the implementation phase of the project. |
| 3    | Review the results of the network evaluation:  
• Assess the impact of network usage, based on the ability to perform the migration in a timely manner.  
• Identify areas of concern.  
• Evaluate how recommended changes from the network evaluation might alter the schedule.  
• Look at the physical and logical (IP address and compname) impact of the Data Mover within the domain and name assignments on client accessibility. |
| 4    | Based on information from the network evaluation, determine whether any enhancements to the network are required to enable the migration effort, including:  
• Enabling a private network for migration transfers only  
• Adding trunking options  
• Upgrading the VNX network interface  
• Upgrading other portions of the internal network infrastructure |
| 5    | Define the limits of the migration effort, including:  
• Number and names of servers involved in the migration  
• Server locations  
• Size of the source file servers  
Understand the number of file systems to be migrated to the Data Mover. |
| 6    | Determine the methodology for migration.  
CDMS might not be able to accomplish all migration requirements. You might need to consider tape, third-party products, or some other option as the case demands. |
Develop a viable strategy and plan for the data migration as this is the most important step in the migration effort. During the design and planning process:

- Determine minimum network usage factors permitted to execute the migration successfully.
  
  A network more than 30% busy might prove to be nearly unusable for migration.

- Determine whether parallel migration streams can be initiated.
  
  Most of the time, if the network allows, processing multiple migration processes on a single Data Mover can be accomplished. On a relatively quiet 100 Mb/s network, you should be able to process three migration streams. Remember, however, that there might be other file systems that can be processed on other Data Movers, or for that matter, other ports on the same Data Mover. The limiting factor is the network.
  
  For a single file system, you might be able to migrate multiple, directory, tree segments within the same file system concurrently. To accomplish directory parallelism, select a subdirectory, give the command the full path to the directory, and start the migration.

- Determine the migration strategies that you want to use for all servers being migrated.
  
  It is likely one-to-one and many-to-one migration strategies are required to fully accomplish the migration. Document the strategy prior to beginning the implementation effort.

- Consider how IP addresses are to be assigned to the Data Movers.
  
  “Step 1: Adding the Data Mover” on page 81 suggests the easiest configuration is to have the Data Mover assume the IP address of the source file server. This is possible when the source file server functions only as a file server, does not provide any other user application support, and does not require access by clients to additional file systems while the migration is in progress. The limitations of this recommendation suggest that most of the time the Data Mover will be assigned a new or different IP address.

  Note: The result of assigning the Data Mover a new address is that all UNIX workstations need to unmount the old file system, mount the new file system with the Data Mover IP address, and ensure that they permanently update the file system tables (fstab or vfstab) to reflect the new location of the NFS data.

- Identify the order in which the file systems should be processed.
  
  Your applications or client needs might determine file system order. If parallel migration streams are to be implemented, the order might become more complex.

- Ensure that the migration schedule takes into account production schedules, periods of downtime or minimal activity, and maintenance windows.
  
  Since the migration of a large file system or share with many small files (1 KB or less) might take several days to complete, there might be periods that do not require onsite monitoring. You need to take into account wait time for migrations to complete.

- Consider intermediate checkpoints in large migration efforts.
  
  These stages provide an opportunity to confirm progress and demonstrate ongoing success.

- Provide for contingency efforts as there might be unforeseen issues that cause the migration to take longer, or to vary from the projected schedule.
  
  Allow for some contingency time in the plan.

- Provide time for client changes, outages, or other interruptions that are necessary to activate UNIX workstation or Windows client use of the VNX.
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Refine the resources and associated costs to perform the migration.</td>
</tr>
<tr>
<td>9</td>
<td>Determine any unusual tasks required to complete the migration. An example might include specific database files that remain locked and require special handling when copied or moved.</td>
</tr>
<tr>
<td>10</td>
<td>Estimate migration time scales, and compare with the available migration window. Information obtained from the network evaluation diskUsage.pl script is invaluable in helping to determine how long the migration of a file system might take. If appropriate, use the <code>backupWrapper.exe</code> utility to give the diskUsage.pl script the necessary backup operator privileges during execution. The diskUsage.pl script provides file system sizing and file size distribution information. However, there are no hard and fast rules. Remember the network ultimately enhances or tempers migration performance. Ensure that you integrate the process into the overall timeline, if you can only migrate data during certain times of the day (for example, 2 a.m. to 6 a.m.).</td>
</tr>
<tr>
<td>11</td>
<td>Determine any server overhead caused by the <code>server_cdms</code> migration command. It is possible to have the source file server and the client on the same computer, however. In this case, you might want to monitor the impact of the migration command on the source file server.</td>
</tr>
<tr>
<td>12</td>
<td>Assess potential risks, and determine risk mitigation strategies. This assessment should identify fallback plans in the event of catastrophic hardware failures, natural disasters, or other unforeseen events.</td>
</tr>
<tr>
<td>13</td>
<td>Define plans for migration verification testing.</td>
</tr>
</tbody>
</table>

To complete the file system migration process, proceed to:
- Chapter 5, “NFS”
- Chapter 6, “CIFS”

**Optional files**

The `server_cdms -start` command, enabling all untouched data to be migrated from the source file server to the Data Mover, can include and/or exclude the migration of certain files.

**The include file**

An include file identifies any high-priority files on the source file server. When an include file is used with the `server_cdms -start` option, only the named files (and any directory structure needed to correctly locate the files) are migrated to the destination. All other files are left on the source.
This list is usually very short, and often contains references to all files with specific extensions. Of particular importance in Windows migrations are OutLook PST (.pst) files and access database (.mdb) files. If it is used, it is normal to have one include file for each disk drive associated with the source file server.

### The exclude file

An exclude file identifies files and directories that should not be migrated from the source file server. If they exist, entries are removed on the MGFS. If files matching the entries in the exclude file have not been migrated yet, the directory tree is migrated to the point where the file entries are found, and they are then removed. All other files are not migrated, and are left on the source.

Although all data from a disk drive on a Windows source file server can be migrated, some elements do not need to be migrated. These elements typically include directories such as the WINNT directory, the system volume information directory, and the recycler (wastebasket directory). You can list files and directories that should not be migrated in the exclude file.
The files can be created by using any text editor whose output is recognized as a text file by the Data Mover. The file format is the same for files:

- Use f to indicate an entry is a file; use d to indicate an entry is a directory. For example:
  
  f subtree_long/* .txt

- You can use a full path or path relative to the thread start path to represent files and directories. The path (if used) is interpreted as being on the Data Mover. In the case where the Data Mover mount point is named differently from the source, the path specified in the include file should be the Data Mover version, and not the source one. Otherwise, the file to be included (or excluded) might not be found.

- # indicates a comment.

- For an exclude file, all matching entries are removed from the migration file system.

- For an include file, only file entries can be added to the include file. If -include is specified, only the matching entries will be migrated.

Include and exclude files can be used separately or together. If include and exclude options are specified for the same migration thread, it will have the following effect:

- All matching include entries are migrated.

- All matching exclude entries are removed.

- All other entries remain offline.

The naming convention for include and exclude files should be the same (for example, include-server1-C.txt or include.txt), and they should reside in the same directory of a file system mounted on <movername>. An example is as follows:

```
$ server_cdms server_2 -start pmacF30 -path /dest1 -log /mgfslog -exclude /amounted_fs/exclude.txt
```

**Note:** You can download include and exclude files by using the Unisphere Data Migration GUI.
The following topics provide step-by-step procedures about how to perform NFS file system migrations and merge multiple file systems:

- **Introduction** ................................................................. 80
- **Summary** ........................................................................... 80
- **One-to-one migration** ..................................................... 81
- **Many-to-one migration** .................................................... 111
- **Correcting GIDs (optional)** ........................................... 117
- **Postmigration testing** ...................................................... 118
Introduction

At this point in the overall NFS migration effort, the planning phase is complete. You should have read Chapter 4, “Planning and Design.” Now it is time to execute the plan.

All mapping from NFS source file servers to target VNX, choices for migration strategies, storage design, and network evaluation have been completed and documented thoroughly.

From this point forward, the migration proceeds as designated in the plan.

Note: CDMS supports NFSv2 and NFSv3 only.

Summary

Table 6 on page 80 summarizes the steps to implement NFS source file server migration.

Table 6  One-to-one migration summary

<table>
<thead>
<tr>
<th>Step</th>
<th>Description and location</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>“Step 1: Adding the Data Mover” on page 81</td>
</tr>
<tr>
<td>2</td>
<td>“Step 2: Preparing file systems for migration” on page 84</td>
</tr>
<tr>
<td>3</td>
<td>“Step 3: Preparing the source file server” on page 89</td>
</tr>
<tr>
<td>4</td>
<td>“Step 4: Starting the CDMS migration service” on page 90</td>
</tr>
<tr>
<td>5</td>
<td>“Step 5: Connecting the file systems” on page 90</td>
</tr>
<tr>
<td>6</td>
<td>“Step 6: Ensuring that all data is migrated” on page 98</td>
</tr>
<tr>
<td>7</td>
<td>“Step 7: Examining the error log” on page 104</td>
</tr>
<tr>
<td>8</td>
<td>“Step 8: Verifying migration completion” on page 105</td>
</tr>
<tr>
<td>9</td>
<td>“Step 9: Converting the migration file system” on page 109</td>
</tr>
</tbody>
</table>

“Many-to-one migration” on page 111 explains how to merge multiple file systems into a single file system on the Data Mover.
One-to-one migration

This section contains step-by-step instructions for migrating NFS file systems from supported clients to a VNX Data Mover.

As a best practice, EMC recommends that the Data Mover assumes the IP address of the original NFS source file server. This strategy avoids problems and complexities in setting up the migration.

However, this strategy might not always be possible. For example, the VNX and the original NFS source file server might be in different subnets of the network, or the file server might need to be retained for other purposes. Then the actions in the following steps might vary somewhat.

Step 1: Adding the Data Mover

If the Data Mover assumes the IP address and hostname of the original server/filer, and the original server/filer does not need to be available on the network for other purposes, you do not have to modify the hosts files, DNS service, NIS, nor client mount points.

Note: If there are multiple IP addresses associated with one hostname in the configuration, the server_cdms <movername> -connect <mgfs> command should use the actual IP address instead of the hostname.

To add the Data Mover to the network, perform the following:

1. If the original server/filer continues to use its original IP address and hostname, and the Data Mover takes a new hostname and IP address on the network, you need to do the following:
   a. Create new names and IP addresses for the Data Mover.
   b. Type the new names and IP addresses in the hosts files, the DNS service, and the appropriate sections of NIS, as required.
   c. Ensure that the UNIX workstations that need to use the Data Mover, as the old server/filer was used, are updated or reconfigured to mount the new name/address.

   Note: If you use an automount server, this task is much simpler.
2. If the Data Mover assumes the IP address and hostname of the original server/filer, but the original server/filer must be available on the network for other purposes, perform the following:

---

**Note:** You can add a new migration by using the Unisphere Data Migration GUI.

---

a. Create a new name and IP address for the old server/filer.

b. Type these new names and IP addresses in the hosts files, the DNS service, and the appropriate sections of NIS, as required.

c. Ensure that the UNIX workstations and user applications on those workstations that need to continue to use the old server are updated or reconfigured to mount the new name/address.

---

**Note:** If you use an automount server, this task is much simpler.

There might be other instances, such as taking over either the hostname or IP address, but not both, or the Data Mover replacing multiple servers with hostnames in different domains, and so forth. These situations would need some combination of name or address changes, customized for the specific circumstances.

3. Before the migration starts, you should attach the Data Mover's network ports to the physical network, if possible, with the agreed IP configuration, and test connectivity with the `server_ping` command.

If the network name or address changes, and connectivity can only be done at the start of the migration, you might need a zone update on DNS before changes are known to all systems in a network. Also, Address Resolution Protocol (ARP) tables (containing IP-address-to-MAC address mappings on a LAN) normally take several minutes to age out. Therefore, the first `server_ping` commands should be sent from ports where the address has changed to ensure that the correct, new MAC addresses are updated in network switches, servers, and so on.
4. Whenever the Data Mover and NFS source file server are attached and IP-addressed, you should verify communication by issuing a server_ping command between the Data Mover and the source file server, and from the server to the Data Mover port, as described in the following table.

**Note:** If DNS service is configured on the Data Mover, you might ping by name instead of by address.

Similarly, check the ability to communicate between one of the UNIX workstations (or client) and the Data Mover.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To verify communication between the source and target servers, use this command syntax:</td>
</tr>
<tr>
<td>$ server_ping {&lt;movername&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = name of the Data Mover</td>
</tr>
<tr>
<td>&lt;interface&gt; = send an ECHO_REQUEST message to a specific host through a specific port to a specified &lt;hostname&gt; or &lt;ip_addr&gt; for a remote host</td>
</tr>
<tr>
<td>&lt;hostname&gt; = name of the remote host being pinged</td>
</tr>
<tr>
<td>&lt;ip_addr&gt; = NFS source file server IP address being pinged</td>
</tr>
<tr>
<td>The ALL option executes the command for all of the Data Movers.</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To verify communications between the server_2 Data Mover and IP address 172.24.67.54, type:</td>
</tr>
<tr>
<td>$ server_ping server_2 172.24.67.54</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 : 172.24.67.54 is alive, time= 0 ms</td>
<td>• server_2 is the name of the Data Mover from which you are pinging.</td>
</tr>
<tr>
<td></td>
<td>• 172.24.67.54 is the IP address of the NFS source file server.</td>
</tr>
</tbody>
</table>
To prepare the VNX to receive migrated data from the source NFS server, perform the following:

Appendix E, “Network Appliance Considerations,” provides information about migrating data from a NetApp file server by using the Network Appliance Snapshot feature.

1. Log in as nasadmin.

2. Use the output of the diskUsage.pl script to determine the minimum size of the MGFS to be created on the Data Mover.

   The final size might be much larger than this size, depending on the future plans.

3. Select one or more disks on the VNX backend where you can create a metavolume.

   For performance reasons, create the MGFS on a striped volume. The choice of location and selection of disk volumes should be preplanned in larger migrations.

   Managing Volumes and File Systems with VNX Automatic Volume Management and Managing Volumes and File Systems for VNX Manually provide more information about creating volumes, file systems, and mount points.

   To be created, the metavolume must be at least as large as the MGFS file system. It can be extended later if this size proves to be inadequate.

   **Note:** You can extend an MGFS by using the Unisphere Data Migration GUI.
### Action

To create a metavolume, use this command syntax:

```
$ nas_volume [-name <name>] -create [-Stripe [stripe_size]] -Meta [volume_name,...]
```

where:
- `<name>` = assigns a volume name. If a name is not specified, one is assigned automatically. The name of a volume is case-sensitive.
- `<stripe_size>` | `Meta` = sets the type for the volume to be either a stripe volume or metavolume (default). If the `-Stripe` option is specified, a stripe depth must be typed in multiples of 8192 bytes. The recommended size is 32,768 bytes (default).
- `<volume_name>` = metavolume specified by name.
- `<GB>` = if size is specified (between 1 and 1024 GB), a volume match is found equal to or greater than the specified size.

**Example:**

To create a metavolume named `mtv1d30` (on which you later create an MGFS) on the `d30` disk drive, type:

```
$ nas_volume -name mtv1d30 -create -Meta d30
```

### Output

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>acl</th>
<th>in_use</th>
<th>type</th>
<th>volume_set</th>
<th>disks</th>
<th>clnt_filesys</th>
</tr>
</thead>
<tbody>
<tr>
<td>122</td>
<td>mtv1d30</td>
<td>0</td>
<td>True</td>
<td>meta</td>
<td>d30</td>
<td>d30</td>
<td>pmacF30</td>
</tr>
</tbody>
</table>

**Note**

- The server displays information on the metavolume you created.
- `mtv1d30` is the metavolume name you created for the MGFS. This is a name of your choice.
- `d30` is the disk where the metavolume is being created. Select enough disks to accommodate the file system being migrated. Be sure to consider the file system layout (striping).

A delay occurs while the MGFS is being created.

**Note:** You can view a current list of migration file systems, and create an MGFS by using the Unisphere Data Migration GUI.
**Action**

To create an MGFS, use this command syntax:

$ nas_fs [-name <name>] [-type mgfs] -create

{<volume_name> | size=<integer>[G|M] pool=<pool>}

[-option <options>]

where:

- `<name>` = assigns the name to the file system
- `<type>` = assigns the file system type. It must be mgfs for a CDMS migration. The default value is uxs.
- `<volume_name>` = creates a file system on the specified metavolume, or the available metavolume with the specified size. A metavolume must be at least 2 MB in size to host a migration file system.
- `<integer>` = new MGFS is greater than or equal to the requested size, but does not exceed 1 terabyte
- `<pool>` = specified file system profile such as symm_std, clar_r1, clar_5_performance, and so forth
- `<options>` = specifies a comma-separated list of characteristics to create a file system

**Example:**

To create an mgfs file system type named `pmacF30` on the `mtv1d30` metavolume, type:

$ nas_fs -name pmacF30 -type mgfs -create mtv1d30

**Output**

id = 30
name = pmacF30
acl = 0
in_use = False
type = mgfs
volume = mtv1d30
profile =
rw_servers =
ro_servers =
symm_devs = 002804000192-0015
disks = d30

**Note:** You can view properties of an MGFS by using the Unisphere Data Migration GUI.
4. Create a mount point on the Data Mover.

Action

To create a mount point on the Data Mover, use this command syntax:

$ server_mountpoint {<movername>|ALL} [-create|delete|.exist] <pathname>

where:

<movername> = Date Mover name to which you want to create a mount point
pathname = directory pathname on which the file system is mounted. All
pathnames must begin with a forward slash (/).
The ALL option executes the command for all of the Data Movers.

Example:
To create a mount point on the server_2 Data Mover with a pathname of
/pmacct3, type:

$ server_mountpoint server_2 -create /pmacct3

Output

server_2 : done

Note

- server_2 is the VNX to which you are migrating data.
- /server1 is the file system mount point to which you are migrating data.

5. Mount the MGFS on the Data Mover.

Action

To mount the MGFS on the Data Mover, use the following command syntax:

$ server_mount <movername> [-Force] [-option <options>] <fs_name> [<mount_point>]

where:

<movername> = name of the Data Mover
<options> = specifies a comma-separated list of mount options
<fs_name> = name of the file system to be mounted
<mount_point> = stipulated mount point for the specified file system

Example:
To mount the pmacF30 file system on the server_2 Data Mover by using the
/pmacct3 mount point, type:

$ server_mount server_2 -option rw pmacF30 /pmacct3

Output

server_2 : done
6. Check that the file system is mounted properly.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To check if the file system is mounted correctly and recognized as a migration file system, use this command syntax:</td>
</tr>
<tr>
<td>$ server_cdms {&lt;movername&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = name of the Data Mover</td>
</tr>
<tr>
<td>&lt;mgfs&gt; = file system name</td>
</tr>
<tr>
<td>Verify that the MGFS special information lines look correct.</td>
</tr>
<tr>
<td>The <strong>ALL</strong> option displays the attributes of all file systems, including the configuration of associated disks.</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To check to see if the pmacF30 file system is mounted properly, type:</td>
</tr>
<tr>
<td>$ server_cdms server_2 -info pmacF30</td>
</tr>
<tr>
<td>You should see <code>rw_servers</code> and MGFS information added to the information output.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 :</td>
</tr>
<tr>
<td>pmacF30:</td>
</tr>
</tbody>
</table>
You must prepare the NFS source file server data for migration to the VNX. If you have assigned the VNX a new IP address, perform the following:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unmount the file systems on all UNIX workstations mapped to the NFS source file server. Applications might receive an Abort, Fail, or Retry? error message if they attempt a file I/O process while the Data Mover is being added. If this occurs, select Retry. The error could occur multiple times until the Data Mover is in place and the shared disk drives are reattached.</td>
</tr>
<tr>
<td>2</td>
<td>Make a note of the original export options parameters because you often need to reproduce them (or their effect) on the export of the migrated file system from the VNX used in a later step.</td>
</tr>
<tr>
<td>3</td>
<td>Unmount and then remount the NFS source file server as read-only to the IP address of the Data Mover. This action ensures data integrity because data updates can only occur on the VNX.</td>
</tr>
</tbody>
</table>
| 4    | Export the file system from the NFS source file server with root access privileges granted to the Data Mover. As an example, in the case of Solaris, the export command is:  
  
  ```
  $ share -F nfs -option ro,root=@128.221.252.54/0 /fs1
  ```
  
  where:
  
  128.221.252.54/0 = Data Mover IP address
  
  /fs1 = NFS source file server
  
  In this example, you can check the command result by issuing a share command on Solaris to display the characteristics of all shared or exported file systems.
  
  Output:
  
  ```
  /sourcefs
  ro=@128.221.252.104/0,root=@128.221.252.104/0 ""
  ```
  
  The following example allows everyone access to the /home directory in the default root volume on a Network Appliance system, but only adminhost and dm3 (VNX Data Mover 3 in this network) have root privileges on the file system:
  
  ```
  $ exportfs -option root=adminhost:dm3
  /vol/vol0/home
  ```
Step 4: Starting the CDMS migration service

Run the following command to start the CDMS migration service.

**Note:** If the CDMS migration service is already started on the Data Mover, you might not have to perform this task.

---

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To start the CDMS migration service, use this command syntax:</td>
</tr>
<tr>
<td>$ server_setup &lt;movername&gt; -Protocol cdms -option start[=&lt;n&gt;]</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = name of the specified Data Mover</td>
</tr>
<tr>
<td>&lt;n&gt; = number of threads for internal migration tasks. Default number of CDMS threads is 32, maximum is 128 threads per Data Mover.</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To start CDMS for the server_2 Data Mover with 25 threads, type:</td>
</tr>
<tr>
<td>$ server_setup server_2 -Protocol cdms -option start=25</td>
</tr>
</tbody>
</table>

---

Step 5: Connecting the file systems

To connect the NFS source file server and MGFS file system, perform the following:

1. Connect the file system (one source) to a subdirectory.

   You can also connect a second NFS source file server, verify that connection, and then check the server log.

   “Many-to-one migration” on page 111 provides details of this process.

   **Note:** This step cannot be interrupted. If a failure occurs, the process may take a significant amount of time before it finishes.
### Action

To connect to the NFS source file server so its content can be replicated to an MGFS subdirectory, use this command syntax:

```bash
$ server_cdms <movername> -connect <mgfs> -type {nfsv2|nfsv3} -path <localpath> -source <srcName>:/<srcPath> [ -option [useRootCred={true|false}] [proto={TCP|UDP}] [nfsPort=<port>] [mntPort=<port>] [mntVer={1|2|3}] [localPort=<port>]
```

Where:

- `<movername>` = name of the Data Mover
- `<mgfs>` = migration file system name
- `nfsv2` | `nfsv3` = NFS source file server type
- `<localpath>` = subdirectory name (that is created if it does not exist; if it exists, it fails) within the mount point of the file system. You cannot connect to the file system mount point, only to the subdirectory.
- `<srcName>:/<srcPath>` = name or IP address of the remote server, and export and path to where the root of this connection should exist.
- `useRootCred={true|false} = default is false. When true, the mount option that ensures the MGFS reads from the source file server where the root access UID=0, GID=0. This action assumes the NFS source file server is exported to allow root access from the specified Data Mover. When false, the MGFS uses the owner’s UID and GID to access data.

**Note:** Values typed that use the `server_cdms` command are case-sensitive. Consequently, a value must be typed in all lowercase or all uppercase letters. For example, if you type `True` using mixed-case letters, the value is interpreted as `false`.

- `proto={TCP | UDP}` = NFS connection, by default, uses TCP communications but can be overridden to use UDP communications
- `nfsPort=<port>` = remote NFS port number in case Portmapper, rpc bind, is not running, and port is not the default of 2049
- `mntPort=<port>` = remote mount port number in case Portmapper, rpc bind, is not running
- `mntVer={1 | 2 | 3}` = version used for mount protocol. By default, NFS V2 uses mount version 2, unless user specified version 1; NFS V3 only uses mount version 3.
### Action

**localPort=<port>** = port number used for NFS services, it needs to be different from the default. By default, the CDMS connection uses a nonprivileged local NFS port (>=1024). If the remote NFS daemon is set up to check the privileged port for clients, you should use this option to specify a port number lower than 1024. For example, on a Solaris machine, if `nfssrv:nfs_portmon=1` appears in the `/etc/system` file, then the NFS daemon on this Solaris machine requires clients’ NFS requests to come from a privileged port. Port 1021 is used as the local port for mount; therefore, the `localPort` option should not be 1021.

The connection cannot be made to the MGFS mount point, only to a previously uncreated path (subdirectory) of the mount point. If the connection command fails, an appropriate error message appears.

Examples:

To connect the `/dest1` subdirectory of the `pmacF30` file system to the `/home1` directory of the source file server IP address 10.100.50.14, type:

```bash
$ server_cdms server_2 -connect pmacF30 -type nfsv3 -path /dest1 -source 10.100.50.14:/home1
```

To connect the `/export1` subdirectory of the `mgfs1` file system to the `/export1` directory of the source file server path `server.domain:/export1`, type:

```bash
$ server_cdms server_2 -connect mgfs1 -type nfsv2 -path /export1 -source server.domain:/export1 -option proto=UDP
```

<table>
<thead>
<tr>
<th>Output</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 : done</td>
<td>• 10.100.50.14 is the IP address of the NFS source file server.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> You can use hostnames as long as DNS is configured correctly. You can also use an IP address.</td>
</tr>
<tr>
<td></td>
<td>• <code>pmacF30</code> is the MGFS name you created when preparing the NFS source file server.</td>
</tr>
<tr>
<td></td>
<td>• <code>pmacF30</code> is the MGFS to which you are migrating.</td>
</tr>
<tr>
<td></td>
<td>• You must define the NFS type, otherwise, the command fails. If you do not supply the protocol type (UDP, TCP), the default value is TCP.</td>
</tr>
</tbody>
</table>
2. Check that the NFS source file server is connected properly on the destination.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
</table>
| To check if the NFS source file server is connected properly to the destination, use this command syntax:  
$ server_cdms \{<movername>|ALL\} -info [<mgfs>]  
where:  
<movername> = name of the Data Mover  
<mgfs> = name of the migration file system  
The ALL option displays the attributes of all file systems, including the configuration of associated disks.  
Example:  
To check if the pmacF30 file system is connected properly to the server_2, type:  
$ server_cdms server_2 -info pmacF30 |

You should see that the first connection information appears in the MGFS lines of the output display.

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
</table>
| server_2:  
pmacF30:  
path = /dest1  
cid = 0  
type = nfs  
source = 10.100.14.50:/home1  
options = |
3. Check the server log to ensure that the connection succeeded properly.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To check the server log contents since the last restart, use this command syntax:</td>
</tr>
<tr>
<td>$ server_log &lt;movername&gt; [-a][-f][-n][-s]</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = Data Mover where the mount command was executed</td>
</tr>
<tr>
<td>-a = complete log</td>
</tr>
<tr>
<td>-f = log growth by entering into an endless loop, pausing, reading the log being generated. The output is updated every second. To exit, press [CTRL] and [C] together.</td>
</tr>
<tr>
<td>-n = log without the timestamp</td>
</tr>
<tr>
<td>-s = time in yyyy-mm-dd format when each command was executed in the log</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To check the server_2 Data Mover’s log content, type:</td>
</tr>
<tr>
<td>$ server_log server_2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>2002-04-09 11:03:54: ADMIN: 4: Command succeeded: connect fsid=22 type=nfsv3 path=/dest2 nfs=10.100.50.15:/home2 proto=TCP</td>
</tr>
</tbody>
</table>

The following example shows the output from the server_cdms <movername> -info <mgfs> command for two sources (remote 0 and remote 1) when they are properly connected to the destination.

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2:</td>
</tr>
<tr>
<td>pmacF30:</td>
</tr>
<tr>
<td>path = /dest1</td>
</tr>
<tr>
<td>cid = 0</td>
</tr>
<tr>
<td>type = nfs</td>
</tr>
<tr>
<td>source = 10.100.14.50:/home1</td>
</tr>
<tr>
<td>options =</td>
</tr>
<tr>
<td>path = /dest2</td>
</tr>
<tr>
<td>cid = 1</td>
</tr>
<tr>
<td>type = nfs</td>
</tr>
<tr>
<td>source = 10.100.14.50:/home2</td>
</tr>
<tr>
<td>options = proto=UDP</td>
</tr>
</tbody>
</table>
4. Export the MGFS so others can use it.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To export the MGFS, use this command syntax:</td>
</tr>
<tr>
<td>$ server_export &lt;movername&gt; [-Protocol nfs [-name &lt;name&gt;] [-ignore] [-option &lt;options&gt;] [-comment &lt;comment&gt;] &lt;pathname&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = Data Mover name you are migrating to and exporting from</td>
</tr>
<tr>
<td>&lt;name&gt; = name as it appears to others. If you do not use a &lt;name&gt;, use server_export &lt;movername&gt; -option [&lt;options&gt;] /&lt;pathname&gt;.</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = name of the Data Mover</td>
</tr>
<tr>
<td>&lt;name&gt; = exports an NFS &lt;pathname&gt; assigning an optional alias for the &lt;pathname&gt; as the &lt;name&gt;</td>
</tr>
<tr>
<td>&lt;options&gt; = normal export options</td>
</tr>
<tr>
<td>&lt;comment&gt; = export table comment</td>
</tr>
<tr>
<td>&lt;pathname&gt; = exports an NFS &lt;pathname&gt; assigning an optional alias for the &lt;pathname&gt; as the &lt;name&gt;</td>
</tr>
</tbody>
</table>

The EMC VNX Command Line Interface Reference for File provides details about the server_cdms command and its options.

Usually, when migrating, these options are a copy or equivalent of the options on the original source export. If these options were restrictive, some permissions or access for the system running the server_cdms command might need to be added.

Example:

To export the /pmac3/dest1 path from the MGFS so other clients can mount it as the /home1 directory, type:

$ server_export server_2 -Protocol nfs -name home1 -option anon=0 /pmac3/dest1

<table>
<thead>
<tr>
<th>Output</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 : done</td>
<td>/pmac3/dest1 is the real pathname inside the server.</td>
</tr>
</tbody>
</table>

The <name> option is useful when you do not want to change the fstab or automount configuration of UNIX workstations.

For example, mount 10.1.1.1:/mnt1 /mnt1 and mount 10.1.1.1:/mnt2 /mnt2 are in the fstab of the client. The data is then consolidated into one migration file system on the VNX. The new mount point is /mnt_new.
The fstab must be modified as follows:

```
mount 10.1.1.1:/mnt_new/mnt1 /mnt1 and
mount 10.1.1.1:/mnt_new/mnt2 /mnt2
```

However, if you use:

```
server_export server_2 -Protocol nfs -name mnt1 -option anon=0
/mnt_new/mnt1
```

and

```
server_export server_2 -Protocol nfs -name mnt2 -option anon=0
/mnt_new/mnt2
```

you do not have to change the fstab. You can access the file system as before.

5. Verify that the file system is exported with an alias name.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To verify that the file system was exported with the alias name, use this command syntax:</td>
</tr>
<tr>
<td>$ server_export {&lt;movername&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = name of the Data Mover</td>
</tr>
<tr>
<td>The ALL option executes the command for all of the Data Movers.</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To verify that the server_2 Data Mover file system was exported with an alias name, type:</td>
</tr>
<tr>
<td>$ server_export server_2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2:</td>
</tr>
<tr>
<td>export &quot;/pmac3/dest1&quot; name=/home1 anon=0</td>
</tr>
<tr>
<td>export &quot;/mp1&quot; anon=0</td>
</tr>
<tr>
<td>export &quot;/mgfslog&quot; anon=0</td>
</tr>
<tr>
<td>export &quot;/&quot; anon=0</td>
</tr>
</tbody>
</table>

6. Mount the servers you unmounted in “Step 3: Preparing the source file server” on page 89 to export from the Data Mover that is now available.

   From this point forward, all files are migrated to the VNX as they are read.

7. Optionally, you can verify the status of the migration with a nas_fs command.
Information appears in the client window. The type should be mgfs. Status displays the amount of free space available and the amount of that space used so far. You can use this information to determine how much of the file system was migrated, and possibly how much still remains.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
</table>
| To verify the status of the migration, use this command syntax:  
$ nas_fs -info [<mgfs>]  
Note that the remote information appears within the MGFS special lines, and is equivalent to the numbers in a df -k command of the source file server.  
where:  
<mgfs> = name of the migration file system  
Verify the MGFS special information lines look correct.  
Example:  
To verify the status of the migration of the pmacF30 file system, type:  
$ nas_fs -info pmacF30 |
### Step 6: Ensuring that all data is migrated

At the beginning of this step, the MGFS contains some directories and files copied from the NFS source file servers by the action of individual clients. However, you still need to migrate all untouched files from the NFS source file server to the VNX. The `server_cdms <movename> -start <mgfs>` command can be run when the file systems are not mounted on the Control Station or the UNIX workstation.

---

**Output**

```bash
# nas_fs -info pmacF30

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>22</td>
</tr>
<tr>
<td>name</td>
<td>pmacF30</td>
</tr>
<tr>
<td>acl</td>
<td>0</td>
</tr>
<tr>
<td>in_use</td>
<td>True</td>
</tr>
<tr>
<td>type</td>
<td>mgfs</td>
</tr>
<tr>
<td>volume</td>
<td>mtv1d30</td>
</tr>
<tr>
<td>pool</td>
<td>symm_std</td>
</tr>
<tr>
<td>member_of</td>
<td>root_avm_fs_group_1</td>
</tr>
<tr>
<td>rw_servers</td>
<td>server_2</td>
</tr>
<tr>
<td>ro_servers</td>
<td></td>
</tr>
<tr>
<td>rw_vdms</td>
<td>=</td>
</tr>
<tr>
<td>ro_vdms</td>
<td>=</td>
</tr>
</tbody>
</table>

**Status**

<table>
<thead>
<tr>
<th>Total KBytes Used</th>
<th>Total inodes Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>KBytes</td>
<td>inodes</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>3856</td>
<td>1062718</td>
</tr>
<tr>
<td>67384</td>
<td>8425</td>
</tr>
<tr>
<td>13144</td>
<td>1645</td>
</tr>
</tbody>
</table>

Command succeeded: mgfs action=query fsid=22

- symm_devs = 000184501314-0023
- disks = d30
- disk=d30 symm_dev=000184501314-0023 addr=c0t2111-03-0
- server=server_2

---

**Note:** Status information for the remote system might not be completely reliable. Many servers give information relative to the entire volume rather than just the mounted or exported file system. However, the displayed Data Mover information is accurate.
Note: EMC recommends you use the server_cdms -start command during a period of low user activity because response time from the source file server might be slow as large amounts of data are migrated. As the command runs, response time improves because more data is being accessed from the VNX rather than from the NFS source file server.
To start the internal migration, run the `server_cdms` command.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To start the CDMS internal migration, use this command syntax:</td>
</tr>
<tr>
<td>$ server_cdms &lt;movername&gt; -start &lt;mgfs&gt; -path &lt;localpath&gt; [-Force] -log &lt;logpath&gt; [-include &lt;include_path&gt;] [-exclude &lt;exclude_file&gt;]</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = name of the Data Mover</td>
</tr>
<tr>
<td>&lt;mgfs&gt; = NFS source file server name from which the files are to be migrated</td>
</tr>
<tr>
<td>&lt;localpath&gt; = local path on the Data Mover, within this MGFS, on which the internal migration thread starts processing</td>
</tr>
<tr>
<td>&lt;logpath&gt; = directory path on a file system mounted or accessible on the Data Mover. Internal migration and error log files are created in this directory. Do not include this directory in the MGFS.</td>
</tr>
<tr>
<td>&lt;include_path&gt; = migrate only entries defined in the &lt;file&gt;</td>
</tr>
<tr>
<td>&lt;exclude_path&gt; = skip all entries defined in the &lt;file&gt;</td>
</tr>
<tr>
<td>The include and exclude files have the same format.</td>
</tr>
<tr>
<td>• Use f to indicate an entry is a file, use d to indicate an entry is a directory.</td>
</tr>
<tr>
<td>• You can use a full path or path relative to the thread start path to represent files and directories.</td>
</tr>
<tr>
<td>• # indicates a comment.</td>
</tr>
<tr>
<td>• For an exclude file, all matching entries are removed from the migration file system.</td>
</tr>
<tr>
<td>• For an include file, only file entries can be added to the include file. If -include is specified, only the matching entries are migrated.</td>
</tr>
<tr>
<td>Include and exclude files can be used separately or together. If include and exclude options are specified for the same migration thread, it has the following effect:</td>
</tr>
<tr>
<td>• All matching include entries are migrated.</td>
</tr>
<tr>
<td>• All matching exclude entries are removed.</td>
</tr>
<tr>
<td>• All other entries remain offline.</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To execute the <code>server_cdms</code> command to force complete migration of the contents of the /home1 directory (the /dest1 subdirectory of pmacF30 file system), type:</td>
</tr>
<tr>
<td>$ server_cdms server_2 -start pmacF30 -path /dest1 -log /mgfslog</td>
</tr>
</tbody>
</table>
The command starts a migration thread, which traverses a directory tree, and forces the migration of files from that tree to the MGFS. You can use include and exclude files separately or concurrently. Chapter 4, “Planning and Design,” provides more information about using include and exclude files.

The -Force option is used only if you need to start a migration thread a second time on the same local path where a previous migration thread had already finished. For example, -Force is needed to start a thread that has no include file (that is, to migrate all remaining files) on a local path where a thread with an include file was already run. Note that if the migration thread status shows FAILED, it does not mean that the entire migration has failed or that the thread has stopped running. It only means that one (or more) of the files touched by the migration thread could not be migrated. The thread may still be running. There will be a message in server_log if the thread completes. For the exact status of the files that failed to migrate, you should examine the thread log files in the log directory. Similarly, if the thread status shows ERROR, it means that the parts of one or more files could not be read for migration. It does not indicate that the entire migration is in error. The thread log files in the log directory provide the exact status of the files that are incompletely migrated.

Note: You can start or stop migration threads by using the Unisphere Data Migration GUI or the CLI. You can also download include and exclude files by using the Unisphere Data Migration GUI or the CLI.

Filenames appear in the client window as files are migrated to the Data Mover.
**Note:** You can view migrations to an MGFS by using the Unisphere Data Migration GUI.

This process might take a significant amount of time, depending upon:

- Migrated file system size
- Type of server on which the source resides
- Network speed
- VNX software version
- Network traffic

**Checking migration progress**

You can check the status of the migration while it is in progress by using the following command syntax from the Control Station:

```
$ server_cdms <movername> -info <mgfs>
```

For example, for the previous example, execute the following command:

```
$ server_cdms server_2 -info pmacF30
```

**Note:** You can view the properties of a migration by using the Unisphere Data Migration GUI.

The command displays connection and thread status. You can check all MGFS file systems or a specified one, or only threads in a particular state. Note that if the migration thread status shows FAILED, it does not mean that the entire migration has failed or that the thread has stopped running. It only means that one (or more) of the files touched by the migration thread could not be migrated. The thread may still be running. There will be a message in server_log if the thread completes. For the exact status of the files that failed to migrate, you should examine the thread log files in the log directory. Similarly, if the thread status shows ERROR, it means that the parts of one or more files could not be read for migration. It does not indicate that the entire migration is in error. The thread log files in the log directory provide the exact status of the files that are incompletely migrated.

**Extending disk space**

If you run out of disk space during the migration, you can extend the Data Mover migration file system while files are being migrated.
Note: You can extend an MGFS by using the Unisphere Data Migration GUI.

Step 7: Examining the error log

Before you run the `server_cdms` command, as described in “Step 8: Verifying migration completion” on page 105, check the migration log file, and convert all errors. The migration log and error log are automatically created by using the following naming convention:

- migLog_path
- migErr_path

```
$ ls -l /nas/rootfs/slot_2/s2ufs2/log
total 16
-rw-r--r-- 1 root bin 0 Sep 30 2003 migErr_demo_mgfs_nfsv3
-rw-r--r-- 1 root bin 10556 Sep 30  2003 migLog_demo_mgfs_nfsv3
```

Note: You can download error log content by using the Unisphere Data Migration GUI.

If the error log is not empty, follow these steps:

1. Extract all pathnames from the error log by using the following command syntax:

   ```
   $ cat <logname>|grep Fullpath
   ```

   where:

   `<logname>` = name of the error log file produced by the `server_cdms` command. The last line of each error description shows the full path of the erroneous file.

2. Mount the source on the local UNIX workstation or the Control Station.

3. For each pathname in the error log file, determine if it can be remotely accessed on the NFS source file server.

4. Do one of the following:

   - If the source directory or file is readable, migrate the file or directory by using the `server_cdms` command on the Data Mover version of this pathname.
   - If the source directory or file is unreadable, there is an error on the source file server. Retain a list of problem source files for this file system.
5. Do one of the following:
   - If you have completely migrated the log file error list’s content, determine if there are any listed problem source files.
   - If you have not completely migrated the content of the log file error list, return to “Step 4: Starting the CDMS migration service” on page 90.

6. If problem source files are present, perform the following:
   a. Deliver the errors corresponding to these files and their pathnames to the customer.
   b. Work with the customer to determine a suitable method to handle these files such as copying by another method, or deleting them on the VNX.

7. When all problem source files have been handled, check to see if the error log is empty:
   - If the log is still not empty, go back to “Step 1: Adding the Data Mover” on page 81.
   - If the log is empty, go to “Step 8: Verifying migration completion” on page 105.

---

**Step 8: Verifying migration completion**

---

**CAUTION**

Do not run a verify command if there are any migration threads running or active on other CDMS connections in the same file system. Any CDMS data transfer to the same file system, even if it is on a different path from the one being verified, causes the verify process to restart from the beginning of the path given in the verify command. Running a CDMS migration thread typically produces near-continuous CDMS transfers. This causes near-continuous restarts of the verify command and therefore, high Data Mover CPU loads, which results in problems for all other users of that Data Mover. Because the verify command is a non-interruptible command, this condition persists until all migration threads are stopped on that file system or the verify command fails. Running a verify command when there are no active migration threads avoids these problems. Normally, a verify command is run only at the end of the migration process, when all migration threads have completed satisfactorily, so this is usually not an issue.
Do not run a verify command if there is another verify command still in progress on a different file system on the same Data Mover. Either one or both commands will fail, and you will have to run them again separately.

Note: The `server_cdms -verify` command verifies the completion of the migration. After you migrate file systems to a Data Mover, verify that the connections migrated successfully.

Optionally, you can verify on a per-connection basis:

- If all the connections you are examining are completely migrated, verification passes.
- If any files, inodes, or directories are not completely migrated, verification fails and an error message appears.

Note: You can view properties of a migration by using the Unisphere Data Migration GUI.

A successful verification changes inode formats, but not the file system type. It remains MGFS. This design enables further connections, if desired, prior to executing the `server_cdms -Convert` command from the Control Station. A successful verification disconnects any currently open migration source connections.

An incomplete, unsuccessful, or failed verification might result in an error message, leaving the file system in MGFS format so that further connections and consolidation migrations are possible. Check in the error log for further information.
To verify the status of a migration, use this command syntax:

```
$ server_cdms <movername> -verify <mgfs> [-path 
({<localpath>|<cid>})]
```

where:

- `<movername>` = name of the Data Mover
- `<mgfs>` = MGFS name to be verified
- `<localpath>` = unique name specified to identify the destination of a particular connection
- `<cid>` = connection ID 0 through 1023. If no cid is supplied, all connections with that MGFS name are verified. The cid for each connection can be seen on the output of a `server_cdms <movername> -info <mgfs>` command.

Examples:

To verify the migration status of the `pmacF30` file system to the `server_2` Data Mover, type:

```
$ server_cdms server_2 -verify pmacF30
```

To verify the migration status of the `mgfs1` file system by using the `/export1` path to the `server_2` Data Mover, type:

```
$ server_cdms server_2 -verify mgfs1 -path /export1
```
The `server_cdms <movername> -info <mgfs>` command allows you to monitor migration status as it displays Data Mover MGFS status information.

“Managing a failed verification or conversion” on page 229 provides more information if the verification does not complete successfully.
Step 9: Converting the migration file system

After the migration has been validated successfully, with all inodes migrated to the Data Mover, you need to run the `server_cdms <movername> -Convert <mgfs>` command to change the migrated MGFS files to UxFS files, removing information maintained by the MGFS while performing the migration.

If successful, disconnect the NFS source file server, or redeploy it for some other purpose.

The `server_cdms <movername> -Convert <mgfs>` command performs verify actions on any files and directories not yet verified by the `server_cdms <movername> -verify <mgfs>` command. A successful conversion makes the file system type UxFS, preventing any further attempt to migrate data into that migration file system. Conversely, an unsuccessful conversion leaves the file system in an MGFS state.

**CAUTION**

Do not run a convert command if there is another convert command still in progress on a different file system on the same Data Mover. Either one or both of the commands fail and have to be run again separately.

**Note:** You can convert a type MGFS to a UxFS by using the Unisphere Data Migration GUI.

The command can be run while the file system is mounted, exported, or in use. It is not an offline operation. Because the file system type is changed during the final part of the command operation, it can cause a momentary delay in access. Therefore, it is recommended as a best practice to convert at a quiet time when there are few outstanding I/O requests on the file system. Avoid converting during periods of heavy I/O use, which may be caused not only by user interaction but also by automated backups or report generation.
“Removing an unwanted connection” on page 232 provides more information if the conversion does not complete successfully.

**CAUTION**

If a CDMS connection to the source system is the reason for a failure of the `server_cdms -Convert` command, it is likely that some prior `server_cdms -verify` command has failed or was not completed. You should first correct all issues on the verify commands and have the connection removed by a flawless verify process for the whole file system before attempting the convert command. Manually removing the connection without a successful verify may force you to delete all incompletely migrated files to succeed with the convert.

### Step 10: Where to go from here

If you want to:

- Learn how to merge multiple file systems into a single file system, go to “Many-to-one migration” on page 111.
- Complete the single file system migration process, go to “Postmigration testing” on page 118.
Many-to-one migration

You can merge several file systems together into a single file system by repeating steps 5, 6, and 7 for each file system that needs to be combined. Each file system might be from one or more NFS source file servers, and might include file systems that reside on different operating systems such as Solaris and AIX.

**Note:** All users must get their identities and privileges from the same NIS.

The merge process can handle up to 1,024 file systems simultaneously. However, it is rarely used for more than four or five file systems migrating in parallel because network bandwidth is a limiting factor.

**Note:** This procedure is similar to the CIFS migration procedure, described in “Many-to-one migration” on page 176.

Any selected NFS source file server follows the procedure as shown in Table 7 on page 111.

**Table 7 Many-to-one migration summary**

<table>
<thead>
<tr>
<th>Step</th>
<th>Description and location</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>“Step 5: Connecting the file systems” on page 111</td>
</tr>
<tr>
<td>6</td>
<td>“Step 6: Ensuring that all data is migrated” on page 112</td>
</tr>
<tr>
<td>7</td>
<td>“Step 7: Examining the error log” on page 112</td>
</tr>
</tbody>
</table>

**Step 5: Connecting the file systems**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Connect the NFS source file server to the target VNX migration file system.</td>
</tr>
<tr>
<td>2</td>
<td>Check the connection to the destination.</td>
</tr>
<tr>
<td>3</td>
<td>Check for connection success by using the <code>server_log &lt;movername&gt;</code> command that displays the log generated by each server.</td>
</tr>
</tbody>
</table>
Step 6: Ensuring that all data is migrated

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Export the target file system for client usage.</td>
</tr>
<tr>
<td>5</td>
<td>Mount active clients.</td>
</tr>
<tr>
<td>6</td>
<td>Verify migration status.</td>
</tr>
</tbody>
</table>

Step 7: Examining the error log

Verify the migrated data.

**Note:** You can download the migration log or error log by using the Unisphere Data Migration GUI.

The `server_cdms <movername> -connect <mgfs> command syntax, as described in “Step 5: Connecting the file systems” on page 90, allows you to merge multiple file systems. All command parameters can be different for each NFS source file server that needs to be migrated, including `type`, `path`, `nfsPort`, and `protocol`.

“Examples” on page 113 shows each element separately in the context of file merging:

```
$ server_cdms <movername> -connect <mgfs> -type {nfsv2|nfsv3} -path <localpath> -source <srcName>:/<srcPath> [-option [useRootCred={true|false}] [proto={TCP|UDP}] [nfsPort=<port>] [mntPort=<port>] [mntVer=1|2|3] [localPort=<port>]]
```

where:

- `<mgfs>` = target migration file system name. This is the only parameter that remains the same for each file system being merged into the MGFS.
-type {nfsv2|nfsv3} = NFS version 3 or NFS version 2 for each NFS source file server.

-path <localpath> = subdirectory name within the mount point of the file system. If the name does not exist, it is created automatically. This is the primary method by which separate file systems are merged. Each additional source file server is added as a subdirectory of the mount point.

For example:

```
/pmacF30
/dest1
/dest2
/dest3
```

nfsPort=<port> = remote mount point information that allows you to identify the same system or a different system as the file system source.

proto={TCP|UDP} = protocol that can be different for each NFS source file server.

localPort=<port> = (optional) might be different on each mount.

**Examples**

The following examples use the `server_cdms` command, consolidating three file systems from three different NFS source file servers into one migration file system. Again, the `<localpath>` option of the `server_cdms <movername> -connect <mgfs>` command accomplishes this migration.

**File system 1:**

Connect the `data1` file system from the first source file server:

```
$ server_cdms server_2 -connect pmacF30 -type nfsv3 -path /dest1
   nfsPort=10.100.50.14:/data1 -option proto=UDP
```

**File system 2:**

Connect the `data2` file system from the second source file server:

```
$ server_cdms server_2 -connect pmacF30 -type nfsv2 -path=/dest2
   nfsPort=10.100.50.15:/data2 -option proto=UDP
```

**File system 3:**

Connect the `data3` file system from the third source file server:

```
$ server_cdms server_2 -connect pmacF30 -type nfsv3 -path /dest3
   nfsPort=10.100.50.16:/data3 -option proto=TCP
```
As with a single file system migration, the `server_cdms <movername> -verify <mgfs>` command verifies connections for multiple source file servers, and changes the file system type to UxFS.

If the connection command fails, an appropriate error message appears.

The following example shows the output from the command for three sources after they are properly connected to the destination:

- remote 0:
- remote 1:
- remote 2:

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>id     = 22</td>
</tr>
<tr>
<td>name   = pmacF30</td>
</tr>
<tr>
<td>acl    = 0</td>
</tr>
<tr>
<td>in_use = True</td>
</tr>
<tr>
<td>type   = mgfs</td>
</tr>
<tr>
<td>volume = mtv1d30</td>
</tr>
<tr>
<td>rw_servers = server_2</td>
</tr>
<tr>
<td>ro_servers =</td>
</tr>
<tr>
<td>status =</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Total Kbytes Used Kbytes Total inodes Used inodes</td>
</tr>
<tr>
<td>Dart cid: 8703632 24 1062718 11</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
<tr>
<td>remote 0: 8703632 67384 1062718 8425</td>
</tr>
<tr>
<td>dest1 nfs=10.100.50.14:/data1 proto=UDP</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
<tr>
<td>remote 1: 8703632 13144 1062718 1645</td>
</tr>
<tr>
<td>dest2 nfs=10.100.50.15:/data2 proto=UDP</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
<tr>
<td>remote 2: 8703632 73144 1062718 9246</td>
</tr>
<tr>
<td>dest3 nfs=10.100.50.16:/data3 proto=TCP</td>
</tr>
<tr>
<td>Command succeeded: mgfs action=query fsid=22</td>
</tr>
<tr>
<td>symm_devs = 000184501314-0023</td>
</tr>
<tr>
<td>disks = d30</td>
</tr>
<tr>
<td>disk=d30 symm_dev=000184501314-0023 addr=c0t2111-03-0</td>
</tr>
<tr>
<td>server=server_2</td>
</tr>
</tbody>
</table>

After the file systems are connected, each one must be exported to the Data Mover sequentially.
The following example shows the exporting of three connected file systems. This is also a good case for the potential need for export aliasing.

```
$ server_export server_2 -Protocol nfs -name data1 -option anon=0 /pmacF30/dest1
server_2 : done
$ server_export server_2 -Protocol nfs -name data2 -option anon=0 /pmacF30/dest2
server_2 : done
$ server_export server_2 -Protocol nfs -name data3 -option anon=0 /pmacF30/dest3
server_2 : done
```

At this point, the UNIX workstation client can mount each file system.

**Parallelization or serialization**

For a single file system, the `server_cdms` command is the preferred method to force all untouched file systems to be migrated to the VNX. However, with multiple file systems connected to a single target migration file system, you have a choice about how to ensure that the data is fully migrated from the NFS source file servers.

If you want to simultaneously execute an individual copy of the `server_cdms` command per connection within that file system, all file systems are migrated at the same time. The migration of multiple file systems can improve data throughput while reducing overall migration time. However, this type of migration might also be more difficult to monitor.

In addition, if any of the migrations are not successful, it might be difficult to determine where the migration failed. Therefore, EMC suggests that in addition to the log file, you ensure to save the output of the `server_cdms` command to a file so any failures might be identified more accurately.

**Note:** The number of parallel streams are limited by the bandwidth of the migration network. As a best practice, keep the number of parallel streams to three on a 100 Mb/s network.

**Sequential file system migration**

Optionally, you can execute the `server_cdms` command against each connection, one at a time. This method might be more manageable, but it increases the overall migration time. The choice for either method is based on the needs and schedule.
The verification process is exactly the same for multiple file systems as it is for a single file system. You can execute the `server_cdms <movername> -verify <mgfs>` command against individual CIDs as the data transfer completes, or you can execute the verification process against all connected file systems involved in the merging effort by omitting the cid from the CLI. Do not run a verify command if there are any migration threads running or active on other CDMS connections in the same file system. Any CDMS data transfer to the same file system, even if it is on a different path from the one being verified, causes the verify process to restart from the beginning of the path given in the verify command. Running a CDMS migration thread typically produces near-continuous CDMS transfers. This causes near-continuous restarts of the verify command and therefore, high Data Mover CPU loads, which results in problems for all other users of that Data Mover. Because the verify command is a non-interruptible command, this condition persists until all migration threads are stopped on that file system or the verify command fails. Running a verify command when there are no active migration threads avoids these problems. Normally, a verify command is run only at the end of the migration process, when all migration threads have completed satisfactorily, so this is usually not an issue.

Do not run a verify command if there is another verify command still in progress on a different file system on the same Data Mover. Either one or both of the commands will fail, and you will have to run them again separately.

An example is as follows:

```
$ server_cdms server_2 -verify pmacF30 [-path {<localpath>|<cid>}]
```

**Note:** You can view migrations to an MGFS by using the Unisphere Data Migration GUI.

With regard to a single file system:

- If all of the current connections are completely migrated, then verification passes.
- If any files or directories are not completely migrated, verification fails.

A successful verification disconnects any currently open migration source connections.

Ensure that you monitor the progress of the verification through the `server_log` command, and the success or failure of the verification through the `server_cdms <movername> -info <mgfs>` command.
Conversion

The MGFS-to-UxFS conversion process is exactly the same for multiple file systems as it is for a single file system. The difference is that the verification process is run against all connected file systems involved in the merging effort. An example is as follows:

$ server_cdms server_2 -Convert pmacF30

Note: You can convert a type MGFS to UxFS by using the Unisphere Data Migration GUI.

Changing the file system type

The final steps are the same as for a single file system, as described in “Step 9: Converting the migration file system” on page 109.

CAUTION

If a CDMS connection to the source system is the reason for a failure of the server_cdms -Convert command, it is likely that some prior server_cdms -verify command has failed or was not completed. You should first correct all issues on the verify commands and have the connection removed by a flawless verify process for the whole file system before attempting the convert command. Manually removing the connection without a successful verify may force you to delete all incompletely migrated files to succeed with the convert.

The file system type changes automatically after successful completion of the conversion process.

Correcting GIDs (optional)

If the NFS source file servers for components of a multiple file system merge were not administered under a common NIS, or if the file systems did not come from a common server, there is a possibility that different group names have identical group ID numbers.

If the previous situation causes a problem, you can use the ch_group.pl script to modify specified GIDs in certain subdirectory trees on the file system.

“ch_group.pl script” on page 249 provides more details about this script.
Postmigration testing

Assuming the migration was successful, verify that all applications and file accessibility issues have been resolved successfully.

Complete the following tasks:

- Ensure that the new VNX systems and UxFS file systems are accessible from various UNIX workstations (or clients).
- If there are specific user applications, verify that they are functioning correctly.
- Verify all necessary operational modifications with the new VNX infrastructure.
- Disconnect, and reconfigure or remove the source file servers from the network.
CHAPTER 6
CIFS

The following topics provide step-by-step procedures about how to perform one-to-one and many-to-one CIFS migrations:

◆ Introduction ...................................................................................... 120
◆ Administrative share methodology ............................................... 121
◆ One-to-one migration ...................................................................... 123
◆ Many-to-one migration ................................................................... 176
◆ Postmigration testing ....................................................................... 212
Introduction

At this point in the overall migration effort, planning should be complete. You should have read “Operational issues and frequently asked questions” on page 66. Now it is time to execute the plan.

All mapping from CIFS source file servers to target VNX systems, choices for migration strategies, storage design, and network evaluation have been completed and documented thoroughly. From this point forward, the migration proceeds as designated in the plan.

**Note:** The VNX supports domain migration (for example, Windows NT 4.0 to Windows 2000 or Windows Server 2003). *Managing a Multiprotocol Environment on VNX* provides more information.

The methodology chosen to fully accomplish a migration is described in “Administrative share methodology” on page 121. This process establishes all connections to each drive's administrative share on the CIFS source file servers. This action simplifies the connection process, permitting all shares and corresponding data on that drive to be migrated to the VNX.

**Note:** Due to the similarities between Windows 2000 domains and Windows Server 2003 domains, these two environments are grouped together in this discussion of CIFS migrations.
Administrative share methodology

Using administrative share methodology, the following three components must be migrated from a CIFS source file server:

- Local groups
- Shares
- Data

Local groups

Local groups are migrated from the CIFS source file server to the Data Mover by using the `lgdup.exe` utility. “lgdup.exe utility” on page 283 and “Step 10: Migrating local group information” on page 155 provide more information.

A local group is a group that can be granted permissions and rights from its own computer to only those resources on its own computer on which the group resides. Computers must be running Windows Professional and member services.

Shares

Share names are migrated from the CIFS source file server to the Data Mover by using the `sharedup.exe` utility. To migrate share names with spaces in them, or into local paths with spaces in them, you can use the asterisk (*) character to replace the space in the `server_cdms <movername> -connect <mgfs> command`. “sharedup.exe utility” on page 284 and “Step 12: Executing the sharedup.exe utility” on page 163 provide more information about this utility.

A share is a resource on the file system, such as a volume, directory, or service, made available to CIFS users on the network.

Data

Data is migrated through connections created between Data Mover’s MGFS and share directories on the CIFS source file server by using the `server_cdms <movername>` command, which is run from the Control Station.

The EMC VNX Command Line Interface Reference for File provides more information about the `server_cdms` command.

To accommodate these connections, the `server_cdms <movername> -connect <mgfs>` command allows the Data Mover to act as a Windows client to a CIFS source file server. This design allows the Data Mover to establish a connection to a share on the source file server. Therefore, it is important to the implementation that these events occur in the following order:
1. The `lgdup.exe` utility migrates the local groups.
2. The CDMS connections are created to link the source file server and the VNX.
3. The `sharedup.exe` utility migrates the share names to the MGFS file system.
4. The `server_cdms <movername>` command migrates all untouched data to the corresponding share names on the MGFS file system.

Using the Administrative Share methodology, this is the basic sequence for CIFS migration. To ensure that the migration of all components is successful, it is important to thoroughly plan for the migration.

Home-to-home level migration is recommended. For example, migrate from the parent directory of the user's home directories; all users are migrated with one connection. Otherwise, separate connections must be made for each user's home directory migration at the MGFS level.

---

### Strategies

The following sections provide the migration specialist with details necessary to execute one of the following strategies:

- **“One-to-one migration” on page 123**
- **“Many-to-one migration” on page 176**

The two strategies associated with many-to-one migration are:

- **“Retained server name merge” on page 180**
- **“New server name merge strategy” on page 207**

These strategies are presented as a set of step scenarios. Use this document and the migration plan as the guide for all technical aspects of the migration.

During a migration, you can always begin another migration from any CIFS source file server to any Data Mover within the domain.
One-to-one migration

One-to-one CIFS migration involves completely taking over the identity of a single member server in a Windows Server 2003, Windows 2000, or Windows NT 4.0 domain. The approach used to migrate from this source file server is known as migration through the use of Administrative Share. This is perhaps the least complex method of migration with CDMS, and is likely to be used the most frequently.

Figure 8 on page 123 depicts a simple-but-typical environment for one-to-one migration.

The scenario described in this section is based on the following source file server assumptions:

- One-to-one migration is performed using the Administrative Share strategy.
- Most migration tasks are invoked by using a Windows client with administrative access to the domain, and a secure, encrypted, remote login application interface access to the Control Station.
- The CIFS source file server is a member server within a single Windows Server 2003, Windows 2000, or Windows NT 4.0 domain.

**Note:** CDMS does not support migration from domain controllers such as DCs, PDCs, or BDCs.

- Multiple disk drives are configured on the CIFS source file server.
- All shares of a source file server are migrated to a Data Mover.
- The migration includes multiple local groups.
- The VNX (one-to-one migration) assumes (or adopts) the name of the CIFS source file server.
- The CIFS source file server is not used (or is used for other purposes) after the migration completes.
- Windows administrative privileges are required to perform the migration.
- Rights are set on the CIFS source file server, the new target (VNX), and the Windows client performing the migration.
- The CIFS source file server does not handle printing functions.
- The migration does not include Microsoft Distributed File Systems (DFSs), nor encrypted files or directories.
- Directories are migrated only if they have shares associated with them at the time of the migration.

---

**VNX assumptions**

The scenario described in this section is based on the following VNX assumptions:

- Unicode is enabled, as described in “Step 6: Configuring the Data Mover” on page 136.
- A single migration file system is created, containing the shares and data of all disk drives from the CIFS source file server, as described in “Step 7: Creating the MGFS file system” on page 142.

---

**Conventions**

The scenario described in this section is based on the following conventions:

- The mount point name of the VNX is the name of the CIFS source file server.
- CIFS source file server drive letters are used for VNX local pathnames (for example, C$, E$, and so on).
Table 8 on page 125 summarizes the steps to implement one-to-one CIFS migration to a VNX.

Perform the following steps only once per migration effort.

<table>
<thead>
<tr>
<th>Step</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>“Step 1: Creating an account” on page 127</td>
</tr>
<tr>
<td>2</td>
<td>“Step 2: Installing components” on page 129</td>
</tr>
<tr>
<td>3</td>
<td>“Step 3: Evaluating the CIFS source file server” on page 130</td>
</tr>
<tr>
<td>4</td>
<td>“Step 4: Identifying high-priority files (optional)” on page 135</td>
</tr>
<tr>
<td>5</td>
<td>“Step 5: Backing up the source file server” on page 136</td>
</tr>
<tr>
<td>6</td>
<td>“Step 6: Configuring the Data Mover” on page 136</td>
</tr>
<tr>
<td>7</td>
<td>“Step 7: Creating the MGFS file system” on page 142</td>
</tr>
<tr>
<td>8</td>
<td>“Step 8: Preparing for migration” on page 147</td>
</tr>
<tr>
<td>9</td>
<td>“Step 9: Setting up the CIFS environment” on page 152</td>
</tr>
<tr>
<td>10</td>
<td>“Step 10: Migrating local group information” on page 155</td>
</tr>
<tr>
<td>11</td>
<td>“Step 11: Creating CDMS connections” on page 158</td>
</tr>
<tr>
<td>12</td>
<td>“Step 12: Executing the sharedup.exe utility” on page 163</td>
</tr>
<tr>
<td>13</td>
<td>“Step 13: Monitoring and waiting” on page 164</td>
</tr>
<tr>
<td>14</td>
<td>“Step 14: Ensuring that all data is migrated” on page 165</td>
</tr>
<tr>
<td>15</td>
<td>“Step 15: Examining the error log” on page 166</td>
</tr>
<tr>
<td>16</td>
<td>“Step 16: Verifying migration completion” on page 169</td>
</tr>
<tr>
<td>17</td>
<td>“Step 17: Converting the MGFS to a UxFS” on page 174</td>
</tr>
<tr>
<td>18</td>
<td>“Step 18: The next step” on page 176</td>
</tr>
</tbody>
</table>
Assume the CIFS source file server name is retained in a one-to-one migration. Figure 9 on page 126 shows the relationship between a Data Mover and one CIFS source file server.

Figure 9  CIFS source file server to Data Mover migration

To start the one-to-one migration process, perform the following:

Note: Ensure that you have read Chapter 4, “Planning and Design,” before you start the Implementation phase.

- Create an account with domain administrator rights from a domain controller or a primary controller
- Install all premigration components such as scripts, utilities, and applications on the Windows client
- Evaluate the source file server directory
- Identify any high-priority files
- Back up the source file server

If you have not performed the previous tasks in the migration environment, complete “Step 1: Creating an account” on page 127 through “Step 6: Configuring the Data Mover” on page 136.

If you have already completed these tasks, go to “Step 7: Creating the MGFS file system” on page 142 to originate the MGFS on the Data Mover, and complete the migration process.
Step 1: Creating an account

Create an account with domain administrator rights from a domain controller (DC) or a primary domain controller (PDC). Create or request a cdms_migrator user account using:

- User Manager for Domains (Windows NT 4.0)

Figure 10 on page 128 and Figure 11 on page 129 illustrate this.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Assign a cdms_migrator account to the Domain Admins group.</td>
</tr>
<tr>
<td>2</td>
<td>Set the Primary Group membership to Domain Admins.</td>
</tr>
<tr>
<td>3</td>
<td>Remove the cdms_migrator account from the Domain Users Group. This step allows restricted access to the CIFS source file server only from the cdms_migrator account when the CDMS procedure denies user access to the source file server in subsequent steps.</td>
</tr>
</tbody>
</table>
For a Windows 2000 or Windows Server 2003 environment

If you are using a DC in a Windows 2000 or Windows Server 2003 environment, review Figure 10 on page 128.

Figure 10  User account on Windows 2000 or Windows Server 2003 system: One-to-one migration
For a Windows NT 4.0 environment

If you are using a PDC in a Windows NT 4.0 environment, review Figure 11 on page 129.

Figure 11  User account on a Windows NT 4.0 system: One-to-one migration

Step 2: Installing components

Install the following components on the Windows client:

- Perl script (ActivePerl 5.6 or later)
- Win32 API extensions for the Perl script
- Appropriate EMC premigration utilities and scripts
- Microsoft Word and Excel 2000 applications

“Adding Win32 API to Perl script (CIFS)” on page 55 provides information about replacing the library file, and installing the Win32 API extensions to the Windows client.

At a minimum, client tools should include the `sharedup.exe`, `lgdup.exe`, and `backupWrapper.exe` utilities, but might also include `dircount.pl`, `diskusage.pl`, `connBuilder.pl`, and `dirprivW.pl` scripts.

Be sure to place all EMC utilities and scripts in the same directory on the Windows client. Be sure the `server_cdms` command is available on the Control Station in the `/nas/bin` directory.
Evaluate the CIFS source file server directory structure for files and directories that should be excluded from the migration. Use the `sharedup.exe` utility and `connBuilder.pl` script to assist with this evaluation. Chapter 4, “Planning and Design,” provides more information on using include and exclude files.

To begin the evaluation, perform the following:

1. Log in to a Windows client with network administrative privileges.
2. In a command window, navigate to the directory where the EMC utilities and scripts are stored on the client.
3. Run the `sharedup.exe` utility to create a basic share listing, as described in the following table.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To run the <code>sharedup.exe</code> utility to create a basic share name listing, use this command syntax:</td>
</tr>
<tr>
<td>C:&gt; sharedup &lt;source&gt;&lt;target&gt; ALL /FO:&lt;outputfile&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;source&gt; = CIFS source file server name</td>
</tr>
<tr>
<td>&lt;target&gt; = reachable target file server name. This operation creates a list of shares on the CIFS source file server, but you must also specify the target file server. Any server name can be used as a dummy for the target server in this case.</td>
</tr>
<tr>
<td>&lt;outputfile&gt; = share list to duplicate on the target server by using selected options</td>
</tr>
<tr>
<td>This filename becomes the input to the <code>connBuilder.pl</code> script.</td>
</tr>
<tr>
<td>Convert the filename from a <code>.txt</code> file to a <code>.doc</code> file as input to the <code>connBuilder.pl</code> script.</td>
</tr>
<tr>
<td>“<code>sharedup.exe</code> utility” on page 284 provides more information about the <code>sharedup.exe</code> utility.</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To run the <code>sharedup.exe</code> utility from the <code>server1</code> source file server to the <code>mypc</code> target server to create a <code>server1_shares.doc</code> share name listing, type:</td>
</tr>
<tr>
<td>C:&gt; sharedup \server1 \mypc ALL /FO:server1_shares.doc</td>
</tr>
</tbody>
</table>
This output file becomes the input to the connBuilder.pl script, which identifies any directories deleted from the exclude file.

4. In the same command window, run the connBuilder.pl script to produce the exclude file that might be required by the server_cdms command.

This action ensures that only specified files and directories are transferred to the VNX:

a. Run the connBuilder.pl script.

The output of the script is placed in the directory in which it is running. Ensure that the output listing of the sharedup.exe utility is also placed in the same directory.

C:\> perl connBuilder.pl <input_filename.doc

where:

<input_filename.doc> = filename produced by the sharedup.exe utility

b. Select Encoded Text in the Convert File box.
The connBuilder.pl script displays a dialog box, as shown in Figure 12 on page 132.

![Figure 12 File Conversion dialog box](image)

Figure 12 File Conversion dialog box

c. Click **OK** to complete the evaluation process.

**Note:** Depending on what Word conventions have been loaded previously, it might be necessary to use the Microsoft’s Word CD to install additional convention packs.

While completing the evaluation process, the connBuilder.pl script displays an Excel spreadsheet in the window. After the script disappears, there might be a need to close the Word and Excel windows. The connBuilder.pl script produces multiple files, depending on the number of drives configured with the source file server. These files end with `.txt`, `.xls`, and `.cmd` extensions. For the Administrative Share migration strategy, however, CDMS is only interested in the files ending with `.txt`.

The connBuilder.pl script uses the following syntax in naming the files:

```
exclude<-servername><-drive.txt>
```

where:
**exclude** = default prefix

<-servername> = name of the CIFS source file server

<-drive.txt> = disk drive letter

For example, the exclude-server1-C.txt filename indicates that the exclude file is for source file server server1, drive C:. This information is important for modifying the exclude file.

5. Modify the pathname of each entry in every exclude file to include the target server name and drive letter.

This change allows the exclude file to properly address the pathname structure that is set up on the VNX by using the sharedup.exe utility.

The following example shows an optional exclude file with the name exclude-server1-C.txt:

d M:\2users
f M:\AUTOEXEC.BAT
f M:\CONFIG.SYS
f M:\IO.SYS
f M:\MSDOS.SYS
f M:\NTDETECT.COM
d M:\RECYCLER
d M:\System Volume Information
f M:\_NavCClt.Log
f M:\arcldr.exe
f M:\arcsetup.exe
f M:\boot.ini
d M:\fmc1_data
f M:\h12_copy_s.bat
f M:\h12_copy_t.bat
f M:\h12_h13_z.bat
f M:\h12_h11_z.bat
f M:\ntldr
d M:\pst

**Note:** The d or f before each entry identifies a directory or file that needs to be excluded from this migration.

To modify the pathname within the exclude file, perform the following:

1. Locate the exclude file by using Windows Explorer, or My Computer from the Windows client.

2. Double-click the exclude file that you want to edit.
This action brings up Microsoft’s Notepad. If not, open the file from within Notepad.

3. From the **Edit** menu, select **Replace**.

4. In the **Replace** dialog box:
   a. Type **M:/** in the **Find what:** box.
   b. Type **M:/server1/c/** in the **Replace with:** box.
   c. Click **Replace All** to complete the modifications.
   d. Click **Cancel** to close the dialog box.

5. Exit Notepad after saving all changes.

Remember to make these modifications on each exclude file. The resulting files become the basis of the exclude lists used with the `server_cdms <movername> -start <mgfs>` command.

Given the previous exclude file, `exclude-server1-C.txt`, the result should appear as follows:

```
d /server1/c/2users
f /server1/c/AUTOEXEC.BAT
f /server1/c/CONFIG.SYS
f /server1/c/IO.SYS
f /server1/c/MSDOS.SYS
f /server1/c/NTDETECT.COM
d /server1/c/RECYCLER

d /server1/c/System Volume Information
f /server1/c/_NavCClt.Log
f /server1/c/arcldr.exe
f /server1/c/arcsetup.exe
f /server1/c/boot.ini
d /server1/c/mcl_data
f /server1/c/h12_copy_s.bat
f /server1/c/h12_copy_t.bat
f /server1/c/h12_h13_z.bat
f /server1/c/h12_h11_z.bat
f /server1/c/ntldr
d /server1/c/pst
```

---

**Note:** The M: drive letter is a value produced through the `connBuilder.pl` script, identifying a mapped drive on the CIFS source file server. This drive letter is important when running the `server_cdms` command later in the process. Although it can be modified in the exclude file to identify another drive letter, the same
drive letter must be used in "Step 15: Converting the MGFS to a UxFS" on page 211 with the net use and server_cdms commands. You are encouraged not to change this letter.

At this time, you can add to the exclude file any other files you do not want to migrate to the Data Mover. However, ensure the format of each entry is consistent with existing entries.

**Note:** For a successful migration, check the details of the CIFS source file server with the backupWrapper.exe utility, and dircount.pl, dirprivW.pl, and diskUsage.pl scripts.

**Step 4:** Identifying high-priority files (optional)

Identify any high-priority files on all drives from the CIFS source file server, and create corresponding include files from the Windows client. Chapter 4, “Planning and Design,” provides more information about using include and exclude files.

You can create these optional include files by using Notepad. Identify the files that should be migrated before all other files. These files are migrated in an online state with all other files remaining in an offline state. This list is usually very short, often containing references to all files with specific extensions. Of particular importance are Outlook PST (.pst) files and access database (.mdb) files. There can be one include file for each disk drive associated with the CIFS source file server. The format of this file is the same as the exclude file. The naming convention for the include file should be identical to the exclude file (for example, include-server1-C.txt), and should reside in the same directory as the exclude files. Remember you can use include files and exclude files concurrently. For example:

```
$ server_cdms server_2 -start mgfs1 -path /export1 -log /mgfs1/logs.txt
```

**Note:** You can download include files by using the Unisphere Data Migration GUI.
An example listing of the include-server1-C.txt file might appear as follows:

f /export1/server1/c/*.pst
f /export1/server1/c/*.mdb

**Note:** Do not use tabs; use spaces only. The d or f before each entry identifies a directory or file that needs to be included in this migration.

If there are no high-priority files, it is still a good idea to include a single entry with an extension that is not in any of the directories of the exclude list. This action forces CDMS to create the directory structure on the VNX.

For example, if a site has no high-priority files, the sample include-server1-C.txt file might contain the following entry:

f /export1/server1/*.none-exist

**Step 5:** Backing up the source file server

Use a reliable method to perform a full backup of the CIFS source file server prior to any data migration. Ensure that the restore method has been tested previously.

**Step 6:** Configuring the Data Mover

Perform the following steps to configure the Data Mover for the migration environment. *Configuring and Managing CIFS on VNX* provides more details:

**Note:** It is most effective to configure the Data Mover through a secure, encrypted, remote login application interface on the Windows client.

1. Synchronize Data Mover time to the Windows domain controller.
   
   If time is not synchronized, Windows 2000 or Windows Server 2003 usernames are not validated by Kerberos.
Note: Kerberos is a security system that authenticates users. It does not provide authorization to services or databases; it establishes identity at login, which is used throughout the session.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To synchronize Data Mover time to the Windows domain controller, use this command syntax:</td>
</tr>
<tr>
<td>$ server_date {&lt;movername&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = name of the Data Mover</td>
</tr>
<tr>
<td>&lt;yymmddhhmm&gt;[&lt;ss&gt;] = required synchronization time. Sets a two-digit number for the year, month, hour, minutes, and seconds in this order where &lt;yy&gt; is the year; the first &lt;mm&gt; is the month; &lt;dd&gt; is the day; &lt;hh&gt; is the hour (in 24-hour system); and the second &lt;mm&gt; is the minute, and &lt;ss&gt; is the second.</td>
</tr>
<tr>
<td>The ALL option executes the command for all of the Data Movers.</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To synchronize the server_2 Data Mover time to the 030110173026 time found on the domain controller, type:</td>
</tr>
<tr>
<td>$ server_date server_2 030110173026</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 : Fri Jan 10 05:30:26 EDT 2003</td>
<td>Set to the time that is synchronized with the domain controller, or specify the NTP server.</td>
</tr>
</tbody>
</table>

Alternatively, the domain controller and the Data Movers can be synchronized to the same Network Time Protocol (NTP) time-server host.

The EMC VNX Command Line Interface Reference for File provides more information about the server_date command.

2. Configure the target Data Mover for either:
   - WINS (Windows NT 4.0)

The following Windows 2000/Windows Server 2003 and Windows NT 4.0 tables illustrate this.
### For a Windows 2000 or Windows Server 2003 environment

**Action**

To configure DNS service for the Data Mover, use this command syntax:

```bash
$ server_dns {<movername>|ALL} [[-protocol {tcp|udp}]
<domainname> {<ip_addr>,...}]
```

where:
- `<movername>` = name of the Data Mover
- `<domainname>` {<ip_addr>,...} = IP address of a name server in that domain (usually on the domain controller)

The **ALL** option executes the command for all of the Data Movers.

**Example:**

To configure DNS service for the `server_2` Data Mover as a member of the `adnative.com` domain on the DNS server from IP address 10.5.44.29, type:

```bash
$ server_dns server_2 adnative.com 10.5.44.29
```

**Output**

```
server_2 : done
```

### For a Windows NT 4.0 environment

**Action**

To configure WINS for the Data Mover, use this command syntax:

```bash
$ server_cifs {<movername>|ALL} -add wins=<ip_addr> [,wins=<ip_addr>...]
```

where:
- `<movername>` = specified Data Mover name
- `-add wins=<ip_addr> [,wins=<ip_addr>...]` = adds WINS servers to the CIFS configuration

The **ALL** option executes the command for all of the Data Movers.

**Example:**

To configure the `server_2` Data Mover to access WINS services from IP address 10.5.44.23, type:

```bash
$ server_cifs server_2 -add wins=10.5.44.23
```

**Output**

```
server_2 : done
```
3. Enable Unicode for the Data Mover.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To enable Unicode for the Data Mover, use this command syntax:</td>
</tr>
<tr>
<td>$ /nas/sbin/uc_config -on -mover &lt;movername&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = specified Data Mover name</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To enable Unicode on the server_2 Data Mover, type:</td>
</tr>
<tr>
<td>$ /nas/sbin/uc_config -on -mover server_2</td>
</tr>
<tr>
<td>Output</td>
</tr>
<tr>
<td>server_2 : done</td>
</tr>
</tbody>
</table>

4. Verify the server security mode.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To verify the server security mode, use this command syntax:</td>
</tr>
<tr>
<td>$ server_cifs {&lt;movername&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = specified Data Mover name</td>
</tr>
<tr>
<td>The <strong>ALL</strong> option executes the command for all of the Data Movers.</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To verify the security mode on the server_2 Data Mover, type:</td>
</tr>
<tr>
<td>$ server_cifs server_2</td>
</tr>
</tbody>
</table>
5. If security mode is not set to NT on the Windows client, use the following command syntax:

   $ server_cifs <movername> -add security=NT

6. Configure migration interfaces between the CIFS source file server and the Data Mover with IP addresses.
To configure interfaces between the CIFS source file server and Data Mover, use this command syntax:

```
$ server_ifconfig {<movername>|ALL} -create -Device <device_name> -name <if_name> -protocol {IP <ipaddr> <ipmask> <ipbroadcast>|ATMLEC <elan>}
```

where:

- `<movername>` = specified Data Mover name
- `<device_name>` = device name containing the specified configuration
- `<if_name>` = usually the same as the device name, but can be any user-chosen identifier
- `<ipaddr>` = assigns the IP protocol with the specified IP address, mask, and broadcast address
- `<ipmask>` = includes the network part of the local address and the subnet, which is taken from the host field of the address
- `<ipbroadcast>` = special destination address specifying a broadcast message to a network
- `<elan>` = creates a virtual device on the specified device where the elan is the emulated LAN name to join for this interface

The `ALL` option executes the command for all of the Data Movers.

Example:

To configure interfaces between the `server_2` Data Mover and the `ana0` source file server device with 10.5.44.89 and 255.255.255.0.10.5.44.255 IP addresses, respectively, type:

```
$ server_ifconfig server_2 -create -Device ana0 -name ana0 -protocol IP 10.5.44.89 255.255.255.0.10.5.44.255
```

Output

```
server_2 : done
```
7. Configure the default gateway, if not already completed.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To configure the default gateway, use this command syntax:</td>
</tr>
</tbody>
</table>
| $ server_route {<movername>|ALL} {-add|-delete} default
   <gateway> |
| where: |
| <movername> = specified Data Mover name |
| <gateway> = adds a default gateway for the specified destination |
| The ALL option executes the command for all of the Data Movers. |
| Example: |
| To configure a 10.5.44.15 IP gateway address for the server_2 Data Mover, type: |
| $ server_route server_2 -add default 10.5.44.15 |
| Output |
| server_2 : done |

8. Set up user authentication, either Internal or External Usermapper.

Configuring VNX User Mapping provides more information on configuring Usermapper.

Create the MGFS file system. Be sure to calculate the size of the migration file system, based on all drives from the CIFS source file server. Include growth factors per local requirements or generally accepted guidelines. The migration file system size calculation also must account for the 8 KB block size of the VNX. When moving data from a Windows environment, there can be as much as double the storage requirements.

Configure the Data Mover by using a secure, encrypted, remote login application interface on the Windows client.

Note: You can add a new MGFS file system by using the Unisphere Data Migration GUI.

Managing Volumes and File Systems with VNX Automatic Volume Management and Managing Volumes and File Systems for VNX Manually provide more information about creating volumes, file systems, and mount points.
Perform the following:

1. Create striped volumes, if required.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To create striped volumes, use this command syntax:</td>
</tr>
<tr>
<td>$ nas_volume [-name &lt;name&gt;] -create [-Stripe [&lt;stripe_size&gt;]</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;name&gt; = name assigned to the volume</td>
</tr>
<tr>
<td>-Stripe &lt;stripe_size&gt;</td>
</tr>
<tr>
<td>If the Stripe option is specified, a stripe depth must be typed in multiples of 8,192 bytes.</td>
</tr>
<tr>
<td>&lt;GB&gt; = if size is specified (between 1 and 1024 GB), a volume match is found equal to or greater than the specified size</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To create 32768 byte stripes on d3, d4, d5, and d6 disks and combine them into one striped volume named stv1, type:</td>
</tr>
<tr>
<td>$ nas_volume -name stv1 -create -Stripe 32768 d3,d4,d5,d6</td>
</tr>
<tr>
<td>Output</td>
</tr>
<tr>
<td>id = 246</td>
</tr>
<tr>
<td>name = stv1</td>
</tr>
<tr>
<td>acl = 0</td>
</tr>
<tr>
<td>in_use = False</td>
</tr>
<tr>
<td>type = stripe</td>
</tr>
<tr>
<td>stripe_size = 32768</td>
</tr>
<tr>
<td>volume_set = d3,d4,d5,d6</td>
</tr>
<tr>
<td>disks = d3,d4,d5,d6</td>
</tr>
</tbody>
</table>
2. Create a metavolume.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To create a metavolume, use this command syntax:</td>
</tr>
<tr>
<td>$ nas_volume [-name &lt;name&gt;] -create [-Stripe [stripe_size]] [-Meta] [(volume_name),...]</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;name&gt; = name assigned to a volume</td>
</tr>
<tr>
<td>-Stripe stripe_size</td>
</tr>
<tr>
<td>&lt;volume_name&gt; = volume table entry name from the set of volumes</td>
</tr>
<tr>
<td>&lt;GB&gt; = volume table entry volume size</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To create a metavolume named mtv1 (on which you later create an MGFS) on the stv1 volume, type:</td>
</tr>
<tr>
<td>$ nas_volume -name mtv1 -create -Meta stv1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>id = 247</td>
<td></td>
</tr>
<tr>
<td>name = mtv1</td>
<td></td>
</tr>
<tr>
<td>acl = 0</td>
<td></td>
</tr>
<tr>
<td>in_use = False</td>
<td></td>
</tr>
<tr>
<td>type = meta</td>
<td></td>
</tr>
<tr>
<td>volume_set = stv1</td>
<td></td>
</tr>
<tr>
<td>disks = d3,d4,d5,d6</td>
<td></td>
</tr>
<tr>
<td>• The default volume is -Meta.</td>
<td></td>
</tr>
<tr>
<td>• Volumes are initially configured as disk volumes.</td>
<td></td>
</tr>
</tbody>
</table>
3. Create the MGFS on the Data Mover.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To create the MGFS, use this command syntax:</td>
</tr>
</tbody>
</table>
| ```
$ nas_fs [-name <name>] [-type <type>] -create
({<volume_name>|size=<integer>[G|M] pool=<pool>})[-option <options>]
```
where:
- `<name>` = name assigned to a file system
- `<type>` = type assigned to the file system. Valid types are uxfs, rawfs, mgfs. This must be mgfs for a CDMS migration. The default value is uxfs.
- `-create` (`{<volume_name>|size=G pool=<pool>}`) = creates a file system on the specified metavolume, or available metavolume with the specified size
- `<options>` = specifies a comma-separated list of characteristics to create a file system

Example:
To create an mgfs file system type named mgfs1 on the mtv1 metavolume, type:
```
$ nas_fs -name mgfs1 -type mgfs -create mtv1
```

<table>
<thead>
<tr>
<th>Output</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>id = 18</td>
<td></td>
</tr>
<tr>
<td>name = mgfs1</td>
<td></td>
</tr>
<tr>
<td>acl = 0</td>
<td></td>
</tr>
<tr>
<td>in_use = False</td>
<td></td>
</tr>
<tr>
<td>type = mgfs</td>
<td></td>
</tr>
<tr>
<td>volume = mtv1</td>
<td></td>
</tr>
<tr>
<td>profile =</td>
<td></td>
</tr>
<tr>
<td>rw_servers =</td>
<td></td>
</tr>
<tr>
<td>ro_servers =</td>
<td></td>
</tr>
<tr>
<td>symm_devs = 002806000209-006,002806000209-007,002806000209-008,002806000209-009</td>
<td></td>
</tr>
<tr>
<td>disks = d3,d4,d5,d6</td>
<td></td>
</tr>
</tbody>
</table>

The default file system type is uxfs.
4. Create a mount point by using the server name of the CIFS source file server.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
</table>
| To create a mount point on the Data Mover, use this command syntax:  
$ server_mountpoint <movername>  
[-create | -delete | -exist] <pathname>  
where:  
<movername> = Data Mover name to which you want to create a mount point  
<pathname> = directory pathname on which the file system is mounted. Up to seven mount points can be created under a pathname. All pathnames must begin with a forward slash (/).  
Example:  
To create a mount point on the server_2 Data Mover with a pathname of /server1, type:  
$ server_mountpoint server_2 -create /server1 |

<table>
<thead>
<tr>
<th>Output</th>
<th>Note</th>
</tr>
</thead>
</table>
| server_2 : done | server_2 is the VNX to which you are migrating data.  
/server1 is the file system mount point to which you are migrating data. |

5. Mount the MGFS on the Data Mover.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
</table>
| To mount the MGFS on the Data Mover, use the following command syntax:  
$ server_mount <movername> [-option <options>]  
<fs_name> [ <mount_point> ]  
where:  
<movername> = name of the Data Mover  
<options> = specifies a comma-separated list of mount options  
<fs_name> = file system name  
<mount_point> = specified mount point. The mount point must begin with a forward slash (/).  
Example:  
To mount the pmacF30 file system on the server_2 Data Mover by using the /pmac3 mount point, type:  
$ server_mount server_2 -option rw pmacF30 /pmac3 |

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 : done</td>
</tr>
</tbody>
</table>
6. Check that the file system is mounted properly.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To check if the file system is mounted correctly and recognized as a migration file system, use this command syntax:</td>
</tr>
<tr>
<td>$ server_cdms {&lt;movername&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = name of the Data Mover</td>
</tr>
<tr>
<td>&lt;mgfs&gt; = file system name</td>
</tr>
<tr>
<td>Verify that the MGFS special information lines look correct.</td>
</tr>
<tr>
<td>The ALL option displays the attributes of all file systems, including the configuration of associated disks.</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To check to see if the pmacF30 file system is mounted properly, type:</td>
</tr>
<tr>
<td>$ server_cdms server_2 -info pmacF30</td>
</tr>
<tr>
<td>You should see rw_servers and MGFS information added to the information output.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 :</td>
</tr>
<tr>
<td>pmacF30:</td>
</tr>
</tbody>
</table>

**Step 8: Preparing for migration**

Perform the following steps to prepare the source file server and Windows client for migration:

1. Set required rights for the lgdup.exe utility migration account on the CIFS source file server and the Windows client.

   To perform this task, use either the:

   
   or

   • User Manager for Domains (Windows NT 4.0)

   Figure 13 on page 148 and Figure 14 on page 148 illustrate this.

   Two rights are required for the lgdup.exe utility:

   • Generate security audits
   
   • Manage auditing and security logs
CIFS

Ensure that these user rights changes are performed once on the CIFS source file server and once on the Windows client.

**Figure 13** Local Security Settings: Windows 2000 or Windows Server 2003 system

**Figure 14** User Rights Policy: Windows NT 4.0 system
2. Using the appropriate administrative tool on the CIFS source file server (for example, Manage Computer in Windows 2000 and Windows Server 2003 or User Manager for Domains in Windows NT 4.0), place the cdms_migrator account in the backup operator group.

3. Log out and log in again on the CIFS source file server and Windows client to ensure that these rights and memberships are invoked correctly.

4. Restrict network access on the CIFS source file server by using either:
       or
   • User Manager for Domains (Windows NT 4.0)

   Figure 15 on page 150 and Figure 16 on page 151 illustrate this.
CIFS

a. If migrating from a Windows 2000 or Windows Server 2003 environment, use the Local Security Policy administrative tool:
   - Assign the **Deny Access to this computer from the network** right to the Domain Users group.

   ![Local Security Policy](image)

   **Figure 15** Restricting access on source file server: Windows 2000 or Windows Server 2003 system

b. If migrating from a Windows NT 4.0 environment, use the User Manager for Domains administrative tool for Windows NT 4.0 systems:
   - Grant **Access to this computer from the network** to the cdms_migrator account.
   - Remove all other groups and users.
Figure 16  Restricting access on source file server: Windows NT 4.0 system

5. Rename the CIFS source file server, retaining the original name for use with the Data Mover.

6. Restart the CIFS source file server.

7. For Windows NT 4.0 environments, remove the original source file server name from the domain by using Server Manager.

   **Note:** Due to the Microsoft domain database update process, removing a server name from the domain can take as long as 45 minutes to complete. Be prepared to wait.
Step 9: Setting up the CIFS environment

Configure the Data Mover by using a secure, encrypted, remote login application interface on the Windows client. Configuring and Managing CIFS on VNX provides more details.

To configure the Data Mover for CIFS migration, perform the following.

1. Verify connections between the Data Mover and the CIFS source file server by performing the following:
   a. Ping the CIFS source file server from the Data Mover.
   b. Ping the Data Mover from the CIFS source file server.

2. Set up the migration Data Mover with the CIFS original source file server name.
   The server name created on the Data Mover is also known as the target server.

For a Windows 2000 or Windows Server 2003 environment

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To set up the Data Mover for a Windows 2000 or Windows Server 2003 environment, use this command syntax:</td>
</tr>
<tr>
<td>$ server_cifs &lt;movername&gt; -add compname=&lt;comp_name&gt;,domain=&lt;full_domain_name&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = Data Mover name being configured</td>
</tr>
<tr>
<td>&lt;comp_name&gt; = Windows 2000 or Windows Server 2003 compatible CIFS server. It can be up to 63 UTF8 characters.</td>
</tr>
<tr>
<td>&lt;full_domain_name&gt; = NetBIOS name is assigned to the specified domain name</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To configure the server_2 Data Mover with a server1 compname, type:</td>
</tr>
<tr>
<td>$ server_cifs server_2 -add compname=server1,domain=adnative.com</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 : done</td>
</tr>
</tbody>
</table>

If the Data Mover emulates multiple Microsoft servers, a more comprehensive command might be necessary. The EMC VNX Command Line Interface Reference for File provides more information about the `server_cifs` command.
For a Windows NT 4.0 environment

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To set up the Data Mover for a Windows NT 4.0 environment, use this command syntax:</td>
</tr>
<tr>
<td>Using the Server Manager for Domains, delete the original server name from the domain and synchronize PDCs and BDCs.</td>
</tr>
<tr>
<td>Using the Server Manager for Domains, add the original server name back into the domain and synchronize PDCs and BDCs.</td>
</tr>
<tr>
<td>Add the NetBIOS name and the WINS server to the migration Data Mover.</td>
</tr>
</tbody>
</table>

```
$ server_cifs <movername> -add netbios=<netbios_name>,domain=<domain_name> [,hidden={y|n}][,[interface=<if_name>],[wins=<ip>][<ip>]] ...
[,-comment <comment>]
```

where:

- `<movername>` = Data Mover name being configured
- `<netbios_name>` = replaces the default NetBIOS name assigned automatically and derived from `<comp_name>`
- `<domain_name>` = NetBIOS name is assigned to the specified domain name
- `<if_name>` = specifies an interface for the CIFS source file server
- `<ip>` = associates one or more WINS IP addresses with each interface
- `<comment>` = assigns a comment enclosed with quotes to the configuration

Example:
To configure the server_2 Data Mover with a server1 NetBIOS name, type:
```
$ server_cifs server_2 -add netbios=server1,domain=ntnative.com,wins=10.5.44.14
```

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 : done</td>
</tr>
</tbody>
</table>
3. For Windows environments, ensure that all servers have joined the domain.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To join the domain in a Windows environment, use this command syntax: $ \texttt{server_cifs} \texttt{&lt;movername&gt;} -Join \texttt{compname=}\texttt{&lt;comp_name&gt;,domain=}\texttt{&lt;full_domain_name&gt;,admin=}\texttt{&lt;admin_name&gt;} where: &lt;movername&gt; = specified Data Mover name &lt;comp_name&gt; = name of the Windows-compatible CIFS server. It can be up to 63 UTF8 characters &lt;full_domain_name&gt; = NetBIOS name is assigned to the specified domain name &lt;admin_name&gt; = login name of the user with administrative rights in the domain. The user is prompted to input a password for the admin account. Example: To allow server_2 to join server1 in the adnative.com domain in a Windows environment, type: $ \texttt{server_cifs} \texttt{server_2} -Join \texttt{compname=}\texttt{server1, domain=}\texttt{adnative.com,admin=}\texttt{administrator}. If in a Windows NT 4.0 environment, add the NetBIOS name to the Domain and identify the WINS server.</td>
</tr>
<tr>
<td>Output</td>
</tr>
<tr>
<td>server_2 : done</td>
</tr>
</tbody>
</table>

If the server is not a member of the domain, the server is not seen as a server in that domain, and any resources the server shares cannot be accessed by users (or clients) in the domain.

For example, server A and server B do not see the servers or resources because server A logged in to the CORP domain and server B logged in to the Engineering domain.

The *EMC VNX Command Line Interface Reference for File* provides more information about the \texttt{server\_cifs} command.
4. Start the CDMS migration service.

If the CDMS migration service is already started on the Data Mover, you might not have to perform this task.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To start the CDMS migration service, use this command syntax:</td>
</tr>
<tr>
<td>$ server_setup &lt;movername&gt; -Protocol -option start[=n]</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = specified Data Mover name</td>
</tr>
<tr>
<td>&lt;n&gt; = number of threads for users. Default number of CDMS threads is 32.</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To start CIFS for the server_2 Data Mover with 25 threads, type:</td>
</tr>
<tr>
<td>$ server_setup server_2 -Protocol cdms -option start=25</td>
</tr>
</tbody>
</table>

Output

server_2 : done

**Step 10: Migrating local group information**

This step requires scripts and utilities on the CIFS source file server, the Windows client, and the target VNX. In addition, ensure that the `server_cdms` command is available on the Control Station in the /nas/bin directory.

To migrate the local group information, perform the following:

1. Set required rights for the cdms_migrator account on the target file server as was done previously for the CIFS source file server and the Windows client.

   To perform this task, use the User Manager for Domains in Windows NT 4.0 administrative tool, as shown in Figure 17 on page 156.

   **Note:** The User Manager for Domains tool must be accessed on the domain controller. If you are in a Windows 2000 or Windows Server 2003 environment, select **Start > Run**, and type the `usrmgr` program name. User Manager for Domains allows you to manage remote domains. Select the **Domain** option from the **User** menu, and then type the target file server name.
Two rights are required for the \texttt{lgdup.exe} utility:

- Generate security audits
- Manage auditing and security log

\textbf{Figure 17} User Rights Policy: Windows NT 4.0 system

2. Place the \texttt{cdms\_migrator} account in the backup operator group on the target file server.

3. On the Windows client, log out and log in again to ensure that these rights and memberships are invoked correctly.

4. From a command window on the Windows client, run the \texttt{lgdup.exe} utility to migrate the local group information.
Note: The *lgdup.exe* utility preserves any groups or domain user accounts. The utility duplicates source file server local rights on the Data Mover, provided that the Data Mover supports the same rights. The VNX supports all standard Windows 2000 and Windows Server 2003 rights.
You should carefully examine the errors in the LGDUP log file because several issues reported in the duplication process by LGDUP do not have any impact on the effectiveness or functionality of the overall migration. For example, the following are acceptable but reported as errors by LGDUP:

- Source local group contains a member that is already configured in the destination local group.
- Source local group contains a SID that cannot be resolved at the destination server (usually from a deleted user or a user from a domain that is not trusted).
- User rights are not copied from a NetApp (Network Appliance servers do not implement Windows local user rights).
- Source local group contains a member that is only a local user on the source server, so cannot exist on the destination server.

However, other errors may affect the functionality of the migration.

⚠️ CAUTION ⚠️

The lgdup.exe utility duplicates the privilege settings of groups and users. However, each Data Mover’s NetBIOS name has one user rights file. In many-to-one migrations, if there are duplicate users on different servers, the last server where the utility was run overwrites the user rights file. Therefore, you must plan which server needs to retain the user rights when running the lgdup.exe utility without the -NOPRIV option.

For servers from which you want to remove the user rights duplication, you must run the utility with the -NOPRIV option.

Create CDMS connections for each CIFS source file server disk drive. Since there are two disk drives (C: and D:) in the following tables, the `server_cdms` command must be run once for each drive on the CIFS source file server.

The WINS server is not used in Windows 2000 or Windows Server 2003 if resolutions are handled by the DNS service.

Configure the Data Mover by using a secure, encrypted, remote login application interface on the Windows client.
For a Windows 2000 or Windows Server 2003 environment

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To create CDMS connections for each CIFS source file server disk drive, use this command syntax:</td>
</tr>
</tbody>
</table>
| $ server_cdms <movername> -connect <mgfs> -type cifs -path <localpath> -netbios <netbios> -source \\
srcServer[.<domain>]<srcShare>[<srcPath>] -admin [domain]<admin_name> [-wins <wins>] |

where:

- `<movername>` = specified Data Mover name
- `<mgfs>` = migration file system name
- `<localpath>` = subdirectory name (created if it does not exist; if it exists, it fails) within the mount point of the file system. You cannot connect to the file system mount point, only to the subdirectory.
- `<netbios>` = NetBIOS name of the Data Mover-CIFS server name (since it can have more than one)
- `<srcServer>[.<domain>]<srcShare>[<srcPath>]` = `<srcServer>` is the CIFS source file server name, `<srcShare>` is the CIFS source file server share name, and `<srcPath>` allows migration that is not at the root of the share. If specified, the root of the migration is `\share\dir1...` instead of just `\share`.
- `<domain><admin_name>` = domain name, and the name of the administrator to connect as [a password is asked interactively when the command is executed to hide (or mask) the password]
- `<wins>` = IP address of the WINS server, only required for Windows NT 4.0.

Example:

If you have a Windows 2000 or Windows Server 2003 source file server configured with two disk drives, C: and D:, the commands would be similar to the following:

For drive C:

```
$ server_cdms server_2 -connect mgfs1 -type cifs -path /c -netbios server1.adnative.com -source \server1_old.adnative.com\c$ -admin adnative.com\cdms_migrator
```

For drive D:

```
$ server_cdms server_2 -connect mgfs1 -type cifs -path /d -netbios server1.adnative.com -source \server1_old.adnative.com\d$ -admin adnative.com\cdms_migrator
```
<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 : done</td>
</tr>
</tbody>
</table>
For a Windows NT 4.0 environment

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To create CDMS connections for each CIFS source file server disk drive, use this command syntax:</td>
</tr>
<tr>
<td>$ server_cdms &lt;movername&gt; -connect &lt;mgfs&gt; -type cifs -path &lt;localpath&gt; -netbios &lt;netbios&gt; -source \&lt;srcServer&gt;[.&lt;domain&gt;]&lt;srcShare&gt;[&lt;srcPath&gt;] -admin [&lt;domain&gt;]&lt;admin_name&gt; [-wins &lt;wins&gt;]</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = specified Data Mover name</td>
</tr>
<tr>
<td>&lt;mgfs&gt; = migration file system name</td>
</tr>
<tr>
<td>&lt;localpath&gt; = subdirectory name (created if it does not exist; if it exists, it fails) within the mount point of the file system. You cannot connect to the file system mount point, only to the subdirectory.</td>
</tr>
<tr>
<td>&lt;netbios&gt; = NetBIOS name of the Data Mover-CIFS server name (since it can have more than one)</td>
</tr>
<tr>
<td>\&lt;srcServer&gt;[.&lt;domain&gt;]&lt;srcShare&gt;[&lt;srcPath&gt;] = &lt;srcServer&gt; is the CIFS source file server name, &lt;srcShare&gt; is the CIFS source file server share name, and &lt;srcPath&gt; allows migration that is not at the root of the share. If specified, the root of the migration is \share\dir... instead of just \share.</td>
</tr>
<tr>
<td>&lt;domain&gt;&lt;admin_name&gt; = domain name, and the name of the administrator to connect as [a password is asked interactively when the command is executed to hide (or mask) the password]</td>
</tr>
</tbody>
</table>

**Note:** The -source and -admin syntax strings must be enclosed in single quotation marks. An example is: ‘\\myserver\myshare\mypath’.

| <wins> = IP address of the WINS server, only required for Windows NT 4.0 |
| Example: |
| If you have a Windows NT 4.0 source file server configured with two disk drives, C: and D:, the commands would be similar to the following: |
| For drive C: |
| $ server_cdms server_2 -connect mgfs1 -type cifs -path /c -netbios server1.adnative.com -source ‘\\server1_old.adnative.com\c$’ -admin ‘adnative.com\cdms_migrator’ -wins 10.5.44.23 |
| For drive D: |
| $ server_cdms server_2 -connect mgfs1 -type cifs -path /d -netbios server1.adnative.com -source ‘\\server1_old.adnative.com\d$’ -admin ‘adnative.com\cdms_migrator’ wins 10.5.44.23 |
If the connection command fails, an appropriate error message appears.

Note: After the server_cdms command is executed, you are prompted for the password of the <admin_name> you are using. In this example, it is cdms_migrator. In addition, if you want to remove user rights, use the -NOPRIV option.

In the event of an error, the server log’s content and “Removing an unwanted connection” on page 232 provide more details.

Check that the CDMS source file server is connected properly to the destination.

To check if the CDMS source file server is connected properly to the destination, use this command syntax:

```
$ server_cdms {<movername>|ALL} -info [<mgfs>]
```

where:

- `<movername>` = name of the Data Mover
- `<mgfs>` = name of the migration file system

The `ALL` option displays the attributes of all file systems, including the configuration of associated disks.

Example:

To check if the `pmacF30` file system is connected properly to the `server_2`, type:

```
$ server_cdms server_2 -info pmacF30
```

You should see that the first connection information appears in the MGFS lines of the output display.

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 :</td>
</tr>
<tr>
<td>pmacF30:</td>
</tr>
<tr>
<td>path    = /dest1</td>
</tr>
<tr>
<td>cid     = 0</td>
</tr>
<tr>
<td>type    = nfs</td>
</tr>
<tr>
<td>source  = 10.100.14.50:/home1</td>
</tr>
<tr>
<td>options =</td>
</tr>
</tbody>
</table>
Step 12: Executing the \sharedup\exe utility

Execute the \sharedup\exe utility against each disk drive from the CIFS source file server, as described in the following table.

Configure the Data Mover by using a secure, encrypted, remote login application interface on the Windows client.

---

**Note:** The local path should correspond to the <localpath> used with the server_cdms <movername> -connect <mgfs> command in “Step 11: Migrating local group information” on page 194.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To execute the \sharedup\exe utility against each disk drive from the CIFS source file server, use this command syntax:</td>
</tr>
<tr>
<td>C:&gt; \sharedup  &lt;source&gt;  &lt;target&gt;  &lt;sourcedrive&gt;  /SD  /P:&lt;mountpointname&gt;&lt;localPath&gt;  /LOG:&lt;logfilename&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;source&gt; = CIFS source file server name</td>
</tr>
<tr>
<td>&lt;target&gt; = target Data Mover name</td>
</tr>
<tr>
<td>&lt;sourcedrive&gt; = drive letter of the CIFS source file server containing shares to be migrated</td>
</tr>
<tr>
<td>/SD = causes the \sharedup\exe utility to transfer all ACLs</td>
</tr>
<tr>
<td>/P: = identifies the path to which the shares are migrated</td>
</tr>
<tr>
<td>&lt;mountpointname&gt; = mount point name of the MGFS (with no slash before mount point)</td>
</tr>
<tr>
<td>&lt;localPath&gt; = local pathname identified with the server_cdms command</td>
</tr>
<tr>
<td>/LOG: = log file optional parameter</td>
</tr>
<tr>
<td>&lt;logfilename&gt; = log filename</td>
</tr>
</tbody>
</table>

“\sharedup\exe utility” on page 284 provides more information about the \sharedup\exe utility.

Example:

If you have a source file server configured with two disk drives, C: and D:, the commands would be similar to the following:

For drive C:

C:\> \sharedup \server1_old \server1  c:  /SD  \/P:server1\c  /LOG:server1-c-shares-log.txt

For drive D:

C:\> \sharedup \server1_old \server1  d:  /SD  \/P:server1\d  /LOG:server1-d-shares-log.txt
The previous example assumes you are moving shares from an entire disk drive to a local path on the VNX to which the drive content is being migrated. The example is not intended for the case where you are moving shares from the same source drives to different Data Movers.

**Note:** There is a relationship between the modified \server1\c file and the sharedup.exe utility where the /P:server1\c option is used from the sharedup.exe utility command line.

---

**Output**

```
SHAREDUP 01.06
Copyright (c) 2004, All Right Reserved,
by EMC Corporation, Hopkinton, MA.
Source server: server1_old    5.0
Target server: server1        5.0
******************************************************************************
SHAREDUP source: server1_old    target: server1
Summary results:
Elapsed time: hours:00,mins: 00,secs:00
Number of share(s) successfully duplicated on drive c: 6
```

```
SHAREDUP 01.06
Copyright (c) 2004, All Right Reserved,
by EMC Corporation, Hopkinton, MA.
Source server: server1_old    5.0
Target server: server1        5.0
******************************************************************************
SHAREDUP source: server1_old    target:: server1
Summary results:
Elapsed time: hours:00,mins: 00,secs:00
Number of share(s) successfully duplicated on drive d: 4
```

At this point, all shares have been exported externally, and are available on the VNX for Windows client access.

**Step 13: Monitoring and waiting**

Check on migration progress before running the `server_cdms` command to copy all file systems to the VNX.
Step 14: Ensuring that all data is migrated

At the beginning of this step, the MGFS contains some directories and files copied from the CIFS source file servers by the action of individual Windows clients.

Note: EMC recommends you use the `server_cdms` command during a period of low user activity because response time from the source file server might be slow as large amounts of data are migrated. As the command runs, response time improves because more data is being accessed from the VNX rather than from the CIFS source file server.

However, you still need to migrate all untouched files from the CIFS source file server to the VNX. The `server_cdms <movername> -start <mgfs>` command can be run when the file systems are not mounted on the Control Station or the Windows client.

To start the internal migration, run the `server_cdms` command, as described in the following table.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To start the CDMS internal migration, use this command syntax:</td>
</tr>
<tr>
<td><code>$ server_cdms &lt;movername&gt; -start &lt;mgfs&gt; -path &lt;localpath&gt; [-Force] -log &lt;logpath&gt; [-include &lt;include_path&gt;] [-exclude &lt;exclude_file&gt;]</code></td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td><code>&lt;movername&gt;</code> = name of the Data Mover</td>
</tr>
<tr>
<td><code>&lt;mgfs&gt;</code> = CIFS source file server name from which the files are to be migrated</td>
</tr>
<tr>
<td><code>&lt;localpath&gt;</code> = local path on the Data Mover, within this MGFS, on which the internal migration thread starts processing</td>
</tr>
<tr>
<td><code>&lt;logpath&gt;</code> = directory path on a file system mounted or accessible on the Data Mover. Internal migration and error log files are created in this directory. Do not include this directory in the MGFS.</td>
</tr>
<tr>
<td><code>&lt;include_path&gt;</code> = migrate only entries defined in the &lt;file&gt;</td>
</tr>
<tr>
<td><code>&lt;exclude_path&gt;</code> = skip all entries defined in the &lt;file&gt;</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To execute the <code>server_cdms</code> command to force complete migration of the contents of the <code>/home1</code> directory (the <code>/dest1</code> subdirectory of <code>pmacF30</code> file system), type:</td>
</tr>
<tr>
<td><code>$ server_cdms server_2 -start pmacF30 -path /home1 -log /mgfslog</code></td>
</tr>
</tbody>
</table>

Note: You can check on the progress of a migration by using the Unisphere Data Migration GUI.
The command starts a migration thread, populates a directory tree, and then begins migrating data to the MGFS. You can use include and exclude files separately or concurrently.

The **-Force** option is used only if you need to start a migration thread a second time on the same local path where a previous migration thread had already finished. For example, **-Force** would be needed to start a thread that had no include file (that is, to migrate all remaining files) on a local path where a thread with an include file had already been run.

**Note:** You can start or stop migration threads and download include and exclude files by using the Unisphere Data Migration GUI or the CLI. You can view migrations to an MGFS by using the Unisphere Data Migration GUI.

Filenames appear in the client window as files are migrated to the Data Mover.

This process might take a significant amount of time, depending upon:

- Migrated file system size
- Type of server on which the source resides
- Network speed
- VNX software version
- Network traffic

**Step 15: Examining the error log**

Before you run the **server_cdms** command, as described in “Step 16: Verifying migration completion” on page 169, check the migration log file, and convert all errors. The migration log and error log are automatically created by using the following naming convention:

- **migLog_path**
- **migErr_path**

```
$ ls -l /nas/rootfs/slot_2/s2ufs2/log
total 16
-rw-r--r-- 1 root bin 0 Sep 30 2003 migErr_demo_mgfs_nfsv3
-rw-r--r-- 1 root bin 10556 Sep 30  2003 migLog_demo_mgfs_nfsv3
```

Output

```
$ server_2 : done
```
Note: You can download error log content by using the Unisphere Data Migration GUI.
If the error log is not empty, follow these steps:

1. Extract all pathnames from the error log by using the following command syntax:
   
   `$ cat <logname> | grep Fullpath`  
   
   where:
   
   `<logname>` = name of the error log file produced by the `server_cdms` command. The last line of each error description shows the full path of the erroneous file.

2. Mount the source on the local UNIX workstation or the Control Station.

3. For each pathname in the error log file, determine if it can be remotely accessed on the NFS source file server.

4. Do one of the following:
   
   • If the source directory or file is readable, migrate the file or directory by using the `server_cdms` command on the Data Mover version of this pathname.
   
   • If the source directory or file is unreadable, there is an error on the source file server. Retain a list of problem source files for this file system.

5. Do one of the following:
   
   • If you have completely migrated the log file error list’s content, determine if there are any listed problem source files.
   
   • If you have not completely migrated the content of the log file error list, return to “Step 4: Identifying high-priority files (optional)” on page 135.

6. If problem source files are present, perform the following:
   
   a. Deliver the errors corresponding to these files and their pathnames to the customer.
   
   b. Work with the customer to determine a suitable method to handle these files such as copying by another method, or deleting them on the VNX.
7. When all problem source files have been handled, check to see if the error log is empty:

- If the log is still not empty, go back to “Step 1: Creating an account” on page 127.
- If the log is empty, go to “Step 16: Verifying migration completion” on page 169.

Step 16: Verifying migration completion

Open a client window to run a continuous server log, and then convert the file system. Monitor and verify migration progress, as described in the following tables.

Configure the Data Mover by using a secure, encrypted, remote login application interface on the Windows client.

⚠️ CAUTION

Do not run a verify command if there are any migration threads running or active on other CDMS connections in the same file system. Any CDMS data transfer to the same file system, even if it is on a different path from the one being verified, causes the verify process to restart from the beginning of the path given in the verify command. Running a CDMS migration thread typically produces near-continuous CDMS transfers. This causes near-continuous restarts of the verify command and therefore, high Data Mover CPU loads, which results in problems for all other users of that Data Mover. Because the verify command is a non-interruptible command, this condition persists until all migration threads are stopped on that file system or the verify command fails. Running a verify command when there are no active migration threads avoids these problems. Normally, a verify command is run only at the end of the migration process, when all migration threads have completed satisfactorily, so this is usually not an issue.

Do not run a verify command if there is another verify command still in progress on a different file system on the same Data Mover. Either one or both of the commands will fail, and you will have to run them again.
Monitoring progress

Action

To monitor the progress of the migration during conversion, use this command syntax:

```
$ server_log <movername> [-a][-f][-n][-s]
```

where:

- `<movername>` = CIFS source file server name
- `-a` = displays the complete log
- `-f` = monitors the growth of the log by entering into an endless loop, pausing, and reading the Data Mover-generated log
- `-n` = displays the log without the timestamp
- `-s` = displays the time in yyyy-mm-dd format when each command in the log was executed

Example:

To check on the migration status of the `server_2` source file server by displaying the migration log, type:

```
$ server_log server_2 -f
```

Output

```
NAS LOG for slot 2:
-------------------
1013710123: ADMIN:  4: Command succeeded: logsys add output disk=root_log_2 bufsz=256
1013710123: ADMIN:  4: Command succeeded: logsys add output event bufsz=2048
1013710123: ADMIN:  4: Command succeeded: bufcache
1013710123: ADMIN:  4: Command succeeded: device isa isa-0
1013710123: KERNEL: 3: PCI BIOS Rev 02.10
1013710123: KERNEL: 4: CMB-100 Motherboard
1013710123: ADMIN:  4: Command succeeded: device pci pci-0
1013710123: ADMIN:  4: Command succeeded: dskdumpconfig full slot=2
1013710123: DRIVERS: 4: scsi-0 (AHA3944AUW Ch: A) @ 1400, irq 7, bus 0, func 0
1013710123: DRIVERS: 4: scsi-1 (AHA3944AUW Ch: B) @ 1800, irq f, bus 0, func 1
1013710125: DRIVERS: 4: scsi-2 (AHA3944AUW Ch: A) @ 2000, irq f, bus 0, func 0
1013710126: DRIVERS: 4: scsi-3 (AHA3944AUW Ch: B) @ 2400, irq f, bus 0, func 1
```
**Note:** Exit by pressing [CTRL] and [C] together.

**Note:** You can monitor the progress of a migration by using the Unisphere Data Migration GUI.

### Verifying migration completion

Do not run a verify command if there are any migration threads running or active on other CDMS connections in the same file system. Any CDMS data transfer to the same file system, even if it is on a different path from the one being verified, causes the verify process to restart from the beginning of the path given in the verify command. Running a CDMS migration thread typically produces near-continuous CDMS transfers. This causes near-continuous restarts of the verify command and therefore, high Data Mover CPU loads, which results in problems for all other users of that Data Mover. Because the verify command is a non-interruptible command, this condition persists until all migration threads are stopped on that file system or the verify command fails. Running a verify command when there are no active migration threads avoids these problems. Normally, a verify command is run only at the end of the migration process, when all migration threads have completed satisfactorily, so this is usually not an issue.

Do not run a verify command if there is another verify command still in progress on a different file system on the same Data Mover. Either one or both of the commands will fail, and you will have to run them again separately.
To verify the status of a migration, use this command syntax:

```
$ server_cdms <movername> -verify <mgfs> [-path
{<localpath>|<cid>})
```

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To verify the status of a migration, use this command syntax:</td>
</tr>
<tr>
<td>$ server_cdms &lt;movername&gt; -verify &lt;mgfs&gt; [-path {&lt;localpath&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = CIFS source file-server name from which files are being migrated</td>
</tr>
<tr>
<td>&lt;mgfs&gt; = migration file system name</td>
</tr>
<tr>
<td>&lt;localpath&gt; = unique name specified to identify the destination of a particular connection</td>
</tr>
<tr>
<td>&lt;cid&gt; = connection ID 0 through 1023. If no cid is supplied, all connections with the specified MGFS name are verified.</td>
</tr>
<tr>
<td>The -verify argument determines whether all files and directories have been migrated successfully. This process might take several minutes, depending on the number of migrated files and directories.</td>
</tr>
<tr>
<td>Examples:</td>
</tr>
<tr>
<td>To verify the migration status of the pmacF30 file system from the server_2 source file server, type:</td>
</tr>
<tr>
<td>$ server_cdms server_2 -verify pmacF30</td>
</tr>
<tr>
<td>To verify the migration status of the mgfs1 file system by using the /export1 path to the server_2 Data Mover, type:</td>
</tr>
<tr>
<td>$ server_cdms server_2 -verify mgfs1 -path /export1</td>
</tr>
</tbody>
</table>
Output

server_2 : done

or

server_2 : failed

For the results of this command, you must look in the server log by using the `server_log server_2` command. For example:

2002-04-09 13:06:34: MGFS: 4: Checking
2002-04-09 13:06:34: MGFS: 4: in-processing: fsid=22 0% done
2002-04-09 13:06:44: MGFS: 4: in-processing: fsid=22 60% done
2002-04-09 13:06:45: MGFS: 4: in-processing: fsid=22 70% done
2002-04-09 13:06:49: MGFS: 4: in-processing: fsid=22 100% done
Identifying the CID

To identify the connections and associated IDs to an MGFS, from the Control Station, type the following command:

```
$ server_cdms <movername> -info
```

where:

```
<movername> = name of the Data Mover
```

For example:

```
$ server_cdms server_2 -info
```

server_2:
CDMS enabled with 128 threads

pmacF30:

mgfs1:
path     = /c
  cid     = 0
  type    = cifs
  source  = \server1_old.adnative.com\c$
  netbios= server1.adnative.com
  admin   = cdms_migrator
path     = /d
  cid     = 1
  type    = cifs
  source  = \server1_old.adnative.com\d$
  netbios= server1.adnative.com
  admin   = cdms_migrator

“Managing a failed verification or conversion” on page 229 provides more information if the verification does not complete successfully.

Step 17: Converting the MGFS to a UxFS

After the migration has been validated successfully, with all inodes migrated to the Data Mover, you need to run the `server_cdms <movername> -Convert <mgfs>` command to change the migrated MGFS files to UxFS files, removing information maintained by the MGFS while performing the migration.

If successful, disconnect the NFS source file server, or redeploy it for some other purpose.
The `server_cdms <movername> -Convert <mgfs>` command performs verify actions on any files and directories not yet verified by the `server_cdms <movername> -verify <mgfs>` command. A successful conversion makes the file system type UxFS, preventing any further attempt to migrate data into that migration file system. Conversely, an unsuccessful conversion leaves the file system in an MGFS state.

Do not run a convert command if there is another convert command still in progress on a different file system on the same Data Mover.

**Note:** You can convert a type MGFS to a UxFS by using the Unisphere Data Migration GUI.

The command can be run while the file system is mounted, exported, or in use. It is not an offline operation. Because the file system type is changed during the final part of the command operation, it can cause a momentary delay in access. Therefore, it is recommended as a best practice to convert at a quiet time when there are few outstanding I/O requests on the file system. Avoid converting during periods of heavy I/O use, which may be caused not only by user interaction but also by automated backups or report generation.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To convert the MGFS to a UxFS, use this command syntax:</td>
</tr>
<tr>
<td><code>server_cdms &lt;movername&gt; -Convert &lt;mgfs&gt;</code></td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td><code>&lt;movername&gt;</code> = name of the Data Mover</td>
</tr>
<tr>
<td><code>&lt;mgfs&gt;</code> = MGFS filename to be converted to UxFS format</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To convert the <code>mgfs1</code> file system on the <code>server_2</code> Data Mover to a UxFS format, type:</td>
</tr>
<tr>
<td><code>$ server_cdms server_2 -Convert mgfs1</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 : done</td>
<td>This command changes the MGFS to a UxFS.</td>
</tr>
</tbody>
</table>

**Note:** You can convert an MGFS file system by using the Unisphere Data Migration GUI.

“Removing an unwanted connection” on page 232 provides more information if there are outstanding connections remaining after the “verify” command and the conversion does not complete successfully.
If a CDMS connection to the source system is the reason for a failure of the server_cdms -Convert command, it is likely that some prior server_cdms -verify command has failed or was not completed. You should first correct all issues on the verify commands and have the connection removed by a flawless verify process for the whole file system before attempting the convert command. Manually removing the connection without a successful verify may force you to delete all incompletely migrated files to succeed with the convert.

If successful, disconnect the CIFS source file server (remove the path created in the server_cdms <movername> connect <mgfs> command), or reuse it for some another purpose.

Step 18: The next step

If you want to:

- Learn how to merge multiple file systems into a single file system, go to “Many-to-one migration” on page 176.
- Complete the single file system migration process, go to “Postmigration testing” on page 212.

Many-to-one migration

This section describes migrations that involve combining shares from two or more CIFS source file servers, and placing them under a single computer name on a Data Mover.

Note: This procedure is similar to the NFS procedure, described in “Many-to-one migration” on page 111.

Benefits

The many-to-one strategy effectively reduces the number of servers to be managed at the customer’s site. As such, there is minimal client disruption because one or more of the computer names from the original servers are removed from the network.

Merge strategies

Depending on what you are attempting to accomplish within the migration, this section describes two ways to merge shares within a single Windows domain from multiple (many-to-one) CIFS source file servers:
Many-to-one migration 177

◆ “Retained server name merge” on page 180
◆ “New server name merge strategy” on page 207

The approach you use to merge multiple servers can follow a number of paths. 
Figure 18 on page 177 represents a typical environment that might be a candidate for merging share data.

Figure 18  Many-to-one migration strategy

Summary

The steps to implement many-to-one retained server name merge CIFS migration are summarized in Table 9 on page 177.

Table 9  Many-to-one retained server name merge migration summary

<table>
<thead>
<tr>
<th>Step</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>“Step 1: Creating an account” on page 181</td>
</tr>
<tr>
<td>2</td>
<td>“Step 2: Installing components” on page 181</td>
</tr>
<tr>
<td>3</td>
<td>“Step 3: Evaluating servers for duplicate shares” on page 181</td>
</tr>
<tr>
<td>4</td>
<td>“Step 4: Evaluating the source file server” on page 182</td>
</tr>
<tr>
<td>5</td>
<td>“Step 5: Identifying high-priority files (optional)” on page 182</td>
</tr>
<tr>
<td>6</td>
<td>“Step 6: Backing up the source file server” on page 182</td>
</tr>
<tr>
<td>7</td>
<td>“Step 7: Configuring the data mover” on page 183</td>
</tr>
</tbody>
</table>
Table 9 Many-to-one retained server name merge migration summary

<table>
<thead>
<tr>
<th>Step</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>“Step 8: Creating the MGFS file system” on page 183</td>
</tr>
<tr>
<td>9</td>
<td>“Step 9: Preparing for migration” on page 187</td>
</tr>
<tr>
<td>10</td>
<td>“Step 10: Setting up the CIFS environment” on page 190</td>
</tr>
<tr>
<td>11</td>
<td>“Step 11: Migrating local group information” on page 194</td>
</tr>
<tr>
<td>12</td>
<td>“Step 12: Creating CDMS connections” on page 197</td>
</tr>
<tr>
<td>13</td>
<td>“Step 13: Executing the sharedup.exe utility” on page 199</td>
</tr>
<tr>
<td>14</td>
<td>“Step 14: Ensuring all data is migrated” on page 206</td>
</tr>
<tr>
<td>15</td>
<td>“Step 15: Converting the MGFS to a UxFS” on page 206</td>
</tr>
<tr>
<td>16</td>
<td>“Step 16: The next step” on page 206</td>
</tr>
</tbody>
</table>

The steps to implement many-to-one new server name merge CIFS migration are summarized in Table 10 on page 178.

Table 10 Many-to-one new server name merge migration summary (page 1 of 2)

<table>
<thead>
<tr>
<th>Step</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>“Step 1: Creating an account” on page 207</td>
</tr>
<tr>
<td>2</td>
<td>“Step 2: Installing components” on page 207</td>
</tr>
<tr>
<td>3</td>
<td>“Step 3: Evaluating servers for duplicate shares” on page 207</td>
</tr>
<tr>
<td>4</td>
<td>“Step 4: Evaluating the source file server” on page 207</td>
</tr>
<tr>
<td>5</td>
<td>“Step 5: Identifying high-priority files (optional)” on page 208</td>
</tr>
<tr>
<td>6</td>
<td>“Step 6: Backing up the source file server” on page 208</td>
</tr>
<tr>
<td>7</td>
<td>“Step 7: Configuring the Data Mover” on page 208</td>
</tr>
<tr>
<td>8</td>
<td>“Step 8: Creating the MGFS file system” on page 208</td>
</tr>
<tr>
<td>9</td>
<td>“Step 9: Preparing for migration” on page 209</td>
</tr>
</tbody>
</table>
Within this context, the one-to-one migration Administrative Share methodology is extended to handle many-to-one requirements that merge multiple source file servers.

There are two possible strategies to merge shares from CIFS source file servers:

- A retained server name merge
- A new server name merge

Table 10  Many-to-one new server name merge migration summary (page 2 of 2)

<table>
<thead>
<tr>
<th>Step</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>“Step 10: Setting up the CIFS environment” on page 210</td>
</tr>
<tr>
<td>11</td>
<td>“Step 11: Migrating local group information” on page 211</td>
</tr>
<tr>
<td>12</td>
<td>“Step 12: Creating CDMS connections” on page 211</td>
</tr>
<tr>
<td>13</td>
<td>“Step 13: Executing the sharedup.exe utility” on page 211</td>
</tr>
<tr>
<td>14</td>
<td>“Step 14: Ensuring all data is migrated” on page 211</td>
</tr>
<tr>
<td>15</td>
<td>“Step 15: Converting the MGFS to a UxFS” on page 211</td>
</tr>
<tr>
<td>16</td>
<td>“Step 16: The next step” on page 212</td>
</tr>
</tbody>
</table>
The retained server name merge strategy configures the Data Mover so it assumes one of the existing CIFS source file server names, as shown in Figure 19 on page 180. All other source file servers are renamed.

![Retained server name merge](CNS-000293)

**Figure 19** Retained server name merge

The new server name merge strategy configures the Data Mover with a new server name, as shown in Figure 20 on page 180. All source file servers are renamed.

![New server name merge](CNS-000294)

**Figure 20** New server name merge
Retained server name merge strategy

This process is very similar to the one-to-one migration strategy with variations for the server name retained (first source file server) and all other server names being consolidated (subsequent servers).

**Note:** Steps 1 through 9 are performed once for the collection of CIFS source file servers being merged while steps 10 through 17 apply to the first CIFS source file server and all subsequent servers. Also note that only differences are documented in this section between the one-to-one and many-to-one retained server name merge migration strategies.

For a successful migration, ensure that you have read Chapter 4, “Planning and Design,” before you start the Implementation phase.

To migrate a group of CIFS source file servers by using the retained server name merge strategy, perform the following:

**Step 1: Creating an account**

Create an account with domain administrator rights from a domain controller (DC) or primary domain controller (PDC). This task is exactly the same as “Step 1: Creating an account” on page 127 of the one-to-one migration strategy.

This step is not necessary if the cdms_migrator user was already added to the Windows domain.

**Step 2: Installing components**

This task installs the Perl script (ActivePerl 5.6 or later), Win32 API extensions for the Perl script, EMC migration utilities, and Microsoft Word and Excel 2000 on the Windows client. This task is exactly the same as “Step 2: Installing components” on page 129 of the one-to-one migration strategy.

**Step 3: Evaluating servers for duplicate shares**

To evaluate all CIFS source file servers for duplicate shares prior to migration, perform the following:

1. Run the `net view` command from a command window on the Windows client against each CIFS source file server.
2. Redirect the output to a text file.
In the following example, three source file servers are being merged into a single server:

C:\> net view \server1 > shares.txt
C:\> net view \server2 >> shares.txt
C:\> net view \server3 >> shares.txt

3. Review the output from the net view commands, identifying duplicate share names by comparing the information from all servers.

You might want to import the files into an Excel spreadsheet to assist in this process. Clearly identify the share name, the source file server name, and the drive on which the share resides. You need to determine how the share names are modified. As a best practice, keep all share names from the first server the same, and modify the share names for all other servers. Document the original, and modified share name, the CIFS source file server name, and the drive on which the share resides. You will use this information in a subsequent step.

**Note:** For a successful migration, check the details of the source file server with the `backupWrapper.exe` utility, and `dircount.pl`, `dirprivW.pl`, and `diskUsage.pl` scripts.

---

**Step 4: Evaluating the source file server**

Evaluate each source file server directory structure for files and directories that might be excluded from the migration. Use the `sharedup.exe` utility and the `connBuilder.pl` script to assist with the evaluation of the directory structure.

This task is exactly the same as “Step 3: Evaluating the CIFS source file server” on page 130 of the one-to-one migration strategy.

**Step 5: Identifying high-priority files (optional)**

Identify any high-priority files on all source file servers and create optional corresponding include files for each disk drive from the source file servers. This task is exactly the same as “Step 4: Identifying high-priority files (optional)” on page 135 of the one-to-one migration strategy.

**Step 6: Backing up the source file server**

Use a reliable method to perform a full backup of the CIFS source file server prior to any migration to a Data Mover.

This task is exactly the same as “Step 5: Backing up the source file server” on page 136 of the one-to-one migration strategy.
Configure the Data Mover for the migration environment by using a secure, encrypted, remote login application interface on the Windows client. If using a different Data Mover, this step is the same as “Step 6: Configuring the Data Mover” on page 136 of the one-to-one migration strategy. If using the same Data Mover, add a new interface, and modify Usermapper for any additional domains.

*Configuring and Managing CIFS on VNX provides complete details.*

Create one MGFS file system for each source file server. Calculate migration file system size based on all drives from each CIFS source file server. Include growth factors per customer or generally accepted guidelines. The size calculation must account for the block size of the VNX. When moving from a Windows environment, there can be as much as double the storage requirements.

Configure the Data Mover by using a secure, encrypted, remote login application interface on the Windows client.

*Managing Volumes and File Systems with VNX Automatic Volume Management and Managing Volumes and File Systems for VNX Manually provide more complete information about the creation of volumes, file systems, and mount points.*

Perform the following:

1. Create striped volumes, if required, as described in the following table.

```markdown
<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To create striped volumes, use this command syntax: $ nas_volume [-name &lt;name&gt;] -create [-Stripe [stripe_size]</td>
</tr>
</tbody>
</table>
```
2. Create the metavolume.

<table>
<thead>
<tr>
<th>Output</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>id = 246</td>
<td>• Stripe depth must be typed in multiples of 8,192 bytes.</td>
</tr>
<tr>
<td>name = stv1</td>
<td>• Default stripe size is 32,768 bytes.</td>
</tr>
<tr>
<td>acl = 0</td>
<td>• CDMS is not supported on Virtual Data Movers.</td>
</tr>
<tr>
<td>in_use = False</td>
<td></td>
</tr>
<tr>
<td>type = stripe</td>
<td></td>
</tr>
<tr>
<td>stripe_size = 32768</td>
<td></td>
</tr>
<tr>
<td>volume_set = d3,d4,d5,d6</td>
<td></td>
</tr>
<tr>
<td>disks = d3,d4,d5,d6</td>
<td></td>
</tr>
</tbody>
</table>

**Action**

To create the metavolume, use this command syntax:

```bash
$ nas_volume [-name <name>] -create [-Stripe [<stripe_size>] | -Meta] {{<volume_name>,...} | size=<GB>}
```

- `<name>` = assigns a volume name (case-sensitive)
- `-Stripe` `<stripe_size>` | `-Meta` = sets the type for the volume to be either a stripe or metavolume (default)
- `<volume_name>` = volume table entry from the set of volumes. Volumes can be specified by name or size.
- `<GB>` = a volume match is found equal to or greater than the specified size

Example:

To create a metavolume named `mtv1` (on which you later create an MGFS) on the `stv1` volume, type:

```bash
$ nas_volume -name mtv1 -create -Meta stv1
```

<table>
<thead>
<tr>
<th>Output</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>id = 247</td>
<td>The default volume is <code>-Meta</code>.</td>
</tr>
<tr>
<td>name = mtv1</td>
<td></td>
</tr>
<tr>
<td>acl = 0</td>
<td></td>
</tr>
<tr>
<td>in_use = False</td>
<td></td>
</tr>
<tr>
<td>type = meta</td>
<td></td>
</tr>
<tr>
<td>volume_set = stv1</td>
<td></td>
</tr>
<tr>
<td>disks = d3,d4,d5,d6</td>
<td></td>
</tr>
</tbody>
</table>
3. Create an MGFS per CIFS source file server.

**Action**

To create an MGFS for each CIFS source file server, use this command syntax:

```
$ nas_fs [-name <name>] [-type <type>] -create
{<volume_name>|size=<integer>[G|M] pool=<pool>} [-option <options>]
```

Default is `-type uxfs`.

where:

- `<name>` = name assigned to the migration file system being created
- `<type>` = type assigned to the migration file system being created. Valid types are `uxfs`, `rawfs`, and `mgfs`. This must be `mgfs` for a CDMS migration. The default value is `uxfs`.
- `-create` `<volume_name>|size=G pool=<pool>` = creates a file system on the specified metavolume, or available metavolume with the specified size
- `<options>` = specifies a comma-separated list of characteristics to create a file system

Example:

To create MGFS file system types named `mgfs1`, `mgfs2`, and `mgfs3` on the `mtv1`, `mtv2`, and `mtv3` metavolumes, type:

```
$ nas_fs -n mgfs1 -type mgfs -create mtv1
$ nas_fs -n mgfs2 -type mgfs -create mtv2
$ nas_fs -n mgfs3 -type mgfs -create mtv3
```

**Output**

```
id = 33
name = mgfs1
acl = 0
in_use = False
type = mgfs
volume = mtv1
profile =
 rw_servers =
 ro_servers =
sym_devs = 002804000192-000D
disks = 10
```

**Note:** You can create a new MGFS by using the Unisphere Data Migration GUI. The default file system type is `uxfs`.

4. Create mount points by using the server names from each of the CIFS source file servers.

The following example shows three source file servers:

```
$ server_mount server_2 mgfs1 /server1
$ server_mount server_2 mgfs2 /server2
$ server_mount server_2 mgfs3 /server3
```

5. Mount the MGFS file systems for all CIFS source file servers being merged on the Data Mover.

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>id = 33</td>
</tr>
<tr>
<td>name = mgfs2</td>
</tr>
<tr>
<td>acl = 0</td>
</tr>
<tr>
<td>in_use = False</td>
</tr>
<tr>
<td>type = mgfs</td>
</tr>
<tr>
<td>volume = mtv2</td>
</tr>
<tr>
<td>profile =</td>
</tr>
<tr>
<td>rw_servers =</td>
</tr>
<tr>
<td>ro_servers =</td>
</tr>
<tr>
<td>sym_devs = 002804000192-000D</td>
</tr>
<tr>
<td>disks = 10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>id = 33</td>
</tr>
<tr>
<td>name = mgfs3</td>
</tr>
<tr>
<td>acl = 0</td>
</tr>
<tr>
<td>in_use = False</td>
</tr>
<tr>
<td>type = mgfs</td>
</tr>
<tr>
<td>volume = mtv3</td>
</tr>
<tr>
<td>profile =</td>
</tr>
<tr>
<td>rw_servers =</td>
</tr>
<tr>
<td>ro_servers =</td>
</tr>
<tr>
<td>sym_devs = 002804000192-000D</td>
</tr>
<tr>
<td>disks = 10</td>
</tr>
</tbody>
</table>
Prepare the source file server and the Windows client for migration. From this point to the end of the migration process, the steps you perform depend on whether you are migrating the first source file server or subsequent servers.

Starting with this step, each CIFS source file server is processed one at a time, in order to limit the amount of offline time for each server. Prior to the actual migration, you should determine the order in which you want to migrate the source file servers.

**Note:** After migration completes, if the CIFS source file server contained any print server functionality, it is not available on the Data Mover.

**For first source file server**

Perform the following:

1. Set required rights for the `lgdup.exe` utility migration account on the:
   - First source file server
   - Windows client

   To perform this task, use either the Local Security Policy in Windows 2000 and Windows Server 2003, or the User Manager for Domains in Windows NT 4.0 administrative tool, as shown in Figure 21 on page 187 and Figure 22 on page 188.

   ![Local Security Settings](image)

   **Figure 21** Local security settings: Windows 2000 or Windows Server 2003 system
Two rights are required for the `lgdup.exe` utility:

- Generate security audits
- Manage auditing and security log

![User Manager for domains: Windows NT 4.0 system](image)

**Figure 22** User Manager for domains: Windows NT 4.0 system

2. Place the `cdms_migrator` account in the backup operator group on the CIFS source file server and target server.

3. On the Windows client, log out and log in again to ensure that these rights and memberships are invoked correctly.

4. Restrict network access on the CIFS source file server by using either the Local Security Policy in Windows 2000 and Windows Server 2003, or User Manager for Domains in Windows NT 4.0 administrative tool on the local source file server:

   - If using the Local Security Policy administrative tool for Windows 2000 and Windows Server 2003 systems, assign the “Deny Access to this computer” from the network right to the Domain Users group.

   - If using the User Manager for Domains administrative tool for Windows NT 4.0 systems:
     
     - Add the `cdms_migrator` account to the “Access this computer from network” right.
     - Remove all other users and groups from this right.
5. Rename the CIFS source file server, retaining the original name for use with the Data Mover.

6. Restart the CIFS source file server.

**For subsequent servers**

Perform the following:

1. Set the required rights for the `lgdup.exe` utility migration account on the CIFS source file server by using either the Local Security Policy in Windows 2000 and Windows Server 2003, or User Manager for Domains in Windows NT 4.0 administrative tool, as shown in Figure 23 on page 194 and Figure 24 on page 195.

Two rights are required for the `lgdup.exe` utility:

- Generate security audits
- Manage auditing and security logs

2. Place the `cdms_migrator` account in the backup operator group on the CIFS source file server and target server.

3. On the Windows client, log out and log in again to ensure that these rights and memberships are invoked correctly.

4. Restrict network access on the CIFS source file server by using either the Local Security Policy in Windows 2000 and Windows Server 2003, or User Manager for Domains in Windows NT 4.0 administrative tool on the CIFS source file server:

   - If using the Local Security Policy administrative tool in Windows 2000 or Windows Server 2003 systems, assign the "Deny Access to this computer from the network" right to the Domain Users Group.
   
   - If using the User Manager for Domains administrative tool in Windows NT 4.0 systems:
     - Add the `cdms_migrator` account to the "Access this computer from network" right.
     - Remove all other users and groups from this right.

5. Restart the CIFS source file server.
Configure the Data Mover for CIFS migration by using a secure, encrypted, remote login application interface on the Windows client. Use the computer name of the first CIFS source file server. When adding a compname or a NetBIOS name to the domain, you should use the NetBIOS/compname aliasing so all other Windows clients do not need to change their mapping names. The old NetBIOS/compname still works, however.

Renaming a NetBIOS name is explained in Managing a Multiprotocol Environment on VNX. The EMC VNX Command Line Interface Reference for File provides information about the server_cifs command.

Configuring and Managing CIFS on VNX provides information about the following steps.

**Note:** This task is done only once for the target Data Mover.

**For first source file server**

Perform the following:

1. Verify connections between the Data Mover and CIFS source file server by performing the following:
   a. Ping the CIFS source file server from the Data Mover.
   b. Ping the Data Mover from the CIFS source file server.
2. Set up the Data Mover with the original source file server name.
   The server name created on the Data Mover is also known as the target server.
For a Windows 2000 or Windows Server 2003 environment

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To set up the Data Mover for a Windows 2000 or Windows Server 2003 environment, use this command syntax:</td>
</tr>
<tr>
<td>$ server_cifs &lt;movername&gt; -add compname=&lt;comp_name&gt;,domain=&lt;full_domain_name&gt; [,netbios=&lt;netbios_name&gt;][,[interface=&lt;if_name&gt; [,wins=&lt;ip&gt;[:&lt;ip&gt;]]]]..]</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = Data Mover name being configured</td>
</tr>
<tr>
<td>&lt;comp_name&gt; = Windows 2000 or Windows Server 2003 compatible CIFS server</td>
</tr>
<tr>
<td>&lt;full_domain_name&gt; = NetBIOS name assigned to the specified domain name</td>
</tr>
<tr>
<td>&lt;netbios_name&gt; = replaces the default NetBIOS name assigned automatically and derived from the comp_name</td>
</tr>
<tr>
<td>,interface=&lt;if_name&gt; [,wins=&lt;ip&gt;[:&lt;ip&gt;]] = specifies an interface for the CIFS server, and associates up to two WINS IP addresses with each interface</td>
</tr>
<tr>
<td>&lt;if_suffix&gt; = Microsoft DNS suffix for the interface. It is, by default, derived from the domain name</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To configure the server_2 Data Mover with a server1 name, type:</td>
</tr>
<tr>
<td>$ server_cifs server_2 -add compname=server1,domain=adnative.com</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 : done</td>
</tr>
</tbody>
</table>
For a Windows NT 4.0 environment

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To set up the Data Mover for a Windows NT 4.0 environment, use this command syntax:</td>
</tr>
<tr>
<td>Using the Server Manager for Domains, delete the original server name from the domain and synchronize PDCs and BDCs.</td>
</tr>
<tr>
<td>Using the Server Manager for Domains, add the original server name back into the domain and synchronize PDCs and BDCs.</td>
</tr>
<tr>
<td>Add the NetBIOS name and the WINS server to the migration Data Mover.</td>
</tr>
<tr>
<td>$ server_cifs &lt;movername&gt; -add netbios=&lt;netbios_name&gt;,domain=&lt;domain_name&gt; [,alias=&lt;alias_name&gt;...],hidden={y</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = Data Mover name being configured</td>
</tr>
<tr>
<td>&lt;comp_name&gt; = name of the Windows-compatible CIFS server. It can be up to 63 UTF8 characters.</td>
</tr>
<tr>
<td>&lt;netbios_name&gt; = replaces the default NetBIOS name assigned automatically and derived from &lt;comp_name&gt;</td>
</tr>
<tr>
<td>&lt;domain_name&gt; = NetBIOS name is assigned to the specified domain name</td>
</tr>
<tr>
<td>[,interface=&lt;if_name&gt; [,wins=&lt;ip&gt;[:&lt;ip&gt;]]]... = specifies an interface for the CIFS source file server, and associates up to two WINS IP addresses with each interface</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To configure the server_2 Data Mover with an ntserver1 NetBIOS name, type:</td>
</tr>
<tr>
<td>$ server_cifs server_2 -add netbios=ntserver1,domain=ntnative.com,wins=10.5.44.14</td>
</tr>
<tr>
<td>Output</td>
</tr>
<tr>
<td>server_2 : done</td>
</tr>
</tbody>
</table>
3. For Windows environments, ensure that all servers have joined the domain.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To join the domain in a Windows environment, use this command syntax:</td>
</tr>
<tr>
<td>$ server_cifs &lt;movername&gt; -Join compname=&lt;comp_name&gt;,domain=&lt;full_domain_name&gt;,admin=&lt;admin_name&gt;</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = specified Data Mover name</td>
</tr>
<tr>
<td>&lt;comp_name&gt; = name of the Windows-compatible CIFS server. It can be up to 63 UTF8 characters.</td>
</tr>
<tr>
<td>&lt;full_domain_name&gt; = NetBIOS name is assigned to the specified domain name</td>
</tr>
<tr>
<td>&lt;admin_name&gt; = login name of user with administrative rights in the domain</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To join a domain for the server_2 Data Mover in a Windows environment with a server1 name in the adnative.com domain, type:</td>
</tr>
<tr>
<td>$ server_cifs server_2 -Join compname=server1,domain=adnative.com ,admin=administrator</td>
</tr>
<tr>
<td>If in a Windows NT 4.0 environment, add the NetBIOS name to the Domain and identify the WINS server.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_2 : done</td>
</tr>
</tbody>
</table>

4. If the CDMS server is not started on the Data Mover, then start the CDMS migration service.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To start the CDMS migration service, use this command syntax:</td>
</tr>
<tr>
<td>$ server_setup &lt;movername&gt; -type {nas</td>
</tr>
<tr>
<td>where:</td>
</tr>
<tr>
<td>&lt;movername&gt; = name of the Data Mover</td>
</tr>
<tr>
<td>cdms = protocol configuration to be managed</td>
</tr>
<tr>
<td>start = starts the protocol configuration</td>
</tr>
<tr>
<td>&lt;n&gt; = number of threads for users. Default number of CDMS threads is 32.</td>
</tr>
<tr>
<td>Example:</td>
</tr>
<tr>
<td>To start CIFS for the server_2 Data Mover with 25 threads, type:</td>
</tr>
<tr>
<td>$ server_setup server_2 -Protocol cdms -option start=25</td>
</tr>
</tbody>
</table>
Step 11: Migrating local group information

For subsequent servers

This step is not performed for subsequent servers. No actions are required.

For first source file server

Perform the following:

1. Set required rights for the `lgdup.exe` utility migration account on the target server.

   To perform this task, use either the Local Security Policy in Windows 2000 and Windows Server 2003, or User Manager for Domains in the Windows NT 4.0 administrative tool, as shown in Figure 23 on page 194 and Figure 24 on page 195.

   ![Local Security Settings](image)

   **Figure 23** Local Security Settings: Windows 2000 or Windows Server 2003 system

   Two rights are required for the `lgdup.exe` utility:

   - Generate security audits
   - Manage auditing and security log

```
server_2 : done
```
2. Place the cdms_migrator account in the backup operator group on the target server.

3. On the Windows client, log out and log in again to ensure that these rights and memberships are invoked correctly.

4. From a command window on the Windows client, run the `lgdup.exe` utility to migrate the local group information.

**Note:** Use the automatic prefixing option on the `lgdup.exe` utility to handle issues with local group names.
For subsequent servers

Perform only “Step 3: Evaluating servers for duplicate shares” on page 181 through “Step 11: Migrating local group information” on page 194 to migrate local group information.
Create CDMS connections for each source file server disk drive. If there are two disk drives, you must run the `server_cdms <movername> -connect <mgfs>` command once for each drive.

**Note:** The WINS server is not used in a Windows 2000 or Windows Server 2003 environment if the resolutions are handled by the DNS service.

Ensure that this command identifies the corresponding file system that was created in “Step 9: Preparing for migration” on page 187 for each of the CIFS source file servers. This command returns the correct source file server name.

**Note:** Because this is a retained server name merge, the target server name remains the same for all connections in this migration.

Configure the Data Mover by using a secure, encrypted, remote login application interface on the Windows client.
Create connections for each CIFS source file server disk drive, as described in the following table.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To create CDMS connections for each CIFS source file server disk drive, use this command syntax:</td>
</tr>
</tbody>
</table>
| $ server_cdms <movername> -connect <mgfs> -type cifs -path <localPath> -netbios <netbios> -source \\
| <srcServer>[.<domain>]<srcShare>[<srcPath>] -admin [ <domain>\] <admin_name> [-wins <wins>] |
| where: |
| <movername> = specified Data Mover name |
| <mgfs> = migration file system name |
| <localpath> = subdirectory name (created if it does not exist; if it exists, it fails) within the mount point of the file system. You cannot connect to the file system mount point, only to the subdirectory. |
| <netbios> = NetBIOS name of the Data Mover-CIFS server name (since it can have more than one) |
| <srcServer>[.<domain>]<srcShare>[<srcPath>] = <srcServer> is the CIFS source file server name, <srcShare> is the CIFS source file share name, and <srcPath> allows migration that is not at the root of the share. If specified, the root of the migration is <srcShare>\dir... instead of just <srcShare>. |
| <domain>\] <admin_name> = domain name, and the name of the administrator to connect as [a password is asked interactively when the command is executed to hide (or mask) the password] |
| ___Note: The -source and -admin syntax strings must be enclosed in single quotation marks. For example: ‘\\myserver\myshare\mypath’. ___ |
| <wins> = IP address of the WINS server, only required for Windows NT 4.0 Example: |
| If you have a CIFS source file server configured with two drives, C: and D:, the commands would be similar to the following: |
| For drive C: |
| $ server_cdms server_2 -connect mgfs1 -type cifs -path /c -netbios server1.adnative.com -source ‘\\server1_old.adnative.com\c$’ -admin ‘adnative.com\cdms_migrator’ |
| For drive D: |
| $ server_cdms server_2 -connect mgfs1 -type cifs -path /d -netbios server1.adnative.com -source ‘\\server1_old.adnative.com\d$’ -admin ‘adnative.com\cdms_migrator’ |
If the connection command fails, an appropriate error message appears.

**For subsequent servers**

This step is identical to creating CDMS connections for each source file server, but remember to modify the `server_cdms` `<movername> -connect <mgfs>` command for the corresponding source file server and disk drives.

**Example:** `server_2` with three disk drives

For drive C:

```
server_cdms server_2 -connect mgfs2 -type cifs -path /c
-netbios server1.adnative.com -source '\server2.adnative.com\c$
-admin 'adnative.com\cdms_migrator'
```

For drive D:

```
server_cdms server_2 -connect mgfs2 -type cifs -path /d
-netbios server1.adnative.com -source '\server2.adnative.com\d$
-admin 'adnative.com\cdms_migrator'
```

For drive E:

```
server_cdms server_2 -connect mgfs2 -type cifs -path /e
-netbios server1.adnative.com -source '\server2.adnative.com\e$
-admin 'adnative.com\cdms_migrator'
```

**Step 13: Executing the sharedup.exe utility**

Execute the `sharedup.exe` utility against each disk drive from the CIFS source file server. The local path should correspond to the `<localPath>` used with the `server_cdms` command in “Step 12: Creating CDMS connections” on page 197.

It is assumed the first CIFS source file server is migrated as is with no changes for duplicate shares.

This example assumes you are moving shares from an entire disk drive to a local path on the VNX to which the drive content is being migrated.

This example is not intended for the case where you are moving shares from the same source drives to different Data Movers.
To perform this task, use a command window on the Windows client.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To execute the <code>sharedup.exe</code> utility against each drive from the CIFS source file server, use this command syntax: C:&gt; sharedup &lt;source&gt; &lt;target&gt; &lt;sourcedrive&gt; /SD /P:&lt;mountpointname&gt;&lt;localPath&gt; /LOG:&lt;logfilename&gt; where: &lt;source&gt; = CIFS source file server name &lt;target&gt; = target Data Mover computer name &lt;sourcedrive&gt; = drive letter of the source file server containing the shares to be migrated /SD = causes the <code>sharedup.exe</code> utility to transfer all ACLs /P: = identifies the path to which the shares are migrated &lt;mountpointname&gt; = mount point name of the MGFS (with no slash before the mount point) &lt;localPath&gt; = the local pathname identified with the <code>server_cdms</code> command /LOG: = log file optional parameter &lt;logfilename&gt; = log filename “<code>sharedup.exe utility</code>” on page 284 provides more information about the <code>sharedup.exe</code> utility. Example: If you have a source file server configured with two drives, C: and D:, the commands would be similar to the following: For drive C: C:&gt; sharedup \server1_old \server1 c: /SD /P:server1\c /LOG:server1-c-shares-log.txt For drive D: C:&gt; sharedup \server1_old \server1 d: /SD /P:server1\d /LOG:server1-d-shares-log.txt</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHAREDUP 01.06 Copyright(c) 2004, All Right Reserved, by EMC Corporation, Hopkinton, MA. Source server:server1_old 4.0 Target server:server1 4.1 EMC-SNAS:T5.1.10.0 ******************************************************* SHAREDUP source:server1_old target:server1 Summary results: Elapsed time: hours:00,mins:00,secs:00 Number of share(s) successfully duplicated on drive c: 6</td>
</tr>
</tbody>
</table>

200  VNX File System Migration Version 2.0 for NFS and CIFS
Note: At this point, all shares have been exported externally, and are available for Windows client access.

For subsequent servers

The syntax of the sharedup.exe utility and corresponding process become much more involved because the sharedup.exe utility now produces an output file you edit, and then use as the input file to a second execution of the sharedup.exe utility. The local path should correspond to the <localPath> used with the server_cdms <movername> -connect <mgfs> command in “Step 12: Creating CDMS connections” on page 211.

The general syntax for the sharedup.exe utility in this context is as follows:

```
C:\> sharedup \<source> \<localcomputername> <sourcedrive> /SD /P:<mountpointname>\<localPath> /FO:<localfilename>.txt
```

where:

- `<source>` = CIFS source file server name
- `<localcomputername>` = Windows client name
- `<sourcedrive>` = drive letter of the CIFS source file server containing the shares to be migrated
- `/SD` = causes the sharedup.exe utility to transfer all ACLs
- `/P:` = identifies the path to which the shares are migrated
<mountpointname> = mount point name of the MGFS (with no slash before mount point)

<localPath> = local pathname identified with the server_cdms command

/FO: = output file option

<localfilename> = name of the local file on the Windows client ending in .txt. This file should be in the same directory as the sharedup.exe utility.

Perform the following:

1. Run the sharedup.exe utility to produce an output file of the share names from each CIFS source file server disk drive.

   If you have a CIFS source file server configured with two drives, C and D, the commands would be similar to the following:

   For drive C:

   C:\> sharedup \server1 \mypc c: /SD /P:server2\c /FO:server2-c-shares.txt

   For drive D:

   C:\> sharedup \server1 \mypc d: /SD /P:server2\d /FO:server2-d-shares.txt

2. Using Notepad on the Windows client, edit the sharedup.exe utility output files to rename duplicate shares identified in “Step 3: Evaluating servers for duplicate shares” on page 181, and then modify the target server.

   To help show what needs to be edited, examples of before and after sharedup.exe utility output are provided in Figure 25 on page 203 and Figure 26 on page 205.
The sharedup.exe utility output before editing

The sharedup.exe utility output file is shown before editing in Figure 25 on page 203.

```plaintext
#SHAREDUP FILE
@Revision:1
@Source:SERVER2
@Target:MYPC
#
# Please leave the above lines intact, SHAREDUP checks them.
#
# File format:
# A comment line must begin with the character '#'.
#
# A share info is stored in one line and must begin with the character ':'.
#
# The ';' character is used as a field separator for the formatted line below:
#
#:<source_share_name>;<target_share_name>;<target_pathname>;<target_comment>
#
# Note: the target_pathname field includes the drive letter.
#
# Please leave intact the source_share_name field.
#
# Maximum lengths of fields:
# - <source_share_name> : 80
# - <target_share_name> : 80
# - <target_pathname>   : 1026
# - <target_comment>   : 255
#
# This file is in UNICODE format.
#
@Drive:C
:sh_scripts;sh_scripts;C:\mgfs25\c\sh_scripts;
a data;a data;C:\mgfs25\c\a;
:winvnc;winvnc;C:\mgfs25\c\tools\winvnc;
:pre release;pre release;C:\mgfs25\c\copytest\pre release;
:ysi$;ysi$;C:\mgfs25\c\ysi;
```

**Figure 25** The sharedup.exe utility output file (before editing)
The sharedup.exe utility output file after editing

The sharedup.exe utility output file is shown after editing in Figure 26 on page 205. The changes are shown in bold text.

```
#SHAREDUP FILE
@Revision:1
@Source:SERVER2
@Target:SERVER1
#
# Please leave the above lines intact, SHAREDUP checks them.
#
# File format:
# A comment line must begin with the character '#'.
# A share info is stored in one line and must begin with the character ':'.
# The ';' character is used as a field separator for the formatted line below:
#
#:<source_share_name>;<target_share_name>;<target_pathname>;<target_comment>
#
# Note: the target_pathname field includes the drive letter.
#
# Please leave intact the source_share_name field.
#
# Maximum lengths of fields:
# - <source_share_name> : 80
# - <target_share_name> : 80
# - <target_pathname>   : 1026
# - <target_comment>   : 255
#
# This file is in UNICODE format.
#
#
@Drive:C
:sh_scripts;sh_scripts;C:\mgfs25\c\sh_scripts;
a data;a data;C:\mgfs25\c\a;
:winvnc;winvnc;C:\mgfs25\c\tools\winvnc;
:pre release;pre release;C:\mgfs25\c\copytest\pre release;
:ysi$;newshare;C:\mgfs25\c\ysi;
```
3. Run the `sharedup.exe` utility again by using the edited output file as the input file to the `sharedup.exe` utility.

The syntax of the `sharedup.exe` utility now reflects the input filename and the target Data Mover computer name:

```
C:\> sharedup \<source> \<target> <sourcedrive> /SD
/P:<mountpointname>\<localPath> /FI:<localfilename>.txt /LOG:<logfilename>
```

where:

- `<source>` = CIFS source file server name
- `<target>` = target Data Mover name
- `<sourcedrive>` = drive letter of the CIFS source file server containing the shares to be migrated
- `/SD` = causes the `sharedup.exe` utility to transfer all ACLs
- `/P:` = identifies the path to which the shares are migrated
- `<mountpointname>` = name of the MGFS mount point (with no slash before mount point)
- `<localPath>` = local pathname identified with the `server_cdms` command
- `<localfilename>.txt` = name of the local file on the Windows client ending in .txt. This file should be in the same directory as the `sharedup.exe` utility
- `/FI:` = input file option
- `/LOG:` = log file optional parameter
- `<logfilename>` = name of the log file

If you have a CIFS source file server configured with two drives, C: and D:, the commands would be similar to the following:

For drive C:

```
C:\> sharedup \server2 \server1 c: /SD /P:server2\c /FI:server2-c-shares.txt /LOG:server2-c-shares-log.txt
```

For drive D:

```
C:\> sharedup \server2 \server1 d: /SD /P:server2\d /FI:server1-d-shares.txt /LOG:server2-d-shares-log.txt
```
Note: At this point, all shares have been exported externally, and are available for Windows client access. Now all clients need to access their shares by using the first source file server name. This action requires that the shares are remapped on these clients. There might be other changes that require updating as well, including desktop icons and URLs.

Step 14: Ensuring all data is migrated

Ensure that all remaining data is migrated to the VNX.

To perform this task, use a command window on the Windows client.

For first file server

This task is exactly the same as “Step 14: Ensuring that all data is migrated” on page 165 in the one-to-one migration.

For subsequent servers

This action also applies to all subsequent servers.

Step 15: Converting the MGFS to a UXFS

Verify and convert the MGFS to a UXFS.

Configure the Data Mover by using a secure, encrypted, remote login application interface on the Windows client.

For first file server

This task is exactly the same as “Step 16: Verifying migration completion” on page 169 and “Step 17: Converting the MGFS to a UXFS” on page 174 in the one-to-one migration.

For subsequent servers

This action also applies to all subsequent servers.

At this point, the CDMS software automatically contacts the Control Station to change the file system type from MGFS to UXFS.

Now the CIFS source file server can be disconnected, or redeployed for another purpose.

Step 16: The next step

If all CIFS source file servers have been migrated successfully, the retained server name merge is complete.

If you want to:
Migrate another source file server to the MGFS file system by using the retained server name merge strategy, go back to “Retained server name merge” on page 180.

Complete the file system migration process, go to “Postmigration testing” on page 212.

If all CIFS source file servers have not been migrated successfully, go to “Step 9: Preparing for migration” on page 187, and continue for each additional source file server.

---

**New server name merge strategy**

The new server name merge process is very similar to the retained server name merge strategy. The difference is that all steps follow the path identified for subsequent servers in the “Retained server name merge” on page 180 with only minor variances.

The following steps describe the tasks required to run a new server name merge.

<table>
<thead>
<tr>
<th>Step 1: Creating an account</th>
</tr>
</thead>
<tbody>
<tr>
<td>You do not need to perform this step if the cdms_migrator user account was already added to the domain.</td>
</tr>
<tr>
<td>Create an account with domain administrator rights from a domain controller (DC) or a primary domain controller (PDC). This task is exactly the same as “Step 1: Creating an account” on page 127 of the one-to-one migration strategy.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 2: Installing components</th>
</tr>
</thead>
<tbody>
<tr>
<td>This task installs the Perl script (ActivePerl 5.6 or later), Win32 API Extensions for the Perl script, EMC Migration Utilities, and Microsoft Word and Excel 2000 on the Windows client. This task is exactly the same as “Step 2: Installing components” on page 181 of the retained server name merge migration strategy.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 3: Evaluating servers for duplicate shares</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluate all CIFS source file servers for duplicate shares prior to migration. This task is exactly the same as “Step 3: Evaluating servers for duplicate shares” on page 181 of the retained server name merge migration strategy.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 4: Evaluating the source file server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluate each source file server directory structure for files and directories that might be excluded from the migration. Use the sharedup.exe utility and connBuilder.pl script to assist with the evaluation of the directory structure.</td>
</tr>
<tr>
<td>This task is exactly the same as “Step 4: Evaluating the source file server” on page 182 of the retained server name merge migration strategy.</td>
</tr>
</tbody>
</table>
Step 5:  
Identifying high-priority files (optional)

Identify any high-priority files on all source file servers and create optional corresponding include files for each disk drive from the source file servers. This task is exactly the same as “Step 5: Identifying high-priority files (optional)” on page 182 of the retained server name merge migration strategy.

Step 6:  
Backing up the source file server

Use a reliable method to perform a full backup of the CIFS source file server prior to any data migration.

This step is exactly the same as “Step 6: Backing up the source file server” on page 182 of the retained server name merge migration strategy.

Step 7:  
Configuring the Data Mover

Configure the Data Mover for the migration environment by using a secure, encrypted, remote login application interface on the Windows client. This task is exactly the same as “Step 7: Configuring the data mover” on page 183 of the retained server name merge migration strategy.

Configuring and Managing CIFS on VNX provides details.

Step 8:  
Creating the MGFS file system

Create one MGFS file system for each source file server. This task is exactly the same as “Step 8: Creating the MGFS file system” on page 183 of the retained server name merge migration strategy.

Configure the Data Mover by using a secure, encrypted, remote login application interface on the Windows client.

Managing Volumes and File Systems with VNX Automatic Volume Management and Managing Volumes and File Systems for VNX Manually provide more information about the creation of volumes, file systems, and mount points.
Step 9: Preparing for migration

Prepare the source file server and the Windows client for migration. Perform the following:

1. Set required rights for the `lgdup.exe` utility migration account on the Windows client by using either the Local Security Policy in Windows 2000 and Windows Server 2003, or User Manager in the Windows NT 4.0 administrative tool, as shown in Figure 27 on page 209 and Figure 28 on page 210.

![Local Security Settings](image)

**Figure 27** Local Security Settings: Windows 2000 or Windows Server 2003 system

Two rights are required for the `lgdup.exe` utility:

- Generate security audits
- Manage auditing and security log
2. Place the cdms_migrator account in the backup operator group on the CIFS source file server and target server.

3. On the Windows client, log out and log in again to ensure that these rights and memberships are invoked correctly.

From this point forward, for all CIFS source file servers, use the steps identified for subsequent servers in “Step 9: Preparing for migration” on page 187 of the retained server name merge migration strategy.

Configure the Data Mover for CIFS migration by using a secure, encrypted, remote login application interface on the Windows client. This task is done only once for the target Data Mover, and gives the Data Mover a unique name within the domain.

This step is exactly the same as “Step 10: Setting up the CIFS environment” on page 190 of the retained server name merge migration strategy.

For subsequent servers

This step is not performed for subsequent servers. No actions are required.
Step 11: Migrating local group information

This task is exactly the same as “Step 11: Migrating local group information” on page 194 of the retained server name merge migration strategy.

To perform this task, use a command window on the Windows client.

Step 12: Creating CDMS connections

Create CDMS connections for each source file server disk drive. This task is exactly the same as “Step 12: Creating CDMS connections” on page 197 of the retained server name merge migration strategy.

Configure the Data Mover by using a secure, encrypted, remote login application interface on the Windows client.

Step 13: Executing the sharedup.exe utility

Execute the sharedup.exe utility against each disk drive from the source file server. This task is exactly the same as “Step 13: Executing the sharedup.exe utility” on page 199 of the retained server name merge migration strategy.

To perform this task, use a command window on the Windows client.

Step 14: Ensuring all data is migrated

Ensure that all remaining data is migrated to the VNX. This task is exactly the same as “Step 14: Ensuring all data is migrated” on page 206 of the retained server name merge migration strategy.

To perform this task, use a command window on the Windows client.

Step 15: Converting the MGFS to a UxFS

Verify and convert the MGFS to a UxFS. This task is exactly the same as “Step 16: Verifying migration completion” on page 169 and “Step 17: Converting the MGFS to a UxFS” on page 174 in the one-to-one migration strategy.

Configure the Data Mover by using a secure, encrypted, remote login application interface on the Windows client.

At this point, the CDMS software automatically contacts the Control Station to change the file system type from MGFS to UxFS.

Now disconnect the CIFS source file server (remove the path created in the server_cdms <movername> -connect <mgfs> command), or redeploy it for some another purpose.

Many-to-one migration
Step 16: The next step

If all CIFS source file servers have been migrated successfully, the new server name merge is complete.

If you want to:

- Learn how to migrate a single file system into an MGFS file system, go to “One-to-one migration” on page 123.
- Complete the multiple file system migration process, go to “Postmigration testing” on page 212.
- Migrate a Windows domain to the domain supporting the CDMS server. Managing a Multiprotocol Environment on VNX provides a description of the VNX system’s domain migration support.

Postmigration testing

Assuming the migration was successful, you need to verify that all application and file accessibility issues have been resolved successfully.

Complete the following tasks:

- Ensure that the new VNX systems and UxFS file systems are accessible from various Windows clients.
- If there are specific applications, verify that they are functioning correctly.
- Verify all operational modifications are necessary with the new VNX infrastructure.
- Disconnect, and reconfigure or remove the source file servers from the network.
CHAPTER 7
Troubleshooting CDMS

The following topics discuss NFS and CIFS problems and associated solutions, how to handle a failed verification or conversion, how to remove an unwanted connection, and a list of common error codes:

- Introduction ...................................................................................... 214
- Problems and solutions ................................................................... 214
- Using the server_cdms -info command ........................................ 218
- Migration suspension (hang) conditions (NFS)........................... 218
- Connection command failures (CIFS) ........................................... 223
- Managing a failed verification or conversion .............................. 229
- Removing an unwanted connection.............................................. 232
- Error codes ........................................................................................ 234
- CDMS server log error messages ................................................... 237
Introduction

This chapter provides guidance for troubleshooting problems you might encounter with CDMS. It discusses the following topics:

- Resolution of selected NFS and CIFS migration problems
- How to use the `server_cdms -info` command to show connection and thread status as well as threads in a particular state
- Causes and solutions to the three main migration suspension conditions
- CIFS connection command failures and solutions
- How to manage a failed verification or conversion
- Steps to remove an unwanted connection
- NFS and CIFS error codes with brief meanings

You can contact the EMC Web Support database for problem information, obtain release notes, or report a VNX technical problem to EMC at EMC Online Support, EMC’s secure extranet site.

The Problem Resolution Roadmap for VNX provides additional details about using EMC Online Support and resolving problems.

“Issue tracker” on page 39 explains how to list bugs for selected EMC products such as VNX.

Problems and solutions

This section contains several NFS and CIFS migration problems and possible solutions.
Table 11 on page 215 summarizes problems you might encounter while performing an NFS migration with CDMS. Use this list as a starting point to solve problems you might be experiencing.

Table 11  NFS problems and solutions  (page 1 of 2)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applications are receiving an Abort, Fail, Retry? error message.</td>
<td>This problem can occur while the Data Mover is being added to the network. If the Data Mover is assuming the IP address of the NFS source file server, users can select Retry until the data can be accessed. The error might occur multiple times until the Data Mover is in place.</td>
</tr>
<tr>
<td>When running the migration script, a Cannot Access, Cannot Open, or Insufficient Privilege error message occurs.</td>
<td>These errors refer to interactions between the migration script client and the Data Mover, not those between the Data Mover and the original NFS source file server. Ensure that you are running the migration script with sufficient user privileges and not as a different or unknown user, and that the VNX has given sufficient privileges or access rights to the script client. Check the options on the <code>server_export</code> command for this file system in the <em>EMC VNX Command Line Interface Reference for File</em>.</td>
</tr>
<tr>
<td>VNX runs out of space during the migration.</td>
<td>Extend the migration file system, and rerun the <code>server_cdms</code> command. If you extend the file system before the file system is full, the command can continue to run.</td>
</tr>
</tbody>
</table>
If there are multiple IP addresses associated with one hostname in the configuration and the `server_cdms <movername> -connect <mgfs>` command fails with the following error message in the server log:

```
nfs server xxxx not found
```

while the user is using hostname in the connection command, then the user should use the correct host IP address.

---

### CIFS

Table 12 on page 217 summarizes problems you might encounter while performing a CIFS migration with CDMS. Use the information in this table as a starting point to solve problems you might experience.
When a Windows 2000 or Windows Server 2003 system is renamed, the server key is not updated on the Key Distribution Center (KDC). This means the Windows 2000 or Windows Server 2003 system and the KDC have different server keys.

An `smb_session_setup` command returns the following error:

0x73 Status_more_processing_required.

**Workaround**

After renaming the Windows 2000 or Windows Server 2003 system, perform the following:

1. Unjoin the domain.
2. Remove the computer account on the domain controller for the system.
3. Rejoin the domain.
Using the server_cdms -info command

The server_cdms <movername> -info <mgfs> command can show connection and thread status. You can use it to display all MGFS file systems or a specific one. It has the ability to show threads in a particular state, and therefore, it is useful for locating ongoing or failed threads.

Examples

Example 1:
Display all MGFS file systems in default format:

$ server_cdms server_2 -info

Example 2:
Display connections and threads for one MGFS file system:

$ server_cdms server_2 -info mgfs1

Example 3:
Display only MGFS file systems in failed state:

$ server_cdms server_2 -info -state failed

Migration suspension (hang) conditions (NFS)

Most migration suspensions (or hangs) occur because of a problem between an NFS source file server and its associated Data Mover. By design, software does not distinguish among different kinds of errors between a source file server and Data Mover. The Control Station that is running the server_cdms command drops erroneous requests, and lets the command retry. If the problem is not temporary, the server.cdms command could retry continuously, resulting in a hang condition.

Causes

There are three main causes of hang conditions:

- Network problems between Data Mover and source file server
- Permission-related hangs
- Stale NFS handles
If you encounter a network problem, try to ping the NFS source file server from the Data Mover by running a `server_ping <movername>` command from the Control Station.

Most permission-related access problems can be identified by running the `dirprivU.pl` script against the file system before starting the migration. From the results of the `dirprivU.pl` script, you can fix or work around these problems before they actually occur on a migration.

By default, a Data Mover uses the file owner’s UID and GID to retrieve data. However, there can be potential problems with using this process, such as:

- **Missing `root=` or `anon=` options (in other words, missing permissions)**

  As a measure of self-protection, NFS only grants `other` privileges to remote users who make NFS requests with root UID identification. NFS grants `root` privileges to requests that come from IP addresses identified by a `root=` option on the `export` or `share` command. Therefore, if a file or directory on the source file server was created by `root` (thus, `root` is owner), the NFS request from the Data Mover to the source file server is only granted `other` privileges, which might not include read permission, unless the Data Mover’s IP address is included in a `root=` option on the source file server’s `export` command. A similar condition can exist between the Data Mover and the Control Station running the `server_cdms` command (that is, `root` access treated as `other`, and `other` does not have read privilege), but in that case, the command halts with a permission denied error, rather than a hang condition.

- **Infrequently, a file might prohibit read access by its owner**

  In this instance, the migration hangs. You need to unmount and remount the MGFS using the `useRootCredential mount` option (and ensure that the source file server grants `root=` option to the Data Mover), change the permission bits on the NFS source file server, or copy the file over manually. For example:

  ```
  $ server_cdms server_2 -connect mgfs1 -type nfsv2 -path /linux23 -source 128.221.252.244:/data -option useRootCred=true proto=UDP
  ```

- **A directory is missing the `x` (execute) attribute**

  ```
  $ server_cdms server_2 -connect mgfs1 -type nfsv2 -path /linux23 -source 128.221.252.244:/data -option useRootCred=true proto=UDP
  ```
Without the `x` attribute, a directory cannot be modified or extended in any manner.

- **Permission is not actually granted**

  In another permission-related hang, files that lack read in the `other` privileges can fail to migrate. The files can hang the `server_cdms` command even though you execute commands that access these files as `root`, not `other`.

  On several Solaris revisions, if you export file systems by using the `share –F nfs –o root=` command, and use a specific IP address as a parameter to the `root=` option, the command is accepted with no error message. However, root privilege is not granted to the identified system. It is treated as any default remote root for NFS access, and only receives “other” privileges. To have a specific single IP address given root privileges, you must use the `@` notation (normally used for subnets or network names).
For example, if the system server_3 had a 128.221.252.4 IP address, the command:

$ share -F nfs -o root=128.221.252.4 /mountpoint

does not grant server_3 root access on the Sun system. Instead, to grant it root access, the command line on the system should be:

$ share -F nfs -o root=@123.234.32.4/0 /mountpoint

The root privilege is also allowed if you use names in the command:

$ share -F nfs -o root=server_3 /mountpoint

The server_3 name with its appropriate address is in the local hosts file on the Sun system. The same holds true of an NFS export command from a Sun system, but only on Solaris.

Note: Using an IP address such as root=123.234.32.4 generally works as expected on aVNX, Network Appliance, and other UNIX variants.

Permission is subject to access control

In some UNIX implementations, the export or share command syntax includes the option of qualifying rw or ro with system names or addresses, for example, rw=sys1: sys2. In this instance, the command actually functions as a VNX would if the export options were -o access=sys1: sys2, rw. In other words, only the named systems are permitted access. The migration fails if the original export is modified for the migration to look like the following:

-option rw=sys1:sys2,root=DataMover IP

It only succeeds if the option looks like the following:

-option rw=sys1:sys2:Data Mover IP,root=Data Mover IP
Stale NFS handles

The Data Mover caches NFS handles. If the source file server changes the mount point or export during the migration, so the handle that the Data Mover caches is invalid, the migration hangs.

You should delete the files/directories affected by the hang, and then disconnect and reconnect the back-end mount of the NFS source file server:

- To delete a file, use the `rm` command on the UNIX workstation.
- To delete a directory, use the `getOnline` utility on the Control Station.

To handle a migration hang problem, perform the following:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
</table>
| 1    | Check all network connections.  
      | “Network problems between Data Mover and source file server” on page 219 provides more details. |
| 2    | Check the UxFS log file to see whether the MGFS executed a message about the Data Mover to the source NFS mount. |
| 3    | Check the file/directory at which the `server_cdms` command stops running, and look at the file owner, permissions, and timestamps associated with this file/directory.  
      | Access this file/directory manually, making sure it is not a client-side issue:  
      | • If the file does not allow the owner read access, disconnect and then reconnect using the `useRootCred` command option.  
      | • If the file’s owner is `root`, verify whether the NFS source file server grants the Data Mover root privilege.  
      | • If it is a directory, check its `x` (execute) permission.  
      | Check the file timestamp on the Data Mover and the NFS source file server. If they do not match, you must bring the file over manually. |
| 4    | If all these solutions do not solve the problem, report a bug. EMC recommends you take a snoop trace between the source file server and Data Mover.  
      | If you have an MGFS log file, this should also be copied and submitted as well.  
      | Appendix D, “Logging,” provides more details. |
Connection command failures (CIFS)

For CIFS migrations, not only must MGFS errors be analyzed, but Server Message Block (SMB) errors must be considered as well. An SMB error has the format of nnnnnnnn, which is similar to all standard CIFS errors.

The following list describes only the most frequent reasons for connection command failure. Go to http://msdn.microsoft.com for descriptions of all messages.

Summary

Table 13 on page 223 describes some common connection command failure messages.

Table 13  CIFS connection command error indications

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usermapper related failure</td>
<td>Usermapper cannot locate the domain used in the</td>
<td>Error c000015b</td>
<td>User login has not been granted.</td>
</tr>
<tr>
<td></td>
<td>connection command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Local path currently in use</td>
<td>The local path used in the connection command is</td>
<td>Error c00000cc</td>
<td>Unable to locate the specified share on the source file server.</td>
</tr>
<tr>
<td></td>
<td>currently in use by another connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time skews too much in a Windows 2000 domain</td>
<td>The CIFS source file server and target VNX do not have synchronized times.</td>
<td>Error 000005B</td>
<td>Status invalid primary group.</td>
</tr>
<tr>
<td>Error c000006d</td>
<td>Incorrect NetBIOS name and domain name, FQDN, or</td>
<td>Error 0000064</td>
<td>Status no such user.</td>
</tr>
<tr>
<td></td>
<td>usname/password.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error c00000be</td>
<td>The name resolution has failed.</td>
<td>Error c0000236</td>
<td>Connecting to a share on a Windows 2000 or 2003-based server by using a DNS alias fails.</td>
</tr>
</tbody>
</table>
This section provides a detailed explanation for errors listed in Table 14 on page 234.

### Usermapper-related failure

**MSDN text**

server_log error:

**Example**

SMB: 3: Usermapper:map2unix no domain conf for req=0

**Resolution**

Usermapper is used in the CDMS migration environment. It cannot locate the domain used in the connection command. If your Usermapper service uses a `usrmap.cfg` file, ensure that the file is correct because the `usrmap.cfg` file is used to specify the UID and GID ranges for each Windows domain it serves.

*Configuring VNX User Mapping* provides more information.

**Example**

SMB: 3: Usermapper RPC Error on (128.221.252100) fffffffe fffffffe

**Resolution**

After you modify the `usrmap.cfg` file, you must update all the old files, and restart Usermapper.

**Local path currently in use**

**MSDN text**

MGFS: 3: Invalid (type,cid), (3,6) should be (3,5)

**Example**

1) First connection with localpath 10_cifs1 succeeded;
   2003-01-17 11:36:08: ADMIN:4: Command succeeded: connect fsid=29 type=CIFS path=/10_cifs1
cifs=\ENG16954.MPFSDOM\ADMIN$\netbios=XFS90028_NT.MPFSDOM
account=MPFSDOM\administrator wins=10.169.0.54 passwd=*  
2) Second connection with the same 10_cifs1 local path failed with this error:
Resolution

The local path used in the server_cdms <movername> -connect <mgfs> command is currently in use by another connection. Use another local path for the new connection, or use the existing connection for migration.

Time skews too much in a Windows 2000 or Windows Server 2003 domain

Example

SMB: 3: SSXAK=c000006d origin=600 stat=d0000,-1765328196

Resolution

An error similar to this one means the CIFS source file server and the target VNX need to have times synchronized.

Error c000006d MSDN text

/*
 * MessageId: STATUS_LOGON_FAILURE
 * MessageText:
 * The attempted logon is invalid. This is either due to a bad username
 * or authentication information.
 */

Example

MGFS: 3: CIFS: Error: logon of user native\admin2 failed: c000006d

Resolution

Ensure that the correct NetBIOS name and domain name are used for migration in a Windows NT 4.0 domain.
Troubleshooting CDMS

**Note:** Renaming a NetBIOS name is explained in *Managing a Multiprotocol Environment on VNX*. The *EMC VNX Command Line Interface Reference for File* provides information about the `server_cifs` command.

Ensure that the correct FQDN is used for migration in a Windows 2000 or Windows Server 2003 domain.

Ensure that the username and password are correct.

**Error c00000be  MSDN text**

```c
/*
* MessageId: STATUS_BAD_NETWORK_PATH
* 
* MessageText: 
* 
* The network path cannot be located.
* */
```

**Example**

MGFS: 3: CIFS: Error: remote server XFS90028_NT.MPFSDOM or local server XFS90029_NT.MPFSDOM is unavailable: c00000be
MGFS: 3: Mount remote fs failed, status = 28
ADMIN:3: Command failed: connect fsid=33 type=CIFS path=/etc1
cifs=\XFS90028_NT.MPFSDOM\s2ufs1\ netbios=XFS90029_NT.MPFSDOM
account=MPFSDOM\administrator passwd=* 

**Resolution**

This error indicates that name resolution has failed. Therefore, the following items must be checked:

- **The Windows NT 4.0 domain**

  If a Windows NT 4.0 server can detect and map these two server shares, check:
  
  - The local hosts file and/or NIS if these two are used.
  
  - If WINS is used for the domain, use `-wins <wins>` in the `server_cdm` `<movername>` `-connect <mgfs>` command.

- **The Windows 2000 or Windows Server 2003 domain**
If a Windows 2000 server or Windows Server 2003 can detect and map these two server shares, check that DNS service is set up correctly, and has these two entries in it.

**Error c000015b**  
MSDN text

```c
/*
 * MessageId: STATUS_LOGON_TYPE_NOT_GRANTED
 * MessageText:
 * A user has requested a type of logon (e.g., interactive or network)
 * that has not been granted. An administrator has control over who might
 * login interactively and through the network.
 */
```

**Example**

```bash
SMB: 3: CIFSCallBackForMGFS is unable to init cifs migrate ctx, status is c000015b
MGFS: 3: CIFS: Error: migrate SD of svr2sh3/ failed: c000015b
MGFS: 3: Mount remote fs failed, status = 28
ADMIN:3: Command failed: connect fsid=78 type=CIFS path=/destpath
cifs=\\k40dvt31s2.dvt_a\svr2sh3\ netbios=k40dvt29s2.dvt_a account=dvt_a\cdmsadmin passwd=* 
```

**Resolution**

Ensure that the user has been permitted to access the CIFS source file server from the network. Use the `usrmgr.exe` utility to verify and add this option, if necessary.

**Error c00000cc**  
MSDN text

```c
/*
 * MessageId: STATUS_BAD_NETWORK_NAME
 * MessageText:
 * {Network Name Not Found}
 * The specified share name cannot be found on the remote server.
 */
```

**Example**

```bash
MGFS: 3: CIFS: Error: connect on share s3mgfs101 failed: c00000cc
MGFS: 3: Mount remote fs failed, status = 28
ADMIN:3: Command failed: connect fsid=23 type=CIFS path=/d2d_1
```
Troubleshooting CDMS

\[
cifs=\"XFS90029_W2K.CDMSW2K.XFS.ENG\"\s3mgfs101
netbios=XFS90028_W2K.CDMSW2K.XFS.ENG
account=CDMSW2K.XFS.ENG\administrator passwd=*
\]

**Resolution**

Ensure the netbiosname.domain for Windows NT 4.0, or the FQDN for Windows 2000 or Windows Server 2003 is correct. Ensure that the share name is spelled correctly.

**Error c000005B MSDN text**

CIFS: ALERT: migrate sd of \Perl\lib\perllocal.pod has unresolved ACLs,
status: c000005b
(CIFS: ALERT:depending on the setting of cifs.acl.mappingErrorAction param, this might no be an issue) <--
plan to get rid of this once the tree is open.

C000005 means [STATUS_INVALID_PRIMARY_GROUP].

**Resolution**

This error occurs when the primary group’s SID is replaced by the primary group SID of the user that is used for migration:

- Generally, this happens when the SID belongs to a group that is not supported on the VNX. The user should ignore the error.
- If the SID belongs to a local group that does not exist on the VNX, the user might not have run the lgdup.exe utility before migration began.

**Error c0000064 MSDN text**


**Resolution**

The user should check if there is a cdmsadmin user in the specified domain.

**Error c0000236 MSDN text**

2004-09-03 11:08:46: MGFS: 3: cannot establish connection between dnsalias.nasdocs.emc.com and celerra4.nasdocs.emc.com with share secondaryshare with user nasdocs.emc.com\Administrator status=c0000236

VNX File System Migration Version 2.0 for NFS and CIFS
Resolution

You receive an error similar to this when you try to create a connection to a secondary storage location that is a Windows 2000 or 2003-based file server and you use a DNS alias to refer to it in the connection creation command.

To resolve this issue, refer to the Microsoft Support website and search the Knowledge Base for article ID 281308.

Be sure you have software version 5.3.23.0, 5.4.20.0, 5.5, 5.6, or later.

Managing a failed verification or conversion

If the verification or conversion does not complete successfully, the error message (which appears on the VNX console and resides in the server log) gives the inode number of the file that failed the verify or convert.

To identify the file and file system that failed verification or conversion, perform the following steps from the VNX Control Station:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
</table>
| 1    | To display the server log output, use this command syntax:  
        `$ server_log <movername>`  
        where:  
        `<movername>` = name of the specified Data Mover  
        Output:  
        A failed verification appears in the server_log file as shown.  
        2004-12-07 14:14:57:MGFS:3:Fsid 157 inode 125887 of connection 6 is offline  
        Note that the file system ID is 157 and that the inode is 125887. |
To identify the file system name by using the file system ID obtained from the server log in step 1, type:

```bash
$ nas_fs -info id=157
```

```
id = 157
name = dmrk_test_fs
acl = 0
in_use = True
type = uxfs
volume = dmrk_test_vol
pool =
  rw_servers = server_3
  ro_servers =
  rw_vdms =
  ro_vdms =
  stor_devs =
    APM00033900124-0011, APM00033900124-0010
  disks = d7, d8
```

To identify the file associated with the inode number provided in the server log, examine the `dmrk_test_fs` file system. There are two ways to accomplish this:

- Explicitly mount the `dmrk_test_fs` file system on the Control Station.
- Look at the `dmrk_test_fs` file system in the Data Mover’s `/nas/rootfs` path after calculating the modified inode number used.
To explicitly mount the file system on the Control Station, perform the following:

a. Export the file system.

    [nasadmin@nsx]$ server_export server_3
    -Protocol nfs -option
    rw=128.222.10.232:10.6.3.140:128.221.252.100,
    root=128.222.10.232:10.6.3.140:128.221.252.100,
    access=128.222.10.232:10.6.3.140:128.221.252.100
    /dmrk_test_fs

    **Note:** Be sure to first examine the current export properties and include them when adding Control Station access. Otherwise, users lose access to the file system.

b. Create a directory where you mount the file system. General practice is to create one under the existing file system path /mnt.

    **Note:** This procedure requires you to be the root user.

    [root@nsx]# cd /mnt
    [root@nsx]# mkdir dmrk_test_fs

c. Mount the file system explicitly on the Control Station.

    [root@nsx]# mount server_3:/dmrk_test_fs
    /mnt/dmrk_test_fs

d. After the file system is mounted, change your working directory to the mounted file system.

    [root@nsx]# cd /mnt/dmrk_test_fs

e. To discover the name of the file that matches the inode number, run the following command:

    [nasadmin@nsx]$ find . -inum 125887 -print
    ./prof.h

    f. You can then recheck your findings by using the following command:

    [nasadmin@nsx]$ ls -il ./prof.h
    125887 -rw-rw-r-- 1 root bin 1411 Nov 18 16:08 ./prof.h

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>To explicitly mount the file system on the Control Station, perform the following:</td>
</tr>
<tr>
<td></td>
<td>a. Export the file system.</td>
</tr>
<tr>
<td></td>
<td>b. Create a directory where you mount the file system. General practice is to create one under the existing file system path /mnt.</td>
</tr>
<tr>
<td></td>
<td>c. Mount the file system explicitly on the Control Station.</td>
</tr>
<tr>
<td></td>
<td>d. After the file system is mounted, change your working directory to the mounted file system.</td>
</tr>
<tr>
<td></td>
<td>e. To discover the name of the file that matches the inode number, run the following command:</td>
</tr>
<tr>
<td></td>
<td>f. You can then recheck your findings by using the following command:</td>
</tr>
</tbody>
</table>
Analyzing the error

To determine why the error occurred, perform the following:

1. Evaluate the file from the `find` command.
2. Fix the problem.
3. Rerun the `server_cdms` command from the Control Station.

Since the `-verify <mgfs>` and `-Convert <mgfs>` options of the `server_cdms` command stops at the first error it encounters, you might need to repeat this troubleshooting process several times until the verification or conversion completes.

If you are unable to run the verification or conversion without errors, contact EMC Customer Service.

Removing an unwanted connection

It is possible to make a connection to any accessible file system or source file server. However, that might not be the connection you want or need right now.
For example:

- The chosen server or file system name might be mistaken.
- The proper connection has access permissions set so you cannot migrate all the data you need.
- The network path fails after connection.

**Note:** You can delete a migration or an MGFS by using the Unisphere Data Migration GUI.

You can remove the connection by using either the `specialCmd` utility or the `server_cdms` command.

### Using the `specialCmd` utility

You must remove the connection in order to reestablish a new one. Therefore, to remove the unwanted connection, perform the following steps from the Control Station:

1. Identify the file system ID (FSID) and the connection ID (CID) for the connection to be removed by using the following command:

   ```
   $ server_cdms <movername> -info <mgfs>
   ```

2. Go to the `/nas/tools/cdms` directory.

3. Remove the connection by using the following command:

   ```
   $ ./specialCmd disconnect {<movername>}{<fsid>}{<cid>}
   ```

   **Note:** If the CID is omitted from the command line, all connections to the file system fsid are removed.

4. To remove the local path, use the following command:

   ```
   $ rmdir <localpathname>
   ```

5. Rerun the migration, as appropriate.

### Using the `server_cdms` command

To remove the unwanted connection without migrating the data, use the following command:

```
$ server_cdms -disconnect <mgfs> {-path <localpath>|-path <cid>|-all}
```
where:

<mgfs> = specified migration file system on the Data Mover
<localpath> = VNX local pathname (or the subdirectory to the mount point)
<cid> = connection identifier (CID) for the source file server to the Data Mover

You must remove the directory after the connection is removed by using the
**server_cdms <movename> -disconnect <mgfs>** command.

## Error codes

During NFS and CIFS migrations, in the source file server log, you might see one of the
decimal error numbers listed in **Table 14 on page 234**. The server log gives you a
better understanding of the error by interpreting the number to a user-friendly text
code, and providing possible reasons for or methods to recover and fix the problem.

<table>
<thead>
<tr>
<th>Error code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>File_OK</td>
</tr>
<tr>
<td>1</td>
<td>File_WouldBlock</td>
</tr>
<tr>
<td>2</td>
<td>File_Too_Big</td>
</tr>
<tr>
<td>3</td>
<td>File_Table_Overflow</td>
</tr>
<tr>
<td>4</td>
<td>File_BadFile</td>
</tr>
<tr>
<td>5</td>
<td>File_InvalidArgument</td>
</tr>
<tr>
<td>6</td>
<td>File_NameTooLong</td>
</tr>
<tr>
<td>7</td>
<td>File_NotFound</td>
</tr>
<tr>
<td>8</td>
<td>File_DuplicateEntry</td>
</tr>
<tr>
<td>9</td>
<td>File_StaleHandle</td>
</tr>
<tr>
<td>10</td>
<td>File_NotDirectory</td>
</tr>
<tr>
<td>11</td>
<td>File_IsDirectory</td>
</tr>
<tr>
<td>12</td>
<td>File_NotEmpty</td>
</tr>
<tr>
<td>13</td>
<td>File_NotFile</td>
</tr>
<tr>
<td>Error code</td>
<td>Meaning</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>14</td>
<td>File_NotLink</td>
</tr>
<tr>
<td>15</td>
<td>File_TooManyLinks</td>
</tr>
<tr>
<td>16</td>
<td>File_CrossDevice</td>
</tr>
<tr>
<td>17</td>
<td>File_NoPermission</td>
</tr>
<tr>
<td>18</td>
<td>File_NotOwner</td>
</tr>
<tr>
<td>19</td>
<td>File_BufferFull</td>
</tr>
<tr>
<td>20</td>
<td>File_EndOfFile</td>
</tr>
<tr>
<td>21</td>
<td>File_Busy</td>
</tr>
<tr>
<td>22</td>
<td>File_Pending</td>
</tr>
<tr>
<td>23</td>
<td>File_IO_Error</td>
</tr>
<tr>
<td>24</td>
<td>File_NoMemory</td>
</tr>
<tr>
<td>25</td>
<td>File_NoSpace</td>
</tr>
<tr>
<td>26</td>
<td>File_IsReadOnly</td>
</tr>
<tr>
<td>27</td>
<td>File_DirtyFileSystem</td>
</tr>
<tr>
<td>28</td>
<td>File_InternalError</td>
</tr>
<tr>
<td>29</td>
<td>File_IncompleteOp</td>
</tr>
<tr>
<td>30</td>
<td>File_Remote</td>
</tr>
<tr>
<td>31</td>
<td>File_RemoteUnreachable</td>
</tr>
<tr>
<td>32</td>
<td>File_ParentDirExported</td>
</tr>
<tr>
<td>33</td>
<td>File_ChildDirExported</td>
</tr>
<tr>
<td>34</td>
<td>File_TooManyExportFS</td>
</tr>
<tr>
<td>35</td>
<td>File_TooManyMountedFS</td>
</tr>
<tr>
<td>36</td>
<td>File_ExportConflict</td>
</tr>
<tr>
<td>37</td>
<td>File_NotExported</td>
</tr>
<tr>
<td>38</td>
<td>File_NetworkInterfaceNotFound</td>
</tr>
</tbody>
</table>
### Table 14 Migration error codes (page 3 of 4)

<table>
<thead>
<tr>
<th>Error code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>39</td>
<td>File_ScsiCfgInvalidChain</td>
</tr>
<tr>
<td>40</td>
<td>File_ScsiCfgChainNotFound</td>
</tr>
<tr>
<td>41</td>
<td>File_ScsiCfgTidMismatch</td>
</tr>
<tr>
<td>42</td>
<td>File_ScsiCfgSymidMismatch</td>
</tr>
<tr>
<td>43</td>
<td>File_ScsiCfgMarkFailed</td>
</tr>
<tr>
<td>44</td>
<td>File_ScsiCfgMarkBadTidlum</td>
</tr>
<tr>
<td>45</td>
<td>File_ScsiCfgMarkInternalErr</td>
</tr>
<tr>
<td>46</td>
<td>File_ScsiCfgMarkDiskErr</td>
</tr>
<tr>
<td>47</td>
<td>File_ScsiCfgNoDrive</td>
</tr>
<tr>
<td>48</td>
<td>File_UmountNotMounted</td>
</tr>
<tr>
<td>49</td>
<td>File_TooManyRoutes</td>
</tr>
<tr>
<td>50</td>
<td>File_RouteEntryInUse</td>
</tr>
<tr>
<td>51</td>
<td>File_Frozen</td>
</tr>
<tr>
<td>52</td>
<td>File_QuotaExceeded</td>
</tr>
<tr>
<td>53</td>
<td>File_NotLocked</td>
</tr>
<tr>
<td>54</td>
<td>File_RangeNotLocked</td>
</tr>
<tr>
<td>55</td>
<td>File_LockConflict</td>
</tr>
<tr>
<td>56</td>
<td>File_LockNotGranted</td>
</tr>
<tr>
<td>57</td>
<td>File_OplockNotGranted</td>
</tr>
<tr>
<td>58</td>
<td>File_LockNotSupported</td>
</tr>
<tr>
<td>59</td>
<td>File_BadPath</td>
</tr>
<tr>
<td>60</td>
<td>File_RPC_ERR</td>
</tr>
<tr>
<td>61</td>
<td>File_InvalidCharacter</td>
</tr>
<tr>
<td>62</td>
<td>File_BadCookie</td>
</tr>
</tbody>
</table>
As of version 5.6, all new event, alert, and status messages are identified by a longer message ID (for example, 13421838337) and are designed to provide detailed information and recommended actions to help you troubleshoot the situation.

In the VNX CLI:

- Use the `nas_message -info <message_id>` command to retrieve the detailed information for a particular error.

In Unisphere:

- Right-click an event, alert, or status message and select to view Event Details, Alert Details, or Status Details.

The *Celerra Network Server Error Messages Guide* provides information about error messages that use an earlier-release message format.
Troubleshooting CDMS
APPENDIX A
Using CDMS Migration Tools

The following topics provide information about syntax and parameters for all CDMS utilities and scripts:

- Introduction ........................................................................................................ 240
- backupWrapper.exe utility .................................................................................. 243
- connBuilder.pl script ...................................................................................... 244
- ch_group.pl script .......................................................................................... 249
- dircount.pl script .......................................................................................... 254
- dirprivU.pl script .......................................................................................... 262
- dirprivW.pl script .......................................................................................... 270
- diskUsage.pl script ......................................................................................... 275
- lgdup.exe utility .............................................................................................. 283
- sharedup.exe utility ......................................................................................... 284
- specialCmd utility ........................................................................................... 286
Introduction

This appendix contains information about available utilities, scripts, and a command to help you manage the CDMS environment. Table 15 on page 241 provides more information about each of these tools.

<table>
<thead>
<tr>
<th>NFS</th>
<th>The following executables and scripts are used during a NFS migration:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>◆ File System Selective GID Change script (ch_group.pl)</td>
</tr>
<tr>
<td></td>
<td>◆ File System Directory Tree Counting script (dircount.pl)</td>
</tr>
<tr>
<td></td>
<td>◆ File System Privilege Checking script (dirprivU.pl)</td>
</tr>
<tr>
<td></td>
<td>◆ Disk Usage script (diskUsage.pl)</td>
</tr>
<tr>
<td></td>
<td>◆ Migration command (server_cdms)</td>
</tr>
<tr>
<td></td>
<td>◆ Special Command utility (specialCmd)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CIFS</th>
<th>The following executables and scripts are used during a CIFS migration:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>◆ BackupWrapper script (backupWrapper.exe)</td>
</tr>
<tr>
<td></td>
<td>◆ Connection Command Builder script (connBuilder.pl)</td>
</tr>
<tr>
<td></td>
<td>◆ File System Directory Tree Counting script (dircount.pl)</td>
</tr>
<tr>
<td></td>
<td>◆ Disk Usage script (diskUsage.pl)</td>
</tr>
<tr>
<td></td>
<td>◆ File System Privilege Checking script (dirprivW.pl)</td>
</tr>
<tr>
<td></td>
<td>◆ Local Group Duplication utility (lgdup.exe)</td>
</tr>
<tr>
<td></td>
<td>◆ Migration command (server_cdms)</td>
</tr>
<tr>
<td></td>
<td>◆ Share Migration utility (sharedup.exe)</td>
</tr>
<tr>
<td></td>
<td>◆ Special Command utility (specialCmd)</td>
</tr>
</tbody>
</table>
Table 15 on page 241 briefly describes the tools that can be run during CDMS migrations.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Application</th>
<th>Description</th>
</tr>
</thead>
</table>
| backupWrapper.exe utility   | CIFS        | • Gives the dircount.pl, dirprivW.pl, and diskUsage.pl Perl scripts necessary backup operator privileges so the script can read all files during CIFS migrations.  
• Runs on Windows clients. |
| connBuilder.pl script       | CIFS        | • Generates a template for connection commands for top-level shares.  
• Creates a list of files that are candidates for the exclude file parameter for the server_cdms command.  
• Runs on Windows clients before migration. |
| ch_group.pl script          | NFS         | • Used in situations where there are duplicate GIDs in migrated, consolidated file systems. One example is after consolidation from several source file servers where the users were not managed under a common NIS.  
• Used to change GID numbers in a directory or subdirectory tree, only if they match a given GID number. All other files remain unchanged.  
• Runs on UNIX workstations. |
| dircount.pl script          | NFS and CIFS| • Gathers information allowing you to see what a directory tree structure on the source file server looks like, and how many files are at each level.  
• Runs on UNIX workstations or Windows clients before migration. |
| dirprivU.pl script          | NFS (U) and CIFS | • Checks file privileges and access on all files on the source file server.  
• Determines what files can be read in the file system, and therefore migrated successfully (use prior to data migration).  
• Uses a directory or pathname to a directory as an input parameter.  
• Runs on UNIX workstations or Windows clients before migration. |
| dirprivW.pl script          | CIFS (W)    |             |
| diskUsage.pl script         | NFS and CIFS| • Estimates the amount of storage space you need on the target VNX Network Server when you migrate a specified directory or file.  
• Provides a list of the amount of data in various file sizes used in estimating migration times.  
• Runs on UNIX workstations before migration. |
<table>
<thead>
<tr>
<th>Tool</th>
<th>Application</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lgdup.exe utility</td>
<td>CIFS</td>
<td>• Runs on each source file server to migrate local groups to Data Movers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Migrates local groups from Data Mover to Data Mover when run on a third Windows host.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Uses the same database for all the local groups from each source file server. If the same local group exists on multiple source file servers,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>only one local group resides on the Data Mover unless the Prefix option is used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• On a WINS server, registers the IP address and NetBIOS name pair.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Runs on Windows clients before migration.</td>
</tr>
<tr>
<td>server_cdms command</td>
<td>NFS and CIFS</td>
<td>• Establishes and removes connections to remote systems.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Allows a user to start on-access migration.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Creates automigration process on the Data Mover.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Checks the state of the MGFS file system, and the automigration process.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Reports if all data has been migrated successfully.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Runs from a Control Station.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The <em>EMC VNX Command Line Interface Reference for File</em> provides more information.</td>
</tr>
<tr>
<td>sharedup.exe utility</td>
<td>CIFS</td>
<td>• Requires an MGFS connection between the source file server and the target Data Mover.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Is system independent. It can be run on a source file server, a third server, and so forth, except on a Data Mover.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• All share names migrated by this tool are made local to the particular NetBIOS name on a Data Mover.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Used in the planning stages to evaluate shares on the source file server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Runs on Windows clients before migration.</td>
</tr>
<tr>
<td>specialCmd utility</td>
<td>NFS and CIFS</td>
<td>• Brings offline directories online for migration.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Disconnects a current operation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Runs from a Control Station.</td>
</tr>
</tbody>
</table>
**backupWrapper.exe utility**

The **backupWrapper.exe** utility, based to run with three Perl scripts, ensures that the necessary backup operator privilege is invoked for the running process specified in the utility's first parameter. Even if a Perl script is run by a user who is a member of the backup operators group, running a program under that user does not mean the program automatically exercises backup operator privileges. The **backupWrapper.exe** utility makes the Perl script use that privilege. This action is performed so the script can attempt to read all the files on the source file server, even files with access rights restricted exclusively to owners, or other such access restrictions.

---

**Tool location**

Windows 2000, Windows Server 2003, and Windows NT 4.0 clients

**Syntax and parameters**

This utility has the following syntax and parameters:

```
C:\> backupWrapper  <perlScript>  [option1] [option2]
```

where:

- `<perlScript>` = dircount.pl, diskUsage.pl, dirprivW.pl
- `<option>` = standard options for each of the supported scripts

**Examples**

Prior to data migration, if you want to gather information about the first 25 directories in the `/temp/user1/files/` directory tree, type:

```
C:\> backupWrapper dircount.pl /temp/user1/files/  [25]
```

Prior to data migration, if you want to estimate the minimum amount of disk storage you will need on the target server when you migrate a specified directory or file, type:

```
C:\> backupWrapper diskUsage.pl -m
```

Prior to data migration, verify that all files can be read from the `/new/test/file1` directory, and therefore allowing a successful migration to the MGFS, type:

```
C:\> backupWrapper dirpriv.pl /new/test/file1
```
connBuilder.pl script

The connBuilder.pl script builds your connection command templates when there are many shares to migrate, eliminating potential typing errors. The script also creates a list of files that are candidates for the -exclude <exclude_path> file parameter of the server_cdms command. This action is for instances where you are migrating a CIFS source file server by moving all data under the server’s administrative shares (disk volumes, for example, C$, E$), but excluding unshared or unwanted files and directories. To run this script correctly, in addition to the Perl script, Microsoft’s Word and Excel applications must be started on the system where this script executes.

---

**Tool location**

Windows 2000, Windows Server 2003, and Windows NT 4.0 clients

**Syntax and parameters**

This script has the following syntax and parameters:

```bash
C:\> perl connBuilder.pl <input_file>.doc [-drive]
```

where:

- `<input_file>` = filename produced by the sharedup.exe utility using the /FO option
- `-drive` = if used, the local path needed in the server_mount commands is /drv/<share_name> instead of /<share_name>

For example:

```
/C/<share_name>, if "@Drive:C" is in the <input_file>.doc file.
```

Manual editing is still necessary in the following situations:

- Input word file
- Command file template
- Exclude file template

---

**Input Word file**

You must add the following entries to the Word <input_file>.doc argument; otherwise, you need to modify the command script created by the connBuilder.pl script.

These entries are case-sensitive, and there is no space after the colon ():

- @Domain:<domain_name>
The connBuilder.pl script creates a template for a command file. The file’s format is similar to that used for the server_mount command on a Control Station that creates the CDMS connection to the source file server. However, the following items in this command template might need to be edited.

Look at an example of the Win2KJ.cmd output file:

```
server_cdms server_2 -connect mgfs1 -type cifs
```
Using CDMS Migration Tools

-path /share1 -netbios DM4.w2kj.domain -source '\\WIN2KJ.w2kj.domain\share1'  
-admin 'w2kj.domain\cdms_migrator'

server_cdms server_2 -connect mgfs1 -type cifs  
-path /share2 -netbios DM4.w2kj.domain -source '\\WIN2KJ.w2kj.domain\share2'  
-admin 'w2kj.domain\cdms_migrator'

---

Note: The password is read interactively from the user.

This </server_name>.cmd command must be run from the Control Station.

- A password file.
  You must create a password file named </passwd> containing the valid password in  
  the same directory as the </server_name>.cmd command.

- The server name.
  You must change it to the corresponding server. The default is </server_2>.

- Share names with spaces.
  You must replace the space with an asterisk (*) because a space is interpreted as  
  the end of the command option string.

- The local path.
  If no -drive option is used, the default local path always uses the same directory  
  name as the remote share. If the -drive option is used, the local path is always the  
  local=/drv/share_dir.
  For example, suppose there is a share named accounting whose directory name is  
  acct on the G: drive. If there is no -drive option, it would be path=/acct; otherwise,  
  it would be path=G/acct. If you do not want to use this naming convention for  
  your local path, you must change it manually.

---

Exclude file template

The exclude file lists all unshared directories and files in the root of the drv$ (for  
example, C$, E$) file system. If you want to connect to the default root shares of that  
file system, using this with the server_cdms command migrates only shared  
directories. If you want to migrate more files or directories, you only need to remove  
the directories or files you want to migrate from this file before using it with the  
server_cdms command. Everything else is excluded during migration. It assumes that  
the MGFS is always mapped to the M: drive on the Windows client for migration.  
Therefore, the exclude file content is always similar to:

- f /path/file_name
Based on how the MGFS is exported or shared, the local path is used for the connection command, and the share is mapped. You need to change all the M: drive notations to the correct drive, and possibly add parent directories for all entries.

For example, if you assume that:

- The file system is exported at the mount point of the /mgfs_share MGFS.
- The system on which you are running has mapped that to the H: drive.
- The local path path=/G is being used to connect to the G$ share directly.

Then the exclude file must have the following modifications:

- f /mgfs_share/G/file_name
- d /mgfs_share/G/dir_name

- d /path/sub_dir
Look at an example of an exclude-WIN2KJ-C.txt exclude file:

```
exclude-WIN2KJ-C.txt
f /mgfs_share/AUTOEXEC.BAT
f /mgfs_share/CONFIG.SYS
d /mgfs_share/Documents and Settings
f /mgfs_share/IO.SYS
d /mgfs_share/Inetpub
f /mgfs_share/MSDOS.SYS
f /mgfs_share/NTDETECT.COM
d /mgfs_share/Perl
d /mgfs_share/Program Files
d /mgfs_share/RECYCLER
d /mgfs_share/SFU
d /mgfs_share/System Volume Information
d /mgfs_share/WINNT
f /mgfs_share/_Argon_.tmp
f /mgfs_share/arcldr.exe
f /mgfs_share/arcsetup.exe
f /mgfs_share/boot.ini
f /mgfs_share/bootfont.bin
f /mgfs_share/cmd.doc
f /mgfs_share/ntldr
f /mgfs_share/pagefile.sys
d /mgfs_share/sharetest
d /mgfs_share/sp3
d /mgfs_share/temp
d /mgfs_share/test share
f /mgfs_share/win2ktowin98.cap
```
ch_group.pl script

The ch_group.pl script changes a GID number if it matches another GID number in a directory or subdirectory tree. All other files remain unchanged, however.

This tool is in the form of a Perl script, and since this script uses the equivalent of a chgrp command, you must be root (superuser) when you run it. If you are running the script against a mounted, remote file system, its server must also have granted remote root rights to your system.

Tool location

UNIX workstations

Syntax and parameters

This script has the following syntax and parameters:

$ perl ch_grp.pl <directory> <old_GID> <new_GID> [-s]

The ch_group.pl script takes three or four parameters; the first three are mandatory and the fourth is optional. If any of the mandatory parameters are missing, the prompt requests that you type it manually. You can run the script with all parameters on the command line, or none of them:

◆ The first parameter, <directory>, is a directory or a pathname to a directory. The directory can be a subdirectory or a mounted remote (NFS) directory. This is the root directory of the tree that is parsed and checked for change candidates. This directory is also checked to see if it should be changed.

◆ The second parameter, <old_GID>, is the number of the original GID. This number is the number to be changed, or the number to check for on the inode.

◆ The third parameter, <new_GID>, is a number for the new GID. This is the number to be set on the inode if the GID originally in the inode matched the second parameter.

◆ The optional fourth parameter, -s, suppress verification output, is used if you are including this command as part of a shell script, for example.

If this parameter is missing or is anything other than -s, the command displays the full verification. If the script has to query for manual entry of any other parameter, the verification output is not suppressed.
Functionality
A CDMS migration can do file system consolidation from different UNIX workstations. Those systems can have their logins, and UID and GID allocation managed from a common NIS. However, if any of those data-source systems had local user logins or multiple NISs, there might be GID duplication in the migrated, consolidated file system.

Example
If on one source file server, the finance group had GID 105, and on another source file server, the engineering group had GID 105, then on the consolidated file system, members of engineering could have access to some finance files, and vice versa. The opposite could also be true—finance for manufacturing could be GID 105 on the manufacturing systems, but HQ finance could be GID 227 on the administrative systems. Now the need is to consolidate the finance departments and their files. Of course, you can export the subdirectories of a consolidated file system in such a way that the individual users would only mount their own group’s subset of the consolidated file system. The finance group, for example, only mounts export A and engineering only mounts export B. However, that situation might not be operationally true indefinitely. Furthermore, if the customer consolidates file systems, then consolidating users or file system access is likely to be part of the overall plan.

Old versus new GIDs
There are many tools available to the UNIX administrator to manage users and groups. These tools can do things such as changing the GID numbers for a set of users. However, all the files ever written in the past by those users still have the old GID in their inodes. The newly renumbered user might only get anybody or other rights on the migrated file because migration preserves inode contents, and the new group does not match the old GID in the inode. Most UNIX workstations have a chgrp command that changes the GID of an individual file. However, if the chgrp command is used recursively (that is, with the –r parameter), it changes every file in the tree structure, not just files belonging to a specific group. Most UNIX implementations do not have a command that gives new GIDs to files only if they match a specific prior one.

“Migrating files with the SetGroupID on execute bit on” on page 307 provides more information.

Note: Linux has a special -from UID:GID option on its chgrp command, but that is an exception. There is not a similar option on the chgrp commands in the Sun operating system or similar systems.
Most UNIX administrators can selectively change user group IDs, but can only modify a GID on files in an overall fashion. All the files in the same directory tree change to the same GID, and that tree cannot include other files that should not be changed. If the files that need a GID modified are scattered in various directories around a file system, there is no easy way to do it.

The ch_group.pl script fulfills the need for changing GIDs only if they match a prior GID in a directory or subdirectory tree without touching other files in that tree. The most likely use of this script might be after consolidation migrations from independent systems.

Invoke the ch_group.pl script from the shell with the parameters you want. For example, assuming your Perl interpreter was in the /etc/bin/perl file, you could include one parameter by using the following command:

```
$ /etc/bin/perl/perl ch_group.pl </mnt/remote1>
```

where:

`</mnt/remote1>` = full path to a directory whose GIDs might need changing

It can be a mounted, remote directory or a subdirectory. In this case, since a mandatory parameter is missing, the script responds by asking for the other two parameters, and includes the verification question:

```
Value of old GID to be changed ? 1003
  New GID value to replace old ? 1010

In a directory tree starting at /mnt/remote1 check all files, and if GID is =1003 make it =1010 ? Proceed ?
```

- If the old GID and new GID in the directory path are correct, type y (lowercase letter y, and then press Enter).
- If the path or GIDs are typed incorrectly, answer n to safely exit the script.

Using --s as the fourth parameter on the command line causes this question to be skipped altogether. The script changes GIDs without prompting you to verify the parameters. This has a certain amount of risk, but can be useful. Suppose, after a file system consolidation for the finance department, the finance users of four separate servers (and four original GIDs) had also been consolidated into a common GID (1010 in the following example). It might be useful to run a shell script such as:

```
for each gi ( 1002 1034 1011 1005 )
```
echo = group $gi =
perl ch_group.pl /mnt/finance $gi 1010 -s
end

You do not have to be present to answer the check question at each run of the script. Of course, you need to be sure every parameter is correct before you run the shell script:

- The ch_group.pl script tries to open the directory named in the initial parameter. The script fails if that name cannot be opened.
- If the name is valid, the directory opens, and GIDs are checked name by name.
- If any of the names are subdirectories, the script proceeds to recursively open the subdirectory.
- If verification output is not suppressed, then for each successful match-and-change, a changed /<pathname> message appears.
- If the GID matched but could not be changed, a Failed to change /<pathname> message appears; however, the Perl script does not stop. It continues on to other files in the file system.

**Note:** Nothing appears on the window if the GID does not match (no change required). Therefore in a large file system, there might be long periods of time with no activity on the window.

- If a directory cannot be opened, a cannot open directory /<pathname> error message appears, and the script stops.
If the script completes successfully, a count message before exiting appears. For example:

In a total of 17865 names, there were 561 changed

If the –s parameter is used, you do not see the verification question upon starting, and there are no changed messages. However, any failure or error messages are still displayed upon exit as well as the count message.

---

**Known issues**

There are known issues associated with the following:

- Perl script
- Access rights

**Perl script**

The `ch_group.pl` script works under any Perl script version 5.6 or later, but might not work under earlier revisions.

**Access rights**

You must have root permissions for the script to be successful.
dircount.pl script

The dircount.pl script gathers information about a file system’s directory tree, which allows you to observe what a directory tree structure looks like, how many files are at each level, and so on. The script should be used before migration begins so you can check the feasibility of partial migrations, splitting workload during a migration, and so forth.

This tool allows you to understand:

- What is the lowest level of subdirectory in the file system?
- Where is the subdirectory located?
- Which directory has the most subdirectories?
- Which directory has the most files?

---

**Tool location**

UNIX workstations, and Windows 2000, Windows Server 2003, and Windows NT 4.0 clients

**Syntax and parameters**

This script has the following syntax and parameters:

**UNIX:**

```
$ perl dircount.pl <directory> [<<#>]
```

**Windows:**

```
C:\> backupWrapper dircount.pl <directory> [<<#>]
```

- The first parameter, `<directory>`, is a directory or a pathname to a directory. It can be a subdirectory, a mounted, remote NFS directory, or a remote CIFS share. If there is no directory parameter, the script fails.

- The optional second parameter, `<#>`, limits the depth of subdirectories examined and parsed in the (first parameter) directory. When this depth limit is reached, the numbers of subdirectories are counted in that directory at that level, but none are entered. The full breadth of the tree (to that level) is traversed, only the depth is limited. If this parameter is missing, a maximum depth of 255 is assumed.

- The log file is automatically created in the current directory, and it retains all command outputs.
Running the script

**UNIX**

For NFS migrations, invoke the dircount.pl script from the shell with a directory name as a parameter.

For example, assuming your Perl interpreter was in the /etc/bin/perl file, use the following command:

```
$ /etc/bin/perl/perl dircount.pl /mnt/<bgdata>
```

If the Perl script was installed as a package under UNIX software, you can invoke it as a command from any directory by running the following command:

```
$ /perl dircount.pl /mnt/<bgdata>
```

where:

`/mnt/<bgdata>` = a directory name (or full path to a directory) whose structure you want to examine

**Windows**

For CIFS migrations, the Perl script must be installed as a package under Windows by using the following command:

```
C:\> dircount.pl <H:\bgdata>
```

where:

`<H:\bgdata>` = a directory name (or full path to a directory) whose structure you want to examine

**Functionality**

The dircount.pl script progresses downward in that directory and counts files. When it comes to a subdirectory, it recursively opens that subdirectory. This action increases the depth level by one. The script works its way along this lower level until it comes to a subdirectory, and so forth. When it locates a directory containing no further subdirectories, it counts the number of files, and outputs the depth and file count. Then the script goes up to the next level and looks through that next higher level directory for a subdirectory. When every subdirectory at that lowest part of the branch of the file system tree is examined, it reports the depth and file and directory counts.

This display line appears in comma-separated variable (CSV) format, so if it is logged or placed into an output file, you can read that file into a spreadsheet for further examination. Then the script goes up to the next level again and looks for a subdirectory to search and report, counting other files during that time.
Eventually, the script parses the entire tree and displays counts at the level from which you started. It also maintains a running count of the number of directories and filenames so you know the totals. It displays that information at the completion of the examination.

When the dircount.pl script begins to output information, you observe information similar to the following data:

```
Depth,Sub-Dir,Names,In /usr
4,0,4,/platform/SUNW,Sun-Blade-100/lib/sparcv9
5,0,7,/platform/SUNW,Sun-Blade-100/lib/picl/plugins
4,1,3,/platform/SUNW,Sun-Blade-100/lib/picl
5,0,4,/platform/SUNW,Sun-Blade-100/lib/abi/sparcv9
4,1,5,/platform/SUNW,Sun-Blade-100/lib/abi
3,3,12,/platform/SUNW,Sun-Blade-100/lib
2,1,5,/platform/SUNW,Sun-Blade-100
4,0,4,/platform/SUNW,Sun-Blade-1000/lib/sparcv9
5,0,5,/platform/SUNW,Sun-Blade-1000/lib/picl/plugins
4,1,3,/platform/SUNW,Sun-Blade-1000/lib/picl
5,0,4,/platform/SUNW,Sun-Blade-1000/lib/abi/sparcv9
4,1,5,/platform/SUNW,Sun-Blade-1000/lib/abi
3,3,12,/platform/SUNW,Sun-Blade-1000/lib
2,1,5,/platform/SUNW,Sun-Blade-1000
4,0,4,/platform/SUNW,Ultra-1/lib/sparcv9
5,0,4,/platform/SUNW,Ultra-1/lib/abi/sparcv9
4,1,5,/platform/SUNW,Ultra-1/lib/abi
3,2,13,/platform/SUNW,Ultra-1/lib
2,1,6,/platform/SUNW,Ultra-1
----
and so on until
----
2,6,77,/vmsys/HELP
3,0,8,/vmsys/OBJECTS/dos
3,0,6,/vmsys/OBJECTS/lp
3,0,8,/vmsys/OBJECTS/mail
3,0,9,/vmsys/OBJECTS/pref
3,0,7,/vmsys/OBJECTS/programs
3,0,4,/vmsys/OBJECTS/spell
2,6,34,/vmsys/OBJECTS
2,0,29,/vmsys/bin
2,0,4,/vmsys/lib
3,0,3,/vmsys/standard/WASTEBASKET
3,0,5,/vmsys/standard/pref
2,2,5,/vmsys/standard
1,5,8,/vmsys
2,0,17,/ucbinclude/sys
2,0,3,/ucbinclude/ufs
1,2,21,/ucbinclude
```
0,30,45,
There are a total of 39389 names in a total of 1628 directories.

Log file: /home/work/new/dircount_usr.log

When the dircount.pl script begins to output information, you observe information similar to the following data:

Depth, Sub-Dir, Names, In C:\>perl
1,0,78,/bin
2,0,37,/eg/aspSamples
2,0,3,/eg/cgi
3,0,23,/eg/Core/cgi
3,0,8,/eg/Core/g
3,0,10,/eg/Core/scan
3,0,6,/eg/Core/sysvipc
3,0,6,/eg/Core/van
2,5,29,/eg/Core
2,0,3,/eg/fork
2,0,10,/eg/IEExamples
2,0,4,/eg/Windows Script Components
2,0,10,/eg/Windows Script Host
1,7,11,/eg
3,0,17,/html/ASPNPerl/img
2,1,5,/html/ASPNPerl
3,0,5,/html/Components/Windows
2,1,4,/html/Components
----

and so on until

----
3,1,30,/site/lin/URI
4,0,4,/site/lib/Win32/API
4,0,10,/site/lib/Win32/OLE
3,2,35,/site/lib/Win32
4,0,3,/site/Win32API/File
4,0,3,/site/Win32API/Registry
3,2,7,/site/Win32API
4,0,3,/site/WWW/RobotRules
3,1,4,/site/WWW
5,0,20,/site/lib/XML/Parser/Encodings
6,0,5,/site/lib/XML/Parser/Expat/bin
6,0,4,/site/lib/XML/Parser/Expat/gennmtab
6,0,4,/site/lib/XML/Parser/Expat/lib
6,0,4,/site/lib/XML/Parser/Expat/sample
6,0,11,/site/lib/XML/Parser/Expat/xmlparse
6,0,19,/site/lib/XML/Parser/Expat/xmltok
6,0,13,/site/lib/XML/Parser/Expat/xmlxlf
5,7,12,/site/lib/XML/Parser/Expar
4,2,6,/site/lib/XML/Parser
3,1,5,/site/lib/XML
4,0,5,/site/lib/XMLRPC/Transport
3,1,5,/site/lib/XMLRPC
2,31,476,/site/lib
1,1,3,/site
0,5,9

There are a total of 4222 names in a total of 371 directories

Log file:
C:/Perl/site/lib/Win32/API/new\dircount_c_Perl.log

Directory structure format is comma-separated so you can redirect it to a file, and import that file to a spreadsheet for sorting or some other type of data manipulation.

Each output line is three numbers and a name, as described in Table 16 on page 258.

### Table 16  Window output format descriptions

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>First number</td>
<td>Depth of subdirectory; start directory level = 0, subdirectory of the start directory = 1, next level subdirectory = 2, and so on.</td>
</tr>
<tr>
<td>Second number</td>
<td>The number of subdirectories in this specific directory at this level. If you are at the furthest level of any branch of the tree, this is 0 (no further subdirectories). Other branches might have more levels, but 0 is the furthest on this part.</td>
</tr>
<tr>
<td>Third number</td>
<td>The number of named inodes, including files, pipes, devices, and so forth, used in this subdirectory at this level.</td>
</tr>
<tr>
<td>Name</td>
<td>The pathname below the initial path to this subdirectory.</td>
</tr>
</tbody>
</table>

If the dircount.pl script executes successfully, eventually it parses the whole subdirectory tree, exhausts all the names it can find, does a quick final calculation, and displays the following message:

There are a total of 1156 names in a total of 67 directories

**Note:** This message does not appear when errors occur that cause the script to fail.

Assuming there were no errors or no inaccessible files, you now have an accurate view of the structure of the directory tree.
Using CDMS Migration Tools

Since commas separate directory structure output variables, you can take the script results and import them into a spreadsheet. The first line can be used as a column head in Microsoft’s Excel or similar spreadsheet applications. A few simple sorts of columns on the spreadsheet let you answer such questions as:

- What is the lowest level of subdirectory?
  
  **ACTION:** Sort by level, descending, and it appears at the top

- Which directory has the most subdirectories?
  
  **ACTION:** Sort by subdirectory, descending, and it appears at the top

- Which directory has the most files?
  
  **ACTION:** Sort by names, descending, and it appears at the top

From the output (sorted by level, descending), you can observe what the tree structure looks like for possible partial migrations, data splitting, and so on. The directory tree-walking algorithm needs to go to the end of each branch before it can begin reporting, so the results come out in the reverse order from the way you would intuitively expect.

For example, the 0 level is reported last. A sort by the directory name often produces a more intuitive-feeling picture of the directory.

**Note:** The count of names (the third column of the spreadsheet), as well as including “.” and “..”, also includes the names of the subdirectories. For example, if the second column of a spreadsheet contains four and the third column contains six, then that directory only contains subdirectories and no other types of files.
### Known issues

There are known issues associated with the following:

- Local UNIX directories
- Special UNIX directory empty
- Perl script
- Access rights
- Long name
- Network Appliance file named .snapshot
- VNX file named .ckpt

#### Local UNIX directories

If you use the dircount.pl script on a local UNIX file system that has remote NFS directories mounted on it, the contents of those remote file systems are included in the directory and filename counts. NFS mounts prevent a user application from being able to tell if the data is local or remote. To get a true estimate of the data structure physically on the local file system, you have to unmount those directories and possibly remove the mount point.

#### Special UNIX directory empty

In some UNIX implementations, if your target file system is local and contains a directory to be used as a mount point, but that does not have anything currently mounted, the dircount.pl script fails. This issue seems to occur on directories on which removable media are to be mounted, for example, /mnt/cdrom. It is possible they are being treated as special cases in the operating system implementation.

#### Perl script

The dircount.pl script works under any Perl script version 5.6 or later, but might not work under earlier revisions. The initial development and testing were done with an implementation of Perl version 5.6 (Active Perl for Microsoft Windows), but there are no version 5.6-specific features used.

#### Access rights

Remember to run the dircount.pl script as a user who has access and permissions to read all directories and subdirectories that you want to survey. If you are logged in as someone with limited access, one of the following occurs:

- Some files might not get counted in the total (with possibly no error to show you the total is wrong).
- The script fails with an unable to open error on a particular directory or file.
If the amount of data or the list of directories appears much smaller than you expected, this might be due to access problems. Some operating systems, depending upon access conditions, allow open access rights, but return no data on read rather than an error. You might need to manually check for access to the file system. Most operating systems show the directory as open, causing an unable to open message to appear.

**Long name**

When the dircount.pl script is parsing through the directories, if the complete filename and pathname becomes a string longer than either the operating system or Perl script can support (whichever comes first—usually around 1000 or 2000 characters long), the script fails.

**Network Appliance file named .snapshot**

If you are measuring directories on a Network Appliance system, the .snapshot history files are in the same volume as the data. Conversely, the equivalents on the VNX and Auspex file servers are in separate volumes.

The dircount.pl script avoids the .snapshot directory area in Network Appliance to prevent contamination of the current file structure with historical structure counts. Therefore, if you are not on a NetApp filer and have real directories whose name includes the string .snapshot, they are skipped and not counted. If you must have them counted, edit the Perl script to remove the test.

**VNX file named .ckpt**

Since this Windows directory is not visible, the dircount.pl script simply skips the .ckpt directory.
dirprivU.pl script

The dirprivU.pl script is used prior to data migration to check that all files can be read and therefore migrated successfully. If unreadable files are located, the size of the file system reported by the diskUsage.pl script is incorrect. This script uses a directory or pathname to a directory as an input parameter. It can be a subdirectory or a mounted remote (NFS) directory. This script produces results for either NFS-mounted or local UNIX files.

---

Tool location
UNIX workstations

Syntax and parameters
This script has the following syntax and parameters:

```
$ perl dirprivU.pl <directory> [-v]
```

The dirprivU.pl script takes two parameters; the first one is mandatory and the second is optional:

- The first parameter, `<directory>`, is the name of a directory (or full path to a directory) that you are going to migrate in the near future. This directory can be an NFS-mounted remote directory or a subdirectory (local or remote). However, because of the way root privileges are handled over NFS protocol, a remote-mounted directory is preferred.

- The optional second parameter, `-v`, following the directory name, suppresses checking for issues that affect migrations by using NFS version 2. In other words, with `-v`, you are presuming an NFS version 3 migration. Without this parameter, the file system is checked for NFS version 2 and NFS version 3 issues. Since NFS version 2 messages are produced if the file handle is greater than 32 bytes or the file length exceeds the unsigned 32-bit integer size, there might be many messages generated on the NFS version 3 system. Therefore, it is recommended that you always use the `-v` parameter unless you are migrating by using NFS version 2.

- The log file is automatically created in the current directory, and it retains all command outputs.
**Functionality**

When the dirprivU.pl script is executed, it progresses downward from the starter directory. When it comes to a file, it checks permission bits, and attempts to open and read 1 byte from the file. All symbolic link files are skipped because they only have an inode, which would be automatically migrated as part of the directory structure. When the script comes to a subdirectory, it checks the permissions, and then recursively opens that subdirectory, progressing downward through the list of files in the subdirectory until it comes to a subdirectory, and so forth. The dirprivU.pl script identifies file system areas that might require attention before you perform a migration. There might be situations that might be troublesome in one set of circumstances, but are not situations guaranteed to fail.

*Appendix C, “NFS Mount and Export Options,” provides more information.* Locating and correcting trouble spots before you migrate data rather than during a migration is preferred. This script does not identify every cause of a migration failure, but does locate the known issues you can correct.

**Subdirectory open successfully**

For each subdirectory that is opened successfully, an x (execute) appears on the window to indicate that the script is running. A new line appears at every 50th x.

**Client cannot error messages**

When the dirprivU.pl script comes to a file or directory it cannot open or read with the current username/password, it displays a client cannot error message together with the operation (open directories, open file, read directories, read file), error, full pathname, and privilege and type bits. This means either the:

- File or directory is corrupt or otherwise unreadable.

  or

- Username does not have the privileges, either locally or granted by the remote export, to open or read this directory or file.

If a file or directory cannot be read at migration time, it cannot be migrated. Therefore, with the exception of locked files, all client cannot messages should be investigated to understand if it is really an unreadable file, or simply an access privilege issue.

**Note:** If there are unreadable files identified by the dirprivU.pl script, several other measurement scripts such as the diskUsage.pl script are also inaccurate by the amount represented by the unreadable files.
For each directory or file containing data (not pipes or link files), the dirprivU.pl script also examines the owner’s privileges. On a CDMS migration, the connection can use either the owner’s UID/GID (default), or optionally read as UID/GID of 1/0; that is root/other. The owner cannot message, similar to the client cannot message, appears if the owner is missing an essential privilege, and the owner is not root. In other words, if the UID is not equal to 0. Therefore, these situations would cause problems if you use a default migration connection. Depending on the number of privilege bit changes, you can work around the issues in the following ways:

- If there are only a few changes, update the privilege bits on the file.
- If there are many changes, use the useRootCred=true option on the server_cdms command of the MGFS. This action causes migration reads from the VNX to come from UID, GID of 0,1 instead of the file owner’s UID and GID. If this method is selected, however, the source file server must also export the file system so the Data Mover IP address is granted root access.

For each directory or file that cannot be used by NFS version 2, for example, the name is too long or the file is too large, a warning message appears. You can suppress these messages by using the -v parameter on a command line.

The dirprivU.pl script checks for the mode bit setting on a file that allows the hard locking (mandatory locking) to be applied to that file at runtime. A file with this lock applied might not be able to be read by other users or process ID (PID). This condition might cause a command to fail, such as the server_cdms command, if another process had the file locked when the command tried to read it. However, the migration should not fail because the application that locked it would presumably read and therefore migrate the file.

**Note:** Any file locking that is advisory can be migrated by using CDMS.

There is a warning message for hard locked files. You can avoid problems with these files if you ensure that certain procedures are followed at migration time.

If any hardlock files are actually locked at the time you run the dirprivU.pl script, you might also see a client cannot open or client cannot read message for that file. For most UNIX implementations, only the PID and host that places the lock can read or write to the file (or locked part) until it is unlocked.
For migrating files with hard locks, perform the following:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ensure that there are no outstanding locks on the file prior to the cutover of the source file server to the VNX. In other words, ensure that just before the cutover, the applications that place the locks have none outstanding, are in a quiescent state, and so on.</td>
</tr>
<tr>
<td>2</td>
<td>Cut over to the VNX. The systems containing the applications that perform the locks can then remount the file system, and run the applications.</td>
</tr>
<tr>
<td>3</td>
<td>If the applications perform new locks on the files, these locks are on the VNX instance of the file, not on the original source. Therefore, the VNX can still read the original source. If the application performs I/O processes to the file, the VNX can satisfy it even if it requires a read from the original source. If locked, no other application (PID, host) can read the file on the VNX.</td>
</tr>
<tr>
<td>4</td>
<td>The <code>server_cdms</code> command fails when it attempts to read locked files.</td>
</tr>
</tbody>
</table>

**Note:** The file is still migrated even if the `server_cdms` command cannot read it because it is locked while some other PID is performing I/O processes to the file. You might need to run the command several times until it passes as most hard locks are only temporary. You might need to run the `server_cdms` command on other directories, find a time when you can ensure that the lock candidate file is not locked, and then run the command over the directory. In any case, either the `server_cdms -verify` or `-Convert` command indicates if a file is not read completely. Consider all possibilities as there might be many alternatives.

**SetGID warning**

For each directory or file that has both the SetGID on execute and the execute privilege on the group bits, a warning message appears. The CDMS migration does not change the GID upon migration. Instead, all files created through the migration connection retain their present GIDs, even if they are migrated into a directory on which the SetGID bit is applied. This functionality exists to retain old GID access on files created prior to the setting of the SetGID bit. This modified UNIX rule applies only to files created on the VNX through the migration connection. The dirprivU.pl script displays a warning so you are aware of this anomaly in the rules. New files created on the VNX through the front rather than the back connection (for example, by performing a tar or cpio to the SetGID directory on the migration Data Mover) have their GIDs changed in accordance with standard UNIX rules.

**Illegal mode bits**

For each directory or file containing an illegal combination of mode bits, the dirprivU.pl script issues a warning message indicating the file or directory might be corrupt.
Files named "." Each directory always contains a self-referential subdirectory named dot ("."). However, on some UNIX workstations, it is possible to create a file named ".". The dirprivU.pl script issues a warning if any such files are discovered. You should rename or remove the files before starting migration.

Running the script This section describes the execution, usage, and warning notes for the dirprivU.pl script:

1. From the UNIX workstation where you intend to run the server_cdms command in a real migration, log in as the user who runs the script.

2. Execute the dirprivU.pl script from the shell by using a directory path as a parameter.

   For example, if your Perl interpreter was in the /etc/bin/perl file, use the following command:

   $ /etc/bin/perl/perl dirprivU.pl /<sapfiles> -v

   If the Perl script was installed as a package under UNIX, you can invoke it as a command from any directory by using the following command:

   $ perl dirprivU.pl /<sapfiles> -v

Sample output

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Owner 789 in Group 30
Cannot read dir /sapi/bin/build
ModeBits = 40000
Owner 789 in Group 30
Has no X privilege dir /sapi/bin/build
ModeBits = 40000
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Hard Lock candidate /sapi/gnu/solaris_2.6/1998_12/gnu/bin/lsof
ModBits = 102555
Hard Lock candidate /sap/gnu/solaris_2.6/1998_12/gnu/bin/lslk
ModeBits = 102555
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Hard Lock candidate /sapi/gnu/solaris_2.6/1998_12/gnu/packages/top-3.5beta8/bin/top
ModeBits = 102711
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Hard Lock candidate /sapi/gnu/hpux_10.20/1997_12/gnu/bin/lsof
ModeBits = 102555
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Owner 1179 in Group 30
Cannot read dir /sapi/home/fhnzlr
ModeBits = 40000
Owner 1179 in Group 30
Has no X privilege dir /sapi/home/fhnzlr
ModeBits = 40000
xxxxxx
Owner 407 in Group 30
Cannot read dir /sapi/home/jlynch
ModeBits = 40000

If the dirprivU.pl script runs successfully, it parses the whole subdirectory tree, exhausts all the names it can find, does a quick final calculation, and displays the following message:

There are a total of 159909 names in a total of 5295 directories
Log file: /home/work/new/dirprivU_sap.log

Known issues

There are known issues associated with the following:

- Perl script
- Access rights
- File systems containing mounts
- Name length
- Network Appliance file named .snapshot
- International multibyte character sets

Perl script

The dirprivU.pl script works under Perl script version 5.6 or later, but might not work under previous revisions.

Access rights

You should run the dirprivU.pl script as a user who has access and permissions to read all directories and subdirectories that you want to migrate. If you are logged in as someone with limited directory access, some files might not be seen, and the script might not return an error message showing the result is wrong. Therefore, if you use the script, you might need to get a root-level password. If the count at the end looks much smaller than you expected, this might be due to access problems. Some UNIX
implementations, depending on access conditions, allow open access rights, but return no data on read rather than an error for unreadable files. You might need to do a manual check for access to the file system.

**File systems containing mounts**

If you are using the dirprivU.pl script on a local UNIX file system with remote NFS directories mounted on it, the contents of those remote file systems are included in the files that the script examines. A user program should not be able to tell if the data is local or remote. However, you should not migrate any file system containing mounts to a VNX because once on a target server, it gets exported so others can use it. UNIX has rules against exporting items you mount.

**Name length**

When the dirprivU.pl script is parsing its way down the directories, the script fails if the complete name (filename and pathname) becomes a string longer than either the operating system or Perl script can support (whichever comes first, usually around 1,000 or 2,000 characters).

**Network Appliance .snapshot file**

If you are measuring directories on a Network Appliance system, the .snapshot directory history files are in the same volume as the data. The equivalents on VNX and Auspex file servers are in separate volumes.

The dirprivU.pl script avoids the .snapshot directory area in the Network Appliance to prevent contaminating the current file structure with historical structure counts. Therefore, if you have real directories, whose name includes the string .snapshot, they are skipped and not counted. If you must count them, edit the Perl script to remove the test.
If the dirprivU.pl script fails because the bytes pragma cannot be located, you are running a version of the Perl script that does not support the Unicode international character set. If you cannot upgrade to an appropriate version of Perl (most implementations of Perl script version 5.6 supported bytes), you can edit the script to comment out the two instances (specification and invocation) of the use of bytes.
**dirprivW.pl script**

The dirprivW.pl script is used prior to a data migration to check that all files can be read and therefore migrated successfully. If any unreadable files are located, the size of the Windows file system reported by the diskUsage.pl script is incorrect. This script uses a directory or pathname to a directory as an input parameter.

**Tool location**

Windows 2000, Windows Server 2003, and Windows NT 4.0 clients

**Syntax and parameters**

This script has the following syntax and parameters:

```
C:\> backupWrapper dirprivW.pl <directory>
```

where:

- `<directory>` = a directory name (or full path to a directory) that you are going to migrate in the near future. It can be a subdirectory or mapping of a remote (CIFS) share. If there is no directory parameter, the script fails.

The log file is automatically created in the current directory, and it retains all command outputs.

**Note:** The -v option is only used for NFS migrations.

When the directory is opened, the name in the first parameter is opened as a directory. Do not end the parameter name with a backslash (\).

**Functionality**

When the dirprivW.pl script is executed, it progresses downward from the first directory in the command. When it comes to a file, it checks attribute bits, and attempts to open and read 1 byte from that file. All symbolic link files are skipped because they only have a directory entry, which would be automatically migrated as part of the directory structure. When the script comes to a subdirectory, it checks the permissions, and then recursively opens that subdirectory, progressing through the list of files in that subdirectory until it comes to another subdirectory, and so on. The dirprivW.pl script can identify file system areas that might need some attention before you perform a migration. There might be situations that are troublesome in one set of circumstances, but are not situations guaranteed to fail.
Locating and correcting trouble spots before you migrate data, rather than during a migration, is the preferred methodology. The dirprivW.pl script does not identify every cause of a migration hang up, but does catch known issues that can be corrected.

For each subdirectory opened successfully, an x (execute) appears on the window to indicate the dirprivW.pl script is running. A new line is displayed at every 50th x.

When the dirprivW.pl script comes to a file or directory it cannot open or read with the current username/password and privileges, it displays a “client <username> on node <system name> cannot” error message together with the operation (such as open directories, open files, read directories, read files), the full pathname, and the error message.

This message means either:

- The file or directory is corrupt or otherwise unreadable.

  or

- The username you are using does not have privileges, either from an ACL or granted by the remote share, to open or read this directory or file.

If a file or directory cannot be read at migration time, it cannot be migrated. Therefore, all client cannot messages, with the exception of encrypted directories, should be investigated to determine if it is really an unreadable file or merely an access privilege issue. Whatever the reason, if there are unreadable files discovered by the dirprivW.pl script, several other measurement scripts (for example, the diskUsage.pl script) are also inaccurate by the amount represented by the unreadable files.

For each directory or file encrypted using Windows 2000 or Windows Server 2003 system encryption, a warning message appears. However, files encrypted by third-party encryption software are not identified as encrypted but migrate successfully. Windows-encrypted files cannot be migrated to the VNX; they must be decrypted prior to migration.

Because the detection of a file or directory's encryption status is performed through the attribute bits on the file header, some older Windows NT systems, which did not allow users to encrypt files, have that bit set on files that are not encrypted. The encrypted warning message is only valid if it is on a Windows 2000 or Windows Server 2003 file system.
This section describes the execution, usage, and warning notes for the dirprivW.pl script:

1. From the Windows client where you intend to run the server_cdms command in a real migration, log in as the user who runs the command.

2. Execute the dirprivW.pl script from the command line window by using a directory path as a parameter.

   The script should be in the same directory as the backupWrapper.exe utility, and the Perl script should be installed as a package on the system by using the following command:
   
   C:\> backupWrapper dirprivW.pl <H:\bgdata>

   where:

   <H:\bgdata> = a directory name (or full path to a directory) whose structure you want to examine

Sample output

   C:\> backupWrapper dirprivW.pl C:\perl

   xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Warning: encrypted file C:\perl/site/lib/Tests/td2/tfd2f1

Warning: encrypted file C:\perl/site/lib/Tests/td3/tfd3f1

Warning: encrypted file C:\perl/site/lib/Tests/td3/tfd3f3

Successful execution

If the dirprivW.pl script runs successfully, it parses the entire subdirectory tree, exhausts all names it can locate, performs a final calculation, and displays the following message:

   There are a total of 4243 names in a total of 375 directories

   Log file: C:/Perl/site/lib/Win32/API/new\dirprivW_c_Perl.log
Known issues

There are known issues associated with the following:

- Perl script
- Access rights
- Name length
- Network Appliance file named .snapshot
- International multibyte character sets

Perl script

The dirprivW.pl script works under Perl script version 5.6 or later, but might not work under previous versions.

Access rights

Running the dirprivW.pl script with the backupWrapper.exe utility requires that you have backup and restore files user rights. The user account must also be a member of administrators or account operators on the source file server. You must execute this script as a user who has access and permissions to read all the directories and subdirectories that you want to survey. If you are logged in as someone with limited access, either:

- Some files might not get counted in the total (with possibly no error to indicate the total is wrong).
  
  or

- The script fails with an unable to open error on a particular directory or file.

If the amount of data or list of the directories scrolled on the window appears much smaller than you expected, this condition might be due to access problems.

Name length

When the dirprivW.pl script is parsing its way down the directories, if the complete name (filename and pathname) becomes a string longer than either the operating system or the Perl script can support (whichever comes first, usually around 1,000 or 2,000 characters long), the script fails.
Network Appliance file named .snapshot

If you are measuring directories on a Network Appliance system, the .snapshot directory history files are in the same volume as the data. Conversely, the equivalents on VNXr and Auspex file servers are in separate volumes. The dirprivW.pl script bypasses the .snapshot directory area in the Network Appliance system to avoid contaminating the current file structure with historical structure counts. Therefore, if you are not on a Network Appliance filer and have real directories whose names include the .snapshot string, they are skipped and not counted. If you must have them counted, edit the Perl script to remove the test.

International multibyte character sets

If the dirprivW.pl script fails because the bytes pragma cannot be located, you are running a version of the Perl script that does not support the Unicode international character set.

If you cannot upgrade to an appropriate version of the Perl script (most implementations of version 5.6 supported bytes), you can edit the script to comment out the two instances (specification and invocation) of the use of bytes.
diskUsage.pl script

The diskUsage.pl script allows you to estimate the minimum amount of storage space you need on the target VNX when you migrate a specified directory or file. It provides a list of the amount of data in various file sizes that can be used in estimating migration times. Appendix B, “Estimating NFS Data Migration Times,” provides more information.

---

**Tool location**

UNIX workstations, and Windows 2000, Windows Server 2003, and Windows NT 4.0 clients

**Syntax and parameters**

The diskUsage.pl script can be run with parameters entered manually or with all parameters on a command line. This process is referred to as running manually or running automatically. EMC recommends you run the diskUsage.pl script manually until you become familiar with its usage.

---

**Running the script manually**

The parameters you can type to run the script automatically are explained in Table 17 on page 280. For manual running, use the following command:

**UNIX:**

$ perl diskUsage.pl -m

**Windows:**

C:> backupWrapper diskUsage.pl -m

where:

- `-m` = specifies the script runs manually

In manual mode, the script prompts you for input. In an example session, the user responses are shown in boldface in “Example (Windows)” on page 280.
The diskUsage.pl script prompts you to type 1, 2, 3, or 4:

- Number 1 specifies you are going to estimate the disk usage for an NFS directory.

- Number 2 specifies you are going to estimate the disk usage for a directory that is shared by using the CIFS (Windows file access) protocol from the VNX. CIFS-shared directories contain more information than NFS-only directories, and if you know this file system is NFS-exported and CIFS-shared after migration, you should use this option to produce a more accurate estimate of the disk usage.

- Number 3 specifies you are going to estimate the disk usage for a specific file. This number could be used to see if the new block size affects an individual file, for example, a large, sparse file.

- Number 4 specifies you want to stop running the script. The example in “Example (UNIX)” on page 278 has entered 1 to estimate the disk usage of an NFS directory.

If you had selected number 1, 2, or 3, the diskUsage.pl script prompts you to type the directory pathname, ./temp2 in “Example (UNIX)” on page 278.

If you had selected number 3, the diskUsage.pl script additionally prompts you to type the name of a file in that directory.

The diskUsage.pl script needs the size of one block in your new file system. For VNX systems, the block size is 8 KB, and is the default value entered when you press Enter.

You should indicate whether you want to remove the hardlink sizes of files before calculating disk usage. A hardlink inode contains file size information, but does not occupy disk space with a corresponding amount of data; a symbolic (or soft) link does not contain file size information. Several UNIX data copy and migration utilities do copy the data associated with a hard-linked file on each occurrence of a hardlink. CDMS re-creates the original file system structure where the data is copied once even though there are multiple, hardlinked inodes. Because the diskUsage.pl script can be used for sizing VNX usage for all migration methods, it offers you a choice on what to do with hardlink sizes.

The default is to include hardlink sizes in the disk usage calculation.

To determine destination size correctly for CDMS, hardlink sizes must not be counted (that is, the data size is counted only once). To omit counting the size of each occurrence of a hardlink, type Y on the menu, or -h on the command line.
Using CDMS Migration Tools

**Entering the pathname for the log file**

The log file is automatically created in the current directory by default, and you can change the path. The file retains all command outputs.

**Running the script automatically**

You can also include all the parameters required by the diskUsage.pl script in a single command line, in which case the script runs without any further interaction required on your part.

Syntaxes for the command line are:

**UNIX:**

```bash
$ ./diskUsage.pl -d <directory> -d <dir> -h
$ ./diskUsage.pl -d <directory> -f <file>
$ ./diskUsage.pl -d <directory> -n <new_size>
$ ./diskUsage.pl -d <directory> -n <new_size> -h
$ ./diskUsage.pl -d <directory> -f <file> -n <new_size>
```

**Windows:**

```bash
C:\> backupWrapper diskUsage.pl -d <dir> -h
C:\> backupWrapper diskUsage.pl -d <dir> -f <file>
C:\> backupWrapper diskUsage.pl -d <dir> -n <new_size>
C:\> backupWrapper diskUsage.pl -d <dir> -n <new_size> -h
C:\> backupWrapper diskUsage.pl -d <dir> -f <file> -n <new_size>
```

where:

- `-f <file>` = used only with the `-d` parameter
- `-h` = recommended parameter if the file system is to be migrated with CDMS
- `-n <new_size>` = new block size; default is VNX-standard 8 KB
- `-d <dir>` = directory name
- `-m` = script runs manually
- `<new_size>` = specifies block size, expressed in KB (kilobytes). If used, the new size must be a value greater than 0 (zero). It is not recommended to use this parameter for the current versions of VNX NAS code because the block size is equal to the default value.
Example (UNIX)

The following example shows the diskUsage.pl script checking the size of the files in the ./temp2 directory, displaying a message each time it performs a check. In addition, there might be a message each time a directory block is added. As shown in the following example, the script completes by displaying a Disk Usage Info table. Table 17 on page 280 provides table entry explanations.

1) Disk usage of a NFS directory.
2) Disk usage of a CIFS directory.
3) Disk usage of a file.
4) Exit.

Please choose from the above menu: 1
Please type directory name: ./temp2
Please type the NEW block size in KB [8]: [Enter]
Do you want to remove hardlink size?[y/n](y): y
Please type the path for log file [/home/work/new]:
checking ----------> temp2/two1
checking ----------> temp2/two2
checking ----------> temp2/ltwo2
checking ----------> temp2/hltwo1
checking ----------> temp2/..hidden
checking ----------> temp2/..dir1
checking ----------> temp2/..dir1/..hidden
checking ----------> temp2/sl_dir
checking ----------> temp2/1
checking ----------> temp2/1/a
checking ----------> temp2/2
checking ----------> temp2/2/b
checking ----------> temp2/a
checking ----------> temp2/b
checking ----------> temp2/c
The output of the `diskUsage.pl` contains an estimate of the migration log size (Estimated MigLog Size section). Consider this log size if you plan to put the logs in the same MGFS file system. Otherwise, arrange enough space on another file system for these logs.

The data above that appears on the screen is saved in a log file with the naming convention `<diskusage-path>.log`. 
Table 17 lists fields and associated descriptions for the Disk Usage Info table.

### Table 17 Disk Usage Info table entry definitions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directory/file:</td>
<td>The pathname of the directory or file to be migrated.</td>
</tr>
<tr>
<td>New block size(byte):</td>
<td>The size, in bytes, of one block in the new (target) file system.</td>
</tr>
<tr>
<td>Real total data size(byte):</td>
<td>The size, in bytes, of the data written within directory or file data to be migrated. This number does not include any metadata associated with the directory or file, or any allowance for the blocks occupied by the data.</td>
</tr>
<tr>
<td>Data size(byte):</td>
<td>The amount of storage, in bytes, required by all blocks in the new (target) file system to hold the directory or file data (not metadata) being migrated.</td>
</tr>
<tr>
<td>Symbolic link size(byte):</td>
<td>The amount of storage, in bytes, required by blocks in the new (target) file system to hold the symbolic link size metadata for the directory or file being migrated.</td>
</tr>
<tr>
<td>Directory size(byte):</td>
<td>The amount of storage, in bytes, required by the blocks in the new (target) file system to hold the directory-size metadata for the directory or file being migrated.</td>
</tr>
<tr>
<td>Total file system size(byte):</td>
<td>The total amount of storage, in bytes, required by blocks in the new (target) file system to hold all migrated file data and metadata. Use this number to size the volume on a VNX.</td>
</tr>
<tr>
<td>Total filesize in (x, y)(byte):</td>
<td>The total amount of storage, in bytes, required by the new (target) file system to hold the migrated files of the specified sizes. This shows you the distribution of file sizes to be migrated. The square bracket at the end of the upper range means the number that precedes the bracket is included in the range. Use this distribution of sizes to estimate migration times. Appendix B, “Estimating NFS Data Migration Times,” provides more information.</td>
</tr>
</tbody>
</table>

**Example (Windows)**

The following example shows the diskUsage.pl script checking the size of the files in the C:\temp directory and displaying a message each time it performs a check. In addition, there might be a message each time a directory block is added.
As shown in this example, it finishes by displaying a Disk Usage Info table. Table 17 on page 280 provides entry explanations.

C:\>perl diskUsage.pl -m
1) Disk usage of a NFS directory.
2) Disk usage of a CIFS directory.
3) Disk usage of a file.
4) Exit.
Please choose from the above menu: 1
Please type directory name: c:\perl
Please type the NEW block size in KB [8]:
Do you want to remove hardlink size?[y/n](n):

Please type the path for log file [C:/]:

checking all ----------> c:\perl/bin
checking dir 1----------> c:\perl\bin
checking all 2----------> bin/a2p.exe
checking !zero file 3--> bin/a2p.exe
checking all 4----------> bin/c2ph.bat
checking !zero file 5--> bin/c2ph.bat
checking all 6----------> bin/config.pl
checking !zero file 7--> bin/config.pl
checking all 8----------> bin/configPPM.pl
checking !zero file 9--> bin/configPPM.pl
checking all 10--------> bin/crc32
checking !zero file 11-> bin/crc32
checking all 12--------> bin/crc32.bat
checking !zero file 13-> bin/crc32.bat
checking all 14--------> bin/dprofpp.bat
checking !zero file 15-> bin/dprofpp.bat
checking all 16--------> bin/exetype.bat
checking !zero file 17-> bin/exetype.bat
checking all 18--------> bin/find2perl.bat
checking !zero file 19-> bin/find2perl.bat
checking all 20--------> bin/GET
checking !zero file 21-> bin/GET
checking all 22--------> bin/GET.bat
checking !zero file 23-> bin/GET.bat

......
checking dir 1----------> lib/XMLRPC
checking all 2----------> XMLRPC/Lite.pm
checking !zero file 3--> XMLRPC/Lite.pm
checking all 4----------> XMLRPC/Test.pm
checking !zero file 5--> XMLRPC/Test.pm
checking all 6----------> XMLRPC/Transport
checking dir 7----------> XMLRPC/Transport
checking all 8----------> Transport/HTTP.pm
checking !zero file 9--> Transport/HTTP.pm
checking all 10--------> Transport/POP3.pm
checking !zero file 11-> Transport/POP3.pm
checking all 12--------> Transport/TCP.pm
checking !zero file ------> Transport/TCP.pm
small dir count = 360
large dir count = 1

====================================================================
*         ************ Disk Usage Info ************* *
* Directory/file: c:\perl/
* New block size(byte): 8192
* Real total data size(byte): 37153056
*
* Considering Meta Data and Block Issue:
* Data size(byte): 53510144
* Symbolic link size(byte): 0
* Directory size(byte): 8880128
* Total file system size(byte): 64430080
*
*         ********* File Size Distribution Info ******** *
* Total file size in (0, 1KB] (byte) : 371071
* Total file size in (1KB, 10KB] (byte) : 5135831
* Total file size in (10KB, 100KB] (byte) : 21641688
* Total file size in (100KB, 1MB] (byte) : 10004466
* Total file size in (1MB, 10MB] (byte) : 0
* Total file size larger than 10MB (byte) : 0
*
*         ******** Estimated MigLog Size *********
* MigLog size is calculated if all entries are migrated successfully
* In case of migration failure, the worse case scenario, log size
* could be 2.5 times larger because of the extra failure messages
* logged.
* MigLog size (byte) : 197784
====================================================================
lgdup.exe utility

The lgdup.exe utility duplicates the local groups and the privileges database from a Windows source file server to a designated VNX. Table 18 on page 283 lists utility options.

Tool location
Windows 2000, Windows Server 2003, and Windows NT 4.0 clients

Syntax and parameters
This utility has the following syntax and parameters:

C:\> lgdup [-r][-p][-s][-v][-l<logFile>][-nopriv] <source> <target>

where:
<logFile> = log filename
<source> = reference local group’s NetBIOS name (\name)
<target> = target NetBIOS name

IMPORTANT
Only members of the Administrators or Account Operators local group on the source file server and target server can execute this utility. You must grant the Generate Security Audits and the Manage Auditing and Security Log privileges to duplicate all the privileges successfully.

Table 18 The lgdup.exe utility options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-r</td>
<td>Replace local groups rather than merge them.</td>
</tr>
<tr>
<td>-s</td>
<td>Do not set or add any member of local groups on a resolve error.</td>
</tr>
<tr>
<td>-v</td>
<td>Use verbose mode.</td>
</tr>
<tr>
<td>-l</td>
<td>Redirect stdout to a file (-l+ to append to the file).</td>
</tr>
<tr>
<td>-p</td>
<td>Prefix the local group name on the target server as &lt;source_name&gt;_lg_name.</td>
</tr>
<tr>
<td>-nopriv</td>
<td>Do not duplicate the privilege settings.</td>
</tr>
</tbody>
</table>

Using EMC Utilities for the CIFS Environment provides further information about the lgdup.exe utility that has been installed on the Windows client.
The sharedup.exe utility duplicates share names from the Windows source file server to the VNX. It migrates share points and security descriptors for those share points. This tool requires that you are a member of the administrator group. Table 19 on page 284 lists utility options.

**Tool location**
Windows 2000, Windows Server 2003, and Windows NT 4.0 clients

**Syntax and parameters**
This utility has the following syntax and parameters:


where:

<source> = NetBIOS name of the source file server
<target> = NetBIOS name of the target server
<srcdrive> = drive letter of the CIFS source file server to select for the duplication (for example, C:). Specify ALL to select all disk drives of the source file server.

**Table 19** The shared.exe utility options (page 1 of 2)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/P:&lt;newrootpath&gt;</td>
<td>Specifies the root path prefixed to the directory of the created shares.</td>
</tr>
<tr>
<td>/R</td>
<td>Replaces the target share if it already exists.</td>
</tr>
<tr>
<td>/SD</td>
<td>Duplicates the security descriptors (SD) of the shares. All the local groups of the source file server must exist on the target server. You must use the lgdup.exe utility before using the sharedup.exe utility.</td>
</tr>
<tr>
<td>/PREFIX</td>
<td>Prefixes the shares on the target server with the source file server name in the format &lt;src_server_name_share_name&gt;.</td>
</tr>
<tr>
<td>/FO:&lt;outputFile&gt;</td>
<td>Creates the list of shares to duplicate on the target server with the given options. Do not apply any action on the target server. Used to create a file for input to the connBuilder.pl script.</td>
</tr>
</tbody>
</table>
Table 19 The shareup.exe utility options (page 2 of 2)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/FO+&lt;outputFile&gt;</td>
<td>Same as the /FO option, but append it to an existing file instead of creating a file. Source and target servers must be included in the file. This option allows concatenation of the shares of different drive letters in a single file.</td>
</tr>
<tr>
<td>/FI:&lt;inputFile&gt;</td>
<td>Specifies the file to use as an input list of shares to create on the target server. This file must have the same format as the file created with the /FO option. The srcdrive, /P, and /Prefix options are ignored.</td>
</tr>
<tr>
<td>/LOG:&lt;path&gt;</td>
<td>Sets the log filename to the path. Erases the file.</td>
</tr>
<tr>
<td>/LOG+:&lt;path&gt;</td>
<td>Sets the log filename to the path. Appends to the file.</td>
</tr>
<tr>
<td>/LOG[+]:&lt;logFile&gt;</td>
<td>Redirects messages to a file.</td>
</tr>
</tbody>
</table>

**Note:** The file built with the /FO option is a Unicode text file. You can edit the file with the WordPad editor.

If any error occurred during the execution, the exit code is not equal to 0.

*Using EMC Utilities for the CIFS Environment* provides more information about the sharedup.exe utility that has been installed on the Windows client.
specialCmd utility

The specialCmd utility allows you to bring an offline directory online, or disconnect a current connection. It is stored in the /nas/tools/cdms directory on the Control Station.

“Removing an unwanted connection” on page 232 provides more information.

<table>
<thead>
<tr>
<th>Tool location</th>
<th>Control Station</th>
</tr>
</thead>
</table>

Syntax 1

This utility has the following syntax and parameters:

```
$ ./specialCmd getOnline <movername> <pathname> [escape]
```

where:

- `<movername>` = name of the Data Mover to search for the pathname
- `<pathname>` = path from the mount point, including the mount point name on the specified Data Mover. When the pathname contains a space, use an asterisk (*) to replace the space, and put the pathname in quotes.
- `escape` = optional, used when the pathname contains a space

Syntax 2

This utility has the following syntax and parameters:

```
$ ./specialCmd disconnect <movername> <fsid> [<cid>]
```

where:

- `<movername>` = name of the Data Mover to search for the pathname
- `<fsid>` = file system ID
- `<cid>` = optional connection ID. If it is missing, all connections are disconnected on this file system.
APPENDIX B
Estimating NFS Data Migration Times

The following topics provide information about how to estimate migration time between a UNIX workstation (client) and a VNX:

- Introduction ................................................................. 288
- Assumptions ................................................................. 289
- Initial time estimate ...................................................... 291
- Network traffic ............................................................. 294
- CPU loading effect ....................................................... 296
- Parallel migration ......................................................... 297
- Gigabit Ethernet interface ............................................ 298
Introduction

This appendix discusses estimating migration times between the NFS migration source file server and a Data Mover, and includes:

◆ Assumptions to accurately estimate NFS data migration times
◆ Initial time estimates by using the diskUsage.pl script, including an example of a Disk Usage Info table, how to perform the calculation for unplanned allowance considerations, and information on small file data migration rates
◆ Network traffic impact on data migration rates, including a method to take traffic into account, and information on using a shared network for migration
◆ CPU loading effects of the target Data Mover on migration times
◆ Parallel migration environments
◆ Gigabit Ethernet interface for migrating small and large files
Assumptions

To estimate NFS export data migration times, assume the following:

- An unused 100 Mb/s Ethernet connection between the migration source and destination
- An unused CPU source file server
- A free Data Mover destination
- A separate, unused 100 Mb/s Ethernet path between the Control Station, running the `server_cdms` command, and the Data Mover
- The UNIX workstation CPU is available for running the `diskUsage.pl` script

Assuming these items are in place, the migration transfer rates in user data in millions or bytes per second are shown in Table 20 on page 289.

**Table 20** Migration transfer rates (page 1 of 2)

<table>
<thead>
<tr>
<th>All volume files =</th>
<th>1 KB</th>
<th>5 KB</th>
<th>10 KB</th>
<th>100 KB</th>
<th>1 MB</th>
<th>10 MB</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetApp, DM 510, NAS 4.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Migration rate</td>
<td>0.158</td>
<td>0.63</td>
<td>0.85</td>
<td>3.15</td>
<td>4.14</td>
<td>4.21</td>
<td>MB/s</td>
</tr>
<tr>
<td>DM free CPU</td>
<td>95%</td>
<td>95%</td>
<td>95%</td>
<td>94%</td>
<td>94%</td>
<td>94%</td>
<td></td>
</tr>
<tr>
<td>Ethernet utility</td>
<td>2%</td>
<td>6%</td>
<td>8%</td>
<td>27%</td>
<td>35%</td>
<td>36%</td>
<td></td>
</tr>
<tr>
<td>NetApp, DM 507, NAS 4.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Migration rate</td>
<td>0.171</td>
<td>0.69</td>
<td>0.95</td>
<td>3.17</td>
<td>3.73</td>
<td>3.68</td>
<td>MB/s</td>
</tr>
<tr>
<td>DM free CPU</td>
<td>91%</td>
<td>91%</td>
<td>90%</td>
<td>90%</td>
<td>89%</td>
<td>89%</td>
<td></td>
</tr>
<tr>
<td>Ethernet utility</td>
<td>3%</td>
<td>7%</td>
<td>9%</td>
<td>27%</td>
<td>31%</td>
<td>31%</td>
<td></td>
</tr>
<tr>
<td>NetApp, DM 507, NAS 2.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Migration rate</td>
<td>0.176</td>
<td>0.706</td>
<td>0.96</td>
<td>3.05</td>
<td>3.53</td>
<td>3.55</td>
<td>MB/s</td>
</tr>
<tr>
<td>DM free CPU</td>
<td>89%</td>
<td>90%</td>
<td>90%</td>
<td>91%</td>
<td>91%</td>
<td>92%</td>
<td></td>
</tr>
<tr>
<td>Ethernet utility</td>
<td>2–3%</td>
<td>7%</td>
<td>9%</td>
<td>21%</td>
<td>28%</td>
<td>29%</td>
<td></td>
</tr>
</tbody>
</table>
### Table 20 Migration transfer rates (page 2 of 2)

<table>
<thead>
<tr>
<th>All volume files =</th>
<th>1 KB</th>
<th>5 KB</th>
<th>10 KB</th>
<th>100 KB</th>
<th>1 MB</th>
<th>10 MB</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sun or VNX, DM 507, NAS 2.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Migration rate</td>
<td>0.170</td>
<td>0.718</td>
<td>0.933</td>
<td>2.53</td>
<td>2.64</td>
<td>2.79</td>
<td>MB/s</td>
</tr>
<tr>
<td>DM free CPU</td>
<td>89%</td>
<td>90%</td>
<td>90%</td>
<td>91%</td>
<td>91%</td>
<td>92%</td>
<td></td>
</tr>
<tr>
<td>Ethernet utility</td>
<td>2–3%</td>
<td>7%</td>
<td>9%</td>
<td>22%</td>
<td>23%</td>
<td>25%</td>
<td></td>
</tr>
<tr>
<td>Auspex, DM 507, NAS 2.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Migration rate</td>
<td>0.102</td>
<td>0.271</td>
<td>0.424</td>
<td>1.06</td>
<td>2.83</td>
<td>2.34</td>
<td>MB/s</td>
</tr>
<tr>
<td>DM free CPU</td>
<td>85%</td>
<td>89%</td>
<td>89%</td>
<td>90%</td>
<td>84%</td>
<td>83%</td>
<td></td>
</tr>
<tr>
<td>Ethernet utility</td>
<td>1–3%</td>
<td>2–4%</td>
<td>4–5%</td>
<td>10%</td>
<td>25%</td>
<td>21%</td>
<td></td>
</tr>
</tbody>
</table>
Initial time estimate

Use the diskUsage.pl script to produce a list of the amount of data in various file sizes on the source file server, as shown in the following example. “diskUsage.pl script” on page 275 provides more information.

Example

==============================================================================
*                         *********** Disk Usage Info ******************* *
*                         * Directory/file: temp2/ *
*                         * New block size(byte): 8192 *
*                         * Real total data size(byte): 7399968253 *
*                         * Considering Meta Data and Block Issue: *
*                         * Data size(byte): 8098611230 *
*                         * Symbolic link size(byte): 0 *
*                         * Directory size(byte): 4411303044 *
*                         * Total file system size(byte): 16210083877 *
*                         * File Size Distribution Info *********** *
*                         * Total file size in (0, 1KB] (byte) : 7270198 *
*                         * Total file size in (1KB, 10KB] (byte) : 279996207 *
*                         * Total file size in (10KB, 100KB] (byte) : 1073680846 *
*                         * Total file size in (100KB, 1MB] (byte) : 2025658935 *
*                         * Total file size in (1MB, 10MB] (byte) : 3007390034 *
*                         * Total file size larger than 10MB (byte) : 1005972033 *
==============================================================================
Performing the calculation

From output of the example, you could estimate that on an otherwise free Network Appliance system, the migration would take the total time (in seconds) specified in Table 21 on page 292.

Note: The decimal point is moved to re-express data in millions of bytes.

Table 21  Calculating migration times

<table>
<thead>
<tr>
<th>Calculation</th>
<th>File size</th>
<th>Seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.270198 / 0.176</td>
<td>For under 1 KB files</td>
<td>41.30</td>
</tr>
<tr>
<td>+ 279.996207 / 0.706</td>
<td>For 1 to 10 KB files</td>
<td>396.60</td>
</tr>
<tr>
<td>+ 1073.680846 / 0.96</td>
<td>For 10 to 100 KB files</td>
<td>1118.42</td>
</tr>
<tr>
<td>+ 2025.658935 / 3.05</td>
<td>For 100 KB to 1 MB files</td>
<td>664.15</td>
</tr>
<tr>
<td>+ 3007.390034 / 3.53</td>
<td>For 1 to 10 MB files</td>
<td>851.95</td>
</tr>
<tr>
<td>+ 1005.972033 / 3.55</td>
<td>For over 10 MB files</td>
<td>283.37</td>
</tr>
</tbody>
</table>

Total time = 3355.79 seconds or 55.92 minutes for about 8 GB of user data.

Unplanned allowances

The calculation in Table 21 on page 292, however, does not make any allowances for any of the following:

- CPU loading at source or destination servers, or a UNIX workstation running the server_cdms command
- Other network traffic:
  - On the Ethernet link, between migration source file server and the VNX Data Mover destination
  - On the Ethernet link, between the UNIX workstation running the migration command and the VNX migration destination Data Mover

For estimating the effect of these items, read:

- “Network traffic” on page 294
- “CPU loading effect” on page 296
- “Parallel migration” on page 297
- “Gigabit Ethernet interface” on page 298
The data migration rate for small files can be approximately 20 times slower than the data rate for large files. For example:

- 50 GB of 10 MB-sized files (in image or engineering drawing storage) would take about four hours to migrate.
- 50 GB of 1 KB-sized files (not uncommon in Web servers or hosting) would take over three days to migrate.

**Note:** A large number of files in one directory might have an impact on the time for migration.

Therefore, the overall data volume or file system size is not a good guide to estimate the duration of a migration. Because you have full access to the files during the migration, however, total migration time for a file system should not be an issue.
Network traffic

The network can slow down the migration in the following cases:

◆ When there is traffic on the path between the original data source and the destination Data Mover port. In a well-planned migration, this path is unique and handles no other traffic.

◆ When the network path is between the Control Station running the `server_cdms` command and the Data Mover port it uses. Anything that slows down the read requests that force the migration, slows down the overall migration. In this situation, it is likely that access to that Data Mover port is from multiple clients (for example, all the users of the old data source file server). Even with a well-planned migration, there is likely to be other traffic on that path.

There is some freedom on the `server_cdms` command path because the read requests occur at intervals. One read must be satisfied before the next request is sent. Therefore, the migration pace is governed by the time it takes the UNIX workstation’s (or client) read to be satisfied by the data migration (over the NFS source file server to the VNX path) needed to cover that read operation.

In practice, the migration is not slowed significantly if the traffic on the client-to-VNX path is low, for example, 2 to 3 percent. The impact is greater when the traffic is higher. However, if there is traffic on the client-to-VNX and VNX-to-source file server links, it is advisable to apply the following estimation method twice (and extend the migration time) to attempt to account for the traffic levels on each side of the network.

Method

Assume that you have the following preexisting usage:

◆ A switched Ethernet network
◆ The current network usage is $P\%$
◆ A data transfer that would take time $T$ on a free network

Therefore, on a network with that preexisting usage, the migration transfer takes approximately time $T_1$, estimated by:

$$T_1 = T \times \frac{100}{(100-P)}$$

Example

If the network is already 5 percent loaded and the transfer is estimated to take 11.73 hours on a free network, it might take at least:

$$= 11.73 \text{ hours} \times \frac{100}{95}$$
= 12.5 hours (with some risk of inaccuracy)

If the network is loaded 30 percent, the time taken would be at least:

= 11.73 hours x (100/70)
= 16.45 hours (with a high risk of inaccuracy)

Using a shared network

Estimating for a shared-Ethernet network (using hubs, not switches), however, is more difficult and less accurate. Therefore, migration over a shared network is not recommended because of timing, collisions, higher network error rates, and so on. Nevertheless, if you need to work out timings, a somewhat inaccurate calculation would be if the current network usage is $P\%$, and you expect the Ethernet usage of the migration to add $M\%$. Then, you have a data transfer migration that would take $T$ on a free network.

Therefore, on a network with this preexisting usage, the transfer takes approximately time $T_1$, estimated by:

$$T_1 = T \times \frac{(100 + P + M)}{(100 - P - M)}$$

This calculation implies the data transfer rate is slow if the network load is 30 percent and even slower if the network load nears 50 percent, which is generally true of hub-based, Ethernet networks.

From the previous examples, assuming the migration causes an additional 15 percent load on a 100 Mb/s Ethernet link, on an originally 5 percent loaded network, a transfer that takes 11.73 hours on a free network now consumes:

= 11.73 x (120/80)
= 17.6 hours

However, on a network currently loaded at 30 percent, it might take:

= 11.73 x (145/55)
= 31 hours

even if the migration worked reliably enough at that higher shared network load.
CPU loading effect

The CPU loading effect is more difficult to estimate. A single migration only uses a couple of threads in a Data Mover, which is only about 10 percent of the Data Mover’s CPU capacity. The CPU usage in the original NFS source file server does not map directly into the Data Mover CPU usage. Since the source file server is running a different operating system, not all the activity in the source file server is associated with use of the file system being migrated.

If the original source file server is lightly loaded, the destination Data Mover is also likely to be lightly loaded. If the original source file server CPU is very busy, it is likely the Data Mover is loaded more heavily. There is some effect, but there is no formula. You might have to make a judgment call on the effect. If the overall Data Mover CPU load is kept below 50 percent, it is probably safe to have the server_cdms command and the normal client use of the file system active simultaneously.
Parallel migration

At currently measured Ethernet usage, it would be unsafe (or slow if the switches had large buffers) to run more than three migrations simultaneously through one 100 Mb/s switched Ethernet link.

At the measured CPU usage rates, it would be unsafe to run more than five simultaneous migrations through one VNX 507 Data Mover, no matter which Ethernet ports were used. However, a VNX 510 Data Mover could support up to eight parallel migrations. There should be fewer simultaneous migrations if the client’s access to those file systems causes a heavy load on the Data Mover CPU.

A safe number of simultaneous migrations for multiple Data Movers to one Symmetrix system is unknown.
Gigabit Ethernet interface

Using a 1-Gigabit Ethernet (GbE) interface does not increase the data transfer rates by 10 times over using a slower 100 Mb/s Ethernet interface. The data rates for GbE migrations can be more difficult to predict because rather than network speed, the following factors have the most impact:

- Data Mover CPU speed
- I/O bus interaction speed on the NFS source file server (which can vary widely)

<table>
<thead>
<tr>
<th>Migrating small files</th>
<th>Small files require the most source file server CPU overhead, and the CPU speed does not vary when GbE is added to the configuration. Therefore, for small files, the data transfer rates are about the same as those for 100 Mb/s Ethernet systems.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migrating large files</td>
<td>For large files, the limitation is system I/O bus interaction. For most current server hardware, this factor limits the maximum transfer rate to 18 to 20 Mb/s (65 to 72 GB/hr) over GbE, and is usually less than 15 Mb/s because of other factors such as fragmentation in the file system.</td>
</tr>
<tr>
<td></td>
<td>Running multiple, parallel migrations from a single source file server does not improve this maximum data rate if the source file server’s storage is on one common I/O bus. The data rate is not improved significantly by using jumbo frames on GbE with a corresponding increase in IP maximums, because the Gigabit Ethernet is not the limiting factor on the migration rate.</td>
</tr>
</tbody>
</table>
APPENDIX C
NFS Mount and Export Options

The following topics provide information about NFS mount and export options:

- Introduction ................................................................. 300
- Obtaining access privileges ............................................ 300
- Constructing read requests .............................................. 301
- Using access options ..................................................... 302
- Migrating files with the SetGroupID on execute bit on ........ 307


**Introduction**

Although NFS file access functionality was developed from the UNIX operating system, it does not behave exactly the same as local UNIX access. All UNIX file access is checked against the UID and GID of a locally logged-in user to learn what access privileges are granted that user. That user might be at a remote location and use secure, encrypted, remote login application to connect to the local system, but the login is still handled locally on the system.

**Obtaining access privileges**

With the NFS protocol, the user is truly remote, and login details are not known to the system providing file access. The login protection could be stronger or weaker than on the local system.

---

**Requesting access**

Each NFS file system request contains a UID and GID of the requestor; the request's source is known by using the IP address. If an NFS server exports a file system without any restrictions or special options, the requestor’s UID and GID are compared to those on the file. Depending on the match (owner, group, or no match), the appropriate access privileges are given or withheld.

---

**Enabling local users**

In the NFS protocol, there is one exception to this process. On all UNIX workstations, there is a special set of access rights granted to local users who can claim superuser or root status. The privilege for a user to become root is usually granted to UNIX system administrators, and normally has a further local password protecting it.

Suppose an NFS server exports a file system without any restrictions or special options. Then, if the UID in the remote NFS request claims to be from a root user, NFS only grants them *other* privileges. “The anon option” on page 305 provides more details.

Generally, the NFS protocol cannot trust any user who dials in from a remote system claiming to be root because their password was not checked on this system.
Constructing read requests

To be able to migrate all files in a file system, the VNX performing the migration must be able to read all files, even if the VNX is not their owner and the files do not have appropriate privileges.

To assist in this situation, the MGFS file system always constructs its read requests to the NFS source file server with either of the following:

- The UID and GID of the owner of the file, since the owner normally has the most access

  or

- The UID and GID set to 0,1 (root, other)

This decision depends on the use of the `useRootCred` option on the `server_cdms` command.

“Step 2: Preparing file systems for migration” on page 84 and the *EMC VNX Command Line Interface Reference for File* provide more information about executing the `server_cdms` command.
Using access options

The migration export (and sometimes the mount) command on the NFS source file server has options such as root=, access=, and anon= that can grant or restrict access and privileges. The effects of these option combinations are described in:

- “Default MGFS: default source” on page 302
- “Default MGFS: source permits root access” on page 303
- “MGFS uses root: default source” on page 304
- “MGFS uses root: source permits root” on page 304
- “The anon option” on page 305
- “The access option” on page 306

However, many sites are extremely restrictive about granting root permission to other systems, and some are hesitant about using anon=0 because there are valid security concerns.

Default MGFS: default source

If the MGFS file system is mounted on the Data Mover as:

```
$ server_cdms server_2 -connect mgfs -type nfsv2 -path nfsdata_new -source sourceIP:/nfsdata -option proto=TCP
```

and if the original source file server exports the file system to be migrated as:

```
$ export -option ro /nfsdata
```

The behavior of files being migrated is as follows:

**Note:** TCP is the default value, and does not need to be specified by the user.

- If the owner (matching UID, GID) is not root (UID=nonzero) and has read privileges, the file is migrated. In other words, it is migrated if there is an r(read) in the first rwx of the group of three (rwxrwxrwx); you would see the display against that filename on an `ls -l` command of that file.
If the owner is root (UID=0) and other has read privileges, the file is migrated. In other words, it is migrated if there is an r in the last rwx of the group of three (rwxrwxrwx); you would see the display against that filename on an ls –l command of that file.

If the owner is root (UID=0) and other does not have read privileges, the file is not migrated, even if the requestor really is root.

If the owner is not root (UID=nonzero) and other does not have read privileges, the file is not migrated. In other words, it is not migrated if there is no r in the first rwx of the group of three (rwxrwxrwx); you would see the display against that filename on an ls –l command of that file.

Generally, this NFS command combination is not recommended because many files and directories are installed or constructed by root, so root remains their owner.

If the MGFS file system is mounted on the Data Mover as:

```
$ server_cdms server_2 -connect pmacF30 -type nfsv3 -path /nfsdata_new -source sourceIP:/nfsdata
```

and if the original source file server exports the file system to be migrated as:

```
$ export –option root=migration_dm_IP,ro /nfsdata
```

the behavior of files being migrated is as follows:

- If the owner (matching UID, GID) is not root (UID=nonzero) and has read privileges, the file is migrated. In other words, it is migrated if there is an r in the first rwx of the group of three (rwxrwxrwx); you would see the display against that filename on an ls –l command of that file.

- If the owner is root (UID=0), the file is migrated. The privileges do not matter; it is possible that r is completely absent from this file’s privileges.

- If the owner is not root (UID=nonzero) and does not have read privilege, the file is not migrated. In other words, it is not migrated if there is no r in the first rwx of the group of three (rwxrwxrwx); you would see the display against that filename on an ls –l command of that file.
This NFS command combination with “MGFS uses root: source permits root” on page 304 is recommended option combinations.

Only the last case cannot migrate, and that might be rare or nonexistent. The dirprivU.pl script warns you about any owners who do not have read privileges on their own files. This is rare in well-maintained file systems.

**MGFS uses root: default source**

If the MGFS file system is mounted on the Data Mover as:

```
$ server_cdms server_2 -connect mgfs -type nfsv2 -path /nfsdata_new -source sourceIP:/nfsdata -option proto=TCP useRootCred=true
```

and if the original source file server exports the file system to be migrated as:

```
$ export -option ro /nfsdata
```

the behavior of files being migrated is as follows:

- **File is migrated**
  - If `other` has read privileges, the file is migrated. In other words, it is migrated if there is an `r` in the last `wx` of the group of three (`rwxrwxrwx`); you would see the display against that filename on an `ls -l` command of that file.

- **File is not migrated**
  - If `other` does not have read privileges, the file is not migrated, even if owner is `root`, and does have the privileges on the owner.

This combination is not recommended.

**MGFS uses root: source permits root**

If the MGFS file system is mounted on the Data Mover as:

```
$ server_cdms server_2 -connect mgfs -type nfsv2 -path /nfsdata_new -source sourceIP:/nfsdata -option proto=TCP useRootCred=true
```

and if the original source file server exports the file system to be migrated as:

```
$ export -option root=migration_dm_IP,ro /original
```

the behavior of files being migrated is as follows:

- **File is migrated**
  - Any file is migrated. The privileges do not matter; it is possible that `r` might be completely absent from this file’s privileges.
This NFS command combination is recommended, especially for poorly maintained file systems where ownership privileges might be random.

The anon option

The UID and GID in a remote NFS request might not match those on the file to which that request applies. Although most users use other to describe the last set of listed privileges, the design of UNIX is different.

A nonmatching remote user is actually assigned an effective local UID. The default on most systems is UID_NOBODY, so other actually gets nobody privileges.

Note: Some UNIX implementations called this privilege anybody.

If a system is not granted the right to have its root claims trusted, then NFS requests from remote root users are also treated locally as nobody. The anon= option on the NFS export overrides this default assignment.

Using anon=0

The anon option most useful for migration is anon=0. It means that a user who is not recognized receives the root UID of 0, and all associated privileges for NFS access. You do not need the root= export option to get root-level access from any MGFS on any Data Mover. Therefore:

- If the NFS source file server is exported with anon=0, a default MGFS mount gets full access to every file, except those files where the nonroot owner UID matches; yet, the owner has no read privileges. This latter case should be rare in a well-maintained file system. In addition, the dirprivU.pl script warns you about any owners who do not have read privileges on their files.

- If the NFS source file server is exported with anon=0 and the MGFS mount uses the useRootCred option, the migration gets full access to every file with remote root = unknown = effective local root ID in each request.

However, after you export with the anon=0 option, anyone who has remote access to that file system has access to every file. It is a security risk. Many customers do not allow this situation, and have policies preventing its use. Although it is useful, it might not be permitted. If this is an issue in which you feel the anon option is needed, consider the following:

- If the NFS source file server is on a physically separate network, connected only to the VNX for the migration, there is no risk from other users access.

- It might be possible to restrict access to only the VNX and some trusted administrative systems with the access= option, and so minimize risk.
Using `anon=<nr.>`

A nonmatching user is assigned an effective local UID, as specified in the option, where `<nr.>` equals a non-zero integer. Some customers use this option to assign unknown, remote users to a guest account and privileges. Often, you see `anon=-1`, which means that an effective local UID is not assigned, so all remote NFS access from an unknown (nonmatching) UID is denied.

The access option

This export option restricts the ability to NFS mount and access a file system to only the systems specified in the hostnames list and netgroups following the `access=` option:

- The access option is applied before all other options so it excludes all other systems from any form of access.
- If you only have one system named in an access list, all other systems cannot access that file system.
- If you observe an export option that looks similar to `-o access=SERVERX, root=SERVERY`, it means that `SERVERY` cannot have access, `root` or otherwise, to the file system.
- If the NFS source file server has any form of access option, you must include the Data Mover in that access list, as well as adding the VNX to `root=` or whatever other options are required.

Sun systems do not have an access option in their export commands. Instead, they permit the use of access lists on the `ro` and `rw` options. The enforcement of such an access list is as strict as if it were on an access option.

For example, a Sun export command such as:

```
$ share -F nfs -o rw=hosta,root=celerraDM /mountpath
```

would actually mean that the Data Mover could not mount or use the file system on `/mountpath`, and severe consequences would result for any attempt at a CDMS migration. The Sun man pages for `share_nfs(1M)` provide the details.
Migrating files with the SetGroupID on execute bit on

The SetGroupID (SGID) mode bit is on for a directory with files that are created with UNIX rules for propagation of the group ID. With this option, files and subdirectories created in that directory inherit the group ID of the directory rather than the GID of the current process. However, the directory being migrated might have had that bit set after files had been created in the directory. These files could bear GID numbers different from those of the directory. However, external clients expect that their original access checking and rights remain unchanged by the migration, even though these are inconsistent with the current status, and the files are being re-created by CDMS.

Therefore, CDMS suspends the UNIX SetGroupID On Execute rule during migration to enable those headers to be migrated unchanged. The original owners see what they expect. The rule remains valid for files and subdirectories created by other UNIX clients using that directory of the MGFS. The rule is only suspended for the internal CDMS migration clients, and only while the migration is in progress (the file system is in the MGFS state). Any files created by regular external clients (normal application or user usage of the file system) create files whose GID is that of the directory, as normal NFS operation requires.
NFS Mount and Export Options
APPENDIX D
Logging

The following topics provide information about how to set up logging and retrieve log files. A sample session on how to set up logging is also provided as a guide:

◆ Introduction ...................................................................................... 310
◆ Setting up ........................................................................................... 310
◆ Sample session .................................................................................. 311
◆ Retrieving log files ........................................................................... 314
Introduction

CDMS can log all inode operations, such as file creation, file removal, and so on. Logging does cause additional overhead, however, and slows the migration effort.

If you want to create an MGFS log file, you must set up the VNX for logging, and specify the location of the log files.

A log file can greatly assist EMC Customer Support Representatives in debugging any CDMS problems.

**Note:** The file system type for the MGFS log must be UxFs.

Setting up

On the Control Station, set up logging by performing the following:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Create a separate UxFs for each log file (one log file for each Data Mover involved in the migration).</td>
</tr>
<tr>
<td>2</td>
<td>Mount the log file system on a Data Mover (or CIFS server). The log must be created on the same Data Mover as the MGFS file system is mounted.</td>
</tr>
<tr>
<td>3</td>
<td>Add a line to the <code>param</code> file specifying where the Data Mover stores the log, and specifying the prefix of the log filename.</td>
</tr>
<tr>
<td>4</td>
<td>Restart the Data Mover.</td>
</tr>
<tr>
<td>5</td>
<td>Perform the migration, starting with step 1 in <em>Chapter 4, “Planning and Design.”</em> When migration begins, logging also starts. Log content can be used for problem analysis.</td>
</tr>
</tbody>
</table>
Sample session

The following is an example of commands you might use to perform the previous steps to set up logging:

1. Create a separate UxFS file system to hold the log files.

   This example session uses a migration size of 11 GB. However, an actual migration would likely involve a much larger size. It is assumed that disk 3 (d3) has space available.

<table>
<thead>
<tr>
<th>Example 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ nas_slice -name disk3 -create d3 11000 1</td>
</tr>
<tr>
<td>Output</td>
</tr>
<tr>
<td>id = 95</td>
</tr>
<tr>
<td>name = disk3</td>
</tr>
<tr>
<td>acl = 0</td>
</tr>
<tr>
<td>in_use = False</td>
</tr>
<tr>
<td>slice_of = d3</td>
</tr>
<tr>
<td>offset(MB) = 1</td>
</tr>
<tr>
<td>size (MB) = 11000</td>
</tr>
<tr>
<td>volume_name = disk3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ nas_volume -name log_vol -create disk3</td>
</tr>
<tr>
<td>Output</td>
</tr>
<tr>
<td>id = 394</td>
</tr>
<tr>
<td>name = log_vol</td>
</tr>
<tr>
<td>acl = 0</td>
</tr>
<tr>
<td>in_use = False</td>
</tr>
<tr>
<td>type = meta</td>
</tr>
<tr>
<td>volume_set = disk3</td>
</tr>
<tr>
<td>disks = d3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ nas_fs -name log -create log_vol</td>
</tr>
</tbody>
</table>

Sample session 311
2. Mount the log file system on the Data Mover (server) doing the migration.

<table>
<thead>
<tr>
<th>Example 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ server_mountpoint server_2 -create /mgfslog</td>
</tr>
<tr>
<td><strong>Output</strong></td>
</tr>
<tr>
<td>server_2 : done</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ server_mount server_2 log /mgfslog</td>
</tr>
<tr>
<td><strong>Output</strong></td>
</tr>
<tr>
<td>server_2 : done</td>
</tr>
</tbody>
</table>

3. Open the param file.

4. Append the line specifying where the Data Mover stores the log plus the prefix of the log filename:

   **Note**: Ensure that you do not add any extra spaces or blank lines into the param file.

   a. Type the following commands:

   $ cd /nas/server/slot_2
   $ ls -l param
b. Using a text editor, open the `param` file:

```
$ vi param
```

c. Append the specified line to:

```
param mgfs logPathPrefix=/mgfslog/mgfs
```

**Note:** Be sure the file system is mounted for the log. When you change the log path, you must remove the old log path line. In addition, after you complete the data migration, you must remove this line from the param file.

5. Restart the Data Mover by typing the following command:

```
$ server_cpu server_2 -reboot now
```

Each time you restart the Data Mover, a new log file is created while the existing log files are retained. The log file has a prefix (in this case, `mgfs`) and a numeric suffix, which represents a timestamp. The larger the number, the more recent the log file.
Retrieving log files

**Note:** You can download migration and error logs through the CDMS GUI.

To retrieve MGFS log files, perform the following:

1. On the Control Station, type the `server_export` command to allow external access to the log file system.

   ```
   $ server_export server_2 -option anon=0 /mgfslog
   
   server_2 : done
   ```

   The *EMC VNX Command Line Interface Reference for File* provides details about `server_export` command options.

2. On the Windows client or UNIX workstation, mount the log file system to a local mount point.

   ```
   C:\> mount 172.24.67.54:/mgfslog /mnt
   
   server_2 : done
   ```

   The directory specified by this mount point contains the log file system.

   If there is any problem with the migration, you need to collect the log files and the panic crash dump file (if any) for EMC Customer Support analysis.
APPENDIX E
Network Appliance Considerations

The following topics provide information that you must know if you are migrating data from a Network Appliance file server to the VNX:

- Introduction ................................................................................................. 316
- Procedure ....................................................................................................... 317
- Example  ......................................................................................................... 318
Introduction

Data migration from a Network Appliance (NetApp) server to the VNX is only supported with the NFS protocol.

If you are migrating data from a Network Appliance file server by using the Network Appliance Snapshot feature, there are additional considerations you must understand, and some extra steps you must perform during migration.

⚠️ CAUTION

The following procedures should only be used for migrations involving Network Appliance source file servers that use the Snapshot feature. Data loss might result if you follow these procedures for other source file servers, or for Network Appliance servers that do not use the Snapshot feature. The Data Movers must be running version 5.6 or later of the software. This procedure must be applied to VNX .ckpt_<ckpt_name> directories.

This procedure brings a .snapshot directory online manually, which allows you to convert the file system to a UxFS format after migration completes.

After conversion, however, the .snapshot directory in the new file system is empty.
Procedure

To migrate a NetApp file system from an NFS source file server with a .snapshot directory, perform the following:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Check the NFS source file server to learn if there is a .snapshot directory at the root level named .snapshot. The <code>diskUsage.pl</code> script avoids the .snapshot directory so it is not counted in the file system predicted size.</td>
</tr>
<tr>
<td>2</td>
<td>Check the software version on the Data Movers to be used for the migration. Ensure that the software version is at least 5.6 or later by typing at the Control Station the <code>server_version &lt;movername&gt;</code> command (where <code>&lt;movername&gt;</code> is the name of the target server).</td>
</tr>
<tr>
<td>3</td>
<td>Locate the <code>specialCmd</code> utility on the Control Station in the <code>/nas/tools/cdms</code> directory.</td>
</tr>
<tr>
<td>4</td>
<td>Create an MGFS file system and corresponding mount point, and mount it.</td>
</tr>
<tr>
<td>5</td>
<td>Execute the <code>server_cdms &lt;movername&gt; -connect &lt;mgfs&gt;</code> command to the Network Appliance source file server, but do not export it yet.</td>
</tr>
</tbody>
</table>
| 6    | If the NFS source file server has a .snapshot directory at the root level of this file system, use the following command syntax:  
  ```bash
  $ specialCmd getOnline {<movername>}|<pathname>/./.snapshot
  <path_to_snapshot_file_system>
  ```  
  where:  
  `<movername>` = Data Mover name  
  `<pathname>` = path from the mount point on the Data Mover  
  The path includes the mount point, the local path, and the .snapshot filename. |
| 7    | Verify that the .snapshot directory is online by using the `server_log` command. |
| 8    | Export from the Data Mover, and mount this MGFS file system on a client host (from client). |
| 9    | Migrate the file system content by using the `server_cdms <movername>` command. |
| 10   | Convert the MGFS file system to a UxFS file system from the Control Station. |
Network Appliance Considerations

Example

The steps in this example correspond to the previous set of steps:

1. Check the NFS source file server for a .snapshot directory.

   The following text is the original source file server structure before the migration. Note that there is a .snapshot directory at the root of this file system, and that it contains subdirectories.

   $ pwd
   /ys10
   $ ls -laR
   .:
   total 16
   drwxr-xr-x 5 root other 512 May 21 13:39 .
   drwxr-xr-x 74 root root 1536 May 21 13:51 ..
   drwxr-xr-x 2 root other 512 May 21 13:39 .snapshot
   -rw-r--r-- 1 root other 15 May 16 16:48 1
   -rw-r--r-- 1 root other 22 May 16 16:48 2
   drwxr-xr-x 2 root other 512 May 21 13:40 dir1
   drwxr-xr-x 2 root other 512 May 21 13:40 dir2
   ./snapshot:
   total 12
   drwxr-xr-x 2 root other 512 May 21 13:39 .
   drwxr-xr-x 5 root other 512 May 21 13:39 ..
   -rw-r--r-- 1 root other 15 May 21 13:39 1
   -rw-r--r-- 1 root other 22 May 21 13:39 2
   -rw-r--r-- 1 root other 15 May 21 13:39 3
   -rw-r--r-- 1 root other 22 May 21 13:39 4
   ./dir1:
   total 8
   drwxr-xr-x 2 root other 512 May 21 13:40 .
   drwxr-xr-x 5 root other 512 May 21 13:39 ..
   -rw-r--r-- 1 root other 15 May 21 10:50 11
   -rw-r--r-- 1 root other 22 May 21 10:50 12
   ./dir2:
   total 8
   drwxr-xr-x 2 root other 512 May 21 13:40 .
   drwxr-xr-x 5 root other 512 May 21 13:39 ..
   -rw-r--r-- 1 root other 15 May 21 13:39 21
   -rw-r--r-- 1 root other 22 May 21 13:39 22

2. Check the software version on the Data Mover to ensure it is version 5.6 or later by typing the following command:

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ server_version server_2</td>
</tr>
</tbody>
</table>
3. Run the specialCmd utility by typing the following commands:

   $ cd /nas/tools/cdms
   $ ./specialCmd getOnline server_2 /mgfs521/.snapshot

4. Ensure that the utility ran successfully by checking the log file.

   An indication of success or failure should appear in the last lines of the file. However, you can always use a server_log server_2|grep –i mgfs command to find the lines:

   $ server_log server_2


   If you type an incorrect directory name, an error message similar to the following appears:

   .specialCmd getOnline server_2 /mgfs521/junk
   server_log server_2
   2001-05-23 17:21:26: MGFS: 3: Cannot find path /mgfs522/junk
   /junk

   **CAUTION**

   Do not read the .snapshot directory before running the specialCmd utility. For example, you should not type a cd .snapshot or ls –alR mgfs521 command. Either of these commands brings the .snapshot directory online, but with all the inodes within it as offline. The getOnline utility, invoked by the specialCmd utility, becomes unusable. In that case, you must use the specialCmd disconnect command to break the connection. Then set online the /mountpoint/localpath directory, delete the localpath directory and all its contents, and rebuild it from the beginning with a new connection by using a server_mount <movername> -option connect command.

   However, you can use an ls –al command at the root of the file system to verify that the .snapshot directory is offline.

   $ ls -al /nas/rootfs/slot_2/mgfs521/fs1
   total 16
   drwxr-xr-x 7 root bin 512 May 21 14:52 .
Network Appliance Considerations

```

drwxr-xr-x 16 root root 512 May 21 11:49 ..
drwxr-xr-x 2 root bin 0 May 21 13:39 .snapshot
-rw-r--r-- 1 root bin 15 May 16 16:48 1
-rw-r--r-- 1 root bin 22 May 16 16:48 2
drwxr-xr-x 2 root bin 0 May 21 13:40 dir1

drwxr-xr-x 2 root bin 0 May 21 13:40 dir2
```

Notice the size of the .snapshot directory is 0 (zero).

The .snapshot directory is online, but everything has been removed in that directory.

```
$ ls -al /nas/rootfs/slot_2/mgfs521/fs1

total 24
  drwxr-xr-x 7 root bin 512 May 21 14:52 .
  drwxr-xr-x 16 root root 512 May 21 11:49 ..
  drwxr-xr-x 2 root bin 512 May 21 14:53 .snapshot
-rw-r--r-- 1 root bin 15 May 16 16:48 1
-rw-r--r-- 1 root bin 22 May 16 16:48 2
  drwxr-xr-x 2 root bin 0 May 21 13:40 dir1
  drwxr-xr-x 2 root bin 0 May 21 13:40 dir2

$ ls -al /nas/rootfs/slot_2/mgfs521/fs1/.snapshot

total 16
  drwxr-xr-x 2 root bin 512 May 21 14:53 .
  drwxr-xr-x 7 root bin 512 May 21 14:52 ..
```

5. Create an MGFS file system and mount point.

Even though only the `nas_fs` command is shown in this example, you also need to use the `nas_disk`, `nas_volume`, `nas_fs`, and `server_mountpoint` commands, as described in the migration procedure.

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$ nas_fs -name mgfs521 -type mgfs -create vol_mgfs521</code></td>
</tr>
</tbody>
</table>

6. Mount the NFS source file server on this Data Mover.
7. Confirm that the remote file system is correct by using one of the following:

- A `server_cdms <movename> -verify <mgfs>` command to verify that the file system name and IP address of the mounted file systems.

- A `server_cdms <movename> -info <mgfs>` command from the Control Station to check the mounted path for that connection.

Do not export it yet.

8. Export the file system by typing the following command:

```
$ server_export server_2 /mgfs521/fs1
```

9. Mount the MGFS on a client.
10. Run the `server_cdms` command.

If you check the `.snapshot` directory before running the command, it is empty, as shown in step 3:

```
$ mount 128.221.252.104:/mgfs521/fs1 /ysi_srv2_mgfs521

$ pwd
/ysi_srv2_mgfs521

$ ls -al .snapshot
total 32
  drwxr-xr-x 2 root other 512 May 21 2001 .
  drwxr-xr-x 7 root other 512 May 21 2001 ..
```

11. Perform file system verification, as appropriate.
Index

CIFS migration 44
NFS migration 42, 43
cid, identifying 174
CIFS
  checklist 44
  creating cdms_migrator account 127
  error codes 234
  migration
    many-to-one 176 to 212
    one-to-one 123 to 176
  overview description 18
  preliminary setup items 50
  premigration tasks 126
  restrictions 30 to 31
  server name merge strategy
    many-to-one new 178
    many-to-one retained 177
  starting migration service 193
  starting service 90
  supported source file servers 20
  troubleshooting 216
client cannot error message
dirprivU.pl script 263
dirprivW.pl script 271
command
  chgrp 249
  export 89, 221
  find 232
  nas_fs 86, 320
  net use 135
  net view for duplicate shares 181
  rm 222
  rmdir 233
  rpcinfo 71
  server_cdms 121
    connections 162
  convert option 69
disk drives 158
  file system merge 113
  info option 68, 321
  server_export 71, 215
  server_log 94, 116, 317
  server_mount 163, 301
  server_mountpoint 321
server_ping 82, 83, 219
server_version 317
share 89, 221
command file template, connBuilder.pl script 245
compname
  configuring Data Mover 190
  Data Mover setup 152
  joining domain 154, 193
  maximum character length 192
configuration
  CIFS-specific guidelines 25 to 26
  common guidelines 24
  Data Mover 210
  Data Mover for CIFS migration 190
  DNS service (Windows 2000) 137
  gateway 142
  helpful tips 25
  international character support (NFS) 51
  migration interfaces 140
  network connections 67
  NFS-specific guidelines 25
  storage requirements 74
  Unicode 53
  WINS (Windows NT 4.0) 137
connBuilder.pl script
  command file template 245
  definition 33, 241
  editing 244
  exclude file template 246
  File Conversion dialog box 132
  file naming syntax 132
  generating exclude file 131
  input word file 244
  Microsoft Excel 2000 spreadsheet 51
  output .txt files 132
  syntax and parameters 244
connection command
  spaces in share name 121
  troubleshooting 223, 226, 228
connections
  creating for disk drives 158, 197, 211
    Windows 2000 159
    Windows 4.0 NT 161
  removing unwanted 232
Control Station, interface options 21
conversion
  failure 229
  I18N 53
  incompletion failure 217
MGFS file systems 109, 117
monitoring progress 170
CPU, loading effects 296

D
data
accessibility by clients 66
updating during migration 35
data migration
calculation 292
completing task 98, 165
estimating times 288
equivalent 35 to 39
Network Appliance 317 to 322
network traffic 294
shared network 295
small files 293
times 76
unplanned allowances 292
using GUI 48
Data Mover
507 18, 297
510 297
adding to network 81
assuming IP address and hostname 81
configuring
for CIFS 136, 152, 210
for migration 208
configuring for CIFS 190
creating mount points 87, 146
enabling Unicode 139
mounting MGFS 87, 146
new host name and IP address 81
rebooting 313
time synchronization 136
Windows 2000 environment configuration 152, 191
Windows NT 4.0 environment configuration 152, 192
default MGFS
default source 302
source permits root access 303
dircount.pl script
access rights 261
ckpt file 261
definition 32, 241
display format definitions 258
displaying directory structure 256, 257
empty special directory 260
known issues 260
local UNIX directories 260
long name 261
Perl script 260
running 255
snapshot directory 261
spreadsheet usage 259
syntax and parameters 254
directory structure, dircount.pl script 256
dirprivU.pl script
accessing files and directories 267
client cannot error messages 263
definition 33, 241, 262
dot (.) files 266
functionality 263
hard lock files 264
illegal mode bits 265
international character sets 269
name length limitation 268
NetApp snapshot directory 268
owner warning messages 264
remote mounted NFS directories 268
running 266
sample output 266, 267
SetGID warning 265
syntax and parameters 262
unreadable directory or file 264
dirprivW.pl script
access rights 273
client cannot error messages 271
definition 33, 241, 270
encrypted directory or file 271
functionality 270 to 271
international character sets 274
known issues 273
name length 273
Perl script 273
running 272
sample output 272
snapshot file 274
syntax and parameters 270
disk drive
creating connections 158, 197
establishing connections 198
disk info table, definitions 280
Index

disk usage, extending 102
diskUsage.pl script
  calculating storage space 49
  definition 32, 241, 275
disk info table 280
entering block size 276
entering directory pathname 276
entering filename 276
estimating directory disk usage 276
example 280
functionality 278
hardlinks 276
log file pathname 277
NFS migration 84
running automatically 277 to 280
running manually 275
syntax and parameters 275
DNS, zone update 82
domain, joining Windows 154, 193

E
EMC application
  BCVs 29
  VNX Replicator 29
EMC tools
  backupWrapper.exe utility 240, 243
  ch_group.pl script 240
  connBuilder.pl script 240, 244
dircount.pl script 240, 254
dirprivU.pl script 240, 262
dirprivW.pl script 240, 270
diskUsage.pl script 240
lgdup.exe utility 240, 283
sharedup.exe utility 240, 284
specialCmd utility 240, 286
enablement, I18N 52
error
analyzing 232
c000005B 228
c0000064 228
c000006d 225
c00000be 226
c0000ccc 227
c000015b 227
time skews 225
error codes, CDMS migration 234
error log
analyzing content 104, 105, 168
examining content 112
server_cdms command 104, 166
evaluation, network 66, 72, 74
exclude file
adding files for exclusion 135
creating 77
edited example 134
editing pathname 133, 134
example 133
modifying pathname 133
purpose 77, 246
running connBuilder.pl script 131
running server_cdms command 134
running sharedup.exe utility 130
template, connBuilder.pl script 246
export
alias name 96
MGFS 95
options 300

F
failure
verification 229
FAQ 66 to 69
file system
  calculating sizes 142
  checking mount 88, 147
  connecting (NFS) 90, 111
  converting 46
  exporting for NetApp 321
  exporting sequentially 115
  merging (NFS) 111
  migration
    sequential 115
    simultaneous 115
size
  determining 49
  extending 49
  unmounting 89
  UxFS 19, 46, 52, 109
  UxFS conversion 216
  verification process 116
  verify export with alias name 96
File System Directory Tree Counting script (dircount.pl) 254
File System Privilege Checking script (dirprivU.pl) 262
files
exclude only 77
high priority 78, 135, 182
high priority for include file 208
include only 77, 135
large 298
migrating with GigE 298
small 293, 298
FQDN, retained server name merge 197
fstab, modifying 95

G
gateway, configuring 142
GID
ch_group.pl script 33, 241, 249
changing with ch_group.pl script 251
checking user’s file access 300
consolidating multiple hosts 25
correcting numbers 117
old versus new 250
SetGID warning 265
SetGroupID On Execute 307
GigE
large files 298
performance improvement 298
small files 298
GUI
running data migration 48
supported functionality 47

H
hard lock files, dirprivU.pl script 264
hardlinks, diskUsage.pl script 276
hardware
installation 23
planning and design phase 70
system requirements 27, 28

I
I18N
conversion 53
definition 52
examples 53
set up process 52
illegal mode bits, dirprivU.pl script 265
implementation

CIFS phase 2 120
CIFS summary of steps
  many-to-one 177, 178
  one-to-one 125
NFS phase 2 81
NFS summary of steps
  many-to-one 111
  one-to-one 80
include file
  creating 135
  definition 78, 135
  example 136
inode
  logging 310
  offline 47
installation
  hardware summary 70
  Microsoft Excel 2000 28, 51
  Microsoft Word 28
  migration tools 45
  Perl script 27, 50
  physical equipment 23
  premigration tools 129
  Software version 5.2 27
  software version 5.2 73
  Win32 API module 28, 55
interface, configuring 140
international character support
  configuring (NFS) 51
dirprivU.pl script 269
dirprivW.pl script 274
support capability 19
intranet, usage 23
IP addresses, multiple 81, 216
Issue Tracker, accessing 39

K
KDC, server key 217
Kerberos, time synchronization 137
known issues
  ch_group.pl script 253
dircount.pl script 260
dirprivU.pl script 267
dirprivW.pl script 273
Index

L
lgdup.exe utility
  definition 33, 242, 283
  migrating local groups 156, 195
  options 283
  overwriting user rights 158
  setting user rights 188, 194, 209
  syntax and parameters 283
  user rights 156
limitations
  CIFS-specific 30, 31
  common 29
  NFS-specific 29
loading, CPU 296
local groups
  administrative share 121
  migrating 155, 156, 211
  migrating information 194, 196
local path currently in use, connection command error 224
log files, retrieving 314
logging
  inode operations 310
  param file 313
  retrieving log files 314
  sample session 311 to 313
  setting up 310
long name, dircount.pl script 261

M
many-to-one migration
  implementation
    CIFS 176
    NFS 112
  summary of steps
    CIFS 177
    NFS 111
merge
  example (NFS) 113
  NFS file systems 111
message
  client cannot 263, 271
  owner warning 264
metavolumes, creating 84, 144, 184
MGFS
  calculating size 183
  creating 85, 142, 145, 183, 185
  creating for NetApp 320
  creating on Data Mover 208
default source 303, 304
exporting 95
extending size 49
mounting on Data Mover 87, 146
source permits root 304
source permits root access 303
uses root 304
Microsoft Excel 2000 spreadsheet
  connBuilder.pl script 51, 132
  installation 28
Microsoft Word, adding convention packs 132
migrating data
  adding Data Mover to network 81
  checking server log 94
  connecting subdirectory to file system 91
  exporting MGFS 95
  preparing NFS source file server 89
  verifying NFS migration status 97
migration
  advisory locking 264
  checking progress 102
  client PC 24
  configuration changes 66
  configuration tips 24
  defining limits 74
  diskUsage.pl script 291
duration variables 102, 166
example 35
executing server_cdms command 100, 165
failed verification 229
hard locked files 264
issues 73
large amount 59
local groups 155, 156, 196
many-to-one implementation
  CIFS 176
  NFS 111
monitoring progress 164, 172
Network Appliance Snapshot feature 50
one-to-one implementation
  CIFS 126
  CIFS definition 123
  NFS 81
online 69
parallel streams 67, 297
partial 172
personnel qualifications 62
posttesting 118, 212
preliminary setup and requirements 50
priority exports and servers 67
process hangs 217
required tasks 23
retained server name merge 181 to 207
sequential 115
SGID bit 307
shared volumes 30, 31
simultaneous 115
site preparation 42
small amount 59
strategy considerations 75
timeline 68
tools 31
  CIFS specific 33
  common 31
  NFS specific 33
transfer rates 289
using GUI 48
verifying status 106
migration phase
  implementation 62
  CIFS 120
  NFS 81
NFS one-to-one implementation 81 to 110
planning and design 62, 70
postmigration testing 62, 118
migration suspension hang condition, NFS 218
mode, security 140
mount points, creating 87, 186
  on Data Mover 146
mount, options 300
multiple file servers, support 21
multiple file systems, conversion process 116, 117
snapshot file 274
source file servers 73
network
  adding Data Mover 81
  configuration of connections 67
data migration times 76
evaluation 66, 72, 74
minimum transfer rate 28
restricting source file server access 149
system requirements 27, 28
throughput speed 22
transfer speed factors 298
network analysis, personnel qualifications 63
network traffic, shared network 295
new server name merge
  definition 180
  implementation 207
NFS
  checklist 42, 43
  error codes 234
  international character set support 51
  migration 81 to 117
  mount and export options 300
  multiple IP addresses 81, 216
  obtaining access privileges 300
  overview description 17
  preliminary setup items 50
  supported functions 19
  supported protocol versions 19
  supported source file servers 19
  troubleshooting 215
NIC, purpose 23
NOPRIV option
  lgdup.exe utility 283
  removing user rights 162
O
offline inodes, definition 47
one-to-one migration
  assumptions (CIFS) 124
  CIFS source file server assumptions 123, 124
  conventions (CIFS) 124
  definition (CIFS) 123
  summary of steps (CIFS) 125
online, migration 69
option
  access 302
Index

anon 305
migration of data 35
NOPRIV 162
OutLook.pst, priority files 67
overview
   CDMS for CIFS 18
   CDMS for NFS 18
   CDMS functionality 34
owner warning error message, dirprivU.pl script 264

P
parallel migration
   definition 297
   streams 67
param file, editing 312
partial migration
   verification 169
   verifying status 106
pathname length, maximum 30
Perl
   adding Win32 API module 55
   backupWrapper.exe utility 58
dircount.pl script 260
dirprivU.pl script 267
dirprivW.pl script 273
installing 28, 207
installing for CIFS migration 129
minimum version 27, 45, 50
personnel, qualifications 62
phase 1, planning and design 70 to 76
phase 2, CIFS implementation 123
phase 2, NFS implementation 80 to 109
phase 3, post-migration testing 118, 212
planning and design
   analysis and detailed planning 74
   assessment 70
   installed hardware 70
privileges, obtaining or access 300

Q
qualifications, migration personnel 62

R
rates, migration transfer 289
read requests, constructing 301
remount, of file system as R/O 89
restrictions
   CIFS-specific 30
   common 29
   NFS-specific 29
retained server name merge
   definition 180
   many-to-one migration 181
   migration implementation 181 to 207
S
scope
   CIFS-specific migration processes 22
   common supported migration processes 21
   NFS-specific migration processes 22
script
   ch_group.pl 249 to 253
   connBuilder.pl 244 to 248
dircount.pl 254 to 261
dirprivU.pl 262 to 269
dirprivW.pl 270 to 274
diskUsage.pl 275 to 280
security
   setting mode 140
   setting user rights 188
   verifying mode 139
   Windows systems 31
sequential migration, definition 115
server
   key, updating 217
   log, checking connection status 94
Server Manager, removing source file server name
   from domain 151
server_cdms command, Control Station location 46
service, starting for CIFS migration 90, 155, 193
SetGID warning, dirprivU.pl script 265
setup
   CDMS migration 50
   l18N 52
SGID bit, migrating files 307
share name
   duplicate 182
   spaces 121
shared network, network traffic 295
shared volume, migration 30
sharedup.exe utility
   definition 33, 242, 284
   editing 201 to 204
executing  199, 205
executing for disk drives  163, 164
options  284
output
  after editing  204
  before editing  203
source file server disk drive  200, 211
syntax and parameters  201, 284
simultaneous migration, definition  115
site preparation
  adding Win32 API  55
  backup considerations  59
  configuring for Unicode  53
determining file system size  49
installing migration tools  45
introduction  42
user translation methods  54
sizes
  calculating file system  142, 183
  of file systems  49
SMB, connection command failures  223
snapshot directory
  dircount.pl script  261
  Network Appliance  316
software
  system requirements  27, 28
source file server
  assumptions for one-to-one CIFS migration  123, 124
  backing up  136
checking connection  93, 162
  data corruption  69
duplicate shares  181, 207
evaluating CIFS directory  130
  international character set  50
options  302
preparing for NFS migration  89
read-only state  24
rebooting  151, 189
renaming  151
restricting user access  149
task reassignment  82
spaces, share names  121
specialCmd utility
  definition  32, 242, 286
options  286
running for NetApp  319
syntax  286
speed, of network  23
spreadsheet
  dircount.pl script  259
  Microsoft Excel 2000  51
stale NFS handles, troubleshooting  222
storage, system requirements  27, 28
streams, parallel migration  67
striped volumes, creating  143, 183
support
  LAN-based NDMP  29
  Windows 2000  22
  Windows NT 4.0  22
system requirements
  CIFS specific  28
  hardware  27, 28
network  27
  NFS specific  27
  software  27, 28
storage  27, 28
testing, postmigration  118
threads
  starting CIFS migration service  193
time
  migration variables  102, 166
  skews, error  225
  synchronization  136
timeline, migration of data  68
tools
  CIFS installation  46
  installation  45
  installing  45, 51
  See migration tools
traffic, network  294
translation, Unicode  53
troubleshooting
  CIFS problems  216
  connection command failures  223 to 229
  error codes  234
defailed verification  229
  NFS
    problems and solutions  214
    NFS migration suspension (hang) conditions  218
    permission-related hangs  219
    stale NFS handles  222

Index

executing  199, 205
executing for disk drives  163, 164
options  284
output
  after editing  204
  before editing  203
source file server disk drive  200, 211
syntax and parameters  201, 284
simultaneous migration, definition  115
site preparation
  adding Win32 API  55
  backup considerations  59
  configuring for Unicode  53
determining file system size  49
installing migration tools  45
introduction  42
user translation methods  54
sizes
  calculating file system  142, 183
  of file systems  49
SMB, connection command failures  223
snapshot directory
  dircount.pl script  261
  Network Appliance  316
software
  system requirements  27, 28
source file server
  assumptions for one-to-one CIFS migration  123, 124
  backing up  136
checking connection  93, 162
data corruption  69
duplicate shares  181, 207
evaluating CIFS directory  130
  international character set  50
options  302
preparing for NFS migration  89
read-only state  24
rebooting  151, 189
renaming  151
restricting user access  149
task reassignment  82
spaces, share names  121
specialCmd utility
  definition  32, 242, 286
options  286
running for NetApp  319
syntax  286
speed, of network  23
spreadsheet
  dircount.pl script  259
  Microsoft Excel 2000  51
stale NFS handles, troubleshooting  222
storage, system requirements  27, 28
streams, parallel migration  67
striped volumes, creating  143, 183
support
  LAN-based NDMP  29
  Windows 2000  22
  Windows NT 4.0  22
system requirements
  CIFS specific  28
  hardware  27, 28
network  27
  NFS specific  27
  software  27, 28
storage  27, 28
testing, postmigration  118
threads
  starting CIFS migration service  193
time
  migration variables  102, 166
  skews, error  225
  synchronization  136
timeline, migration of data  68
tools
  CIFS installation  46
  installation  45
  installing  45, 51
  See migration tools
traffic, network  294
translation, Unicode  53
troubleshooting
  CIFS problems  216
  connection command failures  223 to 229
  error codes  234
defailed verification  229
  NFS
    problems and solutions  214
    NFS migration suspension (hang) conditions  218
    permission-related hangs  219
    stale NFS handles  222
trust relationships, Windows domain 73

U
Unicode
  configuring 53
  editing with WordPad 285
  enabling for Data Mover 139
Unisphere, data migration GUI functionality 47
unmount, of file system 89
update, of data 35
user authentication, setting up 142
User Manager User Rights Policy dialog box 156, 188
user rights
  lgdup.exe utility 187
  modifying 147
  setting for CIFS migration 147
user translation methods, Usermapper 54
Usermapper
  configuring 22
  failure 224
  purpose 54
utility
  installing 45
  sharedup.exe 284 to 285
UxFS
  converting from MGFS 169, 206, 211
  creating for log file 311
  file-system conversion 52
  log format 310
  MGFS log file 310
  NFS data migration 19
  NFS problems and solutions 216

V
verification
  failure 229
  monitoring progress 171
  multiple file systems merge 116
  single file system 116
Virus Checker, migration script 73
VNX HighRoad, configuration 68
volumes
  creating 183
  meta 144
  shared 30, 31
  striped 143

W
Win32 API
  adding to Perl script 55, 57
  Prototype.pm file 58
Windows 2000
  DNS service 137
  security 31
  server key 217
  setting required rights 147
  WINS server 197
Windows NT 4.0
  setting required rights 147
  User Manager User Rights Policy dialog box 149
  WINS 137
Windows XP, CIFS support 20, 22, 31
Windows, security mode 140

X
  xlt.cfg file, editing for international character set support (NFS) 51

Z
  zone, DNS 82