

Data ONTAP™ 5.2 COMMAND REFERENCE

Network Appliance, Inc.
2770 San Tomas Expressway
Santa Clara, CA 95051, USA
Tel: +1 408 367-3000
Fax: +1 408 367-3151
Support tel: (888) 4-NETAPP
Support email: support@netapp.com
Information email: info@netapp.com
Web: <http://www.netapp.com>

Part number 210-02167
September 1998

Copyright

Copyright © 1998 Network Appliance, Inc. All rights reserved. Printed in the U.S.A.

No part of this book covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Network Appliance reserves the right to change any products described herein at any time, and without notice. Network Appliance assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by Network Appliance. The use and purchase of this product do not convey a license under any patent rights, trademark rights, or any other intellectual property rights of Network Appliance.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

Trademarks

Network Appliance, NetCache, Data ONTAP, SmartSAN, SnapCopy, Snapshot, WAFL, Web Filer, FilerView, BareMetal, and SecureShare are trademarks of Network Appliance, Inc. FAServer, NetApp, and the Network Appliance logo are registered trademarks of Network Appliance, Inc.

All other brands or products are trademarks or registered trademarks of their respective holders and should be treated as such.

NAME

arp – address resolution display and control

SYNOPSIS

arp *hostname*

arp -a

arp -d *hostname*

arp -s *hostname ether_address* [**temp**] [**pub**]

arp -s *hostname pvc iface vpi vci* [*aal* [*encap* [*addr*]]]

arp -l **pvc** *iface vpi vci* [*aal* [*encap*]]

arp -x *iface vpi vci*

DESCRIPTION

The **arp** command displays and modifies the tables that the address resolution protocol uses to translate between Internet and Ethernet addresses.

When an ATM interface is present, **arp** also displays and modifies the tables that the ATM address resolution protocol uses to translate between Internet addresses and ATM virtual circuits.

With no flags, **arp** displays the current ARP entry for *hostname*. The host may be specified by name or by number, using Internet dot notation.

OPTIONS

-a Displays all of the current ARP entries.

-d Deletes an entry for the host called *hostname*.

If the ARP entry for this destination host corresponds to an entry in the ATM arp table, the outgoing virtual circuit to this destination host called *hostname* is terminated, and the entry is deleted.

-s Creates an ARP entry for the host called *hostname* with the Ethernet address *ether_address*. The Ethernet address is given as six hex bytes separated by colons. The entry will be permanent if the words following **-s** includes the keyword **temp**. Temporary entries that consist of a complete Internet address and a matching Ethernet address are flushed from the arp table if they haven't been referenced in the past 20 minutes. A permanent entry is not flushed.

If the words following **-s** include the keyword **pub**, the entry will be "published"; i.e., this system will act as an ARP server, responding to requests for *hostname* even though the host address is not its own.

An alternate format of the **arp** command's **-s** option can be used if the words following **-s** include the keyword **pvc**. This sets an ATM arp entry for an

outgoing permanent virtual circuit (PVC). All traffic for the host called *hostname* will be sent via the ATM interface *iface* using VPI *vpi*, VCI *vci*, AAL type *aal*, and will be encapsulated based on encapsulation type *encap*. If encapsulation is specified as **llc_bridged_8023** the argument that follows must be the 6-byte colon-separated destination MAC address.

- I** Attaches the IP layer to an incoming PVC. The words following **-I** must include the keyword **pvc**. All traffic received on the ATM interface *iface* using VPI *vpi*, VCI *vci*, AAL type *aal* and decapsulated based on encapsulation type *encap*, will be handed over to IP.
- x** Detaches the IP layer from an incoming ATM virtual circuit. IP traffic will no longer be accepted on the specified VPI *vpi* and VCI *vci* for the specified interface *iface*.

NOTES

Data ONTAP does not support trailer encapsulations.

The following restrictions apply when using the **arp** command to set ATM arp entries for PVCs:

Currently the specified value of *vpi* must be zero (0) and the specified value of *vci* must be less than 1024.

The specified value of *aal* must be one of 4 or 5. If no value is specified, the default value of 5 is selected.

If no encapsulation is specified, the value of **null** is selected. Encapsulation must be one of:

null (no encapsulation)

llc_routed (IEEE LLC encapsulation for routed PDUs)

llc_bridged_8023 (IEEE LLC encapsulation for Ethernet/802.3 bridged PDUs)

With the **arp -s** and **-I** options to set ATM arp entries for PVCs, the following additional procedures are needed:

The remote ATM host needs a complementary ARP setup, (using syntax appropriate to the host software) to set an outgoing and an incoming PVC to and from the filer.

On the interconnecting ATM switch(es), virtual channels must be assigned to the VPI/VCI entries that correspond to the filer's **arp -s** and **arp -I** commands.

CLUSTER CONSIDERATIONS

In takeover mode, each filer in a cluster maintains its own ARP table. You can make changes to the ARP table on the live filer, or you can make changes to the ARP table on the failed filer using the **arp** command in partner mode. However, the changes you make in partner mode are lost after a giveback.

SEE ALSO

na_ifconfig(1), na_partner(1), RFC1483.

NAME

cf – controls the takeover and giveback operations of the filers in a cluster

SYNOPSIS

cf [**disable** | **enable** | **forcegiveback** | **forcetakeover** | **giveback** [**-f**] | **partner** | **status** | **takeover**]

DESCRIPTION

The **cf** command controls the cluster failover monitor, which determine when takeover and giveback operations take place within a cluster.

The **cf** command is available only if your filer has the cluster license.

OPTIONS

disable	Disables the takeover capability of both filers in the cluster.
enable	Enables the takeover capability of both filers in the cluster.
forcegiveback	Forces the live filer to give back the resources of the failed filer even though the live filer detects an error that would prevent a complete giveback. For example, an error might prevent the failed filer from flushing data in the NVRAM to disk during a giveback. If the live filer detects this error, it does not perform a giveback. However, using the forcegiveback option forces a giveback despite such an error. When the failed filer reboots as a result of a forced giveback, it displays the following message: partner giveback incomplete, some data may be lost
forcetakeover	Forces one filer to take over its partner even though the filer detects an error that would otherwise prevent a takeover. For example, normally, if a detached or faulty ServerNet cable between the filers causes the filers' NVRAM contents to be unsynchronized, takeover is disabled. However, if you enter the cf forcetakeover command, the filer takes over its partner despite the unsynchronized NVRAM contents. This command might cause the filer being taken over to lose client data.
giveback [-f]	Initiates a giveback of partner resources. Once the giveback is complete, the automatic takeover capability is disabled until the partner is rebooted. A giveback fails if there are outstanding CIFS sessions or active system dump processes. If you use the -f option, the cf command terminates the outstanding CIFS sessions and dump processes before attempting a giveback.

partner	Displays the host name of the partner. If the name is unknown, the cf command displays “ partner. ”
status	Displays the current status of the local filer and the cluster.
takeover	Initiates a takeover of the partner.

SEE ALSO**na_partner(1)**

NAME

cifs – summary of cifs commands

SYNOPSIS**Command Summary**

This is a list of the subcommands of the **cifs** command.

cifs access	Modifies share-level Access Control List (ACL) entries.
cifs comment	Displays/modifies the CIFS server description.
cifs lookup	Translates user/group names into SIDs, and vice versa.
cifs restart	Restarts CIFS if it has been shut down with cifs terminate .
cifs setup	Configures CIFS service.
cifs sessions	Displays current configuration and current connections.
cifs shares	Displays/modifies the CIFS exports.
cifs stat	Displays operational statistics.
cifs terminate	Shuts down CIFS, or logs off a single station.
cifs testdc	Tests the filer's connection to domain controllers.

SEE ALSO

**na_cifs_access(1), na_cifs_comment(1), na_cifs_lookup(1), na_cifs_restart(1),
na_cifs_setup(1), na_cifs_sessions(1), na_cifs_shares(1), na_cifs_stat(1),
na_cifs_testdc(1), na_cifs_terminate(1)**

NAME

cifs access – modify share-level access control

SYNOPSIS

cifs access *share* [**-g**] *user rights*

cifs access -delete *share* [**-g**] *user*

DESCRIPTION

The **cifs access** command sets or modifies the share-level Access Control List (“ACL”) of a share.

The *share* argument specifies the share whose ACL is to be modified. The *user* argument specifies the user or group of the ACL entry. *user* can be an NT user or group, if the filer is using NT domain authentication, or it can be a Unix user or group, or it can be the special all-encompassing group **everyone**. The *rights* argument can be specified in either NT or Unix style. NT-style rights are:

No Access

Read

Change

Full Control

Unix-style rights are a combination of **r** for read, **w** for write, and **x** for execute.

If a share-level ACL entry for *user* already exists on the specified share, **cifs access** updates that ACL entry.

To display the current share-level ACL of a share, use Windows Server Manager or the **cifs shares** command.

OPTIONS

-g Specifies that *user* is the name of a Unix group. Use this option when you have a Unix user and group with the same name.

-delete Deletes the ACL entry for *user* on *share*.

EXAMPLES

The following example grants NT Read access to the NT user **ENGINEERING\mary** on the share **releases**.

```
toaster> cifs access releases ENGINEERING\mary Read
```

The following example grants Unix read and execute access to the user **john** on the share **accounting**.

```
toaster> cifs access accounting john rx
```

The following example grants full access to the Unix group **wheel** on the share **sysadmins**.

```
toaster> cifs access sysadmins -g wheel Full Control
```

The following example deletes the ACL entry for **ENGINEERING\mary** on the share **releases**.

```
toaster> cifs access releases -delete ENGINEERING\mary
```

SEE ALSO

na_cifs_shares(1)

NAME

cifs comment – display or change CIFS server description

SYNOPSIS

cifs comment [*newcomment*]

DESCRIPTION

The **cifs comment** command displays or changes the CIFS server description. CIFS clients see the CIFS server description when browsing servers on the network.

If no command-line arguments are given, **cifs comment** displays the current CIFS server description. If you enter a string for the *newcomment* parameter, the current CIFS server description is changed to *newcomment*. If *newcomment* contains spaces, enclose it in double quotation marks.

NAME

cifs lookup – translate name into SID or vice versa

SYNOPSIS

cifs lookup { *name* | *textualsid* }

DESCRIPTION

The **cifs lookup** command translates a Windows NT user or group name into its corresponding textual Windows NT SID (Security ID), or a textual NT SID into its corresponding Windows NT user or group name.

EXAMPLES

```
toaster> cifs lookup mday
SID = S-1-5-21-39724982-1647982808-1376457959-1221
```

```
toaster> cifs lookup NT-DOMAIN\mday
SID = S-1-5-21-39724982-1647982808-1376457959-1221
```

```
toaster> cifs lookup BUILTIN\Administrators
SID = S-1-5-32-544
```

```
toaster> cifs lookup S-1-5-32-544
name = BUILTIN\Administrators
```

```
toaster> cifs lookup nonexistentuser
lookup failed
```

NAME

cifs restart – restart CIFS service

SYNOPSIS

cifs restart

DESCRIPTION

cifs restart restarts CIFS service if it has been terminated by **cifs terminate**.

NAME

cifs sessions – information on current CIFS activity

SYNOPSIS

cifs sessions [*-s*] [*user*]

DESCRIPTION

The **cifs sessions** command displays information about CIFS users who are connected to the filer. If you omit the *user* argument, the command displays a summary of information about the filer and lists the users who are connected to the filer.

EXAMPLES**cifs sessions**

Server Registers as 'CONGA' in group 'NT-DOMAIN'

WINS Server:198.95.227.99

PC style Access Control is being used

Using domain controller NT-DOMAIN-PDC for authentication

```
=====
PC (user)                #shares    #files
HAWLEY-PC    (sam)        1          4
HAWLEY-PC    (wfwg)       2          1
NT40_1       (dave)       2          3
```

If you include the *user* argument, the command displays information about the specified user, along with the names and access level of files that *user* has opened. If you use * as the specified user, the command lists all users.

Executing the command for user **sam** might produce output as follows:

cifs sessions sam

users

shares/files opened

HAWLEY-HOME1 (sam)

ENG-USERS

Read-denyW - \SAM\SRC\FASWARE\PROD\COMMON\HTTPD\httpd_fast.c

HAWLEY-PC (sam)

ENG-USERS

The **-s** option displays security information for a specified connected user. If you do not specify a user or workstation name, the command displays security information for all users.

Executing the command **-s *** might produce the following:

cifs sessions -s *

users

Security Information

WIN-95 (AGuest - nobody[guest])

UNIX uid = 1208

user is a member of group nobody(65535)

NT membership

NT-DOMAIN\Domain Guests

BUILTIN\Guests

User is also a member of Everyone, Network Users

NAME

cifs setup – configure CIFS service

SYNOPSIS

cifs setup

DESCRIPTION

The **cifs setup** command performs the initial configuration of the filer for CIFS. You must have installed the CIFS license before you enter this command. You must run the **cifs setup** command from the console or from a telnet connection; you can't enter the command through **rsh**.

FILES

/etc/cifsconfig.cfg	general configuration information
/etc/cifssec.cfg	NT domain machine account information
/etc/filersid.cfg	local machine SID
/etc/lclgroups.cfg	local NT group information
/etc/usermap.cfg	multiprotocol user map file

CLUSTER CONSIDERATIONS

If your filers in a cluster use WINS and domain controllers, you must configure the filers to use the same WINS servers and domain controllers. Also, the filers must belong to the same domain or workgroup.

SEE ALSO

na_cifs_access(1), **na_partner(1)**, **na_group(5)**, **na_passwd(5)**.

NAME

cifs shares – configure and display CIFS shares information

SYNOPSIS

cifs shares

cifs shares *sharename*

cifs shares –add *sharename path*

[**–comment** *description*]

[**–maxusers** *userlimit*]

[**–forcegroup** *groupname*]

cifs shares –change *sharename*

{ **–comment** *description* | **–nocomment** }

{ **–maxusers** *userlimit* | **–nomaxusers** }

{ **–forcegroup** *groupname* | **–noforcegroup** }

cifs shares –delete *sharename*

DESCRIPTION

cifs shares displays one or more shares, edits a specified share, creates a share, or deletes a share.

Listing shares

To list all shares and their access control lists, use the command **cifs shares** with no arguments. To list a single share and its access control list, use the command **cifs shares** *sharename* where *sharename* name of the share.

```
toaster> cifs shares
Name           Mount Point           Description
----           -
HOME           /vol/vol0/home        Default Share
                  everyone / Full Control
C$             /vol/vol0             Remote Administration
                  BUILTIN\Administrators / Full Control
ENGR           /vol/vol0/engr        Engineering
                  DOMAIN\Engineering / Full Control
NEWS           /vol/vol0/news        News
                  DOMAIN\Guests / No Access
                  everyone / Read

toaster> cifs shares news
Name           Mount Point           Description
----           -
NEWS           /vol/vol0/news        News
                  DOMAIN\Guests / No Access
```

everyone / Read

Creating new shares

To create a new share, use the **-add** option:

```
cifs shares -add sharename path
  [ -comment description ]
  [ -maxusers userlimit ]
  [ -forcegroup groupname ]
```

sharename name of the new share; clients use this name to access the share.

path full path name of the directory on the filer that corresponds to the root of the new share.

-comment *description*
description of the new share. CIFS clients see this description when browsing the filer's shares. If the description includes spaces, it must be enclosed in double quotation marks. If you do not specify a description, the description is blank.

-maxusers *userlimit*
maximum number of simultaneous connections to the new share. *userlimit* must be a positive integer. If you do not specify a number, the filer does not impose a limit on the number of connections to the share.

-forcegroup *groupname*
name of the group to which files to be created in the share belong. The *groupname* is the name of a group in the UNIX group database.

Deleting existing shares

To delete a share, use the **-delete** option:

```
cifs shares -delete sharename
```

sharename is the name of the share to be deleted. A share cannot be deleted if it is in use.

Changing the settings of existing shares

To change the settings of an existing share, use the **-change** option:

```
cifs shares -change sharename
  { -comment description | -nocomment }
  { -maxusers userlimit | -nomaxusers }
  { -forcegroup groupname | -noforcegroup }
```

The settings of a share can be changed at any time, even if the share is in use.

sharename is the name of the existing share that is to be changed.

- comment** *description*
changes the description of the share. For more information about the share description setting, see the **Creating new shares** section, above.
- nocomment** changes the description of the share to an empty string.
- maxusers** *userlimit*
changes the user limit on the share. For more information about the user limit setting, see the **Creating new shares** section, above.
- nomaxusers** removes the user limit on the share.
- forcegroup** *groupname*
changes the forcegroup setting. For more information about the forcegroup setting, see the **Creating new shares** section, above.
- noforcegroup**
specifies that files to be created in the share do not belong to a particular UNIX group. That is, each file belongs to the same group as the owner of the file.

SEE ALSO**na_cifs_access(1)**

NAME

cifs stat – print CIFS operating statistics

SYNOPSIS

cifs stat [*interval*]

DESCRIPTION

The **cifs stat** command has two forms. If you specify the interval, the command continues displaying a summary of CIFS activity until interrupted. The information is for the preceding *interval* seconds. (The header line is repeated in the display every 10 lines.)

If you do not specify the interval, the command displays counts and percentages of all CIFS operations.

EXAMPLE

toaster> **cifs stat 10**

GetAttr	Read	Write	Lock	Open/Cl	Direct	Other
175	142	3	70	115	642	50
0	0	0	0	0	18	0
0	8	0	0	3	8	0
0	10	0	0	0	0	0
0	6	0	0	1	0	0
0	0	0	0	0	0	0

NAME

cifs terminate – terminate CIFS service

SYNOPSIS

cifs terminate [*workstation*] [**-t mins**]

DESCRIPTION

If you stop CIFS service without warning, it is possible that users could lose data in currently opened files because writes that have been cached but not executed will be lost when CIFS service stops. The **cifs terminate** command includes provision to warn users and let them close their files.

cifs terminate has one optional argument, the name of a specific workstation. When you specify a workstation, the command terminates CIFS sessions for that workstation only. If you don't specify a workstation, the command will terminate all open CIFS sessions, and CIFS service will no longer be available to anyone.

The argument **-t mins** permits you to set the number of minutes before the shutdown of CIFS services will take effect. The system immediately sends a notice of the impending shutdown to all the affected workstations. (However, users of Windows 95 and Windows for Workgroups won't see the notification unless they're running **Win-Popup**). There's no guarantee that all users will see the notification, nor that all users will be able to close their open files, but it does provide for a warning where possible.

The shutdown takes effect at the end of the interval you've specified, or when CIFS clients have no open files, whichever comes sooner.

If you execute **cifs terminate** from an **rsh** session, you *must* use the **-t** option. If you enter the command without **-t** from the filer's console or via telnet *and* there are CIFS clients with open files, you'll get a message telling you the number of CIFS clients and the number of open files. Then the system will prompt you to enter the number of minutes until shutdown:

There are currently 23 CIFS users that have 44 open files.

Disconnecting while files are open may cause data loss!!

How many minutes should I wait? [5]:

If you used **cifs terminate** without limiting it to a specific workstation, when the command completes no further CIFS logins will be accepted and CIFS service will be disabled. To restart CIFS service, use the **cifs restart** command.

If you use **cifs terminate -t 0**, CIFS sessions will terminate immediately. A message will be logged, since files that were open may suffer data corruption.

SEE ALSO

na_halt(1), **na_reboot(1)**.

NAME

cifs testdc – test the filer’s connection to Windows NT domain controllers

SYNOPSIS

cifs testdc

DESCRIPTION

The **cifs testdc** command tests the filer’s ability to connect with Windows NT domain controllers. The output of the **cifs testdc** command is useful in the diagnosis of CIFS-related network problems.

EXAMPLE

```
purple> cifs testdc
Using Established configuration
Current Mode of NBT is H Mode

Netbios scope ""
Registered names...
PURPLE < 0> WINS Broadcast
PURPLE < 3> WINS Broadcast
PURPLE <20> WINS Broadcast
PURPLE-1 < 0> WINS Broadcast
PURPLE-1 < 3> WINS Broadcast
PURPLE-1 <20> WINS Broadcast
PURPLE-2 < 0> WINS Broadcast
PURPLE-2 < 3> WINS Broadcast
PURPLE-2 <20> WINS Broadcast
PURPLE-3 < 0> WINS Broadcast
PURPLE-3 < 3> WINS Broadcast
PURPLE-3 <20> WINS Broadcast
PURPLE-4 < 0> WINS Broadcast
PURPLE-4 < 3> WINS Broadcast
PURPLE-4 <20> WINS Broadcast
PURPLE-5 < 0> WINS Broadcast
PURPLE-5 < 3> WINS Broadcast
PURPLE-5 <20> WINS
PURPLE-6 < 0> WINS
PURPLE-6 < 3> WINS
PURPLE-6 <20> WINS
PURPLE-7 < 0> WINS
PURPLE-7 < 3> WINS
PURPLE-7 <20> WINS
PURPLE-8 < 0> WINS
```

```
PURPLE-8 < 3> WINS
PURPLE-8 <20> WINS
PURPLE-9 < 0> WINS
PURPLE-9 < 3> WINS
PURPLE-9 <20> WINS
NT-DOMAIN < 0> WINS
NT-DOMAIN < 3> WINS
NT-DOMAIN <20> WINS
```

Testing Primary Domain Controller

found 2 addresses

trying 192.168.2.14...192.168.2.14 is alive

trying 192.168.2.85...192.168.2.85 is alive

found PDC NT-DOMAIN-BDC

Testing all Domain Controllers

found 4 addresses

trying 192.168.2.14...192.168.2.14 is alive

trying 192.168.2.85...192.168.2.85 is alive

trying 198.95.227.75...198.95.227.75 is alive

trying 192.168.2.14...192.168.2.14 is alive

found DC NT-DOMAIN-BDC

found DC FRENCH40

found DC NT-DOMAIN-BDC

found DC FRENCH40

NAME

date – display or set date and time

SYNOPSIS

date [**-u**] [[[[[*cc*] *yy*] *mm*] *dd*] *hhmm* [*.ss*]]

DESCRIPTION

date displays the current date and time when invoked without arguments.

When invoked with an argument, **date** sets the current date and time; the argument for setting the date and time is interpreted as follows:

<i>cc</i>	First 2 digits of the year (e.g., 19 for 1999).
<i>yy</i>	Next 2 digits of year (e.g., 99 for 1999).
<i>mm</i>	Numeric month. A number from 01 to 12.
<i>dd</i>	Day, a number from 01 to 31.
<i>hh</i>	Hour, a number from 00 to 23.
<i>mm</i>	Minutes, a number from 00 to 59.
<i>ss</i>	Seconds, a number from 00 to 59.

If the first 2 digits of the year are omitted, they default to 19; if all 4 digits of the year are omitted, they default to the current year. If the month or day are omitted, they default to the current month and day, respectively. If the seconds are omitted, they default to 0.

Time changes for Daylight Saving and Standard time, and for leap seconds and years, are handled automatically.

OPTIONS

-u Display or set the date in GMT (universal time) instead of local time.

CLUSTER CONSIDERATIONS

You cannot use the **date** command in partner mode to set the date on the failed filer.

EXAMPLES

To set the current time to 21:00:

date 2100

To set the current time to 21:00, and the current day to the 6th of the current month:

date 062100

To set the current time to 21:00, and the current day to December 6th of the current year:

date 12062100

To set the current time to 21:00, and the current day to December 6th, 1999:

date 9912062100

To set the current time to 21:00, and the current day to December 6th, 2002:

date 200212062100**SEE ALSO**

na_partner(1), na_rdate(1), na_timezone(1)

NAME

df – display free disk space

SYNOPSIS

df [**-i**] [*pathname*]

DESCRIPTION

df displays statistics about the amount of free disk space in one or all volumes on the filer. All sizes are reported in 1024-byte blocks.

The *pathname* parameter is the path name to a volume. If it is specified, **df** reports only on the corresponding volume; otherwise, it reports on every on-line volume.

For each volume, **df** displays statistics about snapshots on a separate line from statistics about the active file system. The snapshot line reports the amount of space consumed by all the snapshots in the system. Blocks that are referenced by both the active file system and by one or more snapshots are counted only in the active file system line, not in the snapshot line.

If snapshots consume more space than has been reserved for them by the **snap reserve** command (see **na_snap(1)**), then the excess space consumed by snapshots is reported as used by the active file system as well as by snapshots. In this case, it may appear that more blocks have been used in total than are actually present in the file system.

With the **-i** option, **df** displays statistics on the number of free inodes.

EXAMPLES

The following example shows file system disk space usage:

```
toaster> df
Filesystem          kbytes  used    avail  capacity  Mounted on
/vol/vol0            4339168 1777824 2561344 41%       /vol/vol0
/vol/vol0/snapshot   1084788 956716  128072  88%       /vol/vol0/snapshot
```

If snapshots consume more than 100% of the space reserved for them, then either the snapshot reserve should be increased (using **snap reserve**) or else some of the snapshots should be deleted (using **snap delete**). After deleting some snapshots, it may make sense to alter the volume's snapshot schedule (using **snap schedule**) to reduce the number of snapshots that are kept on line.

The following example shows file system inode usage for a specified volume:

```
toaster> df -i /vol/vol0
Filesystem          iused    ifree    %iused  Mounted on
/vol/vol0            164591   14313    92%     /vol/vol0
```

You can increase the number of inodes in a file system at any time using the **maxfiles** command (see **maxfiles(1)**).

SEE ALSO

na_maxfiles(1), **na_rc(1)**, **na_snap(1)**

BUGS

On some NFS clients, the **df** command does not follow the NFS protocol specification correctly and may display incorrect information about the size of large file systems. Some versions report negative file system sizes; others report a maximum file system size of 2 GB, no matter how large the file system actually is.

NAME

disk – RAID disk configuration control commands

SYNOPSIS

disk fail *disk_name*

disk remove *disk_name*

disk scrub start

disk scrub stop

disk swap

disk unswap

DESCRIPTION

The **disk fail** command forces a file system disk to fail; the **disk remove** command unloads a spare disk so that you can physically remove the disk from the filer. The **disk scrub** command causes the filer to scan disks for media errors. If a media error is found, the filer tries to fix it by reconstructing the data from parity and rewriting the data. Both commands report status messages when the operation is initiated and return completion status when an operation has completed.

The filer’s “hot swap” capability allows removal or addition of disks to the system with minimal interruption to file system activity. Before you physically remove or add a SCSI disk, use the **disk swap** command to stall I/O activity. After you removed or added the disk, file system activity automatically continues. If you should type the **disk swap** command accidentally, or you choose not to swap a disk at this time, use **disk unswap** to cancel the swap operation and continue service.

If you want to remove or add a fibre channel disk, there is no need to enter the **disk swap** command.

Before you swap or remove a disk, it’s a good idea to run **syconfig -r** to verify which disks are where.

USAGE

disk swap applies to SCSI disks only. It stalls all I/O on the filer to allow a disk to be physically added or removed from a disk shelf. Typically, this command would be used to allow removal of a failed disk, or of a file system or spare disk that was prepared for removal using the **disk fail** or **disk remove** command. Once a disk is physically added or removed from a disk shelf, system I/O will automatically continue.

NOTE: It is important to issue the **disk swap** command only when you have a disk that you want to physically remove or add to a disk shelf, because all I/O will stall *until* a disk is added or removed from

the shelf.

disk unswap undoes a **disk swap** command, cancels the swap operation and continues service.

disk fail *disk_name*

removes the specified file system disk from the RAID configuration, spinning the disk down when removal is complete. **disk fail** is used to remove a file system disk that may be logging excessive errors and requires replacement.

Note that when a file system disk has been removed in this manner, the RAID group to which the disk belongs will enter degraded mode (meaning a disk is missing from the RAID group). If a spare disk at least as large as the disk being removed is available, the contents of the disk being removed will be reconstructed onto that spare disk.

The disk being removed is marked as “broken”, so that if it remains in the disk shelf, it will not be used by the filer as a spare disk, and if it is moved to another filer, it will not be used by that filer as a spare disk.

disk remove *disk_name*

removes the specified spare disk from the RAID configuration, spinning the disk down when removal is complete. You can use **disk remove** to remove a spare disk so that it can be used by another filer (as a replacement for a failed disk or to expand file system space).

disk scrub start starts a RAID scrubbing operation on all RAID groups. The **raid.scrub.enable** option is ignored; scrubbing will be started regardless of the setting of that option (the option is applicable only to scrubbing that gets started periodically by the system).

disk scrub stop stops a RAID scrubbing operation.

SEE ALSO

na_sysconfig(1)

NAME

download – install new version of Data ONTAP

SYNOPSIS

download

DESCRIPTION

download copies Data ONTAP executable files from the **/etc/boot** directory to the filer's boot block on the disks from which the filer boots.

To install a new version of Data ONTAP, extract the files for the new release onto the filer from either a CIFS or an NFS client that has write access to the filer's root directory. For more information about how to install files for the new release, see the upgrade instructions that accompany each release.

After the filer reboots, you can verify the version of the newly installed software with the **version** command.

CLUSTER CONSIDERATIONS

When the filers are not in takeover mode, the **download** command only applies to the filer on which you enter the command. When the filers are in takeover mode, you can enter the **download** command in partner mode on the live filer to download the Data ONTAP executable files from the partner's **/etc/boot** directory to the partner's disks. Under no circumstances does it work for you to enter the **download** command once to download the executable files to both filers in a cluster. Therefore, when you upgrade the software on a cluster, you must enter the **download** command on each filer after installing the system files on each filer. This way, both filers will reboot with the same Data ONTAP(tm) version.

FILES

/etc/boot	directory of Data ONTAP executables
/etc/boot/netapp_4.0-x86	executable for Data ONTAP 4.0 on filers with x86 processors
/etc/boot/netapp_4.0-alpha	executable for Data ONTAP 4.0 on filers with Alpha processors
/etc/boot/netapp-x86	symbolic link to current version of Data ONTAP for filers with x86 processors
/etc/boot/netapp-alpha	symbolic link to current version of Data ONTAP for filers with Alpha processors
/etc/boot/0-x86	first stage boot code and boot FCode for filers with x86 processors
/etc/boot/1-x86	second stage boot code for filers with x86 processors
/etc/boot/fc-hard-alpha	boot FCode for filers with Alpha processors
/etc/boot/1-alpha	second stage boot code for filers with Alpha processors

na_download(1)

User Commands

na_download(1)

SEE ALSO

na_partner(1), na_version(1), na_boot(5).

NAME

dump – file system backup

SYNOPSIS

dump [*options* [*arguments*]] *subtree*

DESCRIPTION

The **dump** command examines files in a subtree and writes to tape the files that need to be backed up. The Data ONTAP **dump** command differs slightly from the standard UNIX **dump**, but the output format is compatible with SunOS 4.x/Solaris 1.x and SunOS 5.x/Solaris 2.x **dump**.

Data ONTAP **dump** can write to its standard output (most useful with **rsh**(1) from a UNIX system), to a remote tape device on a host that supports the **rmt**(8) remote tape protocol or to a local tape drive, connected directly to the system (**na_tape**(4)).

The *subtree* argument specifies a subtree to be dumped. This is one way to allow **dump** to work with remote tape devices that are limited to 2 GB of data per tape file. The specified *subtree* may be in the active file system (e.g. **/home**) or in a snapshot (e.g. **/.snapshot/weekly.0/home**). If the subtree is in the active file system, **dump** creates a snapshot named **snapshot_for_dump.X** where X is a sequentially incrementing integer. This naming convention prevents conflicts between concurrently executing dumps. The dump is run on this snapshot so that its output will be consistent even if the filer is active. If **dump** does create a snapshot, it automatically deletes the snapshot when it completes.

Another way to allow **dump** to work with remote tape devices that are limited to 2 GB of data per tape file is to dump to multiple tape files or “volumes”. The **B** option to **dump** specifies the maximum amount of data to be dumped to one volume; when that much data has been written to one volume, **dump** will start writing to another volume. A list of tape devices can be specified as arguments to the **f** option, and the volumes will be written to the devices in that list, in order. If there are no more devices in the list, **dump** will re-use the last device in the list, after prompting the user to indicate that they’ve put a new tape in that device.

The Data ONTAP **dump** command handling of end of tape is slightly different than that of the standard UNIX dump. Instead of aborting the entire dump if EOT is reached, the Data ONTAP **dump** starts a new volume and continues the dump on the next tape. Specifying a large value for the **B** option will cause dump to utilize the entire tape.

You can enter the **dump** command on a trusted host through **rsh**. It is preferable to enter the **dump** command through **rsh** if the backup takes a significant amount of time. This is because if you enter the **dump** command on the console, the filer does not display the console prompt until the backup is finished. During the time when the filer is backing up data, you do not have console access to the filer.

Another advantage of running the **dump** command through **rsh** is that you can control backups from UNIX shell scripts or crontab entries.

OPTIONS

0-9 Dump levels. A level 0, full backup, guarantees the entire file system is copied. A level number above 0, incremental backup, tells dump to copy all files new or modified since the last dump of a lower level. The default level is 0.

f files Write the backup to the specified *files*. *files* may be:

a list of the names of local tape devices, in the form specified in **na_tape(4)**;

a list of the names of tape devices on a remote host, in the form *host:devices*;

the standard output of the dump command, specified as `-`.

The list may have a single device or a comma-separated list of devices; note that the list must either contain only local devices or only devices on a remote host and, in the latter case, must refer to devices on one particular remote host, e.g.

tapemachine:/dev/rst0,/dev/rst1

Each file in the list will be used for one dump volume in the order listed; if the dump requires more volumes than the number of names given, the last file name will be used for all remaining volumes after prompting for media changes.

Use **sysconfig -t** for a list of local tape devices. See below for an example of a dump to local tape.

For a dump to a tape device on a remote host, *host* must support the standard UNIX **rmt(8)** remote tape protocol.

By default, **dump** writes to standard output.

B blocks Set the size of the dump file to the specified number of 1024-byte blocks. If this amount is exceeded, **dump** will close the current file and open the next file in the list specified by the **f** option. If there are no more files in that list, **dump** will re-open the last file in the list, and prompt for a new tape to be loaded.

It is recommended to be a bit conservative on this option.

This is one way to allow **dump** to work with remote tape devices that are limited to 2 GB of data per tape file.

- u** Update the file `/etc/dumpdates` after a successful dump. The format of `/etc/dumpdates` is readable by people. It consists of one free format record per line: subtree, increment level and `ctime(3)` format dump date. There may be only one entry per subtree at each level. The dump date is defined as the creation date of the snapshot being dumped. The file `/etc/dumpdates` may be edited to change any of the fields, if necessary. See `na_dumpdates(5)` for details.
- b factor** Set the tape blocking factor in k-bytes. The default is 10 KB. If the density is set to greater than 6250 BPI, then the default blocking factor is 32 KB. **NOTE:** Some systems support blocking factors greater than 63KB by breaking requests into 63KB chunks or smaller using variable sized records; other systems do not support blocking factors greater than 63KB at all. When using large blocking factors, always check the system(s) where the potential **restore** may occur to ensure that blocking factor specified in **dump** is supported. Local tape devices restrict the blocking factor to less than, or equal to, 63KB.
- l** Specifies that this is a multi-subtree dump. The directory that is the common root of all the subtrees to be dumped must be specified as the last argument. The subtrees are specified by path names relative to this common root. The list of subtrees is provided from standard in, one item on each line, with a blank line to terminate the list. (If you use this option, you must also use option **n**.)
- n** Specifies the dumpname for a multi-subtree dump. Mandatory for multi-subtree dumps.
- Q** Backs up all files and directories in qtree 0 of the specified volume. Qtree 0 is a qtree that is not created by you with the **qtree** command. In each volume, the files and directories that do not belong to a qtree you create are considered to be in qtree 0. Follow the **Q** option with the path name of a volume (for example, `/vol/vol1`).

EXAMPLES

To make a level 0 dump of the entire file system to a remote tape device with each tape file in the dump being less than 2 GB in size, use:

```
toaster> dump 0ufbB adminhost:/dev/rst0 63 2097151 /
```

To make a level 0 dump of `/home` on a 2 GB tape to a remote tape device, use:

```
toaster> dump 0ufbB adminhost:/dev/rst0 63 2097151 /home
```

To make a level 0 dump of `/home` on a 2 GB tape to a local tape drive (no rewind device, unit zero, highest density) use:

```
toaster> dump 0ufbB nrst0a 63 2097151 /home
```

To make a level 0 dump of the entire file system to a local tape drive (no rewind device, unit zero, highest density), with each tape file in the dump being less than 2 GB in size, without operator intervention, using a tape stacker, with four tape files written per tape, assuming that the dump requires no more than 10GB, use:

```
toaster> dump 0ufbB nrst0a,nrst0a,nrst0a,urst0a,rst0a 63 2097151 /
```

This will:

write the first three files to the norewind device, so that they, and the next dump done after them, will appear consecutively on the tape;

write the next file to the unload/reload device. This will cause the stacker to rewind and unload the tape after the file has been written and then load the next tape.

write the last file to the rewind device, so that the tape will be rewound after the dump is complete.

To back up all files and directories in a volume named **engineering** that are not in a qtree you created, use:

```
toaster> dump 0ufQ rst0a /vol/engineering
```

To run the **dump** command through **rsh**, enter the following command on a trusted host:

```
adminhost# rsh toaster dump 0ufbB adminhost:/dev/rst0 63 2097151 /home
```

CLUSTER CONSIDERATIONS

In takeover mode, the failed filer does not have access to its tape devices. You can, however, back up the failed filer by entering the **dump** command in partner mode on the live filer. The **dump** command writes the data to the tape devices on the live filer.

FILES

/etc/dumpdates dump date record

SEE ALSO

na_partner(1), **na_quota(1)**, **na_rshd(8)**, **na_restore(1)**, **na_snap(1)**, **na_sysconfig(1)**, **na_tape(4)**, **na_dumpdates(5)**

BUGS

Deleting or renaming a snapshot that is currently being backed up is not supported and will lead to **dump** errors.

NOTES

Restore

As stated previously, filer **dump** output format is compatible with SunOS 4.x/Solaris 1.x and SunOS 5.x/Solaris 2.x **dump**. The filer supports a local **restore** command (see **na_restore(1)**), so the restoration process can be performed on the filer. It can also be performed via a **restore** done on an NFS client machine; if such a restore is being done, the client system should be checked to ensure it supports SunOS-compatible **dump/restore** format.

Client Dump and Restore Capability

If a client is to be used for performing filer dump and/or restore, it is important to check what the maximum dump and restore capabilities of your client system are before setting up a dump schedule. There are some client systems which do not support dump and restore of greater than 2 GB while others may support very large dumps and restores. It is especially important to check the **restore** capability of your system when using the filer local tape dump since the filer supports dumps that are greater than 2 GB.

Tape Capacity and Dump Scheduling

Along with the potential 2-GB restriction of **dump** or **restore** on a client system, it is important to consider your tape capacity when planning a dump schedule. For the filer local tape option, the Exabyte 8505 supports an approximate maximum capacity of 10GB per tape using compression. If a client system is used as the target for your dump, the capacity of that tape drive should be checked for dump planning.

If your filer file system exceeds the capacity of the local tape drive or the client system dump/restore, or you choose to dump multiple file system trees to parallelize the restore process with multiple tape drives, you must segment your dump to meet these restrictions.

One way to plan a dump schedule with a UNIX client system is to go to the root mount point of your filer and use the **du** command to obtain sizes of underlying sub-trees on your filer file system. Depending on the restrictions of your client's dump and restore capability or recording capacity of the tape device being used, you should specify a **dump** schedule that fits these restrictions. If you choose to segment your dump, the **norewind** device (see **na_tape(4)**) can be used to dump multiple tape files to one physical tape (again, choose a dump size which meets the criteria of your client restore and capacity of your tape drive).

The following example shows the **du** output from a filer file system on a client that supports dump and restore that are greater than 2 GB:

```
client% du -s *
4108  etc
21608 finance
```

5510100 home
3018520 marketing
6247100 news
3018328 users

You can use a tape device with approximately 10 GB on each tape to back up this filer. The dump schedule for this system can use the **norewind** tape device to dump the *marketing* and *news* subtrees to one tape volume, then load another tape and use the **norewind** tape device to dump *etc*, *finance*, *home* and *users* subtrees to that tape volume.

CIFS Data

The Data ONTAP **dump** command dumps the CIFS attributes and 8.3 name data for each file that is backed up. This data will *not* be backed up by a dump run on an NFS client machine. This data will *not* be restored by a restore run on an NFS client machine. This data will only be restored if a local restore is done of a backup created by the Data ONTAP **dump** command.

NAME

exportfs – export and unexport files or directories

SYNOPSIS

exportfs [**-aiuv**] [**-o options**] [*pathname*]

DESCRIPTION

If no *pathname* is specified, **exportfs** lists all currently exported directories and files. If *pathname* is specified, **exportfs** makes the specified file or directory available or unavailable for mounting by NFS clients.

OPTIONS

- a** Takes the list of path names to be exported or unexported from the */etc/exports* file. If you specify *pathname* in the command when using the **-a** option, the command ignores *pathname*.
- i** Ignores the options in the */etc/exports* file. Without the **-i** option, the **exportfs** command uses the options associated with the *pathname* specified in */etc/exports*.
- u** Unexports the specified path name. If you also include the **-a** option, the command unexports the path names in the */etc/exports* file and ignores *pathname*.
- v** Prints each path name as it is exported or unexported.
- o option**

Specifies a list of comma-separated options that describe how a file or directory is exported. You can specify the option in one of the following formats:

access=hostname[:hostname]...

Give mount access to each host listed. Alternatively, you can specify a netgroup instead of a host in the list. The netgroup must be defined in the */etc/netgroup* file. Whether the hosts can mount *pathname* with root access, read-and-write access, or read-only access depends on how you use the **root**, **rw**, and **ro** options, as described below.

anon=uid

If a request comes from user ID of 0 (root user ID on the client), use *uid* as the effective user ID unless the client host is included in the **root** option. The default value of *uid* is 65534. To disable root access, set *uid* to 65535. To grant root access to all clients, set *uid* to 0.

ro Export the *pathname* read-only. If you do not specify this option, the *pathname* is exported read-write.

rw=hostname[:hostname]...

Export the *pathname* read-only to all hosts not specified in the list and read-write to the hosts in the list. Netgroup names are not allowed in the list.

root=hostname[:hostname]...

Give root access only to the specified hosts. By default, no hosts are granted root access. Netgroup names are not allowed in the list.

When you export a file or directory using the **ro**, **rw**, or **root** option, you can specify that the file or directory be exported to a subnet instead of individual hosts. You cannot export to a subnet when using the **access** option.

Instead of specifying a host name or netgroup name in the **exportfs** command, specify the subnet in one of the following formats:

dotted_IP/num_bits The *dotted_IP* field is either an IP address or a subnet number. The *num_bits* field specifies the size of the subnet by the number of leading bits of the netmask.

“[**network**] *subnet* [**netmask**] *netmask*”

The *subnet* field is the subnet number. The *netmask* field is the netmask.

In UNIX, it is illegal to export a directory that has an exported ancestor in the same file system. Data ONTAP does not have this restriction. For example, you can export both the **/** directory and the **/home** directory. In determining permissions, the filer uses the longest matching prefix.

EXAMPLES

In the following example, all network clients can mount the **/home** directory but only the **adminhost** can mount the **/** directory:

```
exportfs -o access=adminhost,root=adminhost /home
exportfs /
```

The following examples show different forms of the **exportfs** command that export the **/home** directory to the 123.45.67.0 subnet with the 255.255.255.0 netmask:

```
exportfs -o rw=123.45.67.0/24 /home
exportfs -o rw=123.45.67/24 /home
exportfs -o rw="network 123.45.67.0 netmask 255.255.255.0"
exportfs -o rw="123.45.67.0 255.255.255.0"
```

FILES

/etc/exports	directories and files exported to NFS clients
/etc/hosts	host name data base
/etc/netgroup	network groups data base

SEE ALSO

na_exports(5), na_hosts(5), na_netgroup(5)

NOTES

Data ONTAP supports a maximum of 255 host names in each **rw** and **root** option. There is no limit on the number of host names in the list following the **access** option, but the maximum size of the **/etc/exports** file is about 64 KB.

NAME

halt – stop the filer

SYNOPSIS

halt [**-d**] [**-t mins**] [**-f**]

DESCRIPTION

halt flushes all cached data to disk, turns off the non-volatile RAM, and drops into the monitor. Any time you power-off the filer, you should first execute the **halt** command to conserve the batteries on the non-volatile RAM.

NFS clients can maintain use of a file over a **halt** or **reboot** (although experiencing a failure to respond during that time), but CIFS clients cannot do so safely. Therefore, if the filer is running CIFS, the **halt** command invokes **cifs terminate**, which requires the **-t** option. If the filer has CIFS clients and you invoke **halt** without **-t**, it displays the number of CIFS users and the number of open CIFS files. Then it prompts you for the number of minutes to delay. **cifs terminate** automatically notifies all CIFS clients that a CIFS shut-down is scheduled in *mins* minutes, and asks them to close their open files. CIFS files that are still open at the time the filer halts will lose writes that had been cached but not written.

halt logs a message in **/etc/messages** to indicate that the filer was halted on purpose.

OPTION

- d** Dumps system core before halting.
- t mins** Halts after the indicated number of minutes, or after all CIFS files that were open have been closed, whichever is sooner.
- f** Applies only to filers in a cluster. If you enter the **halt -f** command on a filer, its partner does not take over.

CLUSTER CONSIDERATIONS

After you enter the **halt** command on a filer in a cluster, the other filer in the cluster automatically takes over the filer that you halted. If you do not want takeover to happen, use the **halt -f** command.

The **halt** command is not available in partner mode. That is, you cannot enter the **partner halt** command on the live filer after it takes over the failed partner. This is because a filer that has been taken over is no longer running and cannot be halted.

SEE ALSO

na_cifs_terminate(1), **na_partner(1)**, **na_reboot(1)**, **na_savecore(1)**, **na_messages(5)**

NAME

help – print summary of commands and help strings

SYNOPSIS

help [*command ...*]

? [*command ...*]

DESCRIPTION

help prints a summary for each command in its argument list. With no arguments, **help** prints a list of all available Data ONTAP commands.

Full UNIX-style man pages for all filer commands and files are available in the **/etc/man** directory.

FILES

/etc/man directory of UNIX-style manual pages

NAME

hostname – set or display NetApp filer name

SYNOPSIS

hostname [*name*]

DESCRIPTION

hostname prints the name of the current host. The hostname can be set by supplying an argument. This is usually done in the initialization script, **/etc/rc**, which is run at boot time. *name* must exist in the **/etc/hosts** data base.

FILES

/etc/hosts	host name data base
/etc/rc	system initialization command script

SEE ALSO

na_hosts(5), **na_rc(5)**

NAME

httpstat - display HTTP statistics

SYNOPSIS

httpstat [**-tz**] [*interval*]

DESCRIPTION

httpstat displays statistical information about HTTP (HyperText Transfer Protocol) for the filer. It can also be used to reinitialize this information. If no arguments are given, **httpstat** displays statistical information since last reboot, or last zeroed with the **-z** option. If the **-t** option is specified, statistical information since the last reboot is given.

The output consists of the number of GET requests successfully processed (**gets**), rejected requests (**badcalls**), currently open HTTP connections (**open conn.**), and the maximum number of simultaneous connections (**peak conn.**).

If the *interval* argument is specified, **httpstat** will continuously display the summary information for all the statistics. The first line of data displayed contains cumulative statistics. Each subsequent line shows incremental statistics for the *interval* (in seconds) since the last display.

CLUSTER CONSIDERATIONS

In takeover mode, the HTTP statistics displayed reflect the sum of operations that take place on the live filer and the operations that the live filer performs on behalf of the failed filer. The display does not differentiate between the operations on the live filer's disks and the operations on the failed filer's disks.

The HTTP statistics are cumulative; a giveback does not zero out the HTTP statistics. After giving back the failed partner's resources, the live filer does not subtract the statistics about HTTP operations it performed on behalf of the failed filer in takeover mode.

SEE ALSO

na_netstat(1), **na_options(1)**, **na_partner(1)**, **na_sysstat(1)**.

NAME

ifconfig – configure network interface parameters

SYNOPSIS

```
ifconfig interface [ [ alias | -alias ] address ]
          [ netmask mask ] [ broadcast address ]
          [ mediatype type ] [ mtusize size ] [ up | down ]
          [ trusted | untrusted ] [ wins | -wins ]
          [ [ partner | -partner ] address ]
```

ifconfig -a

DESCRIPTION

ifconfig assigns an address to a network interface and configure network interface parameters. **ifconfig** must be used at boot time to define the network address of each network interface present on a machine; it may also be used at a later time to redefine a network interface's address or other operating parameters. When used without optional parameters, **ifconfig** displays the current configuration for a network interface.

The *interface* parameter is the name of the network interface. The name is of the form **en** for Ethernet interfaces, **fn** for FDDI interfaces, and **an** for ATM interfaces, possibly followed by a letter, where *n* is **0** for on-board network interfaces and the expansion slot number for network interfaces plugged into expansion slots. If a card in an expansion slot has more than one network interface, the network interface name will be followed by a letter, indicating which of the network interfaces on that card it is. The network interface name **vh** is used to specify IP virtual host addresses associated with the filer. Only alias addresses (using the *alias* option) may be assigned to the **vh** interface. The network interface name **-a** is special and it does not take any optional parameters. It displays the current configuration for all the network interfaces present.

The *address* is either a host name present in the host name data base **/etc/hosts** or an Internet address expressed in the Internet standard dot notation.

OPTIONS

- broadcast** *address* Specifies the address to use to represent broadcasts to the network. The default broadcast address is the address with a host part of all 1's.
- down** Marks a network interface "down". When a network interface is marked "down" the system will not attempt to transmit messages through that network interface. If possible, the network interface will be reset to disable reception as well. This action does not automatically disable routes using the network interface.
- mediatype** *type* Specifies the Ethernet media type used. Depending on the physical specifications of the Ethernet card the acceptable types are "thick"

(10Base5 AUI), “thin” (10Base2 BNC), “tp” (10Base-T RJ-45 twisted-pair), or “tp-fd” (Full duplex 10Base-T RJ-45 twisted-pair), or “100tx” (100Base-T RJ-45 twisted-pair), or “100tx-fd” (Full duplex 100Base-T RJ-45 twisted-pair), or “auto” (Auto RJ-45 twisted-pair). The default media type is set to “tp” or to “auto” where applicable.

On a 10/100 Mbps auto-negotiable interface, the system will auto-negotiate a 10 Mbps half or full duplex or 100 Mbps half or full duplex link and set the network interface accordingly when it is configured up. If the other end does not support auto-negotiation and full duplex operation is desired, it must be explicitly set using the **mediatype** command.

On a 10/100 Mbps interface, the system will auto-detect a 10 Mbps or 100 Mbps link and set the link speed accordingly when the network interface is configured up. The hardware is not currently capable of auto-detecting full duplex interfaces, so if full duplex operation is desired, it must be explicitly set using the **mediatype** command. Only the 10/100 Mbps interfaces are capable of full duplex operation.

- mtusize** *size* Specifies the MTU (maximum transmission unit) to use for the network interface. This is normally necessary only for non-compliant FDDI or ATM bridges, routers or switches that don't know how to fragment FDDI or ATM size packets.
- netmask** *mask* The mask includes the network part of the local address and the subnet part, which is taken from the host field of the address. The mask can be specified as a single hexadecimal number with a leading 0x, with a dot-notation Internet address, or with a pseudo-network name listed in the network table **/etc/networks**. The mask contains 1's for the bit positions in the 32-bit address that are to be used for the network and subnet parts, and 0's for the host part. The mask should contain at least the standard network portion, and the subnet field should be contiguous with the network portion. A default *netmask* is chosen according to the class of the IP address.
- up** Marks a network interface “up”. This may be used to enable a network interface after an “ifconfig down.” It happens automatically when setting the first address on a network interface. If the network interface was reset when previously marked down, the hardware will be re-initialized.
- alias** Establishes an additional network address for this network interface.

- This is sometimes useful when changing network numbers and one wishes to accept packets addressed to the old network interface. It is required when creating IP virtual host addresses.
- alias** Remove a network address for this network interface.
 - trusted** Specifies that the network to which the network interface is attached is trusted relative to firewall-style security (default).
 - untrusted** Specifies that the network to which the network interface is attached is not trusted relative to firewall-style security.
 - wins** Specifies that the network interface is to be registered with Windows Internet Name Services (default). Such registration is only performed when CIFS is running and at least one WINS server has been configured.
 - wins** Specifies that the network interface is not to be registered with Windows Internet Name Services.
 - partner *address*** Applies only to filers in a cluster. It maps a network interface to *address*, which is an IP address on the partner and is referred to as the partner IP address. In takeover mode, this network interface assumes the identity of the network interface on the Partner filer whose IP address is *address*. For example, **toaster1** and **toaster2** are filers in a cluster. If the IP address of **e8** on **toaster2** is 198.9.200.38, use the following command on **toaster1** if you want **e1** of **toaster1** to assume the identity of **e8** of **toaster2** for the duration of a takeover:
ifconfig e1 partner 198.9.200.38
 Be sure that both the local network interface and the partner's network interface are attached to the same network segment or network switch. Otherwise, after takeover, clients of the failed filer might need to wait an indeterminate amount of time for routing tables to flush before being able to access the data on the failed filer.
 - partner *address*** Applies only to filers in a cluster. It removes the mapping between a network interface and an IP address on the partner.

CLUSTER CONSIDERATIONS

On a filer in a cluster, a network interface performs one of these roles:

A dedicated network interface for the local filer whether or not the filer is in takeover mode. A network interface performs this role if it has a local IP address but not a partner IP address.

A shared network interface for both the local filer and the partner. That is, if the partner fails, the network interface assumes the identity of a network interface on the partner but works on behalf of both the live filer and the partner. A network interface performs this role if it has a local IP address and a partner IP address, which you assign by the **partner** option of the **ifconfig** command.

A standby network interface for the partner. That is, if the partner fails, the network interface works on behalf of the partner. When the filer is not in takeover mode, the network interface is idle. A network interface performs this role if it does not have a local IP address but does have a partner IP address, which you assign by the **partner** option of the **ifconfig** command.

The filer maps a partner IP address to a shared or standby interface when the filer initiates a takeover operation. In takeover mode, all requests destined for the partner IP address are serviced by the shared or standby interface. Also, in partner mode, if a command takes a network interface name as an argument, enter the network interface name of the failed filer. The command is executed on the shared or standby interface on the live filer. Similarly, in partner mode, a command for displaying network interface information displays the network interface name of the failed filer, even though the command is serviced by the shared or standby interface on the live filer.

In addition to assuming the partner IP address in takeover mode, a shared or standby interface assumes the media access control (MAC) address of the partner. This enables the clients of the partner to continue using their address resolution protocol (ARP) tables after the takeover.

These **ifconfig** options are not available in partner mode: **partner**, **-partner**, and **mtusize**.

EXAMPLES

A cluster contains two filers, **toaster1** and **toaster2**. **toaster1** takes over **toaster2** after **toaster2** fails.

The **/etc/rc** file on **toaster1** is as follows:

```
ifconfig e0 192.9.200.37
ifconfig e1 192.9.200.38 partner 192.9.200.41
ifconfig e2 partner 192.9.200.42
```

The **/etc/rc** file on **toaster2** is as follows:

```
ifconfig e7 192.9.200.42
ifconfig e8 192.9.200.41 partner 192.9.200.38
ifconfig e9 partner 192.9.200.37
```

The **e0** interface on **toaster1** is a dedicated interface. It services requests only for address 192.9.200.37. After **toaster1** takes over **toaster2**, this network interface is not available in partner mode.

The **e1** interface on **toaster1** is a shared interface. It services requests for address 192.9.200.38 when **toaster1** is not in takeover mode. When **toaster1** is in takeover mode, the network interface services requests for both addresses 192.9.200.38 and 192.9.200.41. When **toaster1** is in partner mode, this network interface shows up as the **e8** interface in commands that involve network interface names.

The **e2** interface on **toaster1** is a standby interface. It does not service any request when **toaster1** is not in takeover mode. However, after **toaster1** takes over **toaster2**, this network interface services requests for address 192.9.200.42. When **toaster1** is in partner mode, this network interface shows up as the **e7** interface in commands that involve network interface names.

FILES

/etc/hosts	host name data base
/etc/networks	network name data base

BUGS

You cannot configure gigabit Ethernet interfaces and ATM interfaces to be shared interfaces in a cluster. They can be configured as standby interfaces.

SEE ALSO

na_partner(1), **na_hosts(5)**, **na_networks(5)**

NAME

ifstat – display device-level statistics for network interfaces

SYNOPSIS

ifstat [**-z**] **-a** | *interface_name*

DESCRIPTION

The **ifstat** command displays statistics about packets received and sent on a specified network interface or on all network interfaces. The statistics are cumulative since the filer was booted.

The **-z** argument clears the statistics. The **-a** argument displays statistics for all network interfaces including the virtual host and the loopback address. If you don't use the **-a** argument, specify the name of a network interface.

CLUSTER CONSIDERATIONS

In takeover mode, the **ifstat** command displays combined statistics about packets processed by the local network interface and packets processed by the local network interface on behalf of the network interface on the failed filer.

The statistics displayed by the **ifstat** command are cumulative. That is, a giveback does not cause the **ifstat** command to zero out the statistics.

EXAMPLES

The following command displays network statistics for an Ethernet interface named **e7**:

```
ifstat e7
```

The following command displays network statistics for an FDDI interface named **f5**:

```
ifstat f5
```

The following command displays network statistics for the loopback address:

```
ifstat lo
```

The following command displays network statistics for all network interfaces on the filer:

```
ifstat -a
```

SEE ALSO

na_ifconfig(1) **na_partner(1)**.

NAME

license – license Data ONTAP services

SYNOPSIS

license [*service=code*] ...

DESCRIPTION

The **license** command enables you to enter license codes for specific Data ONTAP services. The license codes are provided by Network Appliance. With no arguments, the **license** command prints the current list of licensed services and their codes. It also shows the services that are not licensed for your filer.

The filer is shipped with license codes for all purchased services, so you need to enter the **license** command only after you purchase a new service or after you reinstall the file system.

To disable a license, enter the code **DISABLE**.

All license codes are case-insensitive. Do not leave a space before or after the equal sign in the command.

The following list describes the services you can license:

Enter **nfs** to enable NFS.

Enter **cifs** to enable CIFS.

Enter **http** to enable HTTP.

Enter **volcopy** to enable the **vol copy** commands.

Enter **cluster** to enable clusters.

EXAMPLES

The following example enables NFS:

```
toaster> license nfs=ABCDEFGF
nfs license enabled.
nfs enabled.
```

The following example disables CIFS:

```
toaster> license cifs=DISABLE
unlicense cifs.
cifs will be disabled upon reboot.
```

The following example lists all services that can be licensed:

```
toaster> license
cifs      site BCDEFGH
cluster   site IJKLMNO
http      not licensed
```

volcopy not licensed

CLUSTER CONSIDERATIONS

You must enable the licenses for the same Data ONTAP services on both filers in a cluster, or takeover does not function properly. When you enable or disable a license on one filer in a cluster, the filer reminds you to make the same change on its partner.

You can disable the cluster license only if both of the following conditions are true:

The filer is not in takeover mode.

You used the **cf disable** command to disable cluster failover.

SEE ALSO

na_partner(1)

NAME

logout – use control-D to logout

DESCRIPTION

The filer doesn't have a logout command. (Since the telnet connection and the console are multiplexed into the same session, there would be no way for a logout command to tell which connection to drop.) To log out, type control-D.

Over telnet, typing control-D disconnects the session.

On the console, typing control-D returns the console to the password prompt. If no password is set, control-D has no effect.

SEE ALSO

na_passwd(1)

NAME

maxfiles – increase the number of files the volume can hold

SYNOPSIS

maxfiles [*vol_name* [*max*]]

DESCRIPTION

maxfiles increases the number of files that a volume can hold to *max*. Once increased, the value of *max* can never be lowered, so the new value must be larger than the current value. If no argument is specified, **maxfiles** displays the current value of *max* for all volumes in the system. If just the *vol_name* argument is given, the current value of *max* for the specified volume is displayed.

Because each allowable file consumes disk space, and because the value of *max* can never be reduced, increasing *max* consumes disk space permanently. If **maxfiles** identifies a new size as unreasonably large, it will query the user to verify that the new value is correct.

The filer's **df** command (see **na_df(1)**) can be used to determine how many files have currently been created in the file system.

SEE ALSO

na_df(1)

NAME

mt – magnetic tape positioning and control

SYNOPSIS

mt [**-f** | **-t** *tapedevice*] *command* [*count*] [*command* [*count*] ...]

DESCRIPTION

mt is used to position or control the specified magnetic tape drive supporting the commands listed below. Commands that support a *count* field allow multiple operations to be performed (the rewind, status and offline commands do not support a count field). **mt** will output failure messages if the specified tape drive cannot be opened or if the operation fails.

The **-f** option specifies which tape device to use. Use **sysconfig -t** to list all tape devices on the filer. **-t** has the same effect as **-f**.

USAGE

eof, weof	Writes <i>count</i> end-of-filemarks beginning at the current position on tape.
fsf	Forward spaces over <i>count</i> filemarks. Positions the tape on the end-of-tape side of the filemark.
bsf	Backward spaces over <i>count</i> filemarks. Positions the tape on the beginning-of-tape side of the filemark.
fsr	Forward spaces <i>count</i> records. Positions the tape on the end-of-tape side of the record(s).
bsr	Backward spaces <i>count</i> records. Positions the tape on the beginning-of-tape side of the record(s).
erase	Erases the tape beginning at the current tape position. When the erase completes, the tape is positioned to beginning-of-tape.
rewind	Rewinds the tape, positioning the tape to beginning-of-tape.
status	Displays status information about the tape unit. If you have a tape device that Network Appliance has not qualified, you must use the following command syntax to access the tape device before the filer can register the tape device as a valid clone of a qualified device: mt -f device status After the filer accesses the tape device, you can use the sysconfig -t command to display information about the device emulation.
offline	Rewinds the tape and unloads tape media.
diag	Enables or disables display of diagnostic messages from tape driver. Enabling diagnostic messages can be helpful when attempting to

diagnose a problem with a tape device. Specifying a count of "1" enables display of diagnostic messages, a count of "0" disables diagnostic messages. Diagnostic messages are *disabled* by default.

eom Positions the tape to end of data (end of media if tape is full).

EXAMPLES

The following example uses **mt** to display status information for the no-rewind tape device, unit zero, highest format (density):

```
toaster> mt -f nrst0a status
Tape drive: Exabyte 8505 8mm
Status: ready, write enabled
Format: EXB-8500C (w/compression)
fileno = 0 blockno = 0 resid = 0
```

To skip over a previously created dump file to append a dump onto a no-rewind tape device, use the **fsf** (forward space file) command:

```
toaster> mt -f nrst0a fsf 1
```

CLUSTER CONSIDERATIONS

In takeover mode, the failed filer has no access to its tape devices. If you enter the **mt** command in partner mode, the command uses the tape devices on the live filer.

SEE ALSO

na_partner(1), **na_sysconfig(1)**, **na_tape(4)**.

NAME

ndmpd – manages NDMP service

SYNOPSIS

ndmpd [**on** | **off** | **status** | **probe** [*session*] | **kill** [*session*] | **version** [*version*]]

DESCRIPTION

The **ndmpd** command controls and displays information about the daemon responsible for providing Network Data Management Protocol service.

OPTIONS

- on** Enables NDMP request-handling by the daemon.
- off** Disables NDMP request-handling by the daemon. Processing continues for requests that are already in progress. New requests are rejected. By default, NDMP service is disabled at system startup.
- status** Displays the current state of NDMP service.
- probe** [*session*] Displays diagnostic information about the specified session. If *session* is not specified, information about all sessions is displayed.
- kill** [*session*] Signals the specified session to stop processing its current requests and move to an inactive state. If *session* is not specified, all active sessions are signaled. This allows hung sessions to be cleared without the need for a reboot, since the **off** command waits until all sessions are inactive before turning off the NDMP service.
- version** [*version*] Displays which NDMP versions are supported by the current software release. Can be limited to "version 1 only" by providing an argument. By default (after a reboot), version 1 and 2 are supported.

EXAMPLES

Check supported NDMP versions and set to v1 only:

```
ndmpd version
NDMP server supports v1 & v2.
```

```
ndmpd version 1
ndmpd version
NDMP server supports v1 only.
```

Check the status of all NDMP sessions:

```
ndmpd status
ndmpd ON.
Session: 0
  Active
  tape device:  nrst0m
  data state:   Idle
  data operation: None
  mover state:  Idle
Session: 1
  Inactive
Session: 2
  Inactive
Session: 3
  Inactive
Session: 4
  Inactive
```

The following should be added to the `/etc/rc` file to enable NDMP service at system startup.

```
ndmpd on
```

FILES

`/etc/rc` system initialization command script

SEE ALSO

`na_rc(5)`.

NAME

netstat - show network status

SYNOPSIS

netstat [**-an**]

netstat -mnrs

netstat -i | **-I interface** [**-dn**]

netstat -w interval [**-i** | **-I interface**] [**-dn**]

netstat [**-p protocol**]

DESCRIPTION

The **netstat** command symbolically displays the contents of various network-related data structures. There are a number of output formats, depending on the options for the information presented. The first form of the command displays a list of active sockets for each protocol.

The second form presents the contents of one of the other network data structures according to the option selected.

The third form will display cumulative statistics for all interfaces or, with an *interface* specified using the **-I** option, cumulative statistics for that interface. It will also display the sum of the cumulative statistics for all configured network interfaces.

The fourth form continuously displays information regarding packet traffic on the interface that was configured first, or with an *interface* specified using the **-I** option, packet traffic for that interface. It will also display the sum of the cumulative traffic information for all configured network interfaces.

The fifth form displays statistics about the protocol specified by *protocol*.

OPTIONS

- a** Show the state of all sockets; normally sockets used by server processes are not shown.
- d** With either interface display (option **-i** or an interval, as described below), show the number of dropped packets.
- I interface** Show information only about this interface. When used in the third form with an *interval* specified as described below, information about the indicated *interface* is highlighted in a separated column. (The default interface highlighted is the first interface configured into the system.)
- i** Show the state of interfaces which have been configured.
- m** Show statistics recorded by the memory management routines for the

- network's private pool of buffers.
- n** Show network addresses as numbers. **netstat** normally interprets addresses and attempts to display them symbolically. This option may be used with any of the display formats that display network addresses.
 - p protocol** Show statistics about *protocol*, which is one of **tcp**, **udp**, **ip**, or **icmp**. A null response typically means that there are no interesting numbers to report. The program will complain if *protocol* is unknown or if there is no statistics routine for it.
 - s** Show per-protocol statistics. If this option is repeated, counters with a value of zero are suppressed.
 - r** Show the routing tables. When **-s** is also present, show routing statistics instead.
 - w wait** Show network interface statistics at intervals of *wait* seconds.

DISPLAYS

The default display, for TCP sockets, shows the local and remote addresses, send window and send queue size (in bytes), receive window and receive queue sizes (in bytes), and the state of the connection. For UDP sockets, it shows the local and remote addresses, and the send and receive queue size (in bytes). Address formats are of the form "host.port" or "network.port" if a socket's address specifies a network but no specific host address. If known, the host and network addresses are displayed symbolically according to the data bases */etc/hosts* and */etc/networks*, respectively. If a symbolic name for an address is not known, or if the **-n** option is specified, the address is printed numerically, according to the address family. Unspecified, or "wildcard", addresses and ports appear as "*".

The interface display specified by the **-i** or **-I** options provides a table of cumulative statistics regarding packets transferred, errors, and collisions. The network addresses of the interface and the maximum transmission unit ("mtu") are also displayed. If the interface is currently down, then a "*" is appended to the interface name.

When an *interval* is specified, a summary of the interface information consisting of packets transferred, errors, and collisions is displayed.

The routing table display indicates the available routes and their status. Each route consists of a destination host or network and a gateway to use in forwarding packets. The flags field shows a collection of information about the route stored as binary choices; the flags are:

- 2** Protocol-specific routing flag #2 (for ARP entries, means that the entry is "published").

- C** Use of this route will cause a new route to be generated and used.
- D** The route was created dynamically by a redirect.
- G** The route is to a gateway.
- H** The route is to a host (otherwise, it's to a net).
- L** The route includes valid protocol to link address translation.
- M** The route was modified dynamically by a redirect.
- R** The route has timed out.
- S** The route was manually added with a **route** command (see **na_route(1)**).
- U** The route is usable (“up”).

Direct routes are created for each interface attached to the local host; the gateway field for such entries shows the address of the outgoing interface. The refcnt field gives the current number of active uses of the route. Connection oriented protocols normally hold on to a single route for the duration of a connection while connectionless protocols obtain a route whenever they transmit to a destination. The use field provides a count of the number of packets sent using that route. The interface entry indicates the network interface utilized for the route.

When **netstat** is invoked with the **-w** option and an *interval* argument, it displays a running count of statistics related to network interfaces. An obsolescent version of this option used a numeric parameter with no option, and is currently supported for backward compatibility. This display consists of a column for the primary interface and a column summarizing information for all interfaces. The default primary interface is the first interface configured into the system. The primary interface may be replaced with another interface with the **-I** option. The first line of each screen of information contains a summary since the system was last rebooted. Subsequent lines of output show values accumulated over the preceding interval.

CLUSTER CONSIDERATIONS

Each filer in a cluster maintains its own socket, routing, and interface tables. If a filer is not in partner mode, the **netstat** command displays the information in the tables on the live filer. If a filer is in partner mode, it executes the **netstat** command on behalf of the failed filer, which displays the information in the tables on the failed filer.

However, in takeover mode, counters displayed by the **netstat** command represent the combined statistics of the live filer and the failed filer. For example, from the statistics, you cannot determine how many packets were received on behalf of the live filer and how many packets were received on behalf of the failed filer.

In takeover mode, network interface names used by the failed filer are mapped to network interfaces on the live filer. When you enter the **netstat** command in partner mode, the network interface names displayed are the network interface names on the failed filer.

If you enter the **netstat** command in partner mode, you might see a plus sign (+) appended to some network interface names in the output. The plus sign indicates that the network interfaces are used as shared interfaces.

Statistics displayed by the **netstat** command are cumulative. That is, a giveback operation does not zero out the statistics. After giving back the virtual filer's resources, the live filer does not subtract the statistics about operations it performed on behalf of the failed filer in takeover mode.

FILES

/etc/hosts	host name data base
/etc/networks	network name data base

SEE ALSO

na_ifconfig(1), **na_nfsstat(1)**, **na_partner(1)**, **na_sysstat(1)**, **na_hosts(5)**,
na_networks(5)

NAME

nfs – turn NFS service off and on

SYNOPSIS

nfs [**on** | **off**]

DESCRIPTION

nfs turns NFS service off or on. With no arguments, **nfs** shows the current state of NFS service. **nfs** is normally used in the initialization command script, **/etc/rc**.

FILES

/etc/rc system initialization command script

SEE ALSO

na_rc(5)

NAME

nfsstat – display NFS statistics

SYNOPSIS

nfsstat [*interval*]

nfsstat [**-c**] [**-t**] **-h** [*ip_address* | *host_name*]

nfsstat **-l** [**-t**]

nfsstat **-z**

DESCRIPTION

nfsstat displays statistical information about NFS (Network File System) and RPC (Remote Procedure Call) for the filer. It can also be used to reinitialize this information. If no arguments are given, **nfsstat** displays statistical information since last zeroed with the **-z** option (or since reboot if statistics have not been zeroed).

If the *interval* argument is specified, **nfsstat** continuously displays the summary information for the following NFS requests: getattr, lookup, readlink, read, write, create, remove, and readdir/readdirplus. The first line of data displayed, and every 20th line thereafter, contains cumulative statistics. Each subsequent line shows incremental statistics for the *interval* (in seconds) since the last display.

Per-client statistics can also be collected and displayed by enabling the **nfs.per_client_stats.enable** options (using the **options** command - see **na_options(1)**) and invoking **nfsstat** with the **-h** or the **-l** options. Per-client statistics are collected for up to the first 256 NFS clients that have mounted the filesystem on the given filer.

OPTIONS

-h Displays per-client statistics since last zeroed with the **-z** option (or since reboot if statistics have not been zeroed). The statistics are displayed on a per-client basis, with the IP address and host name (where available) of each client being displayed at the head of each client's block of statistics.

If an optional IP address or host name is specified with the **-h** option, only the statistics associated with this client are displayed.

-l Displays a list of the clients whose statistics have been collected on a per-client basis, along with the total NFS calls for that client since last reboot, or last zeroed with the **-z** option, the count being displayed both as the actual count and as a percentage of calls from all clients.

-z Zeroes (reinitializes) the current statistics. (However, statistics since boot are also retained.)

-c Includes reply cache statistics in the data displayed.

-t Displays the statistics since boot time, rather than since the last time they

were zeroed.

DISPLAYS

The server RPC display includes the following fields, with separate values for TCP and UDP:

calls	The total number of RPC calls received.
badcalls	The total number of calls rejected by the RPC layer (the sum of badlen and xdr call as defined below).
nullrecv	The number of times an RPC call was not available when it was thought to be received.
badlen	The number of RPC calls with a length shorter than a minimum-sized RPC call.
xdr call	The number of RPC calls whose header could not be XDR decoded.

The server NFS display shows the number of NFS calls received (**calls**) and rejected (**badcalls**), and the counts and percentages for the various calls that were made.

BUGS

nfsstat -l reports unexpected percentages, if the **nfs.per_client_stats.enable** option is enabled after the system has been running for a while (typically this option should be enabled at system startup time via the **/etc/rc** file). Resetting the statistics via the **nfsstat -z** command will clear this condition.

nfsstat -h and **nfsstat -l** list some clients that have made no NFS calls to the filer. Also, **nfsstat -z** clears the statistics, but does not clear the list of clients (for the per-client statistics). The same clients will continue to be listed with the **nfsstat -h** or the **nfsstat -l** options even though they may have made no NFS calls after the statistics were cleared.

CLUSTER CONSIDERATIONS

In takeover mode, the **nfsstat** command displays combined statistics for the live filer and the failed filer. From the statistics, you cannot determine how many requests were serviced by the live filer and how many requests were serviced on behalf of the failed filer.

The **-h** and **-l** options display the combined client information for both the live filer and the failed filer.

The NFS statistics are cumulative. That is, a giveback operation does not zero out the NFS statistics. After giving back the failed filer's resources, the live filer does not subtract the statistics about NFS operations it performed on behalf of the failed filer when it was in takeover mode.

SEE ALSO

na_netstat(1), na_options(1), na_partner(1), na_sysstat(1).

NAME

options – display or set filer options

SYNOPSIS

options [*option value*] ...

DESCRIPTION

options is used to change configurable filer software options. If no options are specified, then **options** prints the current value of all available options. The default *value* for most options is **off**, which means that the option is not set. Changing the *value* to **on** enables the option; for most options, the only valid values are **on** (which can also be expressed as **yes**, **true**, or **1**) in any mixture of upper and lower case, and **off** (which can also be expressed as **no**, **false**, or **0**) in any mixture of upper and lower case. The description of the option will indicate the default if it is not **off**, and will indicate what values are allowed if it isn't an on/off option. If it is desired to make an option setting permanent, the necessary **options** command must be placed in the **/etc/rc** file, as options settings are not preserved across system reboots. The legal options are as follows:

autosupport.doit Triggers the autosupport daemon to send email notification immediately. A text word entered as the option is sent in the notification subject line and should be used to explain the reason for the notification.

autosupport.enable

Enables/disables the autosupport email notification feature (see **na_autosupport(8)**). The default is **on** to cause autosupport mail to be sent.

autosupport.from

Defines the user to be designated as the sender of the notification. The default is **autosupport**.

autosupport.mailhost

Defines the list of up to 5 mailhost names. The host names should be entered as a comma-separated list with no spaces in between. The default is an empty list.

autosupport.noteto

Defines the list of recipients for the autosupport short note email. Up to 5 mail addresses are allowed. Enter the addresses as a comma-separated list with no spaces in between. The default is a null list to disable short note emails.

autosupport.to

Defines the list of recipients for the autosupport email notification. Up to 5 mail addresses are allowed. Enter the addresses as a

comma-separated list with no spaces in between. The default is **autosupport@netapp.com**.

- cf.timed.enable** Enables the time daemon on a filer in a clustered pair. The time daemon synchronizes the time between the filers in the pair. Having synchronized times on the filers ensures that network clients continue to see consistent timestamps for their files after a filer in a cluster takes over its partner. This option is applicable only to filers with a cluster license. The default value of this option is **on**.
- cifs.generic_account** Enables a user who has no password account in **/etc/passwd** on the filer or in the NIS password database to get access to the filer, provided that the filer uses a Domain Controller for authentication and the user is in a trusted domain. If the option is set to the name of an account, the user gets the UNIX user ID, group ID, and group set of that account. If the option is blank, generic access is disabled.
- cifs.guest_account** Enables a user to get access to the filer provided that either the filer uses a Domain Controller for authentication and the user is not in a trusted domain, or the filer uses the **/etc/passwd** file or the NIS password database for authentication and the user has no entry in the **/etc/passwd** or the NIS password database. If this option is set to the name of an account in the password database, a user logging into the filer will be assigned to the guest account if their name is not listed in the password database (when using **/etc/passwd** or NIS) or if the user is not from a trusted domain (when using a domain controller). The configured user name will be used for the UNIX user ID, group ID, and group set of the specified account. If the option is blank, guest access is disabled.
- cifs.home_dir** When set to the pathname of a directory, this defines the path to the “homes directory”. The directories under this path should have the names of users as their names. When a user connects to the filer using CIFS and there is a directory name that exactly matches the user name, they will see a share of that name (truncated to 12 characters) that is their “home directory”. Only the user can access the home directory using this share. All other users are denied access.
- cifs.idle_timeout** Specifies the amount of idle time in milliseconds before the filer disconnects a session. An idle session is a session in which a user does not have any files opened on the filer. The value of this option ranges from 60,000 milliseconds to 4,000,000 milliseconds.

cifs.netbios_aliases

Provides a comma-separated list of alternative names for the filer. A user can connect to the filer using any of the listed names.

cifs.oplocks.enable

When **cifs.oplocks.enable** is on (the default), the filer allows clients to use oplocks (opportunistic locks) on files. Oplocks are a significant performance enhancement, but have the potential to cause lost cached data on some networks with impaired reliability or latency, particularly wide-area networks. In general, this option should be disabled only to isolate problems.

cifs.scopeid

NetBIOS scope IDs allow the system administrator to create small workgroups out of a network by partitioning the NetBIOS name space; only clients with the same NetBIOS scope ID as the filer will be able to use the filer as a CIFS server. Normally, the scope ID is a null string, but if the filer is to run in a NetBIOS scope other than the default one, its scope ID must be set to the scope ID of that scope. The scope ID can be changed only when CIFS is not running.

cifs.show_snapshot

By default this option is FALSE. The snapshot directory `~snapshot` is no longer shown at the root of a share. This is a change in behavior from previous versions. Setting this to TRUE will restore the old behavior. On Windows NT 4 or Windows 95 clients, the user can access snapshots by entering `\\filer\share\.snapshot` (or `~snapshot` or `~snapsht`) in the Start->Run menu. Snapshots can also be accessed lower in the share by providing a path to a lower directory. Snapshots can be accessed through DOS on any system by changing to the `~snapsht` directory.

cifs.symlinks.cycleguard

The **cifs.symlinks.cycleguard** option (on by default), eliminates the possibility of traversing directories cyclically during the following of symbolic links. With this option set to on, the filer does not follow symbolic links that contain the "dot" (".") or "dot-dot" ("..") component -- symbolic links that could refer to a directory higher in the same tree. With this option set to off many standard windows apps (such as Find in Win95 / NT4.0) will not operate correctly when a symlink points to a parent directory. This is because they do not understand symbolic links and will repeatedly loop on them. Users should use caution when changing this option.

cifs.symlinks.enable

When **cifs.symlinks.enable** is on (the default), if the object being accessed by a CIFS client is a symbolic link (whether absolute or relative), the filer follows the link with the proviso that the ultimate target turns out to reside within the originating share (thus ensuring that the client has access permission to the target).

dns.domainname

Sets the DNS domainname to the specified domainname.

dns.enable

Enables DNS client on the filer. The DNS domain must be set and the **/etc/resolv.conf** file must exist prior to enabling DNS.

httpd.enable

Enables HTTP access to the filer.

httpd.admin.enable

Enables HTTP access to the administration area of the filer, via a private Network Appliance URL: any URL beginning with **/na_admin** is mapped to the directory **/etc/http**. Thus, a man page on the filer *toaster* with the file name **/etc/http/man/name** can be accessed with the URL **http://toaster/na_admin/man/name**.

httpd.log.max_file_size

Specifies the maximum size that the HTTP log file **/etc/log/httpd.log** can grow to. The default is 2147483647, which is the largest file size that many clients support.

httpd.rootdir

Specifies the complete pathname of the root directory that contains files and subdirectories for HTTP access.

httpd.timeout

Specifies the minimum amount of time (in seconds) before an idle HTTP connection will time out. The default is 900 seconds, which is fifteen minutes.

httpd.timewait.enable

When enabled, the filer will put HTTP connections which have been closed by the client into the **TIME_WAIT** state for one minute, which is twice the maximum segment lifetime (2*MSL). By default, **TIME_WAIT** state is bypassed for HTTP connections.

ip.path_mtu_discovery.enable

Enables/disables path MTU discovery; it is currently used only by TCP. Path MTU discovery, described in RFC 1191, allows a host to discover the “maximum transmission unit”, i.e. the largest link-level packet that can be transmitted, over a path from that host to another host. This means that the filer needn’t choose a conservative packet size for a TCP connection to a host not on the same net as

the filer, but can attempt to discover the largest packet size that can make it to the other host without fragmentation.

nfs.mount_rootonly

When enabled, the mount server will deny the request if the client is not root user using privileged ports. By default, the feature is enabled for more secure access.

nfs.per_client_stats.enable

Enables/disables the collection and display of per-client NFS statistics as described in **na_nfsstat(1)**.

nfs.tcp.enable

When enabled, the NFS server supports NFS over TCP. By default, the feature is enabled; it can be disabled if there is a problem with some client when using NFS over TCP, and that client cannot be configured to use NFS over UDP.

nfs.v2.df_2gb_lim

Causes the filer to return replies to the "file system statistics" NFS version 2 request that show no more than $2^{31} - 1$ (or 2,147,483,647) total, free, or available bytes (i.e., 2GB) on the file system.

Some NFS clients require this option because, if they get return values from the "file system statistics" request with more than the specified number of bytes, they'll incorrectly compute the amount of free space on the file system, and may think that there's no free space on a file system that has more than 2GB free.

nfs.v3.enable

When enabled, the NFS server supports NFS version 3. By default, the feature is enabled; it can be disabled if there is a problem with some client when using NFS version 3, and that client cannot be configured to use NFS version 2.

nfs.webnfs.enable

When enabled, the NFS server supports WebNFS lookups. By default, WebNFS lookups are disabled.

nfs.webnfs.rootdir

Specifies the WebNFS rootdir. Once the rootdir is set, WebNFS clients can issue lookups relative to the rootdir using the public filehandle.

nfs.webnfs.rootdir.set

After specifying the rootdir, this option needs to be enabled for the rootdir setting to take effect. Disabling this option disables the existing rootdir setting.

- nis.domainname** Sets the NIS domain to the specified domainname.
- nis.enable** Enables NIS client on the filer. The NIS domain must be set prior to enabling NIS.
- pcnfsd.enable** Enables/disables the PCNFS (PC)NFS authentication request server (see **na_pcnfsd(8)**). The default is off.
- pcnfsd.umask** Specifies the default umask for files created by (PC)NFS clients. The value of this option is a three-digit octal number, and the digits correspond to the read, write, and execute permissions for owner, group, and other, respectively. The default value is 022, which means that files normally created with mode 666 effectively will have mode 644. ('644' means that the file owner has read and write permissions, but the members of the group and others have only read permission.)
- raid.reconstruct_speed** Specifies the speed at which the RAID reconstruction should occur ranging from the slowest speed **1** to the fastest speed possible **10**. The RAID reconstruction process is given more cpu time as the speed is increased, so increasing the speed of the reconstruction will take away cpu time for network operations. The default speed is **4** which is roughly 40% of the cpu time, though more time may be used if there is idle time available.
- raid.scrub.enable** Enables/disables the RAID scrub feature (see **na_disk(1)**). By default, it is enabled. This option only affects the scrubbing process that gets started from cron. For user requested scrubs, this option is ignored.
- raid.timeout** Set the time in hours, as a number greater than or equal to **1**, that the system will run after a single disk failure has caused the system to go into *degraded mode*. The default is **24**. If the **raid.timeout** option is specified after the system is already in *degraded mode*, the timeout is set to the value specified and the timeout restarted.
- telnet.hosts** Specifies up to 5 clients that will be allowed telnet access to the server. The host names should be entered as a comma-separated list with no spaces in between. Enter a "*" to allow access to all clients; this is the default. Enter a "-" to disable telnet access to the server.
- vol.copy.throttle** Specifies the default speed of all volume copy operations in tenths of full speed. The speed can be a number in the range from 1 to 10, 10 being the highest speed and the default.

waf.maxdirsize Sets the maximum size (in K-Bytes) that a directory can grow to. This is set to 10240 by default; it limits directory size to 10M-Bytes and can hold over 300,000 files. Most users should not need to change this setting. This option is useful for environments where system users may grow a directory to a size that starts impacting system performance. When a user tries to create a file in a directory that is at the limit the system returns a ENOSPC error and fails the create.

waf.root_only_chown

When enabled, only the root user can change the owner of a file. When disabled, non-root users can change the owner of files that they own. When a non-root user changes the owner of a file they own, both the set-UID and set-GID bits of that file are cleared for security reasons. A non-root user is not allowed to give away a file if it would make the recipient overrun its user quota. **waf.root_only_chown** is enabled by default.

Multiple options can be set at once in an **options** command. For example:

```
toaster> options nfs.tcp.enable on nfs.v2.df_2gb_lim on raid.timeout 48
```

sets **nfs.tcp.enable** to **on**, sets **nfs.v2.df_2gb_lim** to **on**, and sets **raid.timeout** to **48**.

CLUSTER CONSIDERATIONS

In general, each filer in a cluster has its own options that are independent of the options of its partner. After a takeover, for most options, the live filer and the virtual filer each uses its own option settings.

The more settings are the same on both filers, the more transparent the takeover. However, a few options must have the same setting on both filers in a cluster for takeover to work properly. This is because in a takeover, the option values for these options on the live filer are used for both filers.

If you change the setting for one of these options on one filer, the filer displays a message reminding you to make the same change on the other filer.

The following list of options must have the same value on both filers in a cluster:

- cf.timed.enable
- cifs.show_snapshot
- dns.domainname
- dns.enable
- httpd.admin.enable
- httpd.timeout
- httpd.timewait.enable
- ip.path_mtu_discovery.enable

```
nfs.per_client_stats.enable
nfs.v2.df_2gb_lim
nfs.v3.enable
nis.domainname
nis.enable
pcnfsd.enable
raid.timeout
raid.reconstruct_speed
vol.copy.throttle
waf.maxdirsize
waf.root_only_chown
```

After takeover, you can use the **options** command in partner mode to modify an option setting for the failed filer. However, the change is lost after the giveback operation.

SEE ALSO

na_disk(1), **na_nfsstat(1)**, **na_partner(1)**, **na_snap(1)**, **na_autosupport(8)**, **na_pcnfsd(8)**.

BUGS

A perfect appliance would need no options (other than, perhaps, a darkness adjustment knob). However, user Nigel Tuffnell reports he likes the **raid.reconstruct_speed** knob that goes from **1** to **10**, but he requests a future enhancement to "go to **11** because it's one better, isn't it?"

NAME

partner – access the data on the partner in takeover mode

SYNOPSIS

partner [*command*]

DESCRIPTION

When one filer in a cluster fails, the other filer takes over. To execute a Data ONTAP command on the failed filer, use the **partner** command on the console of the filer that took over. When a filer executes a command on behalf of its partner, the filer is said to be in ‘partner mode.’

If the filer is currently not in partner mode, the **partner** command without arguments causes the filer to enter partner mode. After the filer enters partner mode, the filer executes each subsequent command on behalf of the partner until you enter the **partner** command again.

Alternatively, you can type **partner** followed by a command. The filer enters partner mode and executes that command on behalf of the partner. After the filer finishes executing the command, it exits partner mode.

In partner mode, you can enter any Data ONTAP command to manage the failed filer, except that you cannot perform the following tasks:

- Halt or reboot the failed filer.
- Set the date on the failed filer.
- Synchronize the date on the failed filer with a time server.
- Set the time zone on the failed filer.
- Initiate a giveback or takeover.
- Disable takeover.
- Unlicense the cluster feature.
- Change the partner interface mapping.
- Change the MTU (maximum transmission unit) to use for a network interface.
- Revert the Data ONTAP(tm) software version to an earlier version.

When in partner mode, the filer changes the console prompt to display the host name of the failed filer, followed by a slash and the host name of the live filer.

After a filer in a cluster fails, it stops running the **telnet** daemon. As a result, you cannot establish a **telnet** session with the failed filer. You can, however, establish a **telnet** session to the filer that has taken over, and enter the **partner** command in the same way as you would on the console.

The **partner** command is available only if your filer has the cluster license.

EXAMPLES

Suppose a cluster contains two filers named **toaster1** and **toaster2**. After **toaster2** fails, **toaster1** takes over. Because you can no longer enter commands on the console of **toaster2**, you must use the **partner** command on **toaster1** to access **toaster2**. For example, to determine the maximum number of files on **toaster2**, enter the following command on **toaster1**:

```
toaster1(takeover)> partner maxfiles
Volume vol0: maximum number of files is currently 241954 (3194 used).
Volume vol1: maximum number of files is currently 241954 (3195 used).
toaster1(takeover)>
```

The following example illustrates how the console prompt on **toaster1** changes after you enter the **partner** command on **toaster1**. The example also shows how to exit partner mode:

```
toaster1(takeover)> partner
toaster2/toaster1> maxfiles
Volume vol0: maximum number of files is currently 241954 (3194 used).
Volume vol1: maximum number of files is currently 241954 (3195 used).
toaster2/toaster1> partner
toaster1(takeover)>
```

The following example shows that the **partner** command toggles between the consoles of two filers in a cluster:

```
toaster1(takeover)> partner partner
toaster1(takeover)>
```

In this example, the first **partner** command causes **toaster1** to execute the second **partner** command on **toaster2**, which returns you to **toaster1**'s console. Consequently, the usual console prompt of **toaster1** is displayed.

NAME

passwd – modify the system password

SYNOPSIS

passwd

DESCRIPTION

passwd changes the filer's password. First it prompts you for the current password. If you type the current password correctly, the filer requests a new password. You must enter the new password twice to avoid typing errors.

The **passwd** command imposes no minimum length or special character requirements. As with any password, it is best to choose a password unlikely to be guessed by an intruder.

If the filer is booted from floppy disk, selection "(3) Change password" enables you to reset the password without entering the old password. This is useful for the forgetful.

CLUSTER CONSIDERATIONS

Each filer in a cluster can have a different password. However, in takeover mode, use only the password set on the live filer to access the consoles of both filers. You do not need to enter the failed filer's password to execute commands in partner mode.

Because the password for the failed filer becomes unnecessary after a takeover, you do not have increased security by assigning different passwords to the filers in a cluster. Network Appliance recommends that you use the same password for both filers.

SEE ALSO

na_partner(1)

NAME

ping – send ICMP ECHO_REQUEST packets to network hosts

SYNOPSIS

ping [*-s*] [*-Rrv*] *host* [*packetsize* [*count*]]

DESCRIPTION

ping uses the ICMP protocol's mandatory ECHO_REQUEST datagram to elicit an ICMP ECHO_RESPONSE from the specified *host*, or gateway. ECHO_REQUEST datagrams have an IP and ICMP header, followed by a *struct timeval* and then an arbitrary number of bytes used to fill out the packet. If *host* responds, **ping** prints “*host* is alive.” Otherwise, **ping** will resend the ECHO_REQUEST once a second. If the *host* does not respond after *count* seconds (default value is 20), **ping** will print “no answer from *host*.”

When the *-s* flag is specified, **ping** sends one datagram per second and prints one line of output for every ECHO_RESPONSE that it receives. **ping** computes the round-trip times and packet loss statistics. When the *count* number of packets have been sent or if the command is terminated with a ^C, the summary statistics is displayed. The default *packetsize* is 56, which translates into 64 ICMP bytes when combined with the 8 bytes of ICMP header.

OPTIONS

- R** Record route. Includes the RECORD_ROUTE option in the ECHO_REQUEST packet and displays the route buffer on returned packets. Note that the IP header is only large enough for nine such routes. Many hosts ignore or discard this option.
- r** Bypass the normal routing tables and send directly to a host on an attached network. If the host is not on a directly-attached network, an error is returned.
- s** Send one datagram every second.
- v** Verbose output. ICMP packets other than ECHO_RESPONSE that are received are listed.

SEE ALSO

na_ifconfig(1), **na_netstat(1)**

NAME

qtree – create and manage qtrees

SYNOPSIS**qtree**

qtree create [*name*]

qtree security [*name* [**unix** | **ntfs** | **mixed**]]

qtree oplocks [*name* [**enable** | **disable**]]

DESCRIPTION

The **qtree** command creates qtrees and specifies attributes for qtrees.

A qtree can be an entire volume or a subset of a volume. It is similar to a partition in that you cannot move files into or out of a qtree. There are, however, two differences between a qtree and a partition:

A qtree is more flexible than a partition because you can change the size of a qtree at any time.

A qtree enables you to apply attributes such as oplocks and security style to a subset of files and directories rather than to an entire volume.

If there are files and directories in a volume that do not belong to any qtrees you create, the filer considers them to be in qtree 0. Qtree 0 can take on the same types of attributes as any other qtrees.

You can use any **qtree** command whether or not quotas are enabled on your filer.

The **qtree** command without any arguments displays the attributes of all quota trees on the filer.

The **qtree create** command creates a qtree. It is equivalent to the **quota qtree** command. If *name* does not begin with a slash (/), the qtree is created in the root volume. To create a qtree in a particular volume, specify *name* in this format: */vol/vol_name/qtree_name*.

A qtree can be created only in the root directory of a volume. By default, a qtree has the same security style as the root directory of the volume and oplocks are enabled. The root directory of a volume, by default, uses the unix security style.

A qtree does not have any restrictions on disk space or the number of files. To impose these restrictions on a qtree, edit the */etc/quotas* file. Refer to the **na_quotas(5)** man page for more information about the file format. To make the changes to the */etc/quotas* file go into effect, use the **quota** command. Refer to the **na_quota(1)** man page for more information about the **quota** command.

If you enter the **qtree create** command without arguments, the command displays all existing qtrees and their attributes.

To delete a qtree, remove it from a client as you would any directory. You can create up to 254 qtrees on a filer.

The **qtree security** command changes the security style for files and directories. Security style means the method the filer uses to determine whether a user has access to a file. If *name* is the path name to a qtree, the security style applies to the files and directories in the specified qtree. The path name to a qtree does not need to end with a slash. If *name* is a path name to a volume, the security style applies to those directories and files in qtree 0. Any new qtree you create inherits the security style from qtree 0 by default. The path name to a volume must end with a slash.

The security style can be one of the following values:

unix The user's UID and GID, and the UNIX-style permission bits of the file or directory determine user access. The filer uses the same method for determining access for both NFS and CIFS requests. If you change the security style of a qtree or a volume from ntfs to unix, the filer disregards the Windows NT permissions that were established when the qtree or volume used the ntfs security style.

ntfs For CIFS requests, Windows NT permissions determine user access. For NFS requests, the filer generates and stores a set of UNIX-style permission bits that are at least as restrictive as the Windows NT permissions. The filer grants NFS access only if the UNIX-style permission bits allow the user access.

If you change the security style of a qtree or a volume from unix to ntfs, files created before the change do not have Windows NT permissions. For these files, the filer uses only the UNIX-style permission bits to determine access.

mixed Some files in the qtree or volume have the unix security style, and some have the ntfs security style. A file's security style depends on whether the permission was last set from CIFS or NFS. For example, if a file currently uses the unix security style and a CIFS user sends a set-ACL request to the file, the file's security style is changed to ntfs. If a file currently uses the ntfs style and an NFS user sends a set-permission request to the file, the file's security style is changed to unix.

If you do not specify unix, ntfs, or mixed in the **qtree security** command, the security style for *name* is displayed. If you omit *name*, the security styles for all qtrees on the filer are displayed.

The **qtree oplocks** command enables or disables oplocks for files and directories in a qtree or in a volume. If *name* is the path name to a qtree, the attribute applies to files and directories in the specified qtree. The path name to a quota tree does not need to end with a slash. If *name* is the path name to a volume, the attribute applies to those files and directories in qtree 0. The path name to a volume must end with a slash.

If the **cifs.oplocks.enable** option is off, oplocks are not sent even if you enable the oplocks on a per-quota-tree basis with the **qtree oplocks** command. The **cifs.oplocks.enable** option is enabled by default.

If you do not specify enable or disable in the **qtree oplocks** command, the oplock attribute for *name* is displayed. If you omit *name*, the oplock attributes for all quota trees on the filer are displayed.

EXAMPLES

The following example sets the security style of a qtree named marketing in the root volume to ntfs:

```
toaster> qtree security marketing ntfs
```

The following example sets the security style of a qtree named engineering in the vol1 volume to ntfs:

```
toaster> qtree security /vol/vol1/engr ntfs
```

The following example sets the security style of the root volume to unix:

```
toaster> qtree security / unix
```

The following example sets the security style of the vol1 volume to unix:

```
toaster> qtree security /vol/vol1/ unix
```

The following example disables oplocks for the engr qtree:

```
toaster> qtree oplocks /vol/vol1/engr disable
```

The following example enables oplocks for the vol1 volume:

```
toaster> qtree oplocks /vol/vol1/ disable
```

The following example displays the security and oplocks attributes for all volumes and qtrees on the filer:

```
toaster> qtree
Volume      Tree      Style      Oplocks
-----
vol0                unix      enabled
vol0      marketing ntfs      enabled
vol1                unix      enabled
vol1      engr      ntfs      disabled
```

SEE ALSO

na_options(1), na_quota(1), na_quotas(5)

NAME

quota – control filer disk quotas

SYNOPSIS

quota [**on** | **off** | **resize**] [*volume*]

quota report [*path*]

quota qtree [*name*]

DESCRIPTION

A quota limits the amount of disk space and the number of files that a particular user or group can consume. A quota can also restrict the total space and files used in a qtree, or the usage of users and groups within a qtree. A request that would cause a user or group to exceed an applicable quota fails with a “disk quota exceeded” error. A request that would cause the number of blocks or files in a qtree to exceed the qtree’s limit fails with an “out of disk space” error.

User and group quotas do not apply to the root user; tree quotas, however, do apply even to root.

The **quota** command controls quotas, and the */etc/quotas* file describes the quotas to impose. All quotas are established on a per-volume basis. For further information on the format of the */etc/quotas* file, refer to the **na_quotas(5)** man page.

With no arguments, the **quota** command indicates whether quotas are on or off in each volume. The following list describes how to use the various **quota** commands:

quota on *volume* activates quotas in the specified volume based on the contents of */etc/quotas*. The volume name may be omitted if the system has only one volume. Changing */etc/quotas* has no effect until the next time **quota on** or **quota resize** is executed. The filer remembers whether quotas are on or off even after a reboot, so **quota on** should *not* be added to */etc/rc*. When quotas are first turned on, the filer scans the file system to determine current file and space usage for each user and group with a quota. This may take several minutes during which quotas are not in effect, although the file system is still accessible. Executing **quota** with no arguments during this period indicates that quotas are initializing and reports how much of the initialization process has completed.

quota off *volume* turns quotas off on the specified volume. The volume name may be omitted if the system has only one volume.

quota resize *volume* adjusts currently active quotas in the specified volume to reflect changes in the */etc/quotas* file. For instance, if you edit an entry

in `/etc/quotas` to increase a user's quota, **quota resize** will cause the change to take effect. The volume name may be omitted if the system has only one volume. **quota resize** can be used only when quotas are already on. Because it does not rescan the file system to compute usage, **quota resize** is faster than turning quotas off and then on again. **quota resize** will apply all updated entries in `/etc/quotas`; however, it will generally ignore newly added entries. A newly added entry will only take effect if the corresponding user or group has an active quota as a result of updating a file subject to default quotas.

quota report prints the current file and space consumption for each user or group with a quota and for each qtree. With a *path* argument, **quota report** displays information about all quotas that apply to the file.

quota qtree name creates a qtree. This command is equivalent to the **qtree create** command. Refer to the **na_qtree(1)** man page for further information about how to create a qtree.

FILES

`/etc/quotas` quota configuration file

SEE ALSO

na_rc(5), **na_rquotad(8)**, **na_qtree(1)**

DIAGNOSTICS

If `/etc/quotas` is incorrectly formatted, or if a specified file doesn't exist, then **quota on** prints a warning and does not turn quotas on.

BUGS

Quotas do not count space consumed by snapshots. Doing so could put users into a state where they couldn't create any new files until all snapshots containing their old files expired, which could take a week or more. That seems like a worse bug.

It is possible for a quota to exceed its limit if files were created before quotas were turned on or during quota initialization.

NAME

raid – raid configuration control commands

SYNOPSIS

raid *command*

DESCRIPTION

The **raid** command supports modification of disk devices in the RAID configuration. The **raid** command supports removal of spare disks, addition of new file system member disks and forced failure of file system disks from a RAID configuration. All commands report status messages when the operation is initiated and will return completion status when an operation has completed.

FAServer 1400 systems support a "hot swap" capability which allows removal or addition of disks to the system with minimal interruption to file system activity. The **raid swap** command is issued prior to physically removing or adding a disk from a FAServer 1400 system to stall I/O activity, once the disk is removed or added, file system activity will automatically continue. If the **raid swap** command is accidentally typed or you choose not to swap a disk at this time, the **raid unswap** command can be issued to cancel the swap operation and continue NFS service.

USAGE

add *disk_id* will add the specified spare disk to the existing RAID configuration for use by the file system. The *disk_id* specified should be a spare disk in the current configuration. Use **sysconfig -r** to check for the existence of spare disks. For systems that support hot swap, a spare disk may be swapped into the system by using **raid swap** at any time and then added to the file system using the **raid add** command. For systems without hot swap support, a spare drive in the existing configuration can be used and added to the file system. Note that in both cases, the disk will be zeroed before being added to the file system and a completion status message will be output once the addition has completed.

swap will stall all I/O on systems which support hot swap to allow a disk to be physically added or removed from a disk shelf. Typically, this command would be used to allow removal of a failed disk or a spare or filesystem disk which was prepared for removal using the **raid remove** or **raid fail** commands, respectively. Once a disk is physically added or removed from a disk shelf, system I/O will automatically continue. **NOTE:** it is important to only issue the **raid swap** command when you have a disk drive that you want to physically remove or add to a disk shelf because all I/O will stall *until* a disk is added or removed from the shelf.

unswap will undo a **raid swap** command, cancel the swap operation and

continue NFS service.

remove *disk_id*

will remove the specified spare disk from the RAID configuration, spinning the disk down when removal is complete. **raid remove** is useful to allow removal of a spare disk for use in another file system (as a replacement for a failed disk or to expand file system space). For systems which do not support "hot swap", the spare disk is spun down and the system could, for example, be quickly halted, the disk removed and the system rebooted and the spare moved to another system for use. For systems which support hot swap, once the disk has been spun down (and the fault "ready for removal" light is on), the **raid swap** command can be issued to allow physical removal of the disk from the system.

fail *disk_id*

will fail the specified file system disk from the RAID configuration, spinning the disk down when removal is complete. **raid fail** is useful to allow removal of a file system disk that may be logging excessive errors and requires replacement. Note that when a file system disk has been removed in this manner, the RAID configuration will enter degraded mode (meaning a disk is missing from the array). For systems which support hot swap, once the disk has been spun down (and the fault "ready for removal" light is on), the **raid swap** command can be issued to allow physical removal of the disk from the system.

scrub start | **stop**

will start or stop a RAID scrubbing operation. **raid scrub start** starts the scrubbing process and **raid scrub stop** stops the scrubbing process. The `raid.scrub.enable` option is ignored.

EXAMPLES

The following example uses **raid** to add spare disk number 4 to the RAID configuration. Note that we first use the **sysconfig** command to display the current RAID configuration and verify that `disk_id` 4 is a spare disk:

```
toaster> sysconfig -r
RAID Disk      DISK_ID#    HA.SCSI#    Used (MB/blks)    Phys (MB/blks)
-----
parity         0           0.0         2000/409600        2040/4177920
data 1         1           0.1         2000/409600        2049/4196352
data 2         2           0.2         1000/204800        1033/2115584
data 3         3           0.3         1000/204800        1033/2115584
spare          4           0.4         1000/204800        1033/2115584
```

```
toaster> raid add 4
Addition of disk 4 to the filesystem has been initiated.
You will be notified when addition is complete.
toaster>
Wed Jun 01 00:39:24 GMT [raid_disk_admin]: Addition of disk 4
to the filesystem has completed successfully.
```

The next example uses **raid remove** to remove spare disk number 3 from the RAID configuration and then the **raid swap** to stall I/O to allow physical removal of the disk from the system:

```
toaster> raid remove 3
Removal and unload of spare disk 3 has been initiated.
You will be notified when unload is complete.
toaster>
Wed Jun 01 22:10:12 GMT 1994 [raid_disk_admin]: Unload of disk
3 has completed successfully.

toaster> raid swap
System has been prepared for a disk swap. NFS service will continue
when a disk is removed or added to the system.
```

Use the "raid unswap" command to continue NFS service if this command was accidentally typed or you choose not to swap a disk at this time.

The following messages are seen when the disk is physically removed from the system, confirming a disk *hot swap* event has occurred and that during the drive unit search no new disk units were found:

```
toaster>
Wed Jun 01 22:10:42 GMT [disk_config_admin]: *** NOTICE ***
A disk has been swapped (removed or added) to a modular
storage shelf. The system will wait 15 seconds and
then check the status of all disk drives.
Wed Jun 01 22:10:57 GMT [disk_config_admin]: *** NOTICE ***
Disk unit status check has completed.
```

SEE ALSO

na_sysconfig(1)

NAME

rdate – set system date from a remote host

SYNOPSIS

rdate *hostname*

DESCRIPTION

rdate sends a request to the time server on *hostname* and sets the local date and time to the value returned by the server. **rdate** will time out if the server doesn't respond in 10 seconds.

rdate can be added to */etc/rc* to automatically synchronize the system time with the time server on each reboot.

CLUSTER CONSIDERATIONS

You cannot use the **rdate** command in partner mode to synchronize the time of the partner with a time server.

FILES

/etc/rc system initialization command script

SEE ALSO

na_date(1), **na_partner(1)**, **na_rc(5)**

NAME

reboot – stop and then restart the filer

SYNOPSIS

reboot [**-d**] [**-t** *minutes*]

DESCRIPTION

reboot halts the filer and then restarts it. **reboot** is commonly used to allow modified configuration files to take effect or to run a newly installed version of Data ONTAP.

NFS clients can maintain use of a file over a **halt** or **reboot** (although experiencing a failure to respond during that time), but CIFS clients cannot do so safely. Therefore CIFS clients should –if possible– be warned to close their open files. If you did not use the **-t** option to specify a maximum delay *and* there are CIFS clients with open files, the **reboot** command displays the number of CIFS users and the number of open CIFS files. Then it prompts you for the number of minutes to delay. CIFS files that are still open at the time the filer halts will lose writes that had been cached but not written.

reboot logs a message in the `/etc/messages` file (see `messages(5)`) file to indicate that the filer was rebooted on purpose.

OPTIONS

- d** Dump system core before rebooting.
- t** *minutes* Reboots after the indicated number of minutes, or after all CIFS files that were open have been closed, whichever is sooner.

CLUSTER CONSIDERATIONS

You cannot use the **reboot** command in partner mode to reboot a failed filer.

If you reboot the live filer that has taken over the failed filer, after the reboot, the live filer reinitiates the takeover process.

SEE ALSO

na_download(1), **na_halt(1)**, **na_partner(1)**, **na_savecore(1)**, **na_setup(1)**

NAME

restore – restore files or file systems from backups made with the `filer's dump` command

SYNOPSIS

restore *key args...*

DESCRIPTION

The **restore** restores files from backup tapes created with the **dump** (see **na_dump(1)**) command. A full backup of a file system may be restored and subsequent incremental backups layered on top of it. The actions of **restore** are controlled by the given *key*, which is a string of characters containing at most one function letter and possibly one or more function modifiers.

The function portion of the key is specified by one of the following letters:

- r** Restores (rebuilds a file system or subtree). The target subtree should be made pristine by removing it from a client of the server or, if the entire file system or all subtrees of the file system are to be restored, by booting from floppy disk and selecting the "Install new file system." option, before starting the restoration of the initial level 0 backup. If the level 0 restores successfully, the **r** key may be used to restore any necessary incremental backups on top of the level 0.

Note that **restore r** will restore *all* files from the dump tape(s).

An example:

restore rf rst0a

Note that **restore** leaves a file **restore_symboltable** in the directory that was dumped to pass information between incremental restore passes. This file should be removed when the last incremental has been restored.

- R** **restore** requests a particular tape of a multi-volume set on which to restart a full restore (see the **r** key above). This is useful if the restore has been interrupted.
- t** Lists the names of the specified files if they occur on the backup. If no file argument is given, then the root directory is listed, which results in the entire content of the backup being listed.
- x** Extracts the named files. If a named file matches a directory whose contents were backed up, the directory is recursively extracted. The owner, modification time, and mode are restored. If no *filename* argument is specified, the backup root directory is extracted. This results in the entire backup being restored.

The following characters may be used in addition to the letter that selects the function desired.

- b** The next argument to **restore** is used as the block size of the media (in

kilobytes). If the **b** option is not specified, **restore** tries to determine the media block size dynamically.

- f** The next argument to **restore** is used as the name of the archive instead of the standard input. If the name of the file is **-**, **restore** reads from standard input.
- s** The next argument to **restore** is a number which selects the file on a multi-file dump tape. File numbering starts at 1.
- D** By default, files will be restored into the directory from which they were dumped. If the **D** option is specified, the next argument to **restore** is the full absolute pathname of a directory into which the files should be restored.
- v** Normally **restore** does its work silently. The **v** (verbose) key causes it to type the name of each file it treats preceded by its file type.
- y** **restore** will not ask whether it should abort the restore if it encounters an error. It will always try to skip over the bad block(s) and continue as best it can.

DIAGNOSTICS

Complains about bad key characters.

Complains if it gets a read error. If **y** has been specified, or the user responds **y**, **restore** will attempt to continue the restore.

If a backup was made using more than one tape volume, **restore** will notify the user when it is time to mount the next volume.

There are numerous consistency checks that can be listed by **restore**. Most checks are self-explanatory or can “never happen”. Common errors are given below.

filename: not found on tape

The specified file name was listed in the tape directory, but was not found on the tape. This is caused by tape read errors while looking for the file, and from using a dump tape created on an active file system.

expected next file *inumber*, **got** *inumber*

A file that was not listed in the directory showed up. This can occur when using a dump created on an active file system.

Incremental dump too low

When doing incremental restore, a dump that was written before the previous incremental dump, or that has too low an incremental level has been loaded.

Incremental dump too high

When doing incremental restore, a dump that does not begin its coverage where the previous incremental dump left off, or that has too high an incremental level has been loaded.

Tape read error while restoring *filename*

Tape read error while skipping over inode *inumber***Tape read error while trying to resynchronize**

A tape (or other media) read error has occurred. If a file name is specified, then its contents are probably partially wrong. If an inode is being skipped or the tape is trying to resynchronize, then no extracted files have been corrupted, though files may not be found on the tape.

resync restore, skipped *num* blocks

After a dump read error, **restore** may have to resynchronize itself. This message lists the number of blocks that were skipped over.

CLUSTER CONSIDERATIONS

In takeover mode, the failed filer cannot access its tape devices. The **restore** command can only restore from the tape devices on the live filer.

FILES

/tmp/rstdir* file containing directories on the tape.

/tmp/rstmode* owner, mode, and time stamps for directories.

restore_symboltable information passed between incremental restores.

SEE ALSO

na_dump(1)

BUGS

A level zero dump must be done after a full restore. **restore** has no control over inode allocation; thus a full restore must be done to get a new set of directories reflecting the new inode numbering, even though the contents of the files is unchanged.

restore does not support the **-i** option for interactive restoring of files. Alternatively, provided your file access is limited to NFS, you could use **restore -i** from a UNIX client. (That alternative won't work correctly for a file system with CIFS clients, because the UNIX restore utility doesn't support the multi-protocol directory structures used by the filer.) If your tape drive is local to the filer, you can use the rmt facility (see **na_rmt(8)**) on the filer to access the local tape drive from the client doing the interactive restore.

SEE ALSO

na_partner(1)

NAME

route – manually manipulate the routing table

SYNOPSIS

route [**-fn**] **add** | **delete** [**host** | **net**] *destination gateway* [*metric*]

DESCRIPTION

route allows the system administrator to manually manipulate the network routing table for the specific host or network specified by *destination*. The *gateway* argument is the next-hop gateway to which packets should be addressed for the corresponding *destination*. The *metric* argument indicates the number of “hops” to the *destination*. The *metric* argument is required for the **add** command; it must be zero if the *destination* is on a directly-attached network, and non-zero if the route is via one or more gateways.

The **add** command adds the specified route for the given *destination* to the routing table. The **delete** command deletes the specified route from the routing table.

Routes to a particular host are distinguished from those to a network by interpreting the Internet address associated with *destination*. The optional keywords **net** and **host** force the destination to be interpreted as a network or a host, respectively. Otherwise, if the *destination* has a “local address part” of INADDR_ANY (i.e., 0), or if the *destination* is the symbolic name of a network, then the route is assumed to be to a network; otherwise, it is presumed to be a route to a host. If the route is to a destination via a gateway, the *metric* parameter should be greater than 0. If *metric* is set to 0, the gateway given is the address of this host on the common network, indicating the interface to be used for transmission.

All symbolic names specified for a *destination* or *gateway* are looked up first as a host name in the */etc/hosts* database. If this lookup fails, then the name is looked up as a network name in the */etc/networks* database. “default” is also a valid destination, which is used if there is no specific host or network route.

OPTIONS

- f** Remove all gateway entries in the routing table. If this is used in conjunction with one of the commands, **route** removes the entries before performing the command.
- n** Prevent attempts to print host and network names symbolically when reporting actions.

DIAGNOSTICS

add [**host** | **net**] *destination:gateway*
The specified route is being added to the table.

delete [**host** | **net**] *destination:gateway*
The specified route is being deleted.

destination gateway done

When the **-f** flag is specified, each routing table entry deleted is indicated with a message of this form.

network unreachable

An attempt to add a route failed because the gateway listed was not on a directly-connected network. The next-hop gateway must be given.

not in table

A delete operation was attempted for an entry which wasn't present in the table.

entry already exists

An add operation was attempted for an existing route entry.

routing table overflow

An add operation was attempted, but the system was unable to allocate memory to create the new entry.

CLUSTER CONSIDERATIONS

In takeover mode, each filer in a cluster maintains its own routing table. You can make changes to the routing table on the live filer, or you can make changes to the routing table on the failed filer using the **route** command in partner mode. However, the changes you make in partner mode are lost after a giveback.

SEE ALSO

na_partner(1), **na_routed(1)**

NAME

routed – network routing daemon

SYNOPSIS

routed on

routed off

routed [-n] status

DESCRIPTION

routed (pronounced “route-D”) uses a variant of the Xerox NS Routing Information Protocol (RIP) to manage selection of the default gateway used for IP network routing. The filer’s **routed** is different from the standard Unix **routed** as it never sends RIP packets, or builds route tables from RIP information, but only snoops for RIP exchanges to determine gateway status; it builds the routing table based on ICMP redirects.

When **routed** is started with the **routed on** command, it reads the **/etc/dgateways** file to create a list of potential default gateways. The **/etc/dgateways** file consists of a series of lines, each in the following format:

gateway metric

where:

gateway is the name or address of a gateway to be used as a potential default gateway.

metric is a metric indicating the preference weighting of the gateway. 1 is the value to use for highest preference, 15 for the least. If no value is specified, *metric* defaults to the value 1.

There can be a maximum of 128 valid entries in the **/etc/dgateways** file – additional ones are ignored, but cause an error message. Duplicate gateway names or addresses are not allowed – only the first one encountered in the file is added to the table, and duplicates produce error messages.

After the list of gateways is created, **routed** selects the one with the lowest metric value to be used as the preferred default route. If there are multiple gateways available with the same metric value, it uses the one named first in the **/etc/dgateways** file.

routed then listens on udp port 520 for routing information packets. When a RIP *request* or *reply* packet is received, **routed** marks the gateway that sent the packet ALIVE. If the gateway has a better metric than the current default gateway, or has the same metric but is listed earlier in **/etc/dgateways**, the current default gateway is changed to the new gateway.

When a gateway is not heard from for 90 seconds, **routed** marks the gateway as DEAD, and if it was the current default gateway, selects a new default gateway if one is available.

In addition, when **routed** is running, it deletes dynamic routes, created by ICMP redirects, every 3 minutes.

USAGE

routed on The route daemon may be turned on at any time with the **routed on** command. This causes **routed** to read the **/etc/dgateways** file, and turn on RIP snooping, dynamic route timeouts, and default gateway selection. If **routed** is already running, this option cause it to reread the **/etc/dgateways** file, and reinitialize. By default, **routed** is invoked at boot time in **/etc/rc**.

routed off The route daemon may be turned off at any time with the **routed off** command. This stops all RIP snooping, default gateway selection, and dynamic route timeouts. The currently selected default gateway is not be deleted when **routed** is turned off.

routed status Displays the status of the default gateway list. This shows whether RIP snooping is active, the current list of default gateways, their metrics, the state of the gateways (ALIVE or DEAD), and the last time each gateway was heard from. The output looks like:

```
maytag> routed status
RIP snooping is on
Gateway Metric State Time Last Heard
alantec1      1 ALIVE   Wed Mar 9 03:38:41 GMT 1994
groucho       1 ALIVE   Wed Mar 9 03:38:41 GMT 1994
192.9.200.66  1 ALIVE   Wed Mar 9 03:38:41 GMT 1994
192.9.200.77  1 ALIVE   Wed Mar 9 03:38:41 GMT 1994
tphub1        2 ALIVE   Wed Mar 9 03:38:41 GMT 1994
192.9.200.32  2 ALIVE   Wed Mar 9 03:38:41 GMT 1994
192.9.200.252 3 ALIVE   Wed Mar 9 03:38:41 GMT 1994
192.9.200.251 4 ALIVE   Wed Mar 9 03:38:41 GMT 1994
192.9.200.250 5 ALIVE   Wed Mar 9 03:38:41 GMT 1994
119 free gateway entries, 9 used
```

OPTIONS

-n If this option precedes **status**, the command displays numeric values for gateway names.

FILES

/etc/rc for default initialization
/etc/dgateways for the list of default gateways.

SEE ALSO

na_netstat(1), na_route(1), na_setup(1), na_dgateways(5), na_rc(5)

DIAGNOSTICS

routed: unable to allocate free entry - too many valid entries were found in the **/etc/dgateways** file. Only the first 128 are used.

routed: duplicate gateway entry not allowed - a duplicate gateway name or address was found in the **/etc/dgateways** file. Only the first one found is used.

routed: unable to open socket - a networking error has prevented **routed** from initializing properly.

BUGS

A default route created with **route add**, either by hand, or in **/etc/rc** may be deleted by **routed** when it starts running, if it knows a better route.

If the “best” entry selected from **/etc/dgateways** is DEAD when **routed** starts up, there may be a period of 90 seconds before a gateway that is ALIVE is selected.

NAME

savecore – save a core dump

SYNOPSIS

savecore

DESCRIPTION

savecore is meant to be called near the end of the initialization file **/etc/rc**. Its function is to save the core dump of the system (assuming one was made) and to write the panic string to **/etc/messages**. **savecore** saves the core dump in two files **/etc/crash/core.n**, and **/etc/crash/core.n-small**, where *n* is determined by the **/etc/crash/bounds** file.

The **-small** core file contains a subset of the memory image that Network Appliance can use for initial troubleshooting. Network Appliance will only need to look at the large core file if the problem cannot be determined by examining the small one.

Before **savecore** writes out a core image, it reads a number from the file **/etc/crash/minfree**. If the number of free kilobytes in the filesystem after saving the core would be less than the number obtained from **minfree**, the core dump is not saved. If **minfree** does not exist, **savecore** always writes out the core file (assuming that a core dump was taken).

FILES

/etc/crash/core.*	saved core files
/etc/crash/core.*-small	saved small core files
/etc/crash/bounds	suffix for next core file
/etc/crash/minfree	free KB in FS to maintain after savecore

CLUSTER CONSIDERATIONS

If the live filer in a cluster has the **savecore** command in its **/etc/rc** file, it can save the cores created by its partner when its partner crashes. The cores are saved to the partner's **/etc/crash** directory.

SEE ALSO

na_partner(1), **na_rc(5)**.

NAME

setup – update filer configuration

SYNOPSIS

setup

DESCRIPTION

setup queries the user for the basic filer configuration parameters such as hostname, IP address, and timezone. It installs new versions of */etc/rc*, */etc/hosts*, */etc/exports*, */etc/resolv.conf*, */etc/hosts.equiv*, and */etc/dgateways* to reflect the new configuration. When **setup** completes, the configuration files have been updated, but their new contents do not take effect until the filer is rebooted (see **na_reboot(1)**). The old contents of the configuration files are saved in **rc.bak**, **exports.bak**, **resolv.conf.bak**, **hosts.bak**, **hosts.equiv.bak**, and **dgateways.bak**.

One piece of information that **setup** requests is the name and IP address for *adminhost*. In */etc/exports*, *adminhost* is granted root access to */* so that it can access and modify the configuration files in */etc*. All other NFS clients are granted access only to */home*. If no *adminhost* is specified, then all clients are granted root access to */*. This is not recommended for sites where security is a concern.

If an *adminhost* is specified, then an additional line is added to the */etc/hosts* file to point the default mailhost to the *adminhost*. This is used by the autosupport daemon (see **na_autosupport(8)**) to send email notification.

If a default gateway is provided to setup, it will be used in */etc/rc* to specify a default route (see **na_route(1)**), and will also be used as the first entry in */etc/dgateways*.

The *hostname* that is provided to **setup** is used to construct default names for all of the configured network interfaces. Ethernet interfaces are given names *hostname-0*, *hostname-1*, and so on. The FDDI interfaces are named *hostname-f0*, *hostname-f1*, and so on, and the ATM interfaces are named *hostname-a0*, *hostname-a1*, and so on.

FILES

<i>/etc</i>	directory of filer configuration and administration files
<i>/etc/rc</i>	system initialization command script
<i>/etc/exports</i>	directories exported by the server
<i>/etc/hosts</i>	host name data base
<i>/etc/hosts.equiv</i>	list of hosts and users with rsh permission
<i>/etc/resolv.conf</i>	list of DNS name servers
<i>/etc/dgateways</i>	list of preferred default gateways for routed
<i>/etc/nsswitch.conf</i>	list of preferred name services

CLUSTER CONSIDERATIONS

After a takeover, you can enter the **setup** command in partner mode to configure the failed filer. However, only the network interfaces on the failed filer that were taken

over appear in the prompts displayed by **setup**. For example, if the **e1** interface on the failed filer was not configured and taken over by the live filer, the **setup** command does not prompt you for the IP address of the **e1** interface.

SEE ALSO

na_ifconfig(1), **na_partner(1)**, **na_reboot(1)**, **na_dgateways(5)**, **na_exports(5)**, **na_hosts(5)**, **na_hosts.equiv(5)**, **na_resolv.conf(5)**, **na_rc(5)**, **na_autosupport(8)**

NOTES

Some Ethernet boards determine the media type automatically. It is not strictly necessary to specify the media type for them, but it is best to do so anyway, in case the board is replaced with one that does not determine media type automatically.

NAME

shelfchk – verify the communication of environmental information between disk shelves and the filer

SYNOPSIS

shelfchk

DESCRIPTION

The **shelfchk** command verifies that the disk shelves and the filer can exchange environmental information. If the environmental information is being exchanged, you can hot-swap disks in the disk shelves.

The **shelfchk** command is interactive. It requires that you type in your responses after observing the LEDs on the disks. Therefore, enter this command from a console that is near the disk shelves.

The **shelfchk** command steps through all the disk host adapters that the filer discovered when it booted. For each host adapter, the **shelfchk** command tries to turn on the disk LEDs on the attached disk shelves. The command waits for confirmation that you have observed the LEDs. If you see that all the LEDs are on, respond “yes” when prompted. If one or more LEDs are off, you respond “no” to the prompt. In this case, a problem exists that might prevent hot swapping on the affected shelves. The **shelfchk** command terminates as soon as you respond “no” to the prompt. It does not continue to test the other disk shelves. A possible cause of disk shelf problems is that the cables for the shelves are not connected properly.

Enter the **shelfchk** command immediately after you install one or more disk shelves. This way, if there are any cabling problems, you can fix them as soon as possible. Also, this command enables you to quickly correlate the disk shelves with their corresponding host adapter. For example, if you intend to have all disk shelves connected to a particular host adapter to be installed in one rack, the **shelfchk** command enables you to see at a glance whether any disk shelves were installed inadvertently in a different rack.

CLUSTER CONSIDERATIONS

The **shelfchk** command checks the status of all disk shelves attached to the filer’s disk adapters. It does not distinguish between the disk shelves owned by the local filer and the disk shelves owned by the partner.

After you set up the cluster for the first time or after you install additional disk shelves to the filers in a cluster, enter the **shelfchk** command on both filers to verify that the disk shelves are attached to the appropriate adapters.

In takeover mode, the **shelfchk** command and the **partner shelfchk** command generate the same result.

EXAMPLE

In the following example, the **shelfchk** command tests the disk shelves of a filer with three host adapters (8a, 8b, and 7a) and finds no problems:

```
toaster> shelfchk
Only shelves attached to ha 7a should have all LEDs ON.
Are these LEDs all ON now? y
Only shelves attached to ha 8a should have all LEDs ON.
Are these LEDs all ON now? y
Only shelves attached to ha 8b should have all LEDs ON.
Are these LEDs all ON now? y
toaster> Fri Aug 22 21:35:39 GMT [rc]: Disk Configuration - No Errors
Identified
```

In the following example, the **shelfchk** command finds an error:

```
toaster> shelfchk
Only shelves attached to ha 9a should have all LEDs ON.
Are these LEDs all ON now? n
*** Your system may not be configured properly. Please check cable connections.
toaster> Mon Aug 25 11:44:34 GMT [rc]: Disk Configuration - Failure
Identified by Operator
```

SEE ALSO

na_partner(1)

NAME

snap – manage snapshots

SYNOPSIS

snap list [*vol_name*]

snap create | **delete** *vol_name name*

snap rename *vol_name from to*

snap sched [*vol_name* [*weeks* [*days* [*hours*[*@list*]]]]]

snap reserve [*vol_name* [*percent*]]

DESCRIPTION

The **snap** family of commands provides a means to create and manage snapshots in each volume.

A snapshot is a read-only copy of the entire file system as of the time the snapshot was created. The filer uses a copy-on-write technique to create snapshots very quickly without consuming any disk space. Only as blocks in the active file system are modified and written to new locations on disk does the snapshot begin to consume extra space.

Snapshots are exported to all CIFS or NFS clients. They can be accessed from each directory in the file system. From any directory, a user can access the set of snapshots from a hidden sub-directory that appears to a CIFS client as **~snapsht** and to an NFS client as **.snapshot**. These hidden sub-directories are special in that they can be accessed from every directory, but they only show up in directory listings at an NFS mount point or at the root of CIFS share.

Each volume on the filer can have up to 20 snapshots at one time. Because of the copy-on-write technique used to update disk blocks, deleting a snapshot will generally not free as much space as its size would seem to indicate. Blocks in the snapshot may be referenced by other snapshots, or by the active file system, and thus may be unavailable for reuse even after the snapshot is deleted.

The **snap** commands are persistent across reboots. Do not include **snap** commands in the **/etc/rc**. If you include a **snap** command in the **/etc/rc** file, the same **snap** command you enter through the command line interface does not persist across a reboot and is overridden by the one in the **/etc/rc** file.

Automatic snapshots

Automatic snapshots can be scheduled to occur weekly, daily, or hourly. Weekly snapshots are named **weekly.N**, where *N* is “0” for the most recent snapshot, “1” for the next most recent, and so on. Daily snapshots are named **daily.N** and hourly snapshots **hourly.N**. Whenever a new snapshot of a particular type is created and the number of existing snapshots of that type exceeds the limit specified by the **sched**

option described below, then the oldest snapshot is deleted and the existing ones are renamed. If, for example, you specified that a maximum of 8 hourly snapshots were to be saved using the **sched** command, then on the hour, **hourly.7** would be deleted, **hourly.0** would be renamed to **hourly.1**, and so on.

USAGE

snap list [*vol_name*]

displays a single line of information for each snapshot. Along with the snapshot's name, it shows when the snapshot was created and the size of the snapshot. If you include the *vol_name* argument, **list** displays snapshot information only for the specified volume. With no arguments, it displays snapshot information for all volumes in the system. The following is an example of the **snap list** output on a filer with two volumes named engineering and marketing.

Volume engineering

<i>%/used</i>	<i>%/total</i>	<i>date</i>	<i>name</i>
0% (0%)	0% (0%)	Nov 14 08:00	hourly.0
50% (50%)	0% (0%)	Nov 14 00:00	nightly.0
67% (50%)	0% (0%)	Nov 13 20:00	hourly.1
75% (50%)	0% (0%)	Nov 13 16:00	hourly.2
80% (50%)	0% (0%)	Nov 13 12:00	hourly.3
83% (50%)	1% (0%)	Nov 13 08:00	hourly.4
86% (50%)	1% (0%)	Nov 13 00:00	nightly.1
87% (50%)	1% (0%)	Nov 12 20:00	hourly.5

Volume marketing

<i>%/used</i>	<i>%/total</i>	<i>date</i>	<i>name</i>
0% (0%)	0% (0%)	Nov 14 08:00	hourly.0
17% (16%)	0% (0%)	Nov 14 00:00	nightly.0
28% (16%)	0% (0%)	Nov 13 20:00	hourly.1
37% (16%)	0% (0%)	Nov 13 16:00	hourly.2
44% (16%)	0% (0%)	Nov 13 12:00	hourly.3
49% (16%)	1% (0%)	Nov 13 08:00	hourly.4
54% (16%)	1% (0%)	Nov 13 00:00	nightly.1
58% (16%)	1% (0%)	Nov 12 20:00	hourly.5

snap create *vol_name name*

creates a snapshot of volume *vol_name* with the specified name.

snap delete *vol_name name*

deletes the existing snapshot belonging to volume *vol_name* that has the specified name.

snap rename *vol_name oldname newname*

gives an existing snapshot a new name. You can use the **snap rename** command to move a snapshot out of the way so that it won't be deleted automatically.

snap sched [*vol_name* [*weeks* [*days* [*hours* [*@list*]]]]]

sets the schedule for automatic snapshot creation. The argument *vol_name* identifies the volume the schedule should be applied to. The second argument indicates how many weekly snapshots should be kept on-line, the third how many daily, and the fourth how many hourly. If an argument is left off, or set to zero, then no snapshot of the corresponding type is created. Daily snapshots are created at 24:00 of each day except Sunday, and weekly snapshots are created at 24:00 on Sunday. Only one snapshot is created at a time. If a weekly snapshot is being created, for instance, no daily or hourly snapshot will be created even if one would otherwise be scheduled. For example, the command

snap sched vol0 2 6

indicates that two weekly snapshots and six daily snapshots of volume *vol0* should be kept on line. No hourly snapshots will be created. For snapshots created on the hour, an optional list of times can be included, indicating the hours on which snapshots should occur. For example the command

snap sched vol0 2 6 8@8,12,16,20

indicates that in addition to the weekly and daily snapshots, eight hourly snapshots should be kept on line, and that they should be created at 8 am, 12 am, 4 pm, and 8 pm. Hours must be specified in 24-hour notation.

With no argument, **snap sched** prints the current snapshot schedule for all volumes in the system. With just the *vol_name* argument, it prints the schedule for the specified volume.

snap reserve [*vol_name*] | [*vol_name percent*]

Sets the size of the indicated volume's snapshot reserve to *percent*. With no *percent* argument, prints the percentage of disk space that is reserved for snapshots in the indicated volume. With no argument, the **snap reserve** command prints the percentage of disk space reserved for

snapshots for each of the volumes in the system. Reserve space can be used only by snapshots and not by the active file system.

SEE ALSO

na_df(1)

BUGS

There is no way to tell how much space will be freed by deleting a particular snapshot or group of snapshots.

The **snap create**, **snap delete**, and **snap list** commands can take 20 or 30 seconds on very large file systems.

NAME

snmp – set and query SNMP agent variables

SYNOPSIS

snmp

snmp authtrap [**0** | **1**]

snmp community [**add** | **delete ro** | **rw**]

snmp contact [*contact*]

snmp init [**1**]

snmp location [*location*]

snmp traphost [**add** | **delete** *hostname* | *ipaddress*]

DESCRIPTION

The **snmp** command is used to set and query configuration variables for the SNMP agent daemon (see **na_snmpd**(8)). If no options are specified, **snmp** lists the current values of all variables.

OPTIONS

In all the following options, specifying the option name alone prints the current value of that option variable. If the option name is followed by one or more variables then the appropriate action to set or delete that variable will be taken. Any variable with an inclusive space or tab must be enclosed in single quotes "".

It is recommended that all **snmp** commands be added to the end of the **/etc/rc** file. The last **snmp** command in the **/etc/rc** file should be:

snmp init 1

This will initialize the SNMP daemon with the values set using the **snmp** command and it will send out a **coldStart** trap as described below.

authtrap [**0** | **1**]

Enable or disable SNMP agent authentication failure traps. To enable authentication traps specify **1**. To disable authentication traps specify **0**. Traps are sent to all hosts specified with the **traphost** option.

community [**add|delete ro|rw** *community*]

Add or delete communities with the specified access control type. Specify **ro** for a read-only community and **rw** for a read-write community. For example, to add the read-only community **private** use the following command:

snmp community add ro private

Currently the SNMP SetRequest PDU is not supported, so all read-write communities will default to read-only. The default community for the filer SNMP

agent is **public** and its access mode is **ro**. Up to a maximum of 8 communities are supported.

contact [*contact*]

Used to set the contact name returned by the SNMP agent as the System.sysContact.0 MIB-II variable.

init [**1**]

With an option of **1** this initializes the snmp daemon with values previously set by the snmp command. It also sends a **coldStart** trap to any hosts previously specified by the **traphost** option. The command:

snmp init 1

should be the last **snmp** command in the filer's **/etc/rc** file.

On a query **init** will return the value 0 if the SNMP daemon has not yet been initialized. Otherwise it will return the value 1.

location [*location*]

Used to set the location name returned by the SNMP agent as the System.sysLocation.0 MIB-II variable.

traphost

[**add** | **delete** *hostname* | *ipaddress*]

To add or delete SNMP managers who will be the recipient of the filer's Trap PDU's. Specify the word **add** or **delete** as appropriate followed by the host name or IP address. If a host name is specified, it must exist in the **/etc/hosts** file. For example, to add the host **alpha** use the following command:

snmp traphost add alpha

No traps will be sent unless at least one trap host is specified. Up to a maximum of 8 trap hosts are supported.

On a query the **traphost** option will return a list of registered trap hosts followed by their IP addresses. If a host name cannot be found in **/etc/hosts** for a previously registered IP address, its name will default to a string representation of its IP address.

EXAMPLES

A typical set of snmp commands in the **/etc/rc** file will look like the following:

```
snmp contact 'Network Manager'  
snmp location 'Bldg 2. Lab 3a'  
snmp community add ro private  
snmp traphost add snmp-mgr1  
snmp traphost add snmp-mgr2
```

snmp init 1**FILES**

/etc/rc startup command script where snmp commands must be added

/etc/hosts hosts name database

CLUSTER CONSIDERATIONS

The files in a cluster can have different settings for the **snmp** options.

SEE ALSO

na_partner(1), **na_autosupport(8)**, **na_snmpd(8)**, **na_rc(5)**

NAME

sysconfig – display filer configuration information

SYNOPSIS

sysconfig [**-d** | **-r** | **-t** | [**-v**] [*slot*]]

DESCRIPTION

sysconfig displays the configuration information about the filer. Without any arguments, the output includes the Data ONTAP(tm) version number and a separate line for each I/O device in the filer. If the *slot* argument is specified, **sysconfig** displays detail information for the specified physical slot; slot 0 is the system board, and slot *n* is the *n*th expansion slot on the filer.

OPTIONS

-d Displays vital product information for each disk.

-r Displays RAID configuration information.

-t Displays device and configuration information for each tape drive.

If you have a tape device that Network Appliance has not qualified, the **sysconfig -t** command output for that device is different from that for qualified devices. If the filer has never accessed this device, the output indicates that this device is a non-qualified tape drive, even though there is an entry for this device in the **/etc/clone_tape** file. Otherwise, the output provides information about the qualified tape drive that is being emulated by this device.

You can enter the following command to access a tape device:

mt -f device status

-v Displays detail information about each I/O device. For SCSI host adapters, the additional information includes a separate line describing each attached disk.

CLUSTER CONSIDERATIONS

During normal operation, the **sysconfig** command displays similar information on a filer in a cluster as the **sysconfig** command on a standalone filer. The output on a filer in a cluster, however, includes disks on both fibre channel loop A and loop B. The information about disks on loop B is for hardware only. That is, the **sysconfig** command only displays information about the adapters supporting the disks on loop B. It does not show the capacity of each disk on loop B or whether a disk on loop B is a file system disk, spare disk, or parity disk.

In takeover mode, the **sysconfig** command provides the same types of information as in normal mode, except that it also displays a reminder that the filer is in takeover mode.

In partner mode, the **sysconfig** command does not display information about any hardware that is attached only to the partner. For example, if you enter the **partner sysconfig -r** command, you can obtain the software information about the disks on the partner. That is, for each disk on the partner, the command output indicates the capacity and whether the disk is a file system, spare, or parity disk. The command output does not include information about the disk adapters on the partner. The information about the disk adapters in the command output is for those on the local filer.

SEE ALSO

na_partner(1), **na_version(1)**, **na_mt(1)**

NAME

sysstat – report filer performance statistics

SYNOPSIS

sysstat [*interval*]

DESCRIPTION

sysstat reports filer performance statistics such as the current CPU utilization, the amount of network I/O, the amount of disk I/O, and the amount of tape I/O. By default, **sysstat** prints a new line of statistics every 15 seconds. The *interval* argument overrides the default causing **sysstat** to report once every *interval* seconds. Use control-C to stop **sysstat**.

EXAMPLE

This is an example of **sysstat** running on a lightly loaded NFS-only filer:

```
toaster> sysstat 1
  CPU  NFS  CIFS  HTTP  Net kB/s  Disk kB/s  Tape kB/s  Cache
      in  out  read write  read write  read write  age
  5%   82   0    0    15  17    16    0    0    0    8
  6%  105   0    0    24  98   100    0    0    0    8
  5%   54   0    0    32  11     0    0    0    0    8
 21%   50   0    0    25  42   120  592    0    0    8
 16%   27   0    0    10  10   144 1008    0    0    8
 17%   90   0    0    64  11    16  104    0  552    8
 15%   96   0    0    65  12     0    0    0  460    8
  5%   60   0    0    30  28    24    0    0    0    8
  1%   60   0    0    32  30    28    0    0    0    8
  4%   57   0    0    46  45    40    0    0    0    8
  5%   66   0    0    23  16     8    0    0    0    8
^C
toaster>
```

From left to right, the columns indicate:

CPU the percentage CPU utilization during the previous *interval* seconds;

NFS the number of NFS operations per second during that time;

CIFS the number of CIFS operations per second during that time;

HTTP the number of HTTP operations per second during that time;

Net kB/s the number of kilobytes per second of network traffic into and out of the server;

Disk kB/s the kilobytes per second of disk traffic being read and written;

- Tape kB/s the number of kilobytes per second of tape traffic being read and written;
- Cache age the age in minutes of the oldest read-only blocks in the buffer cache. Data in this column indicates how fast read operations are cycling through system memory; when the filer is reading very large files (larger than the machine's memory size), buffer cache age will be very low.

CLUSTER CONSIDERATIONS

In takeover mode, the **sysstat** command displays the combined statistics from both the failed filer and the live filer.

The statistics displayed by the **sysstat** command are cumulative; a giveback operation does not zero out the statistics. That is, after giving back its partner's resources, the live filer does not subtract the statistics about operations it performed on behalf of the failed filer in takeover mode.

SEE ALSO

na_netstat(1), **na_nfsstat(1)**, **na_partner(1)**.

NAME

timed – enable and disable the time daemon

SYNOPSIS

timed [**on** | **off**]

DESCRIPTION

This command is applicable only if your filer has the cluster license. The **timed** command determines whether a time daemon (**timed**) runs on the filer. If **timed** is on, the filer synchronizes its time with a time server. In addition, the filers in a cluster synchronize their times with each other. Having synchronized times on the filers ensures that network clients continue to see consistent timestamps for their files after a filer in a cluster has taken over its partner.

Time synchronization takes place once a day. To avoid overburdening the time server, the filer randomly selects the exact time of the synchronization. It can be any time between 10 minutes before midnight and 10 minutes after midnight.

The **timed** command without arguments shows whether the time daemon is running.

To specify up to five time servers, use the **timed.servers** option. No time server is specified by default.

To specify how the filer communicates with the time server, use the **timed.proto** option. You can choose the Network Time Protocol (NTP), which is the default, or the Internet Time Protocol, which is the protocol used by the **rdate** command. For synchronizing the time between filers in a cluster, only NTP is used.

You can modify these options only when the time daemon is not running. For more information about the **timed.servers** and **timed.proto** options, refer to the **na_options(1)** man page.

SEE ALSO

na_options(1), **na_rdate(1)**

NAME

timezone – set the local timezone

SYNOPSIS

timezone [*name*]

DESCRIPTION

timezone sets the system timezone. It is intended to be called in the initialization file **/etc/rc**. The argument *name* specifies the timezone to use. See the system documentation for a complete list of time zone names. If no argument is supplied, the current time zone name is printed.

Each timezone is described by a file that is kept in the **/etc/zoneinfo** directory on the filer. The *name* argument is actually the name of the file under **/etc/zoneinfo** that describes the timezone to use. For instance, the *name* “America/Los_Angeles” refers to the timezone file **/etc/zoneinfo/America/Los_Angeles**. These files are in standard “Arthur Olson” timezone file format, as used on many flavors of UNIX (SunOS 4.x and later, 4.4BSD, System V Release 4 and later, and others).

GMT+13 is to allow DST for timezone GMT+12.

FILES

/etc/zoneinfo directory of time zone information files

CLUSTER CONSIDERATION

In partner mode, you can use the **timezone** command without arguments to display the current time zone. However, you cannot use the **timezone** command to change the time zone.

SEE ALSO

na_partner(1), **na_zoneinfo**(5).

NAME

uptime – show how long system has been up

SYNOPSIS

uptime

DESCRIPTION

uptime prints the current time, the length of time the system has been up, and the total number of NFS operations the system has performed since it was last booted.

The filer runs **uptime** automatically once an hour and automatically logs its output to **/etc/messages**.

EXAMPLE

```
toaster> uptime
8:54am up 2 days 22:23, 3122520 NFS ops
```

CLUSTER CONSIDERATIONS

In partner mode, the **uptime** command displays how long the failed filer has been down and the host name of the live filer.

SEE ALSO

na_netstat(1), **na_nfsstat(1)**, **na_partner(1)**, **na_sysstat(1)**, **na_messages(5)**

NAME

version – display Data ONTAP version

SYNOPSIS

version

DESCRIPTION

version displays the version of Data ONTAP running on the server, and the date when the version was created.

EXAMPLE

```
toaster> version  
NetApp Release 4.0: Thu Oct 31 17:54:48 PST 1996
```

SEE ALSO

na_download(1), **na_sysconfig(1)**

NAME

vif – create and destroy virtual interfaces

SYNOPSIS

vif create [*vif_name*] *interface_name* ...

vif destroy *vif_name*

vif stat *vif_name* [*interval*]

DESCRIPTION

The **vif** command creates and eliminates virtual interfaces. It also displays statistics for a specified virtual interface.

The **vif create** command creates a virtual interface. The name of the virtual interface to be created is specified as *vifn*, where *n* is a number. Make sure that the specified virtual interface name is not already in use. Use the **ifconfig** command to check and see what virtual interface names are being used.

You can specify up to four Ethernet interfaces in the command. The interfaces do not have to be on the same network card. However, some Ethernet switches or routers require that all Ethernet interfaces forming a virtual interface be either half-duplex or full-duplex. Check the documentation that comes with your Ethernet switch or router to see whether you need to configure the filer Ethernet interfaces to be half-duplex or full-duplex.

The **vif destroy** command eliminates an existing virtual interface. You must configure the virtual interface down using the **ifconfig** command before entering the **vif destroy** command.

The **vif stat** command displays the number of packets received and transmitted on each Ethernet interface that makes up the virtual interface. You can specify the time interval, in seconds, at which the statistics are displayed. By default, the statistics are displayed at one-second intervals.

EXAMPLES

The following example creates a virtual interface:

```
toaster> vif vif1 create e0 e7a e6b e8
```

The following example eliminates virtual interface 1:

```
toaster> vif destroy vif1
```

The following example displays statistics about virtual interface 1:

```
toaster> vif stat vif1
```

```
Virtual interface (trunk) vif1
```

```
   e5d           e5c           e5b           e5a
In   Out       In   Out     In   Out     In   Out
```

8637076	47801540		158	159	7023083	38300325	8477195	47223431
1617	9588	0	0	634	3708	919	5400	
1009	5928	0	0	925	5407	1246	7380	
1269	7506	0	0	862	5040	1302	7710	
1293	7632	0	0	761	4416	964	5676	
920	5388	0	0	721	4188	981	5784	
1098	6462	0	0	988	5772	1003	5898	
2212	13176	0	0	769	4500	1216	7185	
1315	7776	0	0	743	4320	530	3108	

SEE ALSO**na_ifconfig(1)**

NAME

vol – commands for managing volumes, displaying volume status, and copying volumes

SYNOPSIS

vol *command argument ...*

DESCRIPTION

The **vol** commands manage a volume, apply options to a volume, or display the status of one or more volumes. Also, some **vol** commands copy volumes on the same filer or between two filers. You can use the volume copy feature only if you have purchased the volcopy license and entered the license code for a filer that is involved in a **vol copy** command.

USAGE

vol create *volname* [**-r** *raidsize*] *ndisks*[*@size*] | **-d** *diskname...*

creates a new volume with the name *volname*. The volume name can contain letters, numbers, and the underscore character(_), but the first character must be a letter or underscore. You can create up to 23 volumes on each filer.

The **-r** *raidsize* argument specifies the maximum number of disks in each RAID group in the volume. The maximum value of *raidsize* is 28. The default value is 14. *ndisks* is the number of disks in the volume, including the parity disks. The disks in this newly created volume come from the pool of spare disks. The smallest disks in this pool join the volume first, unless you specify the *@size* argument. *size* is the disk size in GB, and disks within 1 GB of the specified size are used in this volume.

If you use the **-d** *diskname* argument, the filer creates the volume with the specified spare disks. You can specify a space-separated list of disk names.

vol add *volname ndisks*[*@size*] | **-d** *diskname...*

adds disks to the volume with the name *volname*. Specify the disks in the same way as for the **vol create** command.

When adding disks to a volume, the filer fills up one RAID group with disks before starting another RAID group. Suppose a volume currently has one RAID group of 12 disks and its RAID group size is 14. If you add 5 disks to this volume, it will have one RAID group with 14 disks and another RAID group with 3 disks. The filer does not evenly distribute disks among RAID groups.

vol destroy *volname*

destroys the volume with the name *volname*. The disks originally in the volume become spare disks. Only offline volumes can be destroyed.

vol rename *volname newname*

renames the volume named *volname* to the name *newname*. If the volume named *volname* is referenced in the */etc/exports* file, remember to make the name change in */etc/exports* also so that the affected file system can be exported by the filer after the filer reboots. The **vol rename** command does not automatically update the */etc/exports* file.

vol online *volname*

brings the volume named *volname* online. This command takes effect immediately. The volume specified in this command must be currently offline or foreign. If the volume is foreign, it will be made native before being brought online. A “foreign” volume is a volume that consists of disks moved from another filer and that has never been brought online on the current filer. Volumes that are not foreign are considered “native.” You can also use this command to cancel a **vol offline** command.

vol offline *volname*

takes the volume named *volname* offline. This command takes effect when the filer is rebooted. If you change your mind after entering this command, you can enter **vol online** *volname* before the reboot.

vol options *volname optname optval*

sets the option named *optname* of the volume named *volname* to the value *optval*. The command remains effective after the filer is rebooted, so there is no need to add **vol options** commands to the */etc/rc* file. Some options have values that are numbers. Some options have values that may be **on** (which can also be expressed as **yes**, **true**, or **1**) or **off** (which can also be expressed as **no**, **false**, or **0**). You can use a mixture of uppercase and lowercase characters when typing the value of an option. The **root** option is special in that it does not have a value. To set the **root** option, use this syntax:

vol options *volname root*

The following describes the options and their possible values:

root The volume named *volname* will become the root volume for the filer on the next reboot. This option can be used on one volume only at any given time. The existing root volume will become a non-root volume after the reboot. The only way to remove the root status of a volume is to set the **root** option on another volume.

raidsize *number*

The value of this option is the maximum size of a RAID group within the volume. Changing the value of this option will not cause existing RAID groups to grow or shrink; it will only affect whether more disks will be added to the last existing RAID group and how large new RAID groups will be.

minra on | off

If this option is **on**, the filer performs minimal read-ahead on the volume. By default, this option is **off**, causing the filer to perform very aggressive read-ahead on the volume.

no_atime_update on | off

If this option is **on**, it prevents the update of the access time on an inode when a file is read. This option is useful for volumes with extremely high read traffic, since it prevents writes to the inode file for the volume from contending with reads from other files. It should be used carefully. That is, use this option when you know in advance that the correct access time for inodes will not be needed for files on that volume.

nosnap on | off

If this option is **on**, it disables automatic snapshots on the volume.

nosnapdir on | off

If this option is **on**, it disables the visible **.snapshot** directory that is normally present at client mount points, and turns off access to all other **.snapshot** directories in the volume.

vol status [**-r** | **-v** | **-d**] [*volname*]

displays the status of one or all volumes. If *volname* is used, the status of the specified volume is printed; otherwise the status of all volumes in the filer are printed. By default, it prints a one-line synopsis of the volume, which includes the volume name, whether it is online or offline, other states (for example, partial, degraded, and so on) and per-volume options.

The **-v** flag displays information about each RAID group within the volume.

The **-r** flag displays a listing of the RAID information for that volume.

The **-d** flag displays information about the disks in the specified volume. The types of disk information are the same as those from the **sysconfig -d** command.

vol copy start [**-S** | **-s**] *source destination*

copies all data, including the snapshots, from one volume to another. If you use the **-S** flag, the command copies all snapshots in the source volume to the destination volume. To specify a particular snapshot to copy, use the **-s** flag followed by the name of the snapshot. If you use neither the **-S** nor **-s** flag in the command, the filer creates a snapshot at the time when the **vol copy start** command is executed and copies only that snapshot to the destination volume.

The source volume and destination volume can be on the same filer or different filers. If the source or destination volume is on a filer other than the

one on which you enter the **vol copy start** command, specify the volume name in the *filer_name:volume_name* format.

The filers involved in a volume copy must meet the following requirements for the **vol copy start** command to be completed successfully:

The source volume must be on-line and the destination volume must be off-line.

If data is copied between two filers, each filer must be defined as a trusted host of the other filer. That is, the filer's name must be in the */etc/hosts.equiv* file of the other filer.

If data is copied on the same filer, localhost must be included in the filer's */etc/hosts.equiv* file. Also, the loopback address must be in the filer's */etc/hosts* file. Otherwise, the filer cannot send packets to itself through the loopback address when trying to copy data.

The usable disk space of the destination volume must be greater than or equal to the usable disk space of the source volume. Use the **df pathname** command to see the amount of usable disk space of a particular volume.

Each **vol copy start** command generates two volume copy operations: one for reading data from the source volume and one for writing data to the destination volume. Each filer supports up to four simultaneous volume copy operations.

vol copy abort [*operation_number*]

terminates a volume copy operation. The *operation_number* parameter in the **vol copy abort** command specifies which operation to terminate. If you don't specify an operation number, all volume copy operations are terminated.

vol copy status [*operation_number*]

displays the progress of one or all volume copy operations. The operations are numbered from 0 through 3.

vol copy throttle [*operation_number*] *value*

controls the performance of the volume copy operation. The value ranges from 10 (full speed) to 1 (one-tenth of full speed). The default is 10 (full speed).

You can apply the performance value to an operation specified by the *operation_number* parameter. If you do not specify an operation number in the **vol copy throttle** command, the command applies to all currently active volume copy operations. Use this command to limit the speed of the volume copy operation if you suspect that the volume copy operation is causing

performance problems on your filer.

The **vol copy throttle** command enables you to set the volume copy speed only for a volume copy operation that is in progress. To set the default volume copy speed to be used by future volume copy operations, use the **options** command to set the **vol.copy.throttle** option.

CLUSTER CONSIDERATIONS

Volumes on different filers in a cluster can have the same name. For example, both filers in a cluster can have a volume named vol0.

However, having unique volume names in a cluster makes it easier for you to migrate volumes between the filers in the cluster.

EXAMPLES

vol create vol1 -r 10 20

creates a volume named **vol1** with 20 disks. The RAID groups in this volume can contain up to 10 disks, so this volume has two RAID groups. The filer adds the current spare disks to the new volume, starting with the smallest disk.

vol create vol1 20@9

creates a volume named **vol1** with 20 9-GB disks. Because no RAID group size is specified, the default size (14 disks) is used. The newly created volume contains one RAID group with 14 disks and another group with six disks.

vol create vol1 -d 8a.1 8a.2 8a.3

creates a volume named **vol1** with the specified disks.

vol create vol1 10

vol options vol1 raidsize 5

The first command creates a volume named **vol1** with 10 disks, which belong to one RAID group. The second command specifies that if any disks are subsequently added to this volume, they will not cause any current RAID group to have more than five disks. Each existing RAID group will continue to have 10 disks and no more disks will be added to that RAID group. When new RAID groups are created, they will have a maximum size of five disks.

vol options vol1 root

The volume named **vol1** becomes the root volume after the next filer reboot.

vol options vol1 nosnapdir on

In the volume named **vol1**, the snapshot directory is invisible at the client mount point or at the root of a share. Also, for UNIX clients, the **.snapshot** directories that are normally accessible in all the directories become inaccessible.

vol status -r vol1

displays the RAID information about the volume named **vol1**:

RAID group 0

RAID Disk	HA.DISK_ID	Used (MB/blks)	Phys (MB/blks)
-----	-----	-----	-----
parity	0.3	4000/8192000	4095/8386728
data	0.2	4000/8192000	4095/8386728

vol copy start -s nightly.1 vol0 toaster1:vol0

copies the nightly snapshot named **nightly.1** on the **vol0** volume on the local filer to the **vol0** volume on a remote filer named **toaster1**.

vol copy status

displays the status of all the volume copy operations.

vol copy abort 1

terminates volume copy operation 1.

vol copy throttle 1 10

changes volume copy operation 1 to one-tenth of its full speed.

SEE ALSO

na_partner(1), **na_sysconfig(1)**.

NAME

ypwhich – display the NIS server if NIS is enabled

SYNOPSIS

ypwhich

DESCRIPTION

ypwhich prints the name of the current NIS server if NIS is enabled. If there is no entry for the server itself in the hosts database, then it prints the IP address of the server.

The NIS server is dynamically chosen by the filer.

NAME

tape – information on filer tape interface

DESCRIPTION

The filer supports up to four local tape drives (tape drives connected directly to the system). The tape drive interface follows a UNIX-like device name allowing use of a **rewind**, **norewind** or **unload/reload** device. The format of a filer tape device name is *crstud* where:

- c* use **n** to specify the **norewind** device, use **u** to specify the **unload/reload** device, or no flag to specify the **rewind** device. The **norewind** device will not rewind when the tape device is closed. The **unload/reload** device is used with sequential tape loaders and will unload the current tape volume and attempt to load the next tape volume (note that the server will wait up to one minute for the next volume to become ready before aborting the reload of the next volume). The **rewind** device will rewind the tape volume to beginning-of-tape on close.
- rst** the **rst** portion of the device name is always present and specifies that you are requesting a SCSI tape device.
- u* the logical unit number of the tape drive to use.
- d* the density (or format) to use for tape write operations.

The density specifications for an Exabyte 8505 8mm drive are:

- l** Exabyte 8200 format, no compression
- m** Exabyte 8200 format with compression
- h** Exabyte 8500 format, no compression
- a** Exabyte 8500 format with compression

EXAMPLES

The **sysconfig -t** command will display the supported tape drives on your system and the device names associated with each tape device along with the device's density, or format. The following is an example of the output from a **sysconfig** command on a filer with one tape device attached:

```
toaster> sysconfig -t
Tape drive (0.6) Exabyte 8505 8mm
rst0l - rewind device,          format is: EXB-8200    2.5GB
nrst0l - no rewind device,      format is: EXB-8200    2.5GB
urst0l - unload/reload device,  format is: EXB-8200    2.5GB
rst0m - rewind device,          format is: EXB-8200C  (w/compression)
```

nrst0m	-	no rewind device,	format is: EXB-8200C	(w/compression)
urst0m	-	unload/reload device,	format is: EXB-8200C	(w/compression)
rst0h	-	rewind device,	format is: EXB-8500	5.0GB
nrst0h	-	no rewind device,	format is: EXB-8500	5.0GB
urst0h	-	unload/reload device,	format is: EXB-8500	5.0GB
rst0a	-	rewind device,	format is: EXB-8500C	(w/compression)
nrst0a	-	no rewind device,	format is: EXB-8500C	(w/compression)
urst0a	-	unload/reload device,	format is: EXB-8500C	(w/compression)

SEE ALSO**na_dump(1) na_mt(1), na_sysconfig(1)**

NAME

boot – directory of Data ONTAP executables

SYNOPSIS

/etc/boot

DESCRIPTION

The **boot** directory contains copies of the executable files required to boot the filer. The **download** command (see **na_download(1)**) copies these files from **/etc/boot** into the filer's boot block, from which the system boots.

FILES

/etc/boot	directory of Data ONTAP executables
/etc/boot/netapp_4.0-x86	executable for Data ONTAP 4.0 on filers with x86 processors
/etc/boot/netapp_4.0-alpha	executable for Data ONTAP 4.0 on filers with Alpha processors
/etc/boot/netapp-x86	symbolic link to current version of Data ONTAP for filers with x86 processors
/etc/boot/netapp-alpha	symbolic link to current version of Data ONTAP for filers with Alpha processors
/etc/boot/0-x86	first stage boot code and boot FCode for filers with x86 processors
/etc/boot/1-x86	second stage boot code for filers with x86 processors
/etc/boot/fc-hard-alpha	boot FCode for filers with Alpha processors
/etc/boot/1-alpha	second stage boot code for filers with Alpha processors

SEE ALSO

na_download(1)

NAME

crash – directory of system core files

SYNOPSIS

/etc/crash

DESCRIPTION

If a filer crashes, it creates a core file in the **crash** directory. The core files are very useful for finding and fixing bugs in Data ONTAP, so please notify Network Appliance of any core files on your filer.

See **na_savecore(1)** for more details about how core files are saved.

FILES

/etc/crash/core.*	saved core files
/etc/crash/core.*-small	compact core file.
/etc/crash/bounds	suffix for next core file
/etc/crash/minfree	free KB in FS to maintain after savecore

SEE ALSO

na_savecore(1)

NAME

dgateways – default gateways list

SYNOPSIS

/etc/dgateways

DESCRIPTION

The **/etc/dgateways** file is used by the **routed** command to construct a set of potential default gateways. The file is comprised of a series of lines, each in the following format:

gateway metric

gateway is the name or address of a gateway to be used as a potential default gateway.

metric is a metric indicating the preference weighting of the gateway. 1 is the value to use for highest preference, 15 for the least. If no value is specified, *metric* will default to the value 1.

There can be a maximum of 128 valid entries in the **/etc/dgateways** file - additional ones will be ignored, with an error message being displayed. Duplicate gateway names or addresses are not allowed - only the first one encountered in the file will be added by **routed** to the default gateway table, and the additional ones will produce error messages.

EXAMPLE

Here are typical lines from the **/etc/dgateways** file:

```
main_router 1  
backup_router 2
```

SEE ALSO

na_routed(1), **na_setup(1)**.

NAME

dumpdates – data base of file system dump times

SYNOPSIS

/etc/dumpdates

DESCRIPTION

The **dump** command (see **na_dump(1)**) uses **/etc/dumpdates** to keep track of which subtrees have been dumped and when. Each line in **dumpdates** contains the subtree dumped, the dump level, and the creation date of the snapshot used by **dump**. There is only one entry per subtree at a given dump level. **dumpdates** may be edited to change any of the fields, if necessary.

EXAMPLE

This shows the dumpdate file for a system on which **/home** and **/export** are backed up using **dump**.

```
/home    0 Tue Nov 2 10:56:27 1993  
/export  0 Tue Nov 2 13:51:17 1993  
/export  1 Tue Nov 5 18:31:17 1993  
/home    1 Tue Nov 5 18:45:27 1993
```

FILES

/etc/dumpdates

SEE ALSO

na_dump(1)

NAME

exports – directories and files exported to NFS clients

SYNOPSIS

/etc/exports

DESCRIPTION

The **/etc/exports** file contains a list of directories and files that are exported by the filer. Changes to this file do not take effect until the filer executes the **exportfs** command or the filer is rebooted. When the filer is rebooted, it executes the **exportfs -a** command from the **/etc/rc** script to export all files and directories listed in the **/etc/exports** file.

Each export entry is a line in the following format:

pathname –*option*[,*option*] ...

The following list describes the fields in an export entry:

- pathname* path name of a file or directory to be exported.
- option* the export option specifying how a file or directory is exported. You can specify the option in one of the following formats:
- access=hostname[:hostname]...**
Give mount access to each host listed. Alternatively, you can specify a netgroup instead of a host in the list. The netgroup must be defined in the **/etc/netgroup** file. Whether the hosts can mount *pathname* with root access, read-and-write access, or read-only access depends on how you use the **root**, **rw**, and **ro** options, as described below.
- anon=uid**
If a request comes from user ID of 0 (root user ID on the client), use *uid* as the effective user ID unless the client host is included in the **root** option. The default value of *uid* is 65534. To disable root access, set *uid* to 65535. To grant root access to all clients, set *uid* to 0.
- ro** Export the *pathname* read-only. If you do not specify this option, the *pathname* is exported read-write.
- rw=hostname[:hostname]...**
Export the *pathname* read-only to all hosts not specified in the list and read-write to the hosts in the list. Netgroup names are not allowed in the list.
- root=hostname[:hostname]...**
Give root access only to the specified hosts. By default, no

hosts are granted root access. Netgroup names are not allowed in the list.

In an export entry, you can specify that a file or directory be exported to a subnet instead of individual hosts. The export entry for exporting to subnets can use the **ro**, **rw**, or **root** option; you cannot specify a subnet in the list for the **access** option.

Instead of specifying a host name or netgroup name in the entry, specify the subnet in one of the following formats:

dotted_IP/num_bits The *dotted_IP* field is either an IP address or a subnet number. The *num_bits* field specifies the size of the subnet by the number of leading bits of the netmask.

“[**network**] *subnet* [**netmask**] *netmask*”

The *subnet* field is the subnet number. The *netmask* field is the netmask.

In UNIX, it is illegal to export a directory that has an exported ancestor in the same file system. Data ONTAP does not have this restriction. For example, you can export both the / directory and the **/home** directory. In determining permissions, the filer uses the longest matching prefix.

EXAMPLES

In the following example, all network clients can mount the **/home** directory but only the **adminhost** can mount the / directory:

```
/          -access=adminhost,root=adminhost
/home
```

The following examples show different ways of specifying an export entry that exports the **/home** directory to the 123.45.67.0 subnet with the 255.255.255.0 netmask:

```
/home -rw=123.45.67.0/24
/home -rw=123.45.67/24
/home -rw="network 123.45.67.0 netmask 255.255.255.0"
/home -rw="123.45.67.0 255.255.255.0"
```

BUGS

The filer supports a maximum of 255 host names in each **rw** and **root** option. There is no limit on the number of host names in the list following the **access** option. The maximum size of the **/etc/exports** file is about 64 KB.

FILES

/etc/exports	directories and files exported to NFS clients
/etc/hosts	host name database

SEE ALSO

na_exportfs(1), na_reboot(1), na_hosts(5), na_netgroup(5), na_rc(5)

NAME

group – group file

SYNOPSIS

/etc/group

DESCRIPTION

The **/etc/group** database contains information for each group in the following form:

groupname:password:gid:user-list

The following list describes the required fields:

groupname	The name of the group.
password	The group's password, in an encrypted form. This field may be empty.
gid	An integer representing the group; each group is assigned a unique integer.
user-list	The user list is a comma-separated list of users allowed in the group.

EXAMPLE

Here is a sample group file:

```
project:asderghuIoiyw:12:dan,dave  
myproject::11:steve,jerry
```

SEE ALSO

na_nis(8), **na_pcfnfsd(8)**, **na_nsswitch.conf(5)**, **na_quota(1)**, **na_cifs_access(1)**,
na_cifs_setup(1)

NAME

hosts – host name data base

SYNOPSIS

/etc/hosts

DESCRIPTION

The **hosts** file contains information regarding the known hosts on the network. For each host a single line should be present with the following information:

Internet-address official-host-name aliases

Items are separated by any number of blanks and/or tab characters. A “#” indicates the beginning of a comment; characters up to the end of the line are not interpreted by routines which search the file.

This file may be created from the official host data base maintained at the Network Information Control Center (NIC), though local changes may be required to bring it up to date regarding unofficial aliases and/or unknown hosts.

Network addresses are specified in the conventional “.” (dot) notation. Host names may contain any alphanumeric character, but not field delimiters, newline, or comment characters.

FILES

/etc/hosts

SEE ALSO

na_hostname(1), na_dns(8), na_nis(8)

NAME

hosts.equiv – list of hosts and users with rsh permission

SYNOPSIS

/etc/hosts.equiv

DESCRIPTION

The **hosts.equiv** file contains a list of hosts on which you can enter a filer command through the remote shell protocol (**rsh**).

Hosts specified in this file are considered the trusted hosts of the filer.

Each line in **hosts.equiv** has the following format:

hostname [username]

If the host on which you enter the filer command is a UNIX host, the user name is optional. If the host on which you enter the filer command is a PC, you must enter the user name for that PC in the **/etc/hosts.equiv** file.

If you do not specify a user name for a UNIX host, you must be root on that host to execute a filer command through **rsh**.

If you specify a user name for a UNIX host, the user with that name can execute a filer command through **rsh**. However, because the only local user the filer recognizes is root, if you are a non-root user on the trusted host, you must use the following command syntax on the host to issue a filer command:

rsh -l root toaster command

When you use an **rsh** application on your PC to issue a filer command, specify that you are entering the command as root.

If multiple users on the same host should have access to the filer through **rsh**, enter each user name on a separate line.

EXAMPLE

The following **hosts.equiv** file allows both **root** and **joe_smith** to enter filer commands through **rsh** on a UNIX host named **adminhost**:

adminhost
adminhost joe_smith

If you are root on **adminhost**, use the following command syntax to execute a filer command:

adminhost# **rsh toaster command**

If your user name is **joe_smith**, use the following command syntax to execute a filer command:

adminhost% **rsh -l root toaster command**

SEE ALSO
na_rshd(8)

NAME

httpd.access – authentication controls for HTTP access

SYNOPSIS

/etc/httpd.access

DESCRIPTION

The HTTP daemon can apply authentication controls to individual users or groups on a per directory basis. The file **/etc/httpd.access** specifies the following items for each access-controlled tree:

the path to the tree

the authority required to authenticate access to the tree

the lists of users or groups to who are permitted access when authenticated

The syntax is the same as the access control syntax used by NCSA and Apache. However, the **httpd.access** file only supports a subset of directives supported by NCSA and Apache. You can copy an existing NCSA or Apache access to the file without editing or reformatting.

SYNTAX

The supported directives are:

```
<Directory directory_name>
```

```
</Directory>
```

```
AuthName Title phrase
```

```
require user user_id [, user_id, ...]
```

```
require group group_id [, group_id, ...]
```

where *Title phrase* is a word or phrase that is passed to the authentication dialog as a title for the dialog that prompts the user for a password.

EXAMPLES

The following example restricts access to the file **/home/htdocs/private/bob** so that only user dole can access it, after supplying the required password. The authentication dialog is titled “My private stuff.”

```
<Directory /home/htdocs/private/bob>
```

```
AuthName My private stuff
```

```
<Limit GET>
```

```
require user dole
```

```
</Limit>
```

```
</Directory>
```

The **<Limit GET>** and **</Limit>** directives are not supported, but are retained for format consistency with NCSA and Apache. The file just ignores them.

The following example restricts access to the directory tree **/home/htdocs/private/conspiracy** to the group “guyinblack”, which consists of the users whose IDs are cancer, depththroat, mrx, and skinner. The authentication dialog is titled “Area 51.”

```
<Directory /home/htdocs/private/conspiracy>
AuthName Area 51
<Limit GET>
require group guyinblack
</Limit GET>
</Directory>
```

In this example, “guyinblack” is defined by the following entry in **/etc/httpd.group**:

```
guyinblack: cancer depththroat mrx skinner
```

SEE ALSO

na_httpd.passwd(5), **na_httpd.group(5)**.

BUGS

Only the directives listed above are supported; other directives that may appear in NCSA or Apache access files are ignored.

NAME

httpd.group – names of HTTP access groups and their members

SYNOPSIS

/etc/httpd.group

DESCRIPTION

The file declares the names of groups, and the user IDs of the members of each group, for use by the HTTP daemon in executing the access controls declared in **/etc/httpd.access**.

SYNTAX

group_id1:user_id1 [user_id2 ...]

SEE ALSO

na_httpd.access(5).

NAME

httpd.hostprefixes – configuration of HTTP root directories for virtual hosts

SYNOPSIS

/etc/httpd.hostprefixes

DESCRIPTION

The **httpd.hostprefixes** file maps virtual hosts used in HTTP to corresponding root directories. The same configuration file is used for both IP virtual hosts (discriminated by the IP address used for connecting to the server) and HTTP virtual hosts (discriminated by the **Host:** header used in HTTP requests).

Each virtual host has a corresponding subdirectory within the directory specified by the option **httpd.rootdir**. This subdirectory is called the virtual host root directory. Clients connected to a virtual host can only access files within the virtual host root directory.

In the **httpd.hostprefixes** file, each line consists of a virtual host root directory followed by the names and IP addresses of a virtual host. If you specify an IP address, the virtual host root directory is associated with the given virtual host for IP-level virtual hosting. If you specify a name, the virtual host root directory is associated with the virtual host with that name, using HTTP-level virtual hosting. If the filer can resolve that name to an IP address, which is used for an IP-level host alias (see the **alias** option in **na_ifconfig(1)**), the filer uses that IP address in the same way as it would if you specified the IP address in the **httpd.hostprefixes** file.

If the **/etc/httpd.hostprefixes** file is edited, it is read again by the HTTP server after the changes are saved.

EXAMPLE

This example maps requests sent to **www.customer1.com** to the **customer1** subdirectory of **httpd.rootdir** and requests directed at a host with IP address 207.68.156.58 to the subdirectory **customer2**.

```
/customer1 www.customer1.com  
/customer2 207.68.156.58
```

If the command

```
toaster> ifconfig vh alias www.customer1.com
```

had been issued before the configuration file was read, requests destined for the IP address of **www.customer1.com** would also be mapped to the **/customer1** subdirectory, regardless any the **Host:** header they included.

SEE ALSO

na_ifconfig(1), **na_options(1)**

NAME

httpd.log – Log of HTTP

SYNOPSIS

/etc/log/httpd.log

DESCRIPTION

The HTTP server logs an entry for every file retrieved via HTTP. This log, written to **/etc/log/httpd.log**, is stored in the “Common Log Format,” which is used by many World-Wide Web servers.

Each entry in **/etc/log/httpd.log** consists of one line with seven fields. The fields are, in order:

address	The IP address of the HTTP client requesting the file.
rfc931	This field is always “-”.
authuser	This field is always “-”.
date	The time and date the request was is reported in the format “[Day/Mon/Year:HH:MM:SS]”, which is logged in universal time (GMT) rather than the local time zone.
request	A quoted string is recorded for the method (request type) and file involved in the request.
result	The status code for the request, as defined in RFC 1945, the HTTP protocol specification. (See below.)
bytes	The size of the file in bytes.

Possible values for *result* codes include:

200	Success: the requested file was transmitted.
302	Redirected (see /etc/httpd.translations)
304	Not modified (client cache used)
400	Bad request.
403	Access to file prohibited.
404	File not found.
503	HTTP server disabled.

The size of the log file can be restricted by the option **httpd.log.max_file_size**.

SEE ALSO

na_options(1), **na_httpd.translations(5)**

RFC 1945, ‘‘Hypertext Transfer Protocol -- HTTP/1.0’’

BUGS

Some Web servers report size statistics differently for result codes other than 200. For example, a file size of 0 is often reported for result code 304 (Not modified).

The log file grows automatically and is never reset. It is your responsibility to rotate files and empty the log files regularly.

NAME

httpd.mimetypes – map of file suffixes to MIME Content-Type

SYNOPSIS

/etc/httpd.mimetypes

DESCRIPTION

For HTTP/1.0 and higher protocols, a MIME header is returned in the reply of every GET request. This header includes a "Content-Type" field, whose contents is determined by examining the suffix of the file being transmitted.

The **/etc/httpd.mimetypes** file contains the mapping of filename suffixes to MIME Content-Type. The format of each line is: suffix, Content-Type. Comments are introduced with a "#".

The filer is not shipped with the **/etc/httpd.mimetypes** file. Instead, the filer's system files include a sample file named **/etc/httpd.mimetypes.sample**. Before you start using HTTP, make a copy of **/etc/httpd.mimetypes.sample** and name the copy **/etc/httpd.mimetypes**.

If the file **/etc/httpd.mimetypes** is not installed, the HTTP server looks for the file **/etc/httpd.mimetypes.sample** as a fallback.

EXAMPLE

```
# map .ps files to PostScript
ps      application/postscript
```

NAME

httpd.passwd – file of passwords required for HTTP access

SYNOPSIS

/etc/httpd.passwd

DESCRIPTION

The password file containing the encrypted form of the password that an HTTP client must supply to have access to a file in a controlled-access directory tree, as declared in **/etc/httpd.access**.

The password is encrypted in the regular UNIX style. User of NCSA or Apache can use their **htpasswd** program to generate the user_id:passwd pair.

The HTTP access control does not use the existing CIFS password database on the filer because in http basic authentication, in each request for protected pages, the value of *passwd* is sent over the network in clear text, and without encryption would compromise the user's password.

SYNTAX

user_id1:encrypted_passwd1

used_id2:encrypted_passwd2

...

SEE ALSO

na_httpd.access(5).

NAME

httpd.translations – URL translations to be applied to incoming HTTP requests

SYNOPSIS

/etc/httpd.translations

DESCRIPTION

The HTTP daemon supports four URL translation rules to filter incoming HTTP requests. The HTTP daemon applies each rule in succession, stopping at the first successful **Redirect**, **Pass**, or **Fail** rule:

Map *template result*

Any request which matches *template* is replaced with the *result* string given.

Redirect *template result*

Any request which matches *template* is redirected to the *result* URL. Note that this must be a full URL, e.g., beginning with "http:".

Pass *template* [*result*]

Any request which matches *template* is granted access, and no further rule processing occurs. An optional *result* can be used in place of the matching URL.

Fail *template*

Any request which matches *template* is denied access. Rule processing stops after a matched **Fail**.

Both templates and results may contain wildcards (a star "*" character). The wildcard behaves like a shell wildcard in the *template* string, matching zero or more characters, *including* the slash ("/") character. In the *result* string, a wildcard causes text from the corresponding match in the *template* string to be inserted into the result.

EXAMPLE

This example redirects CGI queries to **cgi-host**, prevents accesses to **/usr/forbidden**, and maps requests to images to a local image directory:

```
#
# Example URL translations
#
Redirect /cgi-bin/* http://cgi-host/*
Fail /usr/forbidden/*
Map /image-bin/* /usr/local/http/images/*
```

NAME

messages – record of recent console messages

SYNOPSIS

/etc/messages

DESCRIPTION

The default behavior of the filer **syslogd** daemon (see **na_syslogd(8)**) is to print all logging messages of priority **info** or higher to the console, and to the **messages** file. A typical message is:

Fri Dec 6 09:16:57 PST 1996 [rc]: NetApp Release 4.0 boot complete.

Every Saturday at 24:00, **/etc/messages** is moved to **/etc/messages.0**, **/etc/messages.0** is moved to **/etc/messages.1**, and so on. Message files are saved for a total of six weeks.

FILES

/etc/messages messages file for current week
/etc/messages.[0-5] messages file for previous weeks

SEE ALSO

na_syslogd(8), **na_syslog.conf(5)**

NAME

netgroup – network groups data base

SYNOPSIS

/etc/netgroup

DESCRIPTION

netgroup defines network wide groups used for access permission checking during remote mount request processing. Each line defines a group and has the format:

groupname member-list

Each element in member-list is either another group name or a triple of the form:

(hostname, username, domainname)

The *hostname* entry must be fully qualified if the specified host is not in the local domain.

The filer can also use the **netgroup** NIS map.

Since the filer uses netgroups only in **/etc/exports** (see **na_exports(5)**), the *username* entry is ignored. The *domainname* field refers to the domain in which the netgroup entry is valid. It must either be empty or be the local domain, otherwise the netgroup entry is ignored. This allows a single **/etc/netgroup** file to be used for filers in multiple domains.

EXAMPLE

This is a typical **netgroup** file:

```
trusted_hosts    (adminhost,,) (zeus,,) (thor,,) (minerva,,)
untrusted_hosts (sleepy,,) (dopey,,) (grumpy,,) (sneezy,,)
all_hosts        trusted_hosts untrusted_hosts
```

With this **netgroup** file it might make sense to modify **/etc/exports** to export / on the filer only to *trusted_hosts*, but to export **/home** to *all_hosts*.

FILES

```
/etc/netgroup
/etc/exports    directories and files exported to NFS clients
/etc/hosts      host name data base
```

SEE ALSO

na_exportfs(1), **na_hosts(5)**, **na_exports(5)**, **na_nis(8)**

BUGS

The only place that netgroups can be used is in the **access=** option of the **exportfs** command (see **exportfs(1)**) and **/etc/exports**.

NAME

networks – network name data base

SYNOPSIS

/etc/networks

DESCRIPTION

The **networks** file contains information regarding the known networks which comprise the Internet. For each network a single line should be present with the following information:

official-network-name network-number aliases

Items are separated by any number of blanks and/or tab characters. A “#” indicates the beginning of a comment; characters up to the end of the line are not interpreted by routines which search the file. This file is normally created from the official network data base maintained at the Network Information Control Center (NIC), though local changes may be required to bring it up to date regarding unofficial aliases and/or unknown networks.

Network number may be specified in the conventional “.” (dot) notation or as a 32 bit integer. Numbers may be specified in decimal (default), octal or hexadecimal. A number is interpreted as octal if it starts with the digit "0". A hexadecimal number must begin with "0x" or "0X." Network names may contain any printable character other than a field delimiter, newline, or comment character.

FILES

/etc/networks

NAME

nsswitch.conf – configuration file for name service switch

SYNOPSIS

/etc/nsswitch.conf

DESCRIPTION

The name service switch configuration file contains the preferred order in which name services will be contacted for name resolution by the filer. For each map, the name services to be used and the lookup order is specified in this file. Currently three name services are supported. They are local files in the /etc directory, NIS and DNS. The maps or "databases" that are supported are hosts, passwd, shadows, group and net-groups. Each line has the form:

map: order of name services

For example:

hosts: files nis dns

passwd: files nis

When trying to resolve a name, the services are contacted one by one, as per the order specified, until the name is successfully resolved. A name resolution failure occurs when no service can successfully resolve the name. When enumerating a map, enumeration happens over all the services specified for the map.

FILES

/etc/nsswitch.conf

SEE ALSO

na_setup(1)

NAME

passwd – password file

SYNOPSIS**/etc/passwd****DESCRIPTION**

The **passwd** file contains basic information about each user's account. It contains a one-line entry for each authorized user, of the form:

username:password:uid:gid:gc~~os~~-field:home_directory:login_shell

Required Fields:

username	The user's login name, not more than eight characters.
password	The user's password, in an encrypted form that is generated by the UNIX passwd function. However, if the encrypted password is stored in /etc/shadow , (see shadow(5)), the password field of /etc/passwd is empty.
uid	A unique interger assigned by the UNIX administrator to represent the user's account; its value is usually between 0 and 32767.
gid	An interger representing the group to which the user has been assigned. Groups are created by the UNIX system administrator; each is assigned a unique integer whose value is generally between 0 and 32767.
gc os -field	The user's real name. The name may be of any length; it may include capital letters as well as lower case, and may include blanks. The name may be empty.
home_directory	The user's home directory. The home directory field may be empty.
login-shell	The default shell launched at login. This field may be empty.

EXAMPLE

Here is a sample **passwd** file when **the /etc/shadow** does not exist:

```
root:bDPu/ys5PBoYU:0:1:Operator:/:bin/csh
dave:Qs5I6pBb2rJDA:1234:12:David:/u/dave:/bin/csh
dan:MNRWDsW/srMfE:2345:23:Dan::
jim:HNRYuuuMfErx:::::
```

If the system keeps the passwords in the **/etc/shadow**, the file **/etc/passwd** would be exactly the same but the password field would be empty.

```
root::0:1:Operator:/:bin/csh
dave::1234:12:David:/u/dave:/bin/csh
```

dan::2345:23:Dan::

jim::::

SEE ALSO

**na_shadow(5), na_options(1), na_nis(8), na_pcfnfsd(8), na_nsswitch.conf(5),
na_quota(1), na_cifs_access(1), na_cifs_setup(1)**

NAME

quotas – quota description file

SYNOPSIS

/etc/quotas

DESCRIPTION

The /etc/quotas file describes disk quotas that go into effect when quotas are enabled. All quotas are established on a per-volume basis. If a volume name is not specified in an entry of the /etc/quotas file, the entry applies to the root volume.

The following sample /etc/quotas file describes different kinds of quotas:

```
# Quota Target      type          disk  files
# -----
mhoward          user          500M  50K
lfine            user@/vol/home 500M
stooges          group@/vol/vol0 750M  75K
/vol/vol0/export tree          750M  75K
mhoward          user@/vol/vol0/export 50M  5K
stooges          group@/vol/vol0/export 100M 10K
*               user@/vol/home 100M  10K
*               group@/vol/vol0 500M  70K
*               user@/vol/vol0/export 20M  2K
*               group@/vol/vol0/export 200M 20K
```

The first non-comment line in the file restricts the user mhoward to 500 MB of disk space and 51,200 files in the root volume. The second line restricts the user lfine to 500 MB of disk space in the home volume, but places no restriction on the number of files he can have. You can leave the file limit blank to indicate that no limit is imposed but you cannot omit the value for disk space.

The next two lines restrict the stooges group and the /vol/vol0/export qtree to 750 MB and 76,800 files each in the root volume.

A user or group is specified by one of the following values:

- a user or group name, which must appear in the password or group database (either in the /etc/passwd or /etc/group file on the filer, or in the password or group NIS map if NIS is enabled on the filer and is being used for the password or group database);

- a numerical user or group ID;

- the pathname of a file owned by that user or group.

The user or group identifier for a user or group quota can be followed by an `@/vol/volume` string, which specifies the volume to which the quota applies. If the string is omitted, the quota applies to the root volume.

A quota of type **tree** can only be applied to a qtree, which is a directory in the root directory of a specified volume. A qtree is created with the **qtree create** or **quota qtree** command.

User and group quotas can be created inside a qtree, so that the user's or group's use of space or files within that qtree is restricted. This is done by specifying the type as **user@tree** or **group@tree** where *tree* is the name of the qtree. In the example above, we first limit overall usage in the qtree `/vol/vol0/export` and then we restrict the user mhoward to 50 MB and 5,120 files under the `/vol/vol0/export` tree. Similarly, the group stooges has been limited to 100 MB of disk space and 10,240 files under the `/vol/vol0/export` tree.

In any operation that creates files or writes to them, all applicable quotas must be satisfied. For example, the user mhoward can write to a file in the `/vol/vol0/export` tree if all of these requirements are met:

- his total disk usage in the root volume does not exceed 500 MB
- his total number of files in the root volume does not exceed 51,200
- his usage within the `/vol/vol0/export` tree does not exceed 50 MB
- his number of files within the `/vol/vol0/export` tree does not exceed 5,120
- the space already in use in the `/vol/vol0/export` tree does not exceed 750 MB
- the number of files in the `/vol/vol0/export` tree does not exceed 768,000

The asterisk (*) in the `/etc/quotas` file specifies a default user or a group quota depending on the type. Any user or group that is not specifically mentioned in the `/etc/quotas` file is subject to the limits of the default user or group. Default user or group quotas can be specified on a per qtree basis or a per volume basis.

Disk and file size limits in the third and fourth columns of the `/etc/quotas` file ends in 'K', 'M', or 'G'. 'K' indicates kilobytes (or kilofiles). That is, it multiplies the limit by 1,024. Similarly, 'M' denotes megabytes (or megafiles) and 'G' denotes gigabytes (or gigafiles). The default for the disk limit is kilobytes.

SEE ALSO

na_qtree(1), na_quota(1), na_rquotad(8)

NAME

rc – system initialization command script

SYNOPSIS

/etc/rc

DESCRIPTION

The command script **/etc/rc** is invoked automatically during system initialization. Since the file has no local editor, **/etc/rc** must be edited from an NFS client with root access to **/etc**. Alternately, you can use the **setup** command to generate a new **/etc/rc** file without using NFS.

EXAMPLE

This is a sample **/etc/rc** file as generated by **setup**:

```
#Auto-generated by setup Tue Jun 2 21:23:52 GMT 1994
hostname toaster.netapp.com
ifconfig e0 'hostname'-0
ifconfig e1a 'hostname'-1
route add default MyRouterBox 1
routed on
timezone Atlantic/Bermuda
savecore
exportfs -a
nfs on
```

FILES

/etc/rc

SEE ALSO

na_exportfs(1), **na_exports(5)**, **na_hostname(1)**, **na_hosts(5)**, **na_ifconfig(1)**,
na_nfs(1), **na_route(1)**, **na_routed(1)**, **na_savecore(1)**, **na_setup(1)**, **na_timezone(1)**,
na_autosupport(8)

NAME

resolv.conf – configuration file for domain name system resolver

SYNOPSIS

/etc/resolv.conf

DESCRIPTION

The resolver configuration file contains information that is read by the resolver routines. The file is designed to be human readable and contains a list of keywords with values that provide various types of resolver information.

The different configuration options are:

nameserver *address*

This specifies the Internet address (in dot notation) of a name server that the resolver should query. Up to 3 name servers may be listed, one per keyword. If there are multiple servers, the resolver queries them in the order listed. When a query to a name server on the list times out, the resolver will move to the next one until it gets to the bottom of the list. It will then restart from the top retrying all the name servers until a maximum number of retries are made.

search *domain-list*

This specifies the search list for host-name lookup. The search list is normally determined from the local domain name; by default, it begins with the local domain name, then successive parent domains that have at least two components in their names. This may be changed by listing the desired domain search path following the **search** keyword with spaces or tabs separating the names. Most resolver queries will be attempted using each component of the search path in turn until a match is found. Note that this process may be slow and will generate a lot of network traffic if the servers for the listed domains are not local, and that queries will time out if no server is available for one of the domains.

The search list is currently limited to six domains with a total of 256 characters.

The keyword and value must appear on a single line, and the keyword (e.g. **nameserver**) must start the line. The value follows the keyword, separated by white space.

FILES

/etc/resolv.conf

SEE ALSO

na_setup(1), **na_rc(5)**, RFC 1034, RFC 1035

NAME

rmtab – remote mounted file system table

SYNOPSIS

/etc/rmtab

DESCRIPTION

/etc/rmtab maintains the list of client mount points between server reboots. The list of client mount points can be obtained by using the **MOUNTPROC_DUMP** remote procedure call, or by using the UNIX **showmount(1)** command. When the server successfully executes a mount request from a client, the server appends a new entry to the file. When the client issues an unmount request, the corresponding entry is marked as unused. When the server reboots, unused entries are deleted from the file.

BUGS

Entries may become stale if clients crash without sending an unmount request. The file may be removed before rebooting the server in which case the server will lose information about any active client mount entries on reboot.

NAME

serialnum – system serial number file

SYNOPSIS

/etc/serialnum

DESCRIPTION

The file **/etc/serialnum** should contain the serial number of your machine. The serial number is found on the back of the machine in the lower right hand corner. You should see a tag that says:

NetworkAppliance SN: xxxx

If the file does not exist on your system, please create it and put the machine's serial number in it. The file should contain a single line that only has the serial number. The file is used to help Network Appliance's customer service group process your autosupport email more efficiently.

FILES

/etc/serialnum

NAME

shadow – shadow password file

SYNOPSIS

/etc/shadow

DESCRIPTION

The **shadow** file provides more secure storage for the user's password (which would otherwise be in **/etc/passwd**). When the password field of an entry in **/etc/passwd** is empty, **/etc/shadow** must contain a corresponding entry with the same user name but a non-empty encrypted password.

username:password:

The following list explains the required fields:

username	The user's login name, not more than eight characters.
password	The user's password, in an encrypted form that is generated by the UNIX passwd function.

There can be other fields in the **/etc/shadow** file following the ":" after the **password**.

EXAMPLE

Here is a sample **shadow password** file entry:

dave:Qs5I6pBb2rJDA:

SEE ALSO

na_passwd(5), **na_options(1)**, **na_nis(8)**, **na_pcnfsd(8)**, **na_nsswitch.conf(5)**

NAME

sm – network status monitor directory

SYNOPSIS

/etc/sm

DESCRIPTION

The network status monitor provides information about the status of network hosts to clients such as the network lock manager. The network status monitor keeps its information in the **/etc/sm** directory.

The **/etc/sm/state** file contains an integer that is incremented each time the filer is booted.

The **/etc/sm/monitor** file contains a list of network hosts the filer is monitoring.

The **/etc/sm/notify** file contains a list of network hosts that made an NLM lock request to the filer. Each time the filer reboots, it tries to notify the hosts of its new state information. You can remove this file if you want the filer to stop notifying the hosts in this file.

BUGS

If the filer cannot resolve a host name in the **/etc/sm/notify** file or if a host in the **/etc/sm/notify** file does not exist on the network any more, the filer logs an error message each time it tries to contact the host. The error message is similar to the following:

```
[sm_recover]: get RPC port for <host=host1,prog=100024,ver=1,prot=17> failed
```

To stop the error messages, remove the **/etc/sm/notify** file.

NAME

syslog.conf – syslogd configuration file

DESCRIPTION

The **syslog.conf** file is the configuration file for the **syslogd** daemon (see **na_syslogd(8)**). It consists of lines with two TAB separated fields:

```
selector      action
```

The *selector* field specifies the types of messages and priorities to which the line applies. The *action* field specifies the action to be taken if a message the **syslogd** daemon receives matches the selection criteria.

The *selector* field is encoded as a *facility*, a period (“.”), and a *level*, with no intervening white-space. Both the *facility* and the *level* are case insensitive.

The *facility* describes the part of the system generating the message, and is one of the following keywords: **auth**, **cron**, **daemon** and **kern**. Here’s a short description of each *facility* keyword:

kern	Messages generated by the filer kernel.
daemon	System daemons, such as the rshd daemon (see na_rshd(8)), the routing daemon (see na_routed(1)), the SNMP daemon (see na_snmpd(8)), etc.
auth	The authentication system, e.g. messages logged for Telnet sessions.
cron	The system’s internal cron facility.

The *level* describes the severity of the message, and is a keyword from the following ordered list (higher to lower): **emerg**, **alert**, **crit**, **err**, **warning**, **notice**, **info**, and **debug**.

Here is a short description of each *level* keyword:

emerg	A panic condition that results in the disruption of normal service.
alert	A condition that should be corrected immediately, such as a failed disk.
crit	Critical conditions, such as hard disk errors.
err	Errors, such as those resulting from a bad configuration file.
warning	Warning messages.
notice	Conditions that are not error conditions, but that may require special handling.

info	Informational messages, such as the hourly uptime message (see na_uptime(1)).
debug	Debug messages used for diagnostic purposes. These messages are suppressed by default.

If a received message matches the specified *facility* and is of the specified *level* (or a higher *level*), the action specified in the *action* field will be taken.

Multiple *selectors* may be specified for a single *action* by separating them with semicolon (“;”) characters. It is important to note, however, that each *selector* can modify the ones preceding it.

Multiple *facilities* may be specified for a single *level* by separating them with comma (“,”) characters.

An asterisk (“*”) can be used to specify all *facilities* or all *levels*.

The special *level* **none** disables a particular *facility*.

The *action* field of each line specifies the action to be taken when the *selector* field selects a message. There are four forms:

- A pathname (beginning with a leading slash). Selected messages are appended to the specified file.
- A hostname (preceded by an at (“@”) sign). Selected messages are forwarded to the **syslogd** daemon on the named host.
- /dev/console. Selected messages are written to the console.
- An asterisk. Selected messages are written to the console.

Blank lines and lines whose first non-blank character is a pound (“#”) character are ignored.

It is recommended that all **/etc/syslog.conf** files include the line

```
*.info      /etc/messages
```

so that all messages are logged to the **/etc/messages** file.

EXAMPLES

A configuration file might appear as follows:

```
# Log all kernel messages, and anything of level err or
# higher to the console.
*.err;kern.*          /dev/console

# Log anything of level info or higher to /etc/messages.
```

```
*.info                                /etc/messages

# Also log the messages that go to the console to a remote
# loghost system called adminhost.
*.err;kern.*                          @adminhost

# The /etc/secure.message file has restricted access.
auth.notice                            /etc/secure.message
```

Also see the sample configuration file in `/etc/syslog.conf.sample`

FILES

`/etc/syslog.conf` The **syslogd** configuration file.
`/etc/syslog.conf.sample` Sample **syslogd** configuration file.

BUGS

The effects of multiple selectors are sometimes not intuitive. For example “daemon.crit,*.err” will select “daemon” facility messages at the level of “err” or higher, not at the level of “crit” or higher.

SEE ALSO

na_syslogd(8), **na_messages(5)**

NAME

zoneinfo – time zone information files

SYNOPSIS

/etc/zoneinfo

DESCRIPTION

The directory **/etc/zoneinfo** contains time zone information files used by the **timezone** command (see **na_timezone(1)**). They are in standard Unix time zone file format as described below.

The time zone information files begin with bytes reserved for future use, followed by six four-byte signed values, written in a “standard” byte order (the high-order byte of the value is written first). These values are, in order:

tz_h_ttisgmtcnt	The number of GMT/local indicators stored in the file.
tz_h_ttisstdcnt	The number of standard/wall indicators stored in the file.
tz_h_leapcnt	The number of leap seconds for which data is stored in the file.
tz_h_timecnt	The number of “transition times” for which data is stored in the file.
tz_h_typecnt	The number of “local time types” for which data is stored in the file (must not be zero).
tz_h_charcnt	The number of characters of “time zone abbreviation strings” stored in the file.

The above header is followed by **tz_h_timecnt** four-byte signed values, sorted in ascending order. These values are written in “standard” byte order. Each is used as a transition time at which the rules for computing local time change. Next come **tz_h_timecnt** one-byte unsigned values; each one tells which of the different types of “local time” types described in the file is associated with the same-indexed transition time. These values serve as indices into an array of structures that appears next in the file; these structures are written as a four-byte signed **tt_gmtoff** member in a standard byte order, followed by a one-byte signed **tt_isdst** member and a one-byte unsigned **tt_abbrind** member. In each structure, **tt_gmtoff** gives the number of seconds to be added to GMT, **tt_isdst** tells whether this time is during a Daylight Savings Time period and **tt_abbrind** serves as an index into the array of time zone abbreviation characters that follow the structure(s) in the file.

Then there are **tzl_leapcnt** pairs of four-byte values, written in standard byte order; the first value of each pair gives the time at which a leap second occurs; the second gives the *total* number of leap seconds to be applied after the given time. The pairs of values are sorted in ascending order by time.

Then there are **tzl_tisstdcnt** standard/wall indicators, each stored as a one-byte value; they tell whether the transition times associated with local time types were specified as standard time or wall clock time. A local time transition specified in standard time ignores any offset due to Daylight Savings Time. On the other hand, a time specified in wall clock time takes the prevailing value of Daylight Savings Time in to account.

Finally there are **tzl_tisgmtcnt** GMT/local indicators, each stored as a one-byte value; they tell whether the transition times associated with local time types were specified as GMT or local time.

SEE ALSO

na_timezone(1)

NAME

autosupport – email notification daemon

SYNOPSIS

Data ONTAP is capable of sending email notification to Customer Support at Network Appliance and/or to other designated addressees in certain situations. The email contains useful information to help them solve or recognize problems quickly and proactively. The system can also be configured to send a short alert notification containing only the reason for the alert to a separate list of recipients. This email is sent only for critical events that might require some corrective action and can be useful for Administrators with alphanumeric pagers that can accept short email messages.

DESCRIPTION

The autosupport mechanism contacts a server system that is listening on the SMTP port (25) to send email. A list of up to 5 mailhosts can be specified and they will be tried in order to send mail out. It sends mail to up to 5 recipient email addresses. To send to more than 5 recipients, create a local exploder and add that as the recipient. The information it sends is described below.

The autosupport mechanism is triggered automatically once a week by the kernel to send information before backing up the messages file. It can also be invoked to send the information through the **options** command. Autosupport mail will also be sent on events that require corrective action from the System Administrator. And finally, the autosupport mechanism will send notification upon system reboot from disk.

The subject line of the mail sent by the autosupport mechanism contains a text string to identify the reason for the notification. The messages and other information in the notification should be used to check on the problem being reported. The following are the cases where mail is sent automatically by the system and the subject line text that identifies the reason for the notification. The events that trigger the short note emails (if a recipient list is configured) are noted below and will contain the subject line reason text string and the time of failure in the email data.

1. Weekly notification is marked "WEEKLY_LOG"
2. Data disk failure notification is marked "DISK_FAIL!!!". This event also sends the short note mail.
3. Spare disk failure notification is marked "SPARE_FAIL!!!". This event also sends the short note mail.
4. Disk scrubbing fixing disk errors is marked "DISK_SCRUB!!!".
5. Failure of a fan in the system is notified with "FAN_FAIL!!!". This event also sends the short note mail.
6. Low NVRAM battery triggers notification with "BATTERY_LOW!!!". This event also sends the short note mail.
7. If a disk shelf reports errors, notification is sent with

- "SHELF_FAULT!!!". This event also sends the short note mail.
8. If one of the power supplies in a redundant power supply unit fails, notification is sent with "POWER_SUPPLY_DEGRADED!!!". This event also sends the short note mail.
 9. If the system shuts down because it has detected that the temperature inside the filer is too high, notification is sent with "OVER_TEMPERATURE_SHUTDOWN!!!". This event also sends the short note mail.
 10. System reboot notification is sent with "REBOOT". This event also sends the short note mail.

The **setup** command does the following for the autosupport feature:

If an **adminhost** is specified, it adds an entry for **mailhost** with the same address as the **adminhost** to the **/etc/hosts** file.

The following information is sent:

1. Generation date and time stamp
2. Software version
3. System ID
4. Hostname
5. SNMP contact name (if specified)
6. SNMP location (if specified)
7. Output from **sysconfig -v**
8. The system serial number, if the system has one.
9. Currently held license codes.
10. Output from **options**
11. Output from **ifconfig -a**
12. Output from **nfsstat -c**
13. Output from **cifs stat**, **cifs sessions**, and **cifs shares**. (Included if CIFS is licensed.)
14. Output from **httpstat**
15. Output from **df**
16. Output from **df -i**
17. Output from **snap sched**
18. Output from **sysconfig -r**
19. The **/etc/messages** file

The autosupport feature is manipulated through the **options** command (see **na_options(1)**).

The options choices are:

autosupport.enable:
on,off

autosupport.mailhost:

Comma-separated list (no spaces)

autosupport.to:

Comma-separated list (no spaces)

autosupport.noteto:

Comma-separated list (no spaces)

autosupport.from:

Local user name

autosupport.doit:

text word describing reason

autosupport.enable: Default is **on**. This option is a switch to enable/disable the autosupport email feature. Customers who wish to disable autosupport permanently will need to set the option in **/etc/rc** with the command

options autosupport.enable off

autosupport.mailhost: Default is **mailhost**. Enter the list of mailhosts separated by "," and no spaces. Up to 5 hosts will be accepted. The autosupport mechanism will try to contact each listed host in turn until it gets a successful SMTP connection. The default mailhost address is set to the **adminhost** address in **/etc/hosts** with a command such as

options autosupport.mailhost mercury,venus,mars

autosupport.to: Default is **autosupport@netapp.com**. Enter the list of recipients separated by "," and no spaces. Up to 5 email addresses may be listed with a command such as

options autosupport.to sysadm,autosupport@netapp.com

autosupport.noteto: Default is an empty list (short note will not be sent). Enter the list of recipients separated by "," and no spaces. Up to 5 email addresses may be listed with a command such as

options autosupport.noteto sysadm1@pager.net,sysadm2@pager.net

autosupport.from: Default is **autosupport**. Enter an user name designated as the sender of the autosupport mail. This allows replies to the mail to be received by a responsible representative at the site.

options autosupport.from sysadm

autosupport.doit: This is a trigger to send the email out. The text word argument to this option is sent in the email subject line. This is used to identify the reason for the notification. To send system information at any time on a running system you can type

a command such as

```
options autosupport.doit SYSTEM_INFO
```

Error conditions are logged through syslog at level LOG_ERR.

CLUSTER CONSIDERATIONS

The autosupport email messages from a filer in a cluster are different from the autosupport email messages from a standalone filer in the following ways:

The subject in the autosupport email messages from a filer in a cluster reads, "Cluster notification," instead of "System notification."

The autosupport email messages from a filer in a cluster contains information about its partner, such as the partner system ID and the partner host name.

In takeover mode, if you reboot the live filer, two autosupport email messages notify the email recipients of the reboot: one is from the live filer and one is from the failed filer.

The live filer sends an autosupport email message after it finishes the takeover process.

SEE ALSO

na_options(1), na_partner(1), na_setup(1), na_hosts(5), na_rc(5), RFC821

NOTES

The autosupport mechanism is enabled by default. When the system boots it will enable the feature. If, for security or other reasons, you wish to disable this feature you should add a line in **/etc/rc** to disable it:

```
options autosupport.enable off
```

If you do keep autosupport enabled, please remember to add the following lines in **/etc/rc** with your name, phone number and site name. For example,

```
snmp contact "John - 415-555-1212"  
snmp location "Network Appliance, Engg Lab"
```

Add the lines with your information even if you do not use SNMP. This information is sent in the notification and will help Network Appliance support contact you proactively in case of a problem.

NAME

DNS – Domain Name System

DESCRIPTION

Domain Name Service provides information about hosts on a network. This service has two parts: a resolver which requests information and a nameserver which provides it.

Data ONTAP supports only the resolver. When the filer needs to resolve a host address, it first looks at the **/etc/nsswitch.conf** (see **na_nsswitch.conf(5)**) file to get the order in which various name services are to be consulted. If the name services before DNS fail in their lookup and DNS is enabled, then the DNS name server is contacted for address resolution.

DNS can be enabled on the filer by running the **setup** command (see **na_setup(1)**) or by manually editing the configuration files as described below. If DNS is enabled by running the **setup** command, then the DNS domain name needs to be entered.

Enabling DNS without the setup command:

1. Create the **/etc/resolv.conf** file (see **na_resolv.conf(5)**) with up to 3 nameservers. Each line contains the keyword **nameserver** followed by the IP address of the server. For example:

```
nameserver 192.9.200.1  
nameserver 192.9.201.1  
nameserver 192.9.202.1
```

2. Edit the **/etc/rc** file (see **na_rc(5)**) to make sure that the option specifying the DNS domain name is set and the option to enable DNS is on. For example:

```
options dns.domainname netapp.com  
options dns.enable on
```

3. Reboot the filer for these changes to take effect. If the above options commands are also entered from the console, the reboot can be avoided.

Enabling DNS with the setup command:

At setup time, one can choose to enable DNS when prompted to do so. **setup** then queries for the Internet addresses of up to three DNS nameservers.

SEE ALSO

na_setup(1), **na_rc(5)**, **na_resolv.conf(5)**, **RFC1034**, **RFC1035**

NAME

NIS – NIS client service

DESCRIPTION

The NIS client service provides information about hosts, user passwords, user groups and netgroups on a network. In NIS terminology, each of the above is referred to as the map and the specific information being looked up is called the key. For example, the hosts map is like the `/etc/hosts` file; it provides a translation from host names to IP addresses. The NIS service typically has two parts: a client component which requests information and a name server which provides it.

Data ONTAP supports only the NIS client. When the filer needs to resolve a key in a given map, it looks at the `/etc/nsswitch.conf` (see `na_nsswitch.conf(5)`) file to figure out the order in which the various databases should be consulted. For example, in case of the hosts map the lookup order may be `file, nis, dns`. This means that the filer will first consult the `/etc/hosts` file. If the host name is not found in the local file, it will then try the NIS service. If the host name is still not found, then it will attempt a DNS lookup.

The NIS client can be enabled on the filer by running the `setup` command (see `na_setup(1)`) or by manually editing the configuration files as described below. If NIS is enabled by running the `setup` command, then the NIS domain name needs to be entered.

Enabling NIS without the setup command:

1. Edit the `/etc/rc` file (see `na_rc(5)`) to make sure that the option specifying the NIS domain name is set and the option to enable NIS is on. For example:

```
options nis.domainname netapp.com
options nis.enable on
```

2. Reboot the filer for these changes to take effect. If the above options commands are also entered from the console, the reboot can be avoided. If the options are entered via the console only, they are not saved across a reboot.

Enabling NIS with the setup command:

At setup time, one can choose to enable NIS when prompted to do so. `setup` then queries for the NIS domain name.

SEE ALSO

`na_setup(1)`, `na_rc(5)`, `na_resolv.conf(5)`, `na_nsswitch.conf(5)`.

NAME

pcnfsd – (PC)NFS authentication request server

DESCRIPTION

pcnfsd provides a personal computer NFS client with the authentication services. This release supports versions 1 and 2 of the PCNFSD protocol.

When **pcnfsd** receives an authentication request, it will register the user by validating the user name and password and returning the corresponding UID and primary GID pair, and the secondary group set for PCNFSD version 2.

It will look up the user in the **/etc/shadow** file, or the **passwd.adjunct** NIS map, if present, to find the user's password. It will look up the user in the **/etc/passwd** file, or the **passwd.byname** NIS map, to find the user's UID and primary GID, and to find the user's password if there is no **/etc/shadow** file or **passwd.adjunct** NIS map.

For a PCNFSD version 2 request, it will scan the **/etc/group** file, or the **group.byname** NIS map, to find all the groups of which the user is a member. It will look up the user in the **auto.home** NIS map, if NIS is enabled, to find the user's home directory; if NIS is not enabled, no home directory will be returned.

FILES

/etc/passwd	This file should be in the format used on many flavors of UNIX (SunOS 4.x and later, 4.4BSD, System V Release 4 and later, and others).
/etc/group	This file should be in the format used on many flavors of UNIX (SunOS 4.x and later, 4.4BSD, System V Release 4 and later, and others).
/etc/shadow	This file should be in the format used on many flavors of UNIX (SunOS 5.x and later, System V Release 4 and later, and others).

SEE ALSO

na_options(1), **na_nis(8)**

BUGS

When the call fails, **pcnfsd** doesn't fake by setting the UID and the GID to acceptable values. Passwords that have been encrypted using Kerberos are not supported.

NAME

rmt – remote magtape protocol module

SYNOPSIS

/etc/rmt

DESCRIPTION

/etc/rmt is a special command that can be used by remote computers to manipulate a magnetic tape drive over a network connection; for example, the UNIX **dump** and **restore** commands often can either use **/etc/rmt** to access a remote tape, or have **rdump** and **rrestore** variants that can do so. **/etc/rmt** is normally run by the **rshd** daemon (see **na_rshd(8)**) as a result of a remote machine making a request to **rshd** to do so.

The **/etc/rmt** command accepts requests specific to the manipulation of magnetic tapes, performs the commands, then responds with a status indication. This protocol is provided by **rmt** commands on many UNIX systems, although UNIX systems may support more commands and may give more different error codes.

All responses are in ASCII and in one of two forms. Successful commands have responses of:

A*number***\n**

number is an ASCII representation of a decimal number. Unsuccessful commands are responded to with:

E*error-number***\n***error-message***\n**

error-number is one of:

2 (ENOENT)

The tape device specified in an open request did not have a valid syntax.

6 (ENXIO)

The tape device specified in an open request does not exist.

5 (EIO)

An I/O error occurred when performing the request.

25 (ENOTTY)

An invalid tape operation was specified in a “perform special tape operation” request.

error-message is a (UNIX-style) error string for the error specified by *error-number*.

The protocol is comprised of the following commands, which are sent as indicated - no spaces are supplied between the command and its arguments, or between its arguments, and **\n** indicates that a newline should be supplied:

Odevice\mode\n

Open the specified *device* using the indicated *mode*. *device* is a tape name of the form described in **na_tape(4)** and *mode* is an ASCII representation of a decimal number specifying how the tape is to be opened:

- 0** read-only
- 1** write-only
- 2** read-write

If a device had already been opened, it is closed before a new open is performed.

Cdevice\n

Close the currently open device. The *device* specified is ignored.

Lwhence\noffset\n

Performs no operation, and returns the value of *offset*; UNIX-style **lseek** operations are ignored on NetApp filer tape devices, just as they are on tape devices on many UNIX systems.

Wcount\n

Write data onto the open device. If *count* exceeds the maximum data buffer size (64 kilobytes), it is truncated to that size. **/etc/rmt** then reads *count* bytes from the connection, aborting if a premature end-of-file is encountered. The response value is the number of bytes written if the write succeeds, or -1 if the write fails.

Rcount\n

Read *count* bytes of data from the open device. If *count* exceeds the maximum data buffer size (64 kilobytes), it is truncated to that size. **/etc/rmt** then attempts to read *count* bytes from the tape and responds with **Acount-read\n** if the read was successful; otherwise an error in the standard format is returned. If the read was successful, the data read is then sent.

Ioperation\ncount\n

Perform a special tape operation on the open device using the specified parameters. The parameters are interpreted as ASCII representations of the decimal values. *operation* is one of:

- 0** write end-of-file marker
- 1** forward space *count* files
- 2** backward space *count* files

- 3 forward space *count* tape blocks
- 4 backward space *count* tape blocks
- 5 rewind the tape
- 6 rewind and unload the tape

The return value is the *count* parameter when the operation is successful.

Any other command causes **/etc/rmt** to close the connection.

DIAGNOSTICS

All responses are of the form described above.

SEE ALSO

na_tape(4), **na_rshd(8)**

NAME

rquotad – remote quota server

DESCRIPTION

The filer supports the remote quota service that allows NFS clients to determine their quota allocation on the server.

SEE ALSO

na_quota(1)

BUGS

The rquota protocol doesn't support group or tree quotas.

NAME

rshd – remote shell daemon

DESCRIPTION

The filer has UNIX-compatible remote shell capability that enables you to execute certain filer commands from a UNIX command line or shell script. It also enables you to use a remote shell application on a PC to enter filer commands.

The **/etc/hosts.equiv** file controls which hosts have access to the filer remote shell. The hosts listed in the **/etc/hosts.equiv** file are called trusted hosts. The filer only accepts commands from the root user if the commands are entered through a remote shell.

To see a list of filer commands that can be executed through **rsh**, enter the **rsh ?** command on the trusted host.

EXAMPLE

The following example shows how to enter the **version** command from a trusted host named “adminhost” through a remote shell:

```
adminhost% rsh -l root toaster version
```

SEE ALSO

na_hosts.equiv(5)

NAME

snmpd – snmp agent daemon

DESCRIPTION

The filer supports an SNMP version 1 (RFC 1157) compatible agent that provides support for both the MIB-II (RFC 1213) management information base for TCP/IP based internets as well as a Network Appliance Custom MIB.

A number of user configurable options for the SNMP agent can be set and queried from the console using the **snmp** command (see **na_snmp(1)**).

Due to weak authentication in SNMP version 1, SetRequest commands that allow the remote setting of configuration variables have been disabled.

MIB-II

Under MIB-II, information is accessible for the **system**, **interfaces**, **at**, **ip**, **icmp**, **tcp**, **udp** and **snmp** MIB-II groups. The transmission and egp groups are not supported.

The **coldStart**, **linkDown**, **linkUp** and **authenticationFailure** traps are implemented. Traps are configured using the **snmp** command.

NETWORK APPLIANCE CUSTOM MIB

The Network Appliance Custom MIB provides a means to obtain detailed information about many aspects of filer operation via SNMP. The Custom MIB can be obtained from Network Appliance's FTP site at **ftp://ftp.netapp.com/pub/netapp/mib/netapp.mib**, or by requesting the MIB on a floppy disk from Network Appliance Technical Support.

The following is a summary of the top-level groups in the Custom MIB and the information they contain:

product

Product-level information such as the software version string and system ID.

sysStat

System-level statistics such as CPU uptime, idle time and aggregate kilobytes received and transmitted on all network interfaces.

nfs

Statistics like those displayed by the **nfsstat** command (see **na_nfsstat(1)**), including statistics for each client if per-client NFS statistics have been enabled using the **nfs.per_client_stats.enable** option (see **na_options(1)**). The per-client NFS statistics are indexed by client IP addresses.

quota

Information related to disk quotas, including the output of the quota report command (see **na_quota(1)**). To access quota information, quotas must be turned on.

filesystems

Information related to the file system, including the equivalent of the **maxfiles** and **df** commands, and some of the information from the **snap list** command (see **na_df(1)**, **na_maxfiles(1)**, **na_snap(1)**).

raid

Information on RAID equivalent to the output of the **sysconfig -r** command (see **na_sysconfig(1)**).

CLUSTER CONSIDERATIONS

In takeover mode, SNMP agents can continue to access the MIBs on both filers in a cluster. However, the counters reported by SNMP are combined counters from both filers. For example, in takeover mode, the SNMP agent can report the number of packets sent or received by both filers, but you cannot determine from the number how many packets are sent or received on each filer.

You can have an application on the network management station set an alarm when a filer has been taken over. The SNMP variable to check is the **netapp.netapp1.sysStat.cf.cfSettings** variable. If this variable is set to **thisNodeDead**, the filer has been taken over.

SEE ALSO

na_df(1), **na_maxfiles(1)**, **na_nfsstat(1)**, **na_options(1)**, **na_partner(1)**, **na_quota(1)**, **na_snap(1)**, **na_snmp(1)**, **na_sysconfig(1)**

NAME

syslogd – log system messages

DESCRIPTION

The **syslogd** daemon logs system messages to the console, log files and other remote systems as specified by its configuration file, **/etc/syslog.conf**. The **syslogd** daemon reads its configuration file when it starts up during the boot procedure, or within 30 seconds after the **/etc/syslog.conf** file is modified. For information on the format of the configuration file, see **na_syslog.conf(5)**.

If **/etc/syslog.conf** does not exist the **syslogd** daemon will output all log messages of priority **info** or higher to the console and to the file **/etc/messages**. To prevent **/etc/messages** from getting too large, the **syslogd** daemon will rotate the contents of **/etc/messages** through the files **/etc/messages.0** through **/etc/messages.5**. This rotation is done once a week. So the log messages of the current week will be saved in the file **/etc/messages** and the message logs of the six weeks prior to that are saved in the files **/etc/messages.0** through **/etc/messages.5**.

To prevent large numbers of repeated messages being logged, the **syslogd** daemon will follow the first instance of a repeated message with the number of times the message was repeated. If a message is repeated over a long time period, the **syslogd** daemon will wait for increasingly longer intervals before logging the number of repeats. The repeat notification interval starts at 30 seconds and moves quickly to 20 minutes.

FILES

/etc/syslog.conf	The configuration file.
/etc/syslog.conf.sample	A sample configuration file.
/etc/messages	Message log file for current week.
/etc/messages.[0-5]	Message log for prior weeks.

CLUSTER CONSIDERATIONS

In takeover mode, the failed filer logs syslog messages to its own **/etc/messages** file and to the **/etc/messages** file on the live filer. The live filer logs its syslog messages only to its own **/etc/messages** file.

Because the **/etc/messages** file on the live filer contains syslog messages from two filers, the filer uses filer names in the syslog messages to indicate the filer from which the syslog message originated.

For example, if **toaster1** takes over **toaster2**, a message from **toaster2** is logged to the **/etc/messages** file on **toaster1**, and the message can be similar to the following:

Wed May 6 18:57:52 GMT [toaster2/toaster1]: raid_disk_admin]: Volume vol7 has been added to the system.

If the name of the failed filer is unknown, the string “partner” is printed instead of a filer name.

SEE ALSO

na_partner(1), na_messages(5), na_syslog.conf(5)