Lucent Technologies
Bell Labs Innovations

GRF® Reference Guide

1.4. Update 2

Part Number: 7820-2037-001
For software version 1.4.20 and later
September, 1999

Lucent Technologies

Customer Service

Customer Service provides a variety of options for obtaining information about Lucent products and services, software upgrades, and technical assistance.

Finding information and software on the Internet

Visit the Web site at `http://www.ascend.com` for technical information, product information, and descriptions of available services.

Visit the FTP site at `ftp.ascend.com` for software upgrades, release notes, and addenda to this manual.

Obtaining technical assistance

You can obtain technical assistance by telephone, email, fax, modem, or regular mail, as well as over the Internet.

Enabling Lucent to assist you

If you need to contact Lucent for help with a problem, make sure that you have the following information when you call or that you include it in your correspondence:

- Product name and model.
- Software and hardware options.
- Software version.
- If supplied by your carrier, Service Profile Identifiers (SPIDs) associated with your line.
- Your local telephone company's switch type and operating mode, such as AT&T 5ESS Custom or Northern Telecom National ISDN-1.
- Whether you are routing or bridging with your Lucent product.
- Type of computer you are using.
- Description of the problem.

Calling Lucent from within the United States

In the U.S., you can take advantage of Priority Technical Assistance or an Advantage service contract, or you can call to request assistance.

Priority Technical Assistance

If you need to talk to an engineer right away, call (900) 555-2763 to reach the Priority Call queue. The charge of $2.95 per minute does not begin to accrue until you are connected to an engineer. Average wait times are less than three minutes.

Other telephone numbers

For a menu of Lucent's services, call (800) 272-363). Or call (510) 769-6001 for an operator.

Calling Lucent from outside the United States

You can contact Lucent by telephone from outside the United States at one of the following numbers:

Telephone outside the United States	(510) 769-8027
Austria/Germany/Switzerland	(+33) 492 96 5672
Benelux	(+33) 492 96 5674
France	(+33) 492 96 5673
Italy	(+33) 492 96 5676
Japan	(+81) 3 5325 7397
Middle East/Africa	(+33) 492 96 5679
Scandinavia	(+33) 492 96 5677
Spain/Portugal	(+33) 492 96 5675
UK	(+33) 492 96 5671

For the Asia Pacific Region, you can find additional support resources at `http://apac.ascend.com`

Obtaining assistance through correspondence

Lucent maintains two email addresses for technical support questions. One is for customers in the United States, and the other is for customers in Europe, the Middle East, and Asia. If you prefer to correspond by fax, BBS, or regular mail, please direct your inquiry to Lucent's U.S. offices. Following are the ways in which you can reach Customer Service:

- Email from within the U.S.—support@ascend.com
- Email from Europe, the Middle East, or Asia—EMEAsupport@ascend.com
- Fax—(510) 814-2312
- Customer Support BBS (by modem)—(510) 814-2302

Write to Lucent at the following address:

Attn: Customer Service
Lucent Technologies Inc.
1701 Harbor Bay Parkway
Alameda, CA 94502-3002

Important safety instructions

The following safety instructions apply to the GRF router models GRF-4-AC, GRF-4-DC, GRF-16-AC, and GRF-16-DC except as noted:

1 Read and follow all warning notices and instructions marked on the product or included in the manual.

2 Do not attempt to service this product yourself, as opening or removing covers and/or components may expose you to dangerous high voltage points or other risks. Refer all servicing to qualified service personnel.

3 The maximum recommended ambient temperature for all GRF router models is 104˚ Fahrenheit (40˚ Celsius). Care should be given to allow sufficient air circulation or space between units when the GRF chassis is installed in a closed or multi-unit rack assembly because the operating ambient temperature of the rack environment might be greater than room ambient.

4 Slots and openings in the GRF cabinet are provided for ventilation. To ensure reliable operation of the product and to protect it from overheating, maintain a minimum of 4 inches clearance on the top and sides of the GRF 400 router, and a minimum of 6 inches on the top and sides of the GRF 1600 router.

5 Installation of the GRF 400 or 1600 in a rack without sufficient air flow can be unsafe.

6 If a GRF router is installed in a rack, the rack should safely support the combined weight of all equipment it supports.
- A fully loaded, redundant-power GRF 400 weighs 38.5 lbs (17.3 kg).
- A fully loaded, single-power GRF 400 weighs 32.5 lbs (14.6 kg).
- A four card, redundant-power GRF 1600 weighs 147 lbs (66.2 kg).
- A four card, single-power GRF 1600 weighs 127 lbs (57.2 kg).

7 The connections and equipment that supply power to GRF routers should be capable of operating safely with the maximum power requirements of the particular GRF model. In the event of a power overload, the supply circuits and supply wiring should not become hazardous.

8 Models with AC power inputs are intended to be used with a three-wire grounding type plug - a plug which has a grounding pin. This is a safety feature. Equipment grounding is vital to ensure safe operation. Do not defeat the purpose of the grounding type plug by modifying the plug or using an adapter.

9 Prior to installation, use an outlet tester or a voltmeter to check the AC receptacle for the presence of earth ground. If the receptacle is not properly grounded, the installation must not continue until a qualified electrician has corrected the problem. Similarly, in the case of DC input power, check the DC ground (s).

10 If a three-wire grounding type power source is not available, consult a qualified electrician to determine another method of grounding the equipment.

11 Models with DC power inputs must be connected to an earth ground through the terminal block Earth/Chassis Ground connectors. This is a safety feature. Equipment grounding is vital to ensure safe operation.

12 Install DC-equipped GRF 400 and 1600 routers only in restricted access areas in accordance with Articles 110-16, 110-17, and 110-18 of the National Electrical Code, ANSI/NFPA 70.

13 Do not allow anything to rest on the power cord and do not locate the product where persons will walk on the power cord.

14 Industry-standard cables are provided with this product. Special cables that may be required by the regulatory inspection authority for the installation site are the responsibility of the customer.

15 When installed in the final configuration, the product must comply with the applicable Safety Standards and regulatory requirements of the country in which it is installed. If necessary, consult with the appropriate regulatory agencies and inspection authorities to ensure compliance.

Wichtige Sicherheitshinweise

Die folgenden Sicherheitshinweise gelten für die GRF-Oberfräsenmodelle GRF-4AC, GRF-4-DC, GRF-16-AC und GRF-16-DC, außer wenn anderweitig angegeben:

1 Lesen und befolgen Sie alle am Produkt angebrachten und im Handbuch enthaltenen Warnhinweise und Anleitungen.

2 Versuchen Sie nicht, dieses Gerät selbst zu warten bzw. die Abdeckung zu öffnen oder Bauteile zu entfernen. Hochspannungsgefahr. Die Wartung muß durch qualifiziertes Fachpersonal ausgeführt werden.

3 Die empfohlene maximale Umgebungstemperatur für alle GRF-Oberfräsenmodelle liegt bei 40º C. Sorgen Sie für gute Belüftung bzw. ausreichenden Abstand zwischen einzelnen Geräten, wenn das GRF-Gehäuse in einem Einzel- oder Mehrfach-Einschubrahmen installiert werden soll, da die Betriebstemperatur in dem Einschubrahmen evtl. höher als die Raumtemperatur sein kann.

4 Schlitze und Öffnungen im GRF-Gehäuse dienen zur Belüftung. Um einen einwandfreien Betrieb des Produktes zu gewährleisten und um Überhitzung vorzubeugen, jeweils oben und an den Seiten der GRF-400-Oberfräse mindestens 10,16 cm und an der GRF-1600-Oberfräse mindesten 15,24 cm Freiraum vorsehen.

5 Bei unzureichender Belüftung ist die Installation eines GRF-400 oder 1600 in einem Einschubrahmen gefährlich.

6 Bei Installation einer GRF-Oberfräse in einem Einschubrahmen, muß dieser das Gesamtgewicht aller darin installierten Geräte sicher tragen können.

- Ein komplett bestückter Redundanzstrom-GRF-400 wiegt 17,3 kg.
- Ein komplett bestückter Einzelstrom-GRF-400 wiegt 14,9 kg.
- Ein mit vier Karten bestückter Redundanzstrom-GRF-1600 wiegt 66,2 kg.
- Ein mit vier Karten bestückter Einzelstrom-GRF-1600 wiegt 57,2 kg.

7 Die Adapter und Geräte, die die GRF-Oberfräsen mit Strom versorgen, sollten auch bei maximaler Stromanforderung des einzelnen GRF-Modells noch sicher laufen. Im Fall einer Stromüberlastung sollten die Versorgungskreise und kabel keine Gefahrenquelle darstellen.

8 Alle mit Netzeingängen versehenen Geräte müssen mit einem vorschriftsmäßigen Stecker bestückt sein. Der Stecker bietet die notwendige Erdung und darf in keiner Weise modifiziert oder mit einem Adapter verwendet werden.

9 Überprüfen Sie vor der Installation mit Hilfe eines Steckdosentestgerätes oder eines Voltmeters die Erdung der Netzsteckdose. Sollte die Steckdose nicht ordnungsgemäß geerdet sein, darf mit der Installation erst fortgefahren werden, wenn ein qualifizierter Elektriker dieses Problem behoben hat. Handelt es sich um einen Gleichstromeingang ist dieser in gleicher Weise auf ordnungsgemäße Erdung zu überprüfen.

10 Ist keine 3polige geerdete Stromquelle vorhanden, beauftragen Sie einen qualifizierten Elektriker damit, das Gerät auf andere Weise zu erden.

11 Bei Modellen mit Gleichstromeingängen muß ein Erdungsdraht entweder an der Klemmleiste oder an einer Gehäuseschraube angeschlossen werden. Hierbei handelt es sich um eine Sicherheitseinrichtung. Die Erdung des Gerätes ist eine wichtige Voraussetzung für den sicheren Betrieb.

12 Die gleichstromausgerüsteten Oberfräsenmodelle GRF-400- und GRF-1600-Oberfräse dürfen nur in Bereichen mit beschränktem Zugang, unter Berücksichtigung der anwendbaren Bestimmungen für Elektroinstallationen sowie der Standards ANSI/NFPA 70 installiert werden.

13 Keine Gegenstände auf das Netzkabel stellen. Das Kabel so verlegen, daß Personen nicht versehentlich darauf treten können.

14 Standardkabel sind im Lieferumfang des Produkts enthalten. Sonderkabel, die evtl. gemäß den örtlichen Bestimmungen für die Installation erforderlich sind, sind vom Kunden zu stellen.

15 Zur Installation in der endgültigen Konfiguration muß das Produkt den am Installationsort geltenden Sicherheitsstandards und bestimmungen entsprechen. Genauere Informationen erhalten Sie ggf. bei den zuständigen Behörden.

Contents

Figures

About This Guide

This guide provides usage information about system commands and configuration files for network administrators who install and configure the GRF. These include the CLI and GRF commands, and the GRF-adapted UNIX commands. Copies of the system configuration files are included in Chapter 2. These are the`.conf` files from the `/etc` directory.

Unless otherwise noted, information in this Guide applies to GRF 400,GRF 1600, and GR-II systems using RMS nodes.

About 1.4 Update 2

The 1.4 GRF manual set is updated to include new features added since software release 1.4.12.

This manual describes the commands and configuration files for GRF® units running software version 1.4.20 and later. User documentation for any commands or files added in a subsequent release will appear in an addendum. Some commands or files may be different in older versions or specialty loads of the software. Unless otherwise noted, the information in this guide applies to GRF 400 and GRF 1600 systems, as well as GRF 400 and GR-II systems using an RMS node.

What is in this guide

This guide contains two chapters:

- Chapter 1, "System Commands," describes the set of GRF-specific and UNIX-like commands that monitor and manage system, memory, and interface functions.
- Chapter 2, "Configuration File Templates," contains the templates for all GRF `/etc/xxx.conf` configuration files.

This guide also includes an index.

What you should know

Configuring and monitoring the GRF requires that a Network Administrator have experience with and an understanding of UNIX systems, and the ability to navigate in a UNIX environment. Knowledge of UNIX, its tools, utilities, and editors is useful, as is experience with administering and maintaining a UNIX system.

Configuring the GRF requires network experience and familiarity with:

- UNIX systems and commands
- IP protocol and routing operations
- IP internetworking

The Network Administrator must understand how TCP/IP internetworks are assembled; what interconnections represent legal topologies; how networks, hosts, and routers are assigned IP addresses and configured into operation; and how to determine and specify route table (routing) information about the constructed internetwork(s). Although not required, a high-level understanding of SNMP is useful.

Documentation conventions

Lucent uses standard documentation conventions, which are as follows:

Convention	**Meaning**
`Monospace text`	Represents text that appears on your computer's screen, or that could appear on your computer's screen.
Boldface	Represents command names or characters that you enter exactly as shown (unless the characters are also in ***italics***—see *Italics*, below). If you could enter the characters but are not specifically instructed to, they do not appear in boldface.
Italics	Represent variable information. Do not enter the words themselves in the command. Enter the information they represent. In ordinary text, italics are used for titles of publications, for some terms that would otherwise be in quotation marks, and to show emphasis.
[ ]	Square brackets indicate an optional argument you might add to a command. To include such an argument, type only the information inside the brackets. Do not type the brackets unless they appear in bold type.
|	Separates command choices that are mutually exclusive.
Key1-Key2	Represents a combination keystroke. To enter a combination keystroke, press the first key and hold it down while you press one or more other keys. Release all the keys at the same time. (For example, Ctrl-H means hold down the Control key and press the H key.)
Note:	Introduces important additional information.
⚠ **Caution:**	Warns that a failure to follow the recommended procedure could result in loss of data or damage to equipment.
⚠ **Warning:**	Warns that a failure to take appropriate safety precautions could result in physical injury.

Documentation set

The GRF 1.4 Update 2 documentation set consists of the following manuals:

- *GRF 400/1600 Getting Started - 1.4 Update 2*
- *GRF Configuration and Management - 1.4 Update2*
- *GRF Reference Guide - 1.4 Update2* (this manual)
- *GRF GateD Manual- 1.4 Update 2*

Related publications

Here are some related publications that you may find useful:

- *Internetworking with TCP/IP,* Volume 1 and 2, by Douglas E. Comer, and David L. Stevens. Prentice-Hall,
- *TCP/IP Illustrated,* Volumes 1 and 2, by W. Richard Stevens. Addison-Wesley, 1994.
- *Interconnections*, Radia Perlman. Addison-Wesley, 1992. Recommended for information about routers and bridging.
- *Routing in the Internet*, by Christian Huitema. Prentice Hall PTR, 1995. Recommended for information about IP, OSPF, CIDR, IP multicast, and mobile IP.
- *TCP/IP Network Administration*, by Craig Hunt. O’Reilly & Associates, Inc. 1994. Recommended for network management information.
- *Essential System Administration*, Æleen Frisch. O’Reilly & Associates, Inc. 1991. Recommended for network management information.

System Commands

1

Chapter 1 describes the GRF system commands in the following order:

Introduction to the GRF user environment

GRF configuration and management tasks are divided between two interfaces:

– a command-line interface (GRF and CLI commands)

– a UNIX shell (GRF and standard UNIX commands)

There is also a maintenance mode in which **maint** commands return media card-specific statistics and configuration data.

Configuring media cards, interfaces, and protocols is performed in the UNIX shell. Most of this is done in the `/etc` directory, using a UNIX editor to edit the `/etc/xxxxxx.conf` configuration files. GRF system memory does not provide space to contain the entire set of UNIX man pages although the GRF man pages are available.

Certain system management tasks are done in the CLI. These include specifying system-wide defaults for dump and executable binary files in the Dump and Load profiles, and media card settings such as FDDI SAS and DAS. Each slot has a Card profile in which those card-specific settings are made for individual media cards. The CLI has a set of profile management commands such as **read**, **write**, and **get**. Additionally, more than 60 of the system commands described in this chapter are available at both the CLI and UNIX prompts.

Chapter 1 includes the CLI commands, the GRF commands, and the GRF-modified UNIX commands. They are arranged alphabetically, so it is convenient to find a command. UNIX commands are not described in this manual unless the command has been adapted to the GRF, **ping** is one example.

At the beginning of each command description, under the command name, is a marker that tells you in which interface you can use a particular command:

(CLI)	-	only from the CLI
(CLI+shell)	-	from both CLI and UNIX shell
(grrmb)	-	only after you enter the **grrmb** command, available from both the CLI and the UNIX shell
(gsm)	-	prompt at which you enter **gsm** subcommands while in the CLI (**gsm** prompt from the UNIX shell includes host domain name)
(shell)	-	only from the UNIX shell

Systems with RMS nodes

If your system uses an RMS node and is upgraded to the current Lucent OS software release, your original commands are still usable. Certain commands operate only on newer GRF routers with an updated control board and no RMS node, those commands are described as such in this manual. If you enter one of those commands, you get a “`may only be used on` GRF” message but nothing else happens. These four commands operate only on RMS-node systems: **grinstall**, **grc**, **grms**, and **pwrfaild**.

A note about grinchd.conf

The CLI has replaced `/etc/grinchd.conf`. The first time you install 1.4 software over a prior release, all the `/etc/grinchd.conf` settings are automatically transferred to their proper places in CLI profiles.

Applying commands to hardware

The GRF 400 has four media card slots, the GRF 1600 and GR-II have 16. Because of this, some examples in this manual use slots 0–3, others use 0–15. Particular slot numbers are not significant in examples.

Slot number systems

Any of three numbering systems can be applied to the chassis slots, decimal is commonly used.

–	decimal in the form: [1-9]	=	slot 1
–	hexadecimal in the form: 0x	=	slot 0x1
–	octal in the form: 0	=	slot 01

To avoid confusion with octal, do not preface decimal numbers with 0.

GRF 400 slot numbers

Slots are numbered top to bottom as shown in Figure 1-1, the control board is always 66:

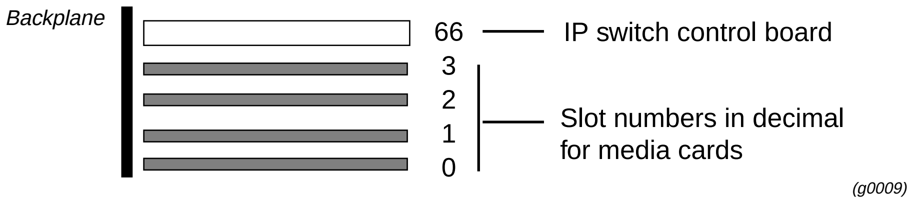


Figure 1-1. Side view of GRF 400 chassis with slots numbered

GRF 1600 slot numbers

The GRF 1600 has 16 media card slots, 0–15, a control board, and a switch board. Slots are numbered left to right as shown in Figure 1-2, the control board is always 66:

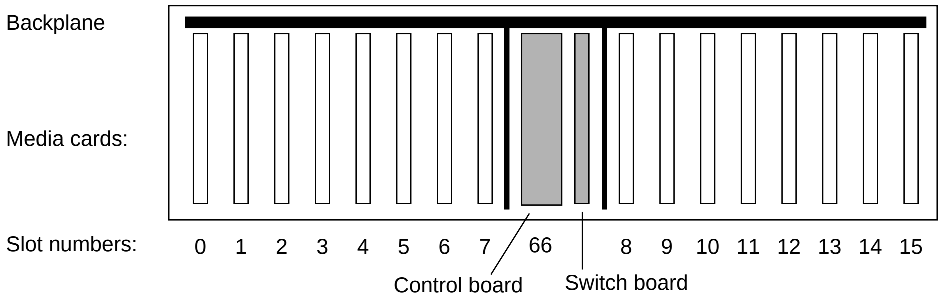


Figure 1-2. Top down view of GRF 1600 chassis with slots numbered

GR-II slot numbers

Slots are numbered left to right as shown in Figure 1-3, the router manager board is always 66:

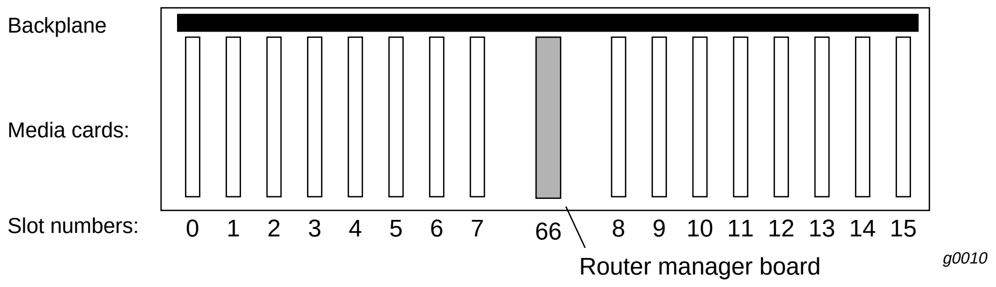


Figure 1-3. Top down view of GR-II chassis with slots numbered

Control board memory

The GRF 400 and GRF 1600 control boards have a different memory architecture from the RMS node. On the GRF systems, the RMS disk and SRAM are replaced by system RAM, internal 85MB flash, and, optionally, external PCMCIA storage devices.

To help you use the GRF memory commands, refer to Figure 1-4 for a diagram of memory organization and components.

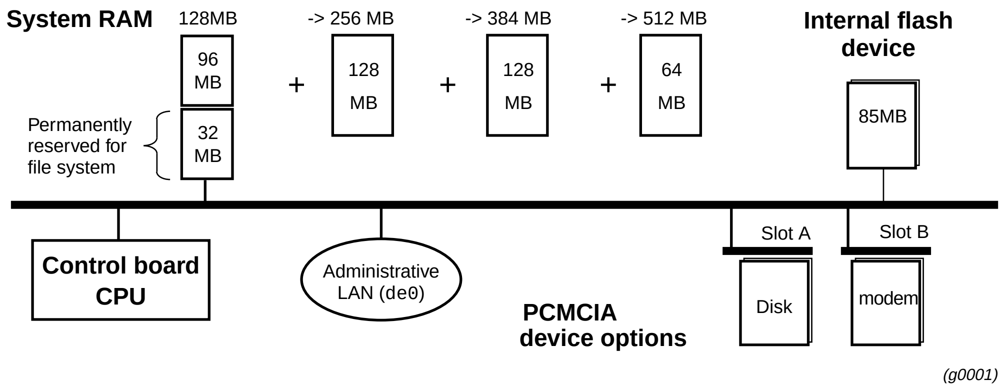


Figure 1-4. GRF memory architecture and options

RAM The GRF is equipped with a minimum of 128MB of system RAM, 32MB of which are permanently reserved for the file system, including configuration files. The remaining 96MB are used by the operating system and user applications such as GateD. System RAM can be upgraded to a maximum of 512MB in increments of 128MB.

In the CLI, use the CLI **mem** command to display the amount of system RAM installed on the GRF control board, enter:

```
super> mem ?
System memory 256 Mbytes
```

You can also determine the amount of RAM by reading the System profile, it is specified in the physical memory field:

```
super> read system
super> list
      .
      .
      .
physical-memory = 256
```

Internal flash The GRF has an internal 85MB ATA flash device from which the system boots. This memory is available for storing different versions of site configurations and operating software.

External flash PCMCIA slots on the control board support various sizes of ATA flash devices. Although external flash can be used to back up and share router configurations among multiple GRF systems, a GRF cannot boot from an external flash device. Local logging and dumping to external flash is also supported. The **grwrite** command writes files from system RAM to internal flash, **grsnapshot** copies files between internal and external flash devices. Commands for flash device management are discussed in the *GRF Configuration and Management* manual.

PCMCIA devices PCMCIA slot A is used for a portable external disk device. A PCMCIA modem device can operate in either slot A or B.

Command conventions

- In the CLI, all commands are lower case.
- In the UNIX shell, commands are case sensitive.
- Use a space to separate the command from any arguments.
- Expressions in italics are variables: *slot number*
- Expressions enclosed in brackets are optional: [*ipadd*] (broadcast ignore)

grrmb command summary

In this series, the user starts the GR##> prompt, displays a list of commands, and then exits **grrmb** to return to the CLI prompt:

```
super> grrmb
GR 66> ?
Lucent Commands
        ?
        bignore     (broadcast ignore on, off)
        fan
        maint -[hi|fd] [<port>:]<data> [,<data>[...]]
        power
        port <port>
        temp
GR 66> quit
super>
```

Here are brief descriptions of **grrmb** commands; **reset**, **echo**, **help**, and **ver** are not available.

`?`

Lists **grrmb** command set.

`bignore`

Displays broadcast ignore status, on or off.

`fan`

Displays chassis fan RPMs or on/off status.

Output for a GRF 400:

```
GR 02> fan
Fan 0 : Curr 75 Last 9 Diff 66
Fan 1 : Curr 229 Last 162 Diff 67
Fan 2 : Curr 179 Last 112 Diff 67
```

Output for a GRF 1600, 1 = on, 2 = off:

```
GR 15> fan
Fan 0 : Curr 1 Last 0 Diff 1
Fan 1 : Curr 1 Last 0 Diff 1
```

`maint`

Operates media-specific commands, a different set for each type of media. Descriptions are in the media configuration chapters in the GRF *Configuration and Management* manual.

`power`

Displays on/off status of GRF 1600 power supplies:

```
GR 15> power
power {1|2} {on|off}
port number
Sets GRF slot number, from 0–3 or 0–15.
temp
Returns current temperatures measured at the control board:

                       LT     HT    TEMP
                   ----------------------
 Measured Junction     55 C   60 C   34 C
Calc. Ext. Ambient     49 C   54 C   28 C
```

? or help
(CLI)

help and its alias **?** return a list of all registered commands authorized by user's security profile. Use **help** or **?** followed by a specific command name to obtain description or usage information.

Permission level: user

Syntax

`help, ?`

`help` *`command-name`*

`?` *`command-name`*

Example

```
super> ?  whoami
whoami  - output the user profile name associated with this session
super>
```

aitmd
(shell)

The **aitmd** program is a daemon process that is responsible for managing IP tunnels using the Ascend Tunnel Management Protocol (ATMP). **aitmd** can support a large number of ATMP tunnels to many home networks (HN). At present the GRF functions as a home agent operating in gateway mode. That is, traffic from a mobile node that has been tunneled to the GRF by a foreign agent (FA) is de-encapsulated and forwarded directly to the appropriate home network.

Messages are logged via **syslog** for major events such as system start up and reconfiguration, and for operation errors such as failed negotiation attempts.

aitmd communicates with the frame relay daemon, **fred**, to get the status of gateway circuits to the home networks. **aitmd** command options are used primarily for debugging.

By default, **aitmd** is not running. To enable ATMP, the administrator creates the file `/etc/aitmd.run`. This is done in the UNIX shell with the command:

```
# touch /etc/aitmd.run
```

Save the configuration change with:

```
# grwrite -v
```

The daemon starts within 15 seconds and restarts automatically on future reboots. The file is removed normally with **rm** and the removal again saved with **grwrite**. Removing `/etc/aitmd.run` does not kill the daemon, it only prevents **aitmd** from being restarted.

The daemon loads the ATMP configuration from the file `/etc/aitmd.conf.` The configuration can be changed on the fly by editing the configuration file and then sending the process a HUP signal. Please see Chapter 2 for a template of the `/etc/aitmd.conf` file.

Syntax

```
aitmd [-F]  [-f config_file]  [-h]  [-l log_file]  [-L logopt]  [-n]
[-q]  [-Q]  [-d]  [-P pvc_fake_state]
```

Options

`-F`

Sets **aitmd** to run in the foreground, used during interactive debugging.

By default, the daemon disconnects from the controlling tty and forks itself into the background. The **`-F`** flag is useful during debugging and to prevent a shell wrapper from losing track of the daemon.

`-d`

Enables debug output

`-f` *`config_file`*`]`

Loads the configuration from the named file rather than the default `/etc/aitmd.conf.`

`-h`

The help flag, prints a brief usage message.

`-l` *`log_file`*

Sends log messages to the named file.

This implicitly enables file logging in addition to the **syslog** logging.

`-n`

Instructs **aitmd** to check the syntax of the `/etc/aitmd.conf` configuration file.

`-q`

Sets operation to quiet mode in which only major log messages are generated.

`-Q`

Sets operation to very quiet mode in which no log messages are generated.

`-P` *`pvc_fake_state`*

This option tells the daemon to not check the real state of the ATMP circuits, and to assume that all circuits are in the given state. ***pvc_fake_state*** must be set to either up or down. This option is sometimes useful for testing configuration files for **aitmd** when the circuits are not actually up.

`-L` *`logoption`*

Sets a logging option.

The arguments for a logging option are a pair of letters such as **LN**. The first letter is the name of the logging category or control that the option applies to, and the second is the option setting.

You can group logging options. For example, if you are debugging a complicated problem and want to run the daemon in the foreground, disable syslogging, enable logging to **stderr,** and see all logging categories, use this group of options (no spaces are needed between the grouped options **TsTfLA**):

```
# /usr/sbin/aitmd -F -L TsTFLA
```

The command and options can also be entered as:

```
# /usr/sbin/aitmd -F -L Ts -L TF -L LA
```

Logging can be targeted to go to **syslog**, or to a file, or both. If a file name is not specified, then "file" log output goes to **stderr**. The second letter is typically an on/off toggle, with upper case denoting on and lower case denoting off.

-L Ts	disable logging to syslog
-L TS	enable logging to syslog, default
-L Tf	disable logging to file or **stderr**, default
-L TF	enable logging to stderr or to a file, implied by -f, described above

If **aitmd** generates large bursts of logging output to **syslog**, such as when dumping state information in response to a SIGINFO signal, the output may be lost. This problem is not present if **aitmd** is logging to a file.

Logging can be enabled and disabled by individual message category. There are four categories of messages: notice, operational error, debug, and internal errors. The category can also be wildcarded to apply the control to all categories at once.

These messages are syslogged at the **syslog** LOG_NOTICE logging level and include non-error events such as start up and reconfiguration.

-L Ln	disable logging notice messages
-L LN	enable logging notice messages, default

These errors are syslogged at the **syslog** LOG_WARNING logging level and are transient errors that can occur during normal operations, such as hosts being unreachable or authentication failures.

-L Lo	disable logging operational errors
-L LO	enable logging operation errors, default

These messages should never be seen during normal operations and usually indicate a bug. They are syslogged at the **syslog** LOG_ERR logging level.

-L Ld	disable debug log messages, default
-L LD	enable debug log messages

These are syslogged at the **syslog** LOG_DEBUG logging level.

-L Le	disable internal error log messages
-L LE	enable internal error log messages, default
-L La	disable all log message categories
-L LA	enable all log message categories

Additional options are available to tailor what information is displayed in the log messages.

These controls can be applied separately to each category, or to all categories at once. The first letter denotes the message category:

n for notices

o for operational errors

d for debug messages

e for internal errors

A to apply the control to all categories at once (wildcard)

The examples below all use the "**n**" (notice) category, but any category, or the wildcard **A**, could be used.

-L nt	do not include timestamp in notice log messages
-L nT	include timestamps in notices
-L np	do not include process IDs (pids) in notices
-L nP	include timestamps in notices

Syntax check aitmd -n

The **aitmd -n** command checks the syntax of the ATMP configuration file. It describes the error and reports the number of the line at which the error occured. By default, the configuration file is `/etc/aitmd.conf`. If the **-f** option is used in conjunction with the **-n** option, then the file specified with the **-f** option is checked. This example shows errors found for GRF `box1`:

```
# aitmd -n
Sep  8 14:44:07 box1 aitmd: Checking syntax for config file :
"/etc/aitmd.conf"
Sep  8 14:44:07 box1 aitmd: WARNING: No default_foreign_agent
declaration found in file
Sep  8 14:44:07 box1 aitmd: SYNTAX ERROR: Invalid addr value '10.9..1'
at line 19
Sep  8 14:44:07 box1 aitmd: SYNTAX ERROR: Invalid character encountered
‘&’ on line 47
Sep  8 14:44:07 box1 aitmd: 14:44:07 (pid=2456) OpError: Syntax
error(s) found in aitmd config file.
```

Signals

The tunnel management daemon **aitmd** handles a number of different signals. Signals are sent from the UNIX shell using the **kill** command.

HUP

This signal reloads the configuration file and reconfigures ATMP. If a home network or foreign agent entry is deleted from the configuration file, then any tunnels associated with the agent or network are shut down. The status of the home network circuits is also rechecked. Tunnels that are negotiated before the reconfiguration and are valid in the new configuration continue to operate without interruption.

INFO

This signal causes **aitmd** to log its current configured state. If a large number of tunnels are configured, this can take some time, and **aitmd** does not process new messages until this logging is complete.

TERM

This signal shuts down all tunnels, terminates **aitmd**, and exits. Note that the tunnel manager is started by **grstart** and a "shell wrapper” exists to restart the daemon. Normally **aitmd** is automatically restarted after receiving this signal.

auth
(CLI)

This command selects and authenticates a User profile. Use this command to increase or decrease the permissions of the current login.

Permission level: user

Syntax

```
auth user-name
```

Example

If you supply the proper password for that User profile, the GRF enables the privileges in that profile and then displays the system prompt again. Note that the User profile may specify its own system prompt, which is a useful way to flag certain permissions levels; for example:

```
GRF> auth admin
Password:
admin>
```

If you supply the wrong password at the prompt, you’ll see this message:

```
Login incorrect
User:
```

Enter the user name again and the Password prompt is displayed.

biosver
(CLI+shell)

The **biosver** command prints out the BIOS release date. The date is used to determine which BIOS version is installed on a GRF system. The command does not work on RMS node systems.

Permission level: system

Syntax

```
biosver
```

Example

```
super> biosver
  Date BIOS was built: 04/17/98
```

bredit
(CLI+shell)

The **bredit** utility is used to access and edit the `/etc/bridged.conf` configuration file.

bredit opens the configuration file in the **vi** editor. After you make changes, you exit the file with the **vi** exit file **:q** command. Chapter 2 includes a copy of the `/etc/bridged.conf` file.

At this point **bredit** runs a script in which you are asked if you want to make the changes permanent. The script also gives you the option of signaling **bridged** to re-read the updated file immediately. When this option is taken, **bridged** restarts as if it was stopped and restarted for the first time. If you change the file in **vi** but do not choose either of the script options, **bredit** tells you that your changes were not committed.

If **bridged** is not running when **bredit** is used, the user is given the option of saving changes to the configuration in `/etc/bridged.conf` so that the next time **bridged** is started, the new changes take effect.

Syntax

```
bredit
```

brinfo
(CLI+shell)

The **brinfo** command is used to retrieve bridging interface information for administrative debugging and other situations where a simple checking of bridge port information is needed.

brinfo prints the list of bridge groups and the bridge ports (underlying interfaces) that are members of the specified group(s). If no groups are specified, all groups are reported on by default. **brinfo** gets its information directly from the BSD kernel whereas **brstat** gets its information from **bridged**.

Syntax

```
brinfo bridge_group | all
```

brstat
(CLI+shell)

The **brstat** command provides a snapshot of state information directly from **bridged**. A short lag occurs between the time a request is made and when an active **bridged** returns the information. **brinfo** gets its information directly from the BSD kernel whereas **brstat** gets its information from **bridged**.

Syntax

```
brstat
```

cd
(CLI)

Lists the field in the current profile. After a profile is read, you can view the contents of that profile by using the **cd** command (**cd** is an alias for the **list** command). A second use is to move back (or up) in a profile. The **cd ..** command moves you back to the level you were viewing prior to the last **list** or **cd** entered.

See also: **list**, **get**, **ls**

Permission level: user

Syntax

```
cd [field-name] [field-index] [...]
```

```
cd .. [ .. ]
```

Example

cd displays the fields in the User default profile after the profile is read:

```
super> read user default
USER/default read

super>cd
name* = default
password = ""
auth-method = { PASSWD { "" 1645 udp "" } { 5500 udp 5510 tcp +
active-enabled = yes
allow-system = no
allow-update = no
allow-password = no
allow-debug = no
prompt = GRF=>
log-display-level = none
```

clear
(CLI)

The **clear** command clears the screen and moves the prompt to the top line of the screen.

Permission level: user

Syntax

```
clear
```

csconfig
(shell)

csconfig is a UNIX command that sets a PCMCIA slot interface on (up) or off (down), and reports general interface and device status. This command is useful for remote management of PCMCIA devices just to verify the status of device and slot interface readiness. If an external device fails or otherwise cannot be reached, error messages are sent to the `gr.console` log, they are not handled by **csconfig**.

Syntax

```
csconfig slot_number [up | down]
```

```
csconfig -a [up | down]
```

Options

`-a`

Apply command to all (both) slots.

`down`

Set PCMCIA interface off, disable link to kernel.

`slot_number`

PCMCIA slots are A and B, numbers 0 and 1, respectively.

`up`

Set PCMCIA interface on, enable link to kernel.

Examples

Turn the interface in slot A (number 0) off before removing the PCMCIA card:

```
# csconfig 0 down
```

Use the command to determine the status of the interface link to the kernel (up or down) and the state of the slot (empty or running). In this example, the interface can communicate with the kernel (up), but slot A is empty. When the interface is up, the slot is ready for a device to be inserted.

```
# csconfig 0
Slot 0: flags=0x5<UP,EMPTY>
```

In the next example, the interface is up and slot B contains an ATA flash device:

```
# csconfig 1
Slot 1: flags=0x3<UP,RUNNING>
        Attached device: wdc1
        Manufacturer Name: "SunDisk"
        Product Name: "SDP"
        Additional Info1: "5/3 0.6"
        Function ID: 4 (PC card ATA)
        Assigned IRQ: 11
        Assigned I/O port1: 0x3d0-0x3df
```

Use the **-a** option to return status for both slots, or to set both slots either up or down:

```
# csconfig -a
Slot 0: flags=0x5<UP,EMPTY>
Slot 1: flags=0x5<UP,EMPTY>
```

csctl
(shell)

The **csctl** is a UNIX utility for **csctld**, and is rarely needed by users. The **csctl** utility is used to configure and control PCCARD slots. If no arguments are specified, a list of possible PCMCIA device configurations are displayed.

Syntax

`csctl [-acdlqr] [ conf_file ]`

Options

`-a`
: Add a new device.

`-c`
: Check a configuration file.

`conf_file`
: If ***conf_file*** is not specified, `/etc/pccard.conf` is used, if required.

`-d`
: Run a daemon other than **csctld**.

`-l`
: Load in a new configuration.

`-q`
: Be quiet when the device is not configured.

`-r`
: Reset the configuration.

date
(CLI)

Returns current time and date.

Permission level: update

Syntax

`date`

Example

```
super> date
Tue Aug 12 19:02:35 1999
```

delete (CLI)

The delete command is used to remove an instance of a profile from local storage. The specification of ***profile-index*** or ***profile-type*** is not optional. Only indexed profiles can be deleted. The Load, Dump, and System profiles cannot be deleted since only one of each exists.

⚠ **Caution:** After a deletion, the profile is permanently gone, there is no undelete or undo.

Permission level: update

Syntax

```
delete profile-type profile-index
```

Examples

You are always queried when you issue a **delete** command:

```
super> delete user operator
Delete profile USER/operator? [y/n] y
USER/operator deleted
```

If the profile does not exist, you get a message:

```
super> delete user operator
error: specified profile not found
super>
```

You can delete a field in a profile:

```
super> read card 1
CARD/1 read
super> delete port 1
Delete ports/1? [y/n] y
ports/1 deleted
```

dir
(CLI)

Returns a list of the top level or main-line profiles.

Permission level: user

Syntax

```
dir [ profile-type [ profile-index ] ]
```

Example

To see the list of available profile types:

```
super> dir
CARD                   Card info
DUMP                   System dump information
LOAD                   System load information
SYSTEM                 System info
USER                   Administrative user accounts
```

A **dir** request for a profile of a specific type returns the information for every known instance of that profile type.

In the output, the first column is the size of the profile in bytes, the second and third columns are the date and time the profile was last modified, and the fourth column lists the indices of the profiles.

```
super> dir user
92  10/09/96 11:19:31  default
103 10/09/96 11:19:31  admin
106 10/09/96 11:19:31  super
```

A **dir** request for a specific profile:

```
super> dir user default
92  10/09/96 11:19:31  default
```

exit
(CLI)

This command exits a user from the command-line interface (CLI), same as **quit**.

Syntax

```
exit
```

Permission level: user

fan
(grrmb)

This **grrmb** command displays the RPMs for GRF 400 chassis fans and the fan on/off status for GRF 1600 chassis.

Syntax

```
fan
```

Examples

Here is output for a GRF 400:

```
GR 02> fan
Fan 0 : Curr 75 Last 9 Diff 66
Fan 1 : Curr 229 Last 162 Diff 67
Fan 2 : Curr 179 Last 112 Diff 67
```

You will see marked changes if you execute **fan** several times in a row. This is normal because the GRF 400 fans are constantly changing speed.

```
GR 1> fan
Fan 0 : Curr 127 Last 66 Diff 61
Fan 1 : Curr 206 Last 144 Diff 62
Fan 2 : Curr 194 Last 132 Diff 62
GR 1> fan
Fan 0 : Curr 176 Last 87 Diff 89
Fan 1 : Curr 7 Last 172 Diff 91
Fan 2 : Curr 254 Last 163 Diff 91
```

Here is what you will see if fan 0 has stopped:

```
GR 1> fan
Fan 0 : Curr 0 Last 0 Diff 0
Fan 1 : Curr 206 Last 144 Diff 62
Fan 2 : Curr 194 Last 132 Diff 62
```

Here is output for a GRF 1600, 0 is off, 1 is on:

```
GR 15> fan
Fan 0 : Curr 1 Last 0 Diff 1
Fan 1 : Curr 1 Last 0 Diff 1
```

fastboot
(CLI+shell)

On RMS node-based systems, **fastboot** reboots the system. File systems are not checked at reboot time when you use **fastboot** and there is no command query. This allows the management software to load faster.

On GRF systems with the new control board, **fastboot** is the same as **reboot**.

Permission level: system

Syntax

```
fastboot [ -d -i -n -q ]
```

Options

`-d`

If the **-d** option is specified, the system will make a ``crash dump" (debugging image) before halting or rebooting.

`-i`

The **-i** option is only available on the GRF system.

This option is an override flag to ignore file system check. The GRF system has a built-in safety feature where it prevents the administrator from accidentally rebooting the system without saving the configuration files in `/etc`. It determines this by running the **grwrite** command. If the **grwrite** command exits with a non-zero exit status, then a warning message is printed and the shutdown operation is aborted. The **-i** option is used to override this safety feature.

This option is not applicable on a non-GRF system.

`-n`

If the **-n** option is specified, the file system cache is not flushed.

This option is not recommended for use.

`-q`

If the **-q** option is specified, the system is halted or restarted quickly and ungracefully, and only the flushing of the file system cache is performed.

This option is not recommended for use.

flashcmd
(CLI+shell)

The **flashcmd** command is used to execute a command against a specified flash device.

flashcmd mounts the flash device, performs the specified command against it, and then unmounts the flash device. If no device is specified, the default action is to use the device from which the system was loaded.

flashcmd performs a substitution of any %R found in the command string with the string specified in the [**-r** ***revision***] option. If the [**-r** ***revision***] option is not specified, the Revision information is retrieved by the **getver** command.

Permission level: system

Syntax

```
flashcmd [-P] [-S] [-A] [-B] [-d device_descriptor ]  [-w]  [-r revi-
sion] command
```

Options

`-P`

Specifies the internal primary device, `/dev/wd0a.` (default)

`-S`

Specifies the internal secondary device, `/dev/wd1a.`

`-A`

Specifies the PCMCIA device in slot A, `/dev/wd2a.`

`-B`

Specifies the PCMCIA device in slot B, `/dev/wd3a.`

`-d` *`device_descriptor`*

Use the specified device descriptor, this option enables another device to be acted upon.

`-w`

Mounts the flash device as writable, the default is to mount the device as read-only.

`-r` *`revision`*

Specifies the software release and version against which to perform the command.

This option is in the form ***release,version***. For example, `1.4.x,default` or `1.4.x,atmtestB`. The currently-running version is assigned by **flashcmd** to be **,default**.

`command`

The ***command*** can be any UNIX command. Ones typically used include copy (**cp**), determine free space (**df**), remove file (**rm**), and list contents (**ls**).

`%R`

Substitutes the ***revision*** specified with the **-r** option in the ***command*** executed (see examples)

Examples

1 At this command

```
# flashcmd -w rm -rf /flash/tmp
```

flashcmd performs the following actions:

- mounts the internal flash device
- removes the `/flash/tmp` directory
- unmounts the device

2 At this command:

```
# flashcmd -A -r 1.4.x,default
    cp /flash/etc.%R/grifconfig.conf /etc/grifconfig.conf
```

flashcmd performs the following actions:

- mounts the flash device contained in the external PCMCIA slot A using the **mountf** command
- executes the **cp** command to copy the file from the directory `/flash/etc.1.4.x,default/grifconfig.conf` on a flash device to the `/etc/grifconfig.conf` file on RAM

  Note that the `%R` “shorthand” translates to the specified release, `1.4.x,default`.
- unmounts the flash device using the **umountf** command

3 This command shows that %R represents the current system:

```
# flashcmd echo %R
1.4.x,default
```

4 These two commands perform the same actions:

```
# flashcmd ls /etc.%R
# flashcmd ls /etc/1.4.x,default
```

gdc
(CLI+shell)

Monitors and manages the GateD routing daemon.

Permission level: system

Syntax

```
gdc  [-q ] [-n ] [-d ]  [-O  size ] [-c coresize ] [-f filesize ]
        [-m datasize ]  [-s stacksize ] [-t seconds ] command
```

Options

`-q`

Run quietly. With this option informational messages which are normally printed to the standard output are suppressed and error messages are logged via **syslogd** instead of being printed to the standard error output.

`-n`

Run without changing the kernel forwarding table. Useful for testing, and when operating as a route server which does no forwarding.

`-c` *`coresize`*

Sets the maximum size of a core dump a GateD started with **gdc** will produce. Useful on systems where the default maximum core dump size is too small for GateD to produce a full core dump on errors.

`-d`

Not available.

`-f` *`filesize`*

Sets the maximum file size a GateD started with **gdc** will produce. Useful on systems where the default maximum file dump size is too small for GateD to produce a full state dump when requested.

`-m` *`datasize`*

Sets the maximum size of the data segment of a GateD started with **gdc**. Useful on systems where the default data segment size is too small for GateD to run.

`-O` *`size`*

Used with **gdc start** to change the allowed number of open file descriptors.

This version of GateD removes a restriction on the number of open file descriptors which limited GateD to approximately 50 simultaneous adjacencies and/or BGP peering sessions. The default is now 320 open file descriptors, enough for roughly 300 adjacencies or peers.

`-s` *`stacksize`*

Sets the maximum size of stack of a GateD started with **gdc**. Useful on systems where the default maximum stack size is too small for GateD to run.

`-t` *`seconds`*

Specifies the time in seconds which **gdc** will spend waiting for GateD to complete certain operations, in particular at termination and startup. By default this value is set to 10 sec.

Commands

The following commands cause signals to be delivered to GateD for various purposes:

`backout`

Back out to the previous `/etc/gated.conf` file.

`BACKOUT`

Back out without questions.

`checkconf`

Check the current `/etc/gated.conf` file for syntax errors.

`checknew`

Check the new `/etc/gated.conf` file for syntax errors.

`createconf`

Create a new `/etc/gated.conf` file.

`COREDUMP`

Sends an abort signal to GateD, causing it to terminate with a core dump.

`dump`

Signal GateD to dump its current state into the file `/var/tmp/gated_dump`.

Caution: Note that a **gdc** dump will create a `/var/tmp/gated_dump` file under the uid of `nobody` and a gid of `none`. This prevents a `/var/tmp/gated_dump` file from ever filling the file system to more than 95% full.

Also note, it may be impossible on a Lucent GRF to edit (vi) the `/var/tmp/gated_dump` file due to file system size restriction of the RAM-based file system. It is recommended that `more | grep | ftp` to another machine be used to access or edit the file.

`interface`

Signal GateD to recheck the interface configuration.

`KILL`

Cause GateD to terminate ungracefully. Normally useful when the daemon has hung.

`modeconf`

Clean up `/etc/gated.conf` file modes.

`newconf`

Rotate the new `/etc/gated.conf` file into place.

`reconfig`

Signal GateD to reread the `/etc/gated.conf` file configuration file, reconfiguring its current state as appropriate.

`restart`

Stop and then start GateD.

`rmcore`

Remove existing GateD core file.

`rmdump`

Remove existing GateD dump file.

`rmparse`

Remove existing GateD parse error file.

`running`

Determine whether GateD is running.

`start`

Start GateD.

`stop`

Signal GateD until it stops.

`term`

Signal GateD to terminate after shutting down all operating routing protocols gracefully. Executing this command a second time should cause GateD to terminate even if some protocols have not yet fully shut down.

`toggletrace`

If GateD is currently tracing to a file, cause tracing to be suspended and the trace file to be closed. If GateD tracing is currently suspended, cause the trace file to be reopened and tracing initiated.   This is useful for moving trace files.

`version`

Display GateD version information.

get (CLI)

Displays the contents of the current working profile read into local memory. The **get** command retrieves the names and contents of fields within profiles without changing the user's location within the tree. Using **get**, you can look at a field in another profile. **ls** is an alias for **get**.

See also: **cd**, **list**, **ls**

Permission level: user

Syntax

```
get [. | profile-type [ profile-index ] ] [ field-name field-index ... ]
```

Example

Use **get** to check the contents of one profile when you are at a lower level of another profile:

```
super> read card 3
CARD/3 read
super> list .
card-num* = 3
media-type = no-media
debug-level = 0
hssi-frame-protocol = Frame-Relay
sonet-frame-protocol = PPP
ether-verbose = 0
ports = < { 0 {off on 10 3 }{single off}{"" "" 1 sonet inter-
nal-oscillat+
load = { 0 < > 1 0 0 }
dump = { 0 < > off off }config = { 0 0 1 1 4 0 }
icmp-throttling = { 0 10 10 2147483647 10 10 }

super> list ports 0
port_num = 0
cisco-hdlc = { off on 10 3 }
fddi = { single off }
sonet = { "" "" 1 sonet internal-oscillator 0 207 }
hssi = { 0 16-bit }
ether = { autonegotiate }
hippi = {1 32 no-mode 999999 4 incremental 5 300 10 10 03:00:0f:c0
                     disabled di+
super> list hssi
source-clock = 0
CRC-type = 16-bit
```

Here is where you use **get** to look in to another profile:

```
super> get card 1 ports 0 hssi
source-clock = 0
CRC-type = 32-bit
```

get operates on the last profile read when a dot (.) is used in place of the profile type and profile index:

```
super> read user default
USER/default read
super> get .
name* = default
password = ""
auth-method = {PASSWD {"" 1645 udp ""}{5500 udp 5510 tcp /var/ace}}
active-enabled = yes
allow-system = no
allow-update = no
allow-password = no
allow-debug = no
prompt = GRF=>
log-display-level = none
```

get operates on the last profile read when a dot (.) is used in place of the profile type and profile index and a field name is specified:

```
super> read user default
USER/default read

super> get . auth-method
auth-type = PASSWD
rad-auth-client = { "" 1645 udp "" }
securid-auth-client = { 5500 udp 5510 tcp /var/ace }
```

getver
(CLI+shell)

The **getver** command reports the release and version of the operating system that is currently running, or the release and version number of the system that is set to run after the next boot. Use the **setver** command to specify (set) the release and version of the next system to be run.

Permission level: system

See also: **setver**, **grfins**

Syntax

```
getver  [-c]  [-n]  [-s]
```

Options

The **getver** command supports the following options:

`-c`

Reads the `/etc/release` file in system RAM and returns the release name and version of the currently running system. (default)

`-n`

Reads the `/flash/etc/release` file in the internal flash drive and returns the release name and version of the system to be installed during the next boot.

`-s`

Returns the release name and version in a binary or compact form for use in other scripts

Example

This pair of commands returns the names of the current and the “next-boot” operating systems:

```
# getver
Current Revision: 1.4.12  Version: default

# getver -n
Next Revision: 1.4.12  Version: fdditest

# getver -s
1.4.12,default
```

graps
(shell)

The **graps** command provides a way to manage the working and protection channel selection. Use the command to change the APS settings. (These settings can also be changed in the SONET Card profile.) **graps** provides standard APS options, for example, those defined by Bellcore R5-89.

Syntax

```
graps -p port
```

where ***port*** is the card's slot number.

Example

The **graps** command returns three pieces of information:

– which channel is active or, more specifically, on which channel is the receive data path fully connected. WORKING indicates the receive data path is active on the WORKING channel, upper interface A. PROTECTION indicates the receive data path is active on the PROTECTION channel, lower interface B.

– the current CSR setting

– the last APS command (1 through 6 below) entered

Then **graps** prompts you to enter another command or to quit.

```
# graps -p 6

APS channel selection: WORKING
      APS channel CSR: Do not revert
     Last APS command: Lockout protection channel

Please enter command (? for help): ?
Commands:
   1) Clear all other switch commands
   2) Lockout protection channel
   3) Forced switch to protection channel
   4) Forced switch to working channel
   5) Manual switch to protection channel
   6) Manual switch to working channel
   q) quit
```

Priority and other APS interactions require that the user understand APS 1+1 before changing the default settings. Please use the Bellcore Technical Reference, *Synchronous Optical Network (SONET) Transport Systems: Common Generic Criteria (*TR-NWT-000253 Issue 2, December 1991) or the ANSI T1.105-1988 publication, "*Digital hierarchy: optical interface rates and formats specifications*" for information about configuring the channels.

grarp
(CLI+shell)

grarp supports address resolution for all media. The command maintains a mapping of IP addresses to physical addresses in the `/etc/grarp.conf` configuration file and displays the file's contents. In addition to the information presented here, a man page is also available.

The `/etc/grarp.conf` file is a statically-configured global address resolution table. Each media card maintains its own address resolution table. The default behavior of the **grarp** command when displaying or deleting entries is to use the media card tables. Please see Chapter 2 for a template of the `/etc/grarp.conf` file.

The **gria** command that mapped IP addresses to physical HIPPI card interfaces is deleted. It is replaced by **grarp**.

Note: Sites do not have to recreate **gria** entries. A built-in script automatically copies any existing **gria** command information into the site's new `/etc/grarp.conf` file.

BSD maintains one ARP table for all its media. The GRF media cards have their own separate ARP tables. The default behavior of **grarp** when displaying or deleting entries is to use all the tables. The **-i** and **-p** options restrict the action to the BSD kernel or to one of the cards.

Permission level: system

Syntax

```
grarp [-n] [-i ifname] [-p port] hostname [server]
grarp -a [-n] [-i ifname] [-p port]
grarp -r [-n] [-i ifname]
grarp -d hostname [-i ifname] [-p port] [server]
grarp -z -i ifname
grarp -s -i ifname [-e] hostname phys_addr [temp] [pub] [trail]
[server]
grarp -f filename  [-i ifname] [-p port]
```

Command options

`hostname`

Displays the current ARP entry for *hostname*. The hostname may be specified by name or address, using Internet dot notation. The optional server modifier (ATM only) is used to get a local ATMARP server entry. (The ATMARP clients and local servers operate independently; the client at a given interface only has the entries that it requests from the local server.) The default is to get the client entry.

`-a`

Displays the current ARP entries for the specified interface or port. If no interface or port is specified, **grarp** displays the ARP entries for all interfaces. The **-a** option does not display local ATMARP server entries.

When the `-i` *`ifname`* option is used with **-a**, **grarp** displays ARP entries for the specified interface named in the `gx0yz` format. When the `-p` *`port`* option is used with **-a**, **grarp** displays ARP entries for the specified slot named in the GRIT *`port`* address of the form 0:<*slot*>:<*interface*>, for example 0:2:11.

`-r`

(ATM only) Displays the current ATMARP server entries for the local server at the specified interface. If no interface is given, **grarp** displays the ATMARP entries for all local ATMARP servers.

`-d` *`hostname`*

Deletes the ARP entry for *`hostname`*. The server modifier (ATM only) directs **grarp** to delete the ATMARP entry from the local server at the specified interface. The server modifier requires the ***-i ifname*** option. (The default is to delete the client's ATMARPentry.)

`-z -i` *`ifname`*

(ATM only) Flushes the ATMARP server entries from the local server at the specified interface, including those that **grarp** adds.

`-s -i` *`ifname hostname phys_addr`* `[temp] [pub] [trail] [server]`

Creates an ARP entry for the host called ***hostname*** with the physical address ***phys_addr***. The physical address is given as six hex bytes separated by colons for Ethernet and FDDI and in other formats as appropriate to the media.

The entry is permanent unless the modifier **temp** is given in the command. If the modifier **pub** is given, the entry is "published", meaning that this system will act as an ARP server and respond to requests for ***hostname*** even though the host address is not its own. Do not use the **pub** modifier to add local ATMARP server entries. The modifier **server** (ATM only) is used to add the ATMARP entry to a local server. The modifier **trail** indicates that trailer encapsulations may be sent to this host. The **-i** option must be used with this command to specify the interface for which the entry will be created.

`-f` *`filename`*

Causes the file ***filename*** to be read and multiple entries to be set in the ARP tables. Entries in the file should be of the form:

```
   ifname hostname phys_addr [temp] [pub] [server] [trail]
```

with argument meanings as given above.

Additional options

`-b`

Makes the command address Bridge ARP table entries on the media cards instead of the regular ARP table. The Bridge ARP table contains copies of the ARP table entries of other media cards that are doing bridging.

`-e`

(ATM only) Specifies that the ATM address is a CCITT E.164-style address instead of the default NSAPA format. If an ATM sub-address is given, this option is overridden and the address and sub-address are treated as E.164 and NSAPA respectively.

`-n`

Show network addresses as numbers (normally **grarp** interprets addresses and attempts to display them symbolically).

`-i` *`ifname`*

This is the GRF interface name in the format `gx0yz.` The ***ifname*** argument is the interface name as you would see it in column one of the output of **netstat -i**.

`-p` *`port`*

The ***port*** argument is a simple slot number or a GRIT address of the form 0:<*slot*>:<*interface*>, for example 0:2:11.

BSD maintains one ARP table for all its media. The GRF media cards have their own separate ARP tables. The default behavior of **grarp** when displaying or deleting entries is to use all the tables. The **-i** and **-p** options restrict the action to the BSD kernel or to one of the cards.

Address formats

The format of the ***phys_addr*** argument given with the **-s** command depends on the hardware type ***phys_addr***:

FDDI

- six hexadecimal bytes separated by colons.

HIPPI

- a 32-bit unsigned integer in C language convention (prefix "0x" for hexadecimal, "0" for octal, a non-zero digit for decimal).

ATM format optionss:

1. Specifies address and an optional subaddress as a single string with "+" separating them. The NSAP address and subaddress are each a series of up to 20 hexadecimal bytes,
   two digits each, with optional "." separators between the bytes.
2. Uses VPI/VCI values for a PVC in the form `nn/mm` where `nn` is VPI and `mm` is VCI.
3. With **-e** command, address is in CCITT E.164 format.

grass
(CLI+shell)

The **grass** command is used to enable and disable various internal IP service settings in the operating system. The `/etc/grass.conf` file should not be edited by the user.

grass allows an administrator to choose what application level services are run, and to quickly enable/disable them as needed. If possible, **grass** attempts to activate/deactivate the service immediately. The user is informed of the actions taken by the program. Changes made using **grass** must be saved using **grwrite** to save the changes for the next reboot.

By default, **grass** displays the current status of all managed services as viewed by their configuration files. Services are listed in the `/etc/services` file.

Permission level: system

Syntax

```
grass   [+/-s service_name]   [+/-g group_name ]    [+/-1 ]
```

Options

`+/-s` *`service_name`*

Activates/deactivates the named IP service. The service is listed in the `/etc/grass.conf` configuration file, and disabled or enabled as required. This option can be repeated to add or subtract multiple services from the set of services to change.

`+/-g` *`group_name`*

Activates/deactivates a group of IP services. The group name is listed in the `/etc/grass.conf` file and each service in the group is enabled/disabled. This option can be repeated to add or subtract multiple groups of services from the set of services to change.

`+/-l`

Lists the services and groups from `/etc/grass.conf.` For each service, information is displayed:

- name of the service
- file in which service entry is located
- script run to change service status “on the fly”
- the regular expression string used to locate the service in the given file Group entries contain the list of services managed by the group.

Examples

Line options are cumulative. For example, this command disables all services except **telnet**:

```
# grass -g all +s telnet
```

This command disables **telnet**, and prevents remote access to the system:

```
# grass -s telnet
```

Enacting changes

If a script to dynamically start/stop the given service is not available, the user will need to reboot the system to enact changes.

etc/services

The `/etc/services` file maps the protocol names to the TCP/IP ports.

When you modify the `/etc/services` file, those changes will get saved to flash memory when you do a **grwrite**. On RMS node systems, your changes to `/etc/services` are saved and archived by the **grc** command. However, subsequent software upgrades will install the new release version of `/etc/services`, overwriting any changes you may have made. Please be sure to record your changes to this file as you will need to add them again when you upgrade.

Here is a portion of the /`etc/services` file:

```
# Network services, Internet style
#       @(#)services    8.1 (Berkeley) 6/9/93
gritlog         16/grit
gritboot        17/grit
gritpclog       18/grit
grinch          19/grit
echo            20/grit
discard         21/grit       sink null
daytime         22/grit
chargen         23/grit       ttytst source
time            24/grit
trap            25/grit
ioctl           26/grit       BSD kernel ioctl routines
filtercomm      27/grit       Filterd known port
#
tcpmux         1/tcp               # TCP port multiplexer (RFC1078)
echo            7/tcp
echo            7/udp
discard         9/tcp         sink null
discard         9/udp         sink null
systat          11/tcp        users
daytime         13/tcp
daytime        13/udp
netstat         15/tcp
chargen         19/tcp        ttytst source
chargen         19/udp        ttytst source
ftp             21/tcp
telnet          23/tcp
smtp            25/tcp        mail
time            37/tcp        timserver
time            37/udp        timserver
rlp             39/udp        resource      # resource location
nameserver      42/tcp        name          # IEN 116
whois           43/tcp        nicname
domain          53/tcp        nameserver    # name-domain server
domain          53/udp        nameserver
```

gratm
(CLI+shell)

The **gratm** program configures ATM OC-3c and OC-12c media cards with specific information needed for ATM operations. This includes information for configuring:

- physical interfaces for SVC signalling
- permanent virtual circuits (PVCs)
- rate queues to support traffic shaping

gratm reads configuration information from `/etc/gratm.conf` and uses the **grinch** command to download the configuration information to the appropriate media card. Please see Chapter 2 for a template of the `/etc/gratm.conf` file.

Permission level: system

Syntax

```
gratm [-f file] card [...] [-n]
```

Options

`-f` *`file`*

Reads the ATM physical interface configuration from the specified program file instead of the default configuration file `/etc/gratm.conf.`

`card`

Uses the first four components of the interface name, `ga0y,` to identify a specific ATM media card as to slot location.

`-n`

No execute: prints the **grinch** commands that would be used to configure the kernel or media card, but does not actually execute them.

Examples

The **gratm** command can be used to check the syntax of the `/etc/gratm.conf` configuration file before the configuration is used to initialize ATM media cards.

To parse and check the syntax of `/etc/gratm.conf`, enter:

```
# gratm
```

Use this command to parse and check the syntax of `/etc/gratm.conf` and print the **grinch** commands that would be run by **gratm** to configure the ATM card in slot 3. The **-n** option executes without actually performing any configuration actions, enter:

```
# gratm -n ga03
```

When parsing the configuration file, **gratm** reports any configuration errors it detects. If errors are detected, no actual configuration of ATM cards is performed.

grbist
(CLI+shell)

Caution: The **grdiag** command runs hardware diagnostics that cover all but the actual interface connectors on the media card. For most cases, **grdiag** replaces **grbist**. Note that **grdiag** does not interrupt any other system operations.

Permission level: system

See also: **grdiag**

Syntax

```
grbist [-p portcard ]
```

grburn

This command is now obsolete, refer to the **grflash** command.

grc
(CLI+shell)

The **grc** script provides commands to collect, archive, and restore system configuration files on systems with RMS nodes. GRF systems without RMS nodes use **grsnapshot**.

See also: **grwrite**, **grsite**, **grsnapshot**

Syntax

```
grc [info] [list] [restore] [save] [-a archive] [-d directory] [-f
listfile] [-F] [-q] [-v]
```

Usages

These are the options appropriate for each **grc** command:

```
grc info [-q] [-a archive] [-d directory] [-f file]
grc save [-Fqv] [-a archive] [-d directory] [-f file]
grc restore [-qv] [-a archive] [-d directory] [-f file]
grc list [-qv] [-a archive] [-d directory] [-f file]
```

Commands

`info`

Prints information about the system configuration archive.

This includes the date and time the archive was created, the user who created the archive, and any comments supplied at creation.

`list`

Lists the configuration files in the archive.

`restore`

Extracts all configuration files from the archive. The files are placed relative to the current directory, or to the directory specified by the **-d** option.

`save`

Saves the configuration files in an archive. Prompts the user for an optional comment to be stored in the archive and used to describe the contents. You might add a comment such as, "Config before adding new FDDI card to slot 3".

Options

Options are specified after the appropriate **grc** command. **grc** responds with an appropriate usage message if an inappropriate option is used with a command.

`-a` *`archive`*

Writes information *to* (for the **save** command) or reads information *from* (for the **restore** command) the specified archive instead of the default floppy diskette drive of `/dev/fd0.`

`-d` *`directory`*

Specifies the directory relative to which **grc** will save or restore the configuration files. This option has **grc** do a **cd** to ***directory*** prior to the operation.

`-F`

Used with the **save** command to format the floppy diskette that will hold the archive (uses the **fdformat** command).

`-f` *`listfile`*

Reads the list of files to be archived, extracted or listed from ***listfile***. Files are specified relative to the current working directory or the directory specified by the **-d** option, and are listed one per line. A line in a ***listfile*** that begins with a **#** (pound sign) is considered a comment and is ignored.

`-q`

Quiet mode, suppresses the usual chatty messages that describe what **grc** is doing.

`-v`

Verbose mode. When used with **restore** or **save** commands, the option lists files that are being extracted from, or written to, the archive. When used with the **list** command, the option prints out a verbose listing (uses **-v** option of **tar**).

The default list of configuration files not only includes router-specific files, such as `/etc/grifconfig.conf`, but also includes typical configuration files that are subject to modification from time to time, such as `/etc/passwd`.

Note: Any changes you make to the `/etc/services` file are overwritten when you install a new software release. Record these changes and add them back after the upgrade.

```
etc/aitmd.conf           etc/gritd.conf         etc/namedb
etc/bridged.conf         etc/grlamap.conf       etc/netstart
etc/crontab              etc/group              etc/networks
etc/fstab                etc/grppp.conf         etc/passwd
etc/filterd.conf         etc/grpvc.conf         etc/printcap
etc/gated.conf           etc/grroute.conf       etc/pwd.db
etc/gettytab             etc/grstart            etc/rc
etc/grarp.conf           etc/hostname.txt       etc/rc.local
etc/gratm.conf           etc/hosts              etc/resolv.conf
etc/grclean.conf         etc/hosts.equiv        etc/services
etc/grclean.logs.conf    etc/localtime          etc/snmpd.conf
etc/grfr.conf            etc/master.passwd      etc/spwd.db
etc/grifconfig.conf      etc/motd               etc/syslog.conf
etc/grinchd.conf         etc/named.boot
```

Figure 1-5. List of files archived by the grc command

A site can specify alternate file(s) to archive on the command line, or in a ***listfile*** specified by the **-f** option. If alternate files are specified both in the **-f** option file and on the command line, the **grc** script archives any file specified in either ***listfile*** or the command, that is, **grc** will operate on a union of the two lists.

All files are specified relative to the directory from which the archive will be created (usually /). For example, the password file is specified `etc/passwd`.

By default, **grc** operates on a floppy diskette in the RMS node's floppy diskette drive (`/dev/fd0`). An alternate archive (e.g., a file) can be specified using the **-a** option. A man page is available.

You must log in as root to use **grc**.

grcard
(CLI+shell)

Use the **grcard** command to display the status of media cards, including slot number, hardware type, and current operating status.

Permission level: system

Syntax

```
grcard   [-dv]
```

Options

`-d`

Specifies the debug option (not available).

`-v`

Specifies the verbose option to add column headers to command output.

States

A media card will be in one of these operating states:

Power-up	- initial state of a card at system power on
Boot-requested	- card has requested its run-time code
Dumping	- card is being dumped
Loading	- card is receiving run-time code
Configuring	- card has requested its configuration tables
Running	- card is configured and operating
Not-responding	- card does not respond to requests from management software, or the media card is not installed in the slot
Panic	- card has encountered a system fault
Hold-reset	- card is being held in reset state
State unknown	- card’s state cannot be determined

The “Q” type of media card can be indicated in a display. For example, FDDI/Q and ATM/Q cards appear as FDDI_V2, and ATM_OC3_V2, respectively. V2 does not indicate /Q versions, as HSSI_V1, SONET_V1, and ETHER_V1 are all /Q level versions.

As shown in this output from a GRF 400, the information displayed includes slot number, hardware type, and current operational status:

```
#  grcard -v
Slot    HWtype  State
----    ------  -----
0       HSSI_V1 running
1       HIPPI_V1 not-responding
2       ATM_OC3_V2      running
3       FDDI_V2 running
```

These are the names **grcard** displays for other cards: `ETHER_V1, SONET_V1, DEV1_V1, ATM_OC12_V1,` and `ATM_OC12_V2.`

grclean (CLI+shell)

grclean is an internal program that compresses, archives, and manages dumps and log files, and saves them to a specified file name.

Software release 1.3.11 changed the contents of `/etc/grclean.logs.conf`. If you upgrade to 1.4 from a 1.3.9 or earlier software release, the previous `/etc/grclean.logs.conf` file is renamed to `/etc/grclean.logs.conf.old`, and a new copy of `/etc/grclean.logs.conf` is installed. You may see the message describing this transfer if you are watching the console as **grfins** operates

If you have never made any changes to `/etc/grclean.logs.conf`, the upgrade has no effect. However, if you did change `/etc/grclean.logs.conf` in the past, then you must now make those changes again as soon as possible after the 1.4 update procedure is finished. Cut and paste just your changes into the 1.4 version of the file, take care not to overwrite the new sections in the file.

This excerpt from `/etc/grclean.logs.conf` shows GRF-specific logs **grclean** now manages:

```
##############################################
size=10000
logfile=/var/log/gritd.packets
size=150000
logfile=/var/log/gr.console
size=11000
logfile=/var/log/gr.boot
size=150000
logfile=/var/log/grinchd.log
logfile=/var/log/gr.conferrs
logfile=/var/log/mib2d.log
logfile=/var/tmp/gated.rip
logfile=/var/log/mibmgrd.log
logfile=/var/log/cli.log
logfile=/var/log/fred.log
```

Permission level: system

Syntax

`/usr/nbin/grclean [-f` *`configfile`*`] [-l` *`log`*`] -n -v`

Options

`-f` *`configfile`*

Dump to ***configfile*** instead of default, `/etc/grclean.conf.`

`-l` *`log`*

Log **grclean** events to ***log*** instead of the default file, `/var/log/grclean.log.`

`-n`

Do not execute, just show actions.

`-v`

Display in verbose mode.

grconslog
(CLI+shell)

The **grconslog** command displays the messages logged to the `/var/log/gr.console` file. Various options exist to filter the messages displayed. If no options are specified on the command line, **grconslog** prints the message associated with every entry in `/var/log/gr.console`.

A common option for monitoring GRF events is **grconslog -vf**. It is usual practice to telnet into the GRF, enter **grconslog -vf** and keep the window open to monitor system events as they are reported. Use the abort or equivalent key to quit the log.

Note: **grconslog** sometimes hangs when the log file fills up. Use Control-C (or the abort or equivalent key) and then restart **grconslog**.

The **grconslog** command runs normally when logging is configured to an external PCMCIA flash device or to an NFS-mounted disk. The `/var/log` directory has a pointer to the external device. The *GRF 400/1600 Getting Started* and GRF *Configuration and Management* manuals describe how to configure PCMCIA devices.

When logging is done remotely via **syslogd**, items are logged out to the **syslog** server. Note that **grconslog** runs locally. You must run **grconslog** on the remote **syslog** server or use the **tail -f.**

Permission level: system

Syntax

```
grconslog  [- number] [-a] [-A] [-t]  [-T] [-v] [-V] [file]
[-{b|B} slot]  [-d]  [-e]  [-f]  [-h]  [-p]  [-{r|R} regex]
```

Options

`-` *`number`*

Print only the last ***number*** of lines in `/var/log/gr.console`. The default is 10 lines. (Refer to the **tail** man page.)

`-a`

Print all lines, even blank lines and screen prompts.

`-A`

Print all lines, including blank lines, screen prompts, date stamp, sending media card, and media card hardware type (effectively equivalent to **grconslog -atv**).

`-b` *`slot`*

Only print the messages for the media card in the specified ***slot***. This lets you check what has been reported about a single card, you are saved reading every line in the general log.

`-B` *`slot`*

Only print the messages for media cards *not* in the specified ***slot***. This option lets you avoid the entries for a card that is generating lots of messages.

`-d`

Print the date stamp.

`-e`

Expand: When a message is put in `/var/log/gr.console` to indicate the last message has been repeated *n* times, the last message is actually printed out *n* times. Without **-e**, *"last message repeated* `n` *times*" is printed.

`-f`

Causes **grconslog** not to stop when end of file is reached, but to wait for additional data to be appended to the input. (Refer to the **tail** man page.)

`-h`

Help function, prints out all options and then exits.

`-p`

Print slot number of message originator.

`-r` *`regex`*

Print all messages that contain a specified regular expression (***regex***). (Refer to the **perl** and **grep** man pages.)

`-R` *`regex`*

Print messages that do not contain a specified regular expression (***regex***). (Refer to the **perl** and **grep** man pages.)

`-t`

Print the hardware type of the message originator.

`-T`

Print the message type.

`-v`

Verbose mode: print the slot number of message originator and print the date stamp (effectively equivalent to **grconslog -dp**).

`-V`

Version: print version number of **grconslog** and print `/etc/motd` to return the current version level of system software.

`file`

Instead of filtering `/var/log/gr.console`, **grconslog** filters ***file*: If *file*** is '-', then it filters standard input.

grdebug (CLI+shell)

This command controls operating system actions in response to the state of a media card. By default, the **grdebug** command prints the current configuration of the actions for the media card specified via the **-p** option.

Caution: Use these options with care; media card performance can be adversely affected.

Permission level: system

Syntax

```
grdebug [-C] [-E] [-H] [-I] [-P] [-R] [-T] [-U] -p slot_number [on|off]
```

Options

`slot_number`

Chassis slot number, from 0 to 3 or 0 to 15.

`-C`

Specifies that the *"reset upon configuration errors"* action will be configured on, configured off, or displayed.

`-E`

Specifies that the *"echo (ping) when silent"* action will be configured on, configured off, or displayed.

`-H`

Specifies that the *"hold card in reset upon configuration errors"* action will be configured on, off, or displayed.

`-I`

Specifies that the *"initialize card upon start up"* action will be configured on, configured off, or displayed.

`-P`

Specifies that the *"reset card upon panic"* action will be configured on, configured off, or displayed.

`-p` *`slot_number`*

Specifies that the bits for the indicated media card will be configured on, configured off, or displayed.

`-R`

Specifies that the *"reset card when hung"* action will be configured on, configured off, or displayed.

`-T`

Specifies that the *"time out card for echo or reset"* action will be configured on, configured off, or displayed.

`-U`

Specifies that the *"update card during normal operation"* action will be configured on, configured off, or displayed.

Examples

To turn off all monitoring activity for a media card in slot 2, enter:

```
# grdebug -p 2 off
```

To prevent a media card in slot 2 from automatically having its configuration updated and from being reset when it does not respond to a **ping** command, enter:

```
# grdebug -R -U -p 2 off
```

grdiag
(shell)

The **grdiag** program invokes low-level hardware diagnostics on specified GRF media cards. Users can run a set of internal BIST-level diagnostics to verify media card hardware. A media card that fails this set of diagnostics must be replaced.

The command operates on GRF 400 and GRF 1600 routers as well as on the GR-II. HIPPI media cards do not support the diagnostic set invoked by **grdiag**.

grdiag can be executed at any time since it affects the operation of only the target card or cards. You can run diagnostics on one card or on all the chassis cards at the same time. The length of time needed for the diagnostic to run depends on the type of media card, how many times it will run, and how many cards are being tested at one time.

Caution:
The **grdiag** program reboots the media card. For this reason, it is very important that you save any configuration changes before you run **grdiag**. Unsaved media card changes will be lost.

The "Management Commands and Tools" chapter in the *GRF Configuration and Management* manual describes how to use **grdiag**.

Permission level: system

Syntax

```
grdiag
```

There are no options for **grdiag**.

grdinfo
(shell)

The **grdinfo** utility is to be used in conjunction with Customer Support staff. The **grdinfo** utility enables the site to use a single command to collect a comprehensive set of debug and configuration information for the GRF. **grdinfo** options specify the type of information collected, including logs, dumps, media card statistics, protocol statistics, and control board data. Target data can be obtained at the system level or at the card level.

You can execute **grdinfo** while the GRF is running although there will be an impact on performance while the information is collected. This is a diagnostic tool. If the media cards are busy forwarding data and are unable to respond to statistics requests, or if not enough disk space is available, you will get an error message reporting the condition. In most cases, **grdinfo** stops and ends.

Some files **grdinfo** creates are very large. As a result, the collection process can interfere with system operations. Data is saved in compressed TAR files for ease in transfers to Technical Support.

Information specified by the specified option or options is collected and compressed into a `grdinfo.tar.gz` file in the `/var/tmp/grdinfo` directory. One **grdinfo** file is collected at a time. The size of a particular `grdinfo.tar.gz` file will vary widely and can tax system file system resources. The **grdinfo** utility is intended to collect information, not to store it. After you use **grdinfo** to collect the needed information, copy the data to external storage such as a file server, and then clean up the`/var/tmp/grdinfo` directory. It is suggested that you move the output file off the GRF for analysis. Do not extract the data out of the `grdinfo.tar.gz` output file while it is on the GRF.

Syntax

`grdinfo -card=`*`slot`*` | all`

This command returns configuration and state information for a specific media card or for all installed cards.

`grdinfo -config`

Collects system configuration data, including Card, System, Dump, and Load profiles, and all `/etc/*.conf` configuration files, including `/etc/gated.conf`.

`grdinfo -log`

Collects all log files from the local `/var/log` directory

`grdinfo -dump`

Collects all dump files from the local `/var` directory, including media card, utility, and kernel dumps

`grdinfo -system`

Collects control board (RMS) data

`grdinfo -frame`

Collects system-wide Frame Relay status, configurations, and statistics

`grdinfo -bridge`

Collects system-wide bridging status, configurations, and statistics

`grdinfo -dr`

The dynamic routing option is not available in this release.

`grdinfo -all`

Collects and combines all the data the other options collect (not recommended)

Refer to the “Management Commands and Tools” chapter in the *GRF Configuration and Management* manual for examples specific usage procedures.

gredit
(CLI)

Opens the `/etc/filterd.conf` or the `/etc/gated.conf` configuration file, but brings the file up in **vi** ready to edit.

Permission level: system

Syntax

```
gredit filterd
```

```
gredit gated
```

grfddi
(CLI+shell)

This utility command sets the dual and single attachment connections for an FDDI media card. **grfddi** commands are not maintained across system reboots. Permanent settings must be made in the appropriate Card profile. After configuring or reconfiguring the attachment interface, you must reset the media card for the change to take effect.

Permission level: system

Syntax

```
grfddi  [-p slot_number ]  [ -u | -l ]  [ -s | -d ]
```

Options

slot_number

Chassis slot number, from 0 to 3 or 0 to 15.

`-u`

Specifies upper pair of interfaces, A0 and B0.

`-l`

Specifies lower pair of interfaces, A1 and B1.

`-s`

Sets specified interface pair as two single attach.

`-d`

Sets specified interface pair as one double attach.

Example 1

To set the lower pair of interfaces on FDDI media card in slot 3 to dual attach and set the upper pair to two single attach interfaces, first enter:

```
# grfddi -p3 -l -d
Configuring lower ports on slot 3 to be dual-attached.
2.12.2.4.4.3.4.1:  OK
2.12.2.4.4.4.4.1:  OK
Port card in slot 3 must be reset for change to take effect.
```

Then enter:

```
# grfddi -p 3 -u -s
Configuring upper ports on slot 3 to be single-attached.
2.12.2.4.4.1.4.1:  OK
2.12.2.4.4.2.4.1:  OK
Port card in slot 3 must be reset for change to take effect.
```

Example 2

Two commands are needed to set all interfaces on FDDI media card in slot 1 as single attach interfaces:

```
# grfddi -p 1  -l  -s
# grfddi -p 1  -u  -s
```

grfins
(CLI+shell)

The **grfins** command installs a given release of software on non-RMS node systems and up/downgrades the configuration files as required.

Use **grwrite** to save any configuration changes made since last boot.

```
# grwrite
```

The **grfins** command unpacks zipped files, **flashcmd** cleans up the flash device:

```
    # grfins --source=/flash/tmp --release=1.4.12 --activate
    # flashcmd -w rm -rf /flash/tmp
```

A reboot installs the files on system RAM, enter:

```
    # shutdown -r now
```

Permission level: system

Syntax

`grfins --source=`*`source_loc`*` --release=`*`release_number`*` `*`options`*

`--source=`*`source_loc`*

Specifies the source of files to be installed, required, there is no default.

`--release=`*`release_name`*

Provides the release name as seen in the new release binary, must specify full release name, for example, `1.4.12.`

Options

`--activate`

Executes the **setver** command when finished to make this installation activate on the next boot.

`--downgrade`

Use this option instead of **--activate** when you are moving back to an older system, back to 1.3.8 from 1.4.12, for example.

`--help`

Shows a short help message.

`--move`

When placing the `.TAR.gz` and `root.gz` files, moves them instead of using a copy command.

`--no`

Answers no to all queries.

One query is whether configuration information should be saved in `/etc/profs`. With the **--no** option, no information is saved in that directory.

`--savesite`

Does not delete the site file for this release, this is the default if the user is installing over an existing release.

`--srcflash`

Selects an alternate source flash device. When this option is used, the device is mounted on `/mnt`. Thus, the **`--source=`** value should start with `/mnt/` as a path to the source `.gz` files.

`--target=`*`device_name`*

Specifies the target device on which release files will be loaded.

The target can be one of the following:

P - internal primary flash

S - internal secondary flash

A - PCMCIA slot A

B - PCMCIA slot B

`--verbose`

Prints out more event information during disk creation

`--yes`

Answers yes to all queries. One query is whether configuration information should be saved in `/etc/profs`. With the **--yes** option specified, information is saved in that directory.

grflash
(CLI+shell)

The **grflash** command burns a new program into flash memory on all media cards. (This command replaces **grburn**.)

Warning: The **grflash** command must only be used with the direction of Customer Support staff.

Permission level: system

Syntax

```
grflash  -p port_num  [-dh] [-v] [-E] [-T timeout ] file -t
{ a | aq | a2 | e | f | h | hl | r | rl | r2 | rx | s | d }
```

`port_num`
: Chassis slot number, from 0 to 3 or 0 to 15.

`file`
: Location of new program file.

Options

`-p port_num`
: Directs ***file*** to a specified port.

`-d`
: Specifies debug option (not available).

`-h`
: Turns on hash marks so they are displayed after every segment is sent.

`-v`
: Specifies verbose option, displays number of segments sent so far and total number of segments to be sent.

`-E`
: Places card in a series of states, checks for error conditions.

`-T timeout`
: Specifies the ***timeout*** number of seconds to wait for a packet from the card or control board, default is 60 seconds.

`-t { a | aq | a2 | e | f | h | hl | r | rl | r2 | rx | s | d }`

Specifies type of file being sent:

`aq`	ATM/Q boot loader
`a2`	ATM/OC-12 boot loader
`e`	Ethernet runtime program
`f`	FDDI runtime program
`h`	HIPPI runtime program

`hl`	HIPPI loader
`o`	SONET runtime program
`r`	RMB runtime program
`rl`	RMB loader
`r2`	RMB secondary loader
`rx`	RMB xilinx
`s`	HSSI runtime program
`d`	DEV1 runtime program

grfr
(shell)

The **grfr** program configures and manages Frame Relay parameters on GRF media cards.

The **grfr** command can be executed at any time to add, modify, or delete PVC configurations as well as to display various types of status. Using the **-c ccl** and **-c crl** commands, you can add or delete Frame Relay links, or modify a link parameter, without resetting the media card. On-the-fly configuration is described in the Frame Relay configuration chapter of the GRF *Configuration and Management* manual.

The default configuration file used by **grfr** and by **fred**, the Frame Relay daemon, is `/etc/grfr.conf`. Please see Chapter 2 for a template of the `/etc/grfr.conf` file.

Note: The **grfr -c dsc** and **grfr -c dlc** command have replaced **grfr -c dss** and **grfr -c dls**.

Syntax

```
grfr -c command   [-n ]  [-s slot]  [-l port]  [-i dlci]
[-d debug-level]  [-f grfr-file]  [-g grif-file]
```

Options

`-c` *`command`*

The **`-c`** flag is required when using a **grfr** command, commands begin on the next page.

`-n`

No execute mode, only verifies the `/etc/grfr.conf` configuration file.

`-s` *`slot`*

Specifies the number of the GRF chassis slot in which the card resides.

`-l` *`port`*

Specifies the port number, also called the link number.

`-i` *`dlci`*

Specifies the DLCI or channel identifier.

`-d` *`debug-level`*

Specifies the number of the debug level, range is 0–4:

0	=	lowest level
1	=	useful when bringing up system, default
2	=	displays packets received and sent
3	=	displays packets received and sent, packet content
4	=	returns protocol-level information

`-f` *`grfr-file`*

Retrieve Frame Relay configuration information from the named file instead of from the default `/etc/grfr.conf` file.

`-g` *`grif-file`*

Retrieve interface configuration information from the named file instead of from the default `/etc/grifconfig.conf` file

Display commands

Display commands are prefaced with the **-c** flag and begin with the letter **d**:

`-c dsc`

Display system configuration and status.

`-c dlc`

Display link configuration and status.

`-c dpc`

Display PVC configuration and status.

`-c dic`

Display interface configuration and status.

`-c dss`

Display system status.

`-c dls`

Display link status.

`-c dps`

Display PVC statistics

`-c dbs,` Display board status

Configuration commands

The **grfr** configuration commands are prefaced with the **-c** flag and start with the letter **c**.

In commands that enable a link or PVC, the second letter is **e.** In commands that disable a link or PVC, the second letter is **d**.

Configuration commands require links or PVCs to be specified. Use one or more of these options in the order given: **-l *port***, **-s *slot***, **-i *DLCI***. (The **-n *name*** option is not available.)

`-c cel`

Enable link, enable link on slot 3, port 1: `# grfr -c cel -s 3 -l 1`

`-c cdl`

Disable link, disable link on slot 3, port 0: `# grfr -c cdl -s 3 -l 0`

You can add or delete Frame Relay links, or modify a link parameter, without resetting the media card using the **-c ccl** and **-c crl** commands:

`-c ccl`

Configure a new link. If the specified link is already configured, the link will be configured as specified in the `/etc/grfr.conf` file.

Configure a new link on slot 5, port 0 `# grfr -c ccl -s 5 -l 0`

`-c crl`

Remove a link and all underlying PVCs, must specify slot and port.

Remove the link on slot 5, port 0 `# grfr -c crl -s 5 -l 0`

Note: The **grfr -c ccl** command does not restore the link to state before a **grfr -c crl** command is executed. When a link is removed using the **grfr -c crl** command, the link and all underlying PVCs are removed. A subsequent **grfr -c ccl** command reestablishes the link but not the PVCs. The **fred** daemon needs to be restarted in order to reestablish all of the PVCs on the link.

grfr can also be used to configure new PVCs that have been added to the configuration file or to disable PVCs currently in the file.

`-c cep`

Enable PVC, requires PVC to be specified: `# grfr -c cep -s 3 -l 0 -i 888`

`-c cdp`

Disable PVC, requires PVC to be specified: `# grfr -c cdp -s 3 -l 0 -i 888`

`-c ccp`

Configure PVC, requires the configuration file and the PVC DLCI to be specified:

`# grfr -c ccp -f /etc/grfr.conf -i` *`dlci`*

`-c crp`

Remove PVC, requires the PVC DLCI to be specified: `# grfr -c crp -i` *`dlci`*

`–c crs`

Resets the PVC statistics reported by `grfr –c dps` to zero, user may optionally specify slot, port, and DLCI `# grfr –c crs [-s` *`slot`*`] [-l` *`port`*`] [-i` *`dlci`*`]`

The debug level can be displayed and set with **grfr**:

`-c ddl`

Display debug level: `# grfr -c ddl`

`-c csd`

Set debug level, requires the **-d** option to specify the new debug level.

Debug levels are 0–4: `# grfr -c csd -d 3`

Examples

The **grfr -c dps** command displays the statistics for configured Frame Relay PVCs. Enter:

```
# grfr -c dps
```

```
C O N F I G U R E D   P V C s   S T A T s:
=========================================
(S=Slot, P=Port, R=receive, T=Transmit)
(TP=Transmitted Packets, TO=Transmitted Octets)
Name    S/P/DLCI  Type   R-Packets  R-Octets  T-Packets T-Octets  TP-Dropped TO-Dropped
----     ------   ----    -------   --------  --------   -------  ---------   ------
2:0:0    02:0:0    Switch  2793      81044       2791     39074       0          0
2:0:160  02:0:160  ATMP    10266     1038795     10225    573187      0          0
2:0:491  02:0:491  Route   10270     863275      10279    1327565     0          0
```

The **grfr -c dlc** command shows the status of configured links and their current parameters. Enter:

```
# grfr -c dlc
```

```
   C O N F I G U R E D  L I N K S :
   =================================
   Name:       S/P:  LMI:     Link:    Autogrif: N391: N392: N393: T391: T392: Status:
   ----        ---   ---      ----     --------  ----  ----  ----  ----  ----  ------
   2:0         2 /0  ANNEX-D  UNI-DTE  None      6     3     4     10    15    Active
   Total: 1 links configured
```

grifconfig
(shell)

The **grifconfig** command initializes and manipulates the configuration of GRF network interfaces.

By default, **grifconfig** reads a table of interfaces from the file `/etc/grifconfig.conf` and compares these entries to those found in the kernel. Addresses that are no longer defined for an interface are removed and new ones are added. An interface is not modified unless an IP adddress, netmask, and (optionally) a broadcast or destination address has been modified. Please see Chapter 2 of this manual for a template of `/etc/grifconfig.conf`.

Syntax

```
grifconfig [-InR] [-f file] [interface [...]]
grifconfig -c [-InR] interface ifaddress [netmask] [address]
               [arguments  [...]]
```

`interface`

The GRF interface name in the format `gx0yz`.

`ifaddress`

The Internet address of the interface. Use a single hyphen as a place-holder if no address is desired (optional).

`netmask`

The netmask for the interface (optional). Use a single hyphen as a place-holder if no netmask is desired, but a broadcast or destination address must be specified in the next field.

`address`

For interfaces on broadcast media (such as FDDI), the broadcast address for the interface (optional).

For interfaces on non-broadcast network media (such as ATM and HIPPI), the destination address of a point-to-point interface. If address is not specified or is the place-holder argument - (a single hyphen) on an entry for a non-broadcast interface, the interface will be treated as a multi-access interface (it is attached to an ATM or HIPPI switch rather than a single host).

`arguments`

Additional arguments can be used to specify an MTU value or other descriptor for the interface (optional). See the **ifconfig** man page for other argument options.

Options

`-c`

Specifies that instead of reading its arguments from the configuration file, **grifconfig** configures a single interface from the command-line arguments.

`-f` *`file`*

Requests that the table of interfaces will be read from the specified file instead of the default configuration file `/etc/grifconfig.conf`. A file name of `-` specifies that the table of interfaces will be read from standard input.

`-I`

This option is obsolete and remains only for historical purposes.

`-n`

Specifies a no execute option so that commands which would be used are printed but not actually executed.

`-R`

Instructs **grifconfig** to call the **grroute** command to initialize the kernel with the remote routes that are accessible via gateways on the network attached to the interface being configured.

To make changes to the configuration file take efect, run the **grifconfi**g **-f /etc/grifconfig.conf** command to initialize the new entries. The **gratm gx0y** command parses the file.

grinch
(CLI+shell)

The **grinch** command sends one or more copies of a GRINCH configuration request to a specified media card.

grinch interprets its command-line arguments as a list of values that specify the contents of each four-octet word of the packet data. Values are specified in standard C notation: a leading '0x' indicates a hexadecimal number; a leading '0' indicates an octal number; everything else is a decimal number. By default, **grinch** will send the packet to the media card specified by the **-p** option, or to the media card specified by the value of the GRID_PORT environment variable if the **-p** option is not used.

Settings made using **grinch** commands are temporary. This is because the CLI program, **ncli**, writes changes made via the CLI to **grinch** variables, these settings are permanent. Changes made with **grinch** commands do not get written to **ncli**, therefore, these changes are temporary and do not survive system reboot.

Permission level: system

Syntax

```
grinch   [-A]  [-c count]  [-D]  [-d]  [-f file]  [-h hwtype]
[-i interval] [-p slot_number]  [-q]  [-v]  [-x]  value [. . . ]
```

Options

`-B`

Specifies that the value command-line arguments (following the options) will specify the values of individual bytes in the packet, not entire words. The arguments for each byte must be presented in network byte order.

`-c` *`count`*

Specifies that ***count*** copies of the packet will be sent to the destination media card. By default, the packets will be sent one second apart.

`-d`

Specifies the debug option (not currently implemented).

`-h` *`hwtype`*

Specifies that hardware type field in the GRID header should be set to value for ***hwtype*** .

`-i` *`interval`*

Specifies each copy of packet is sent interval seconds apart.

`-p` *`slot_number`*

Specifies that the packet or group of packets will be sent to the destination media card in ***slot_number.***

`-q`

Specifies the quiet option in which **grinch** does not print any information about received packets.

`-v`

Sets verbose option in which **grinch** prints contents of any unexpected packets received (usually ignored).

`-x`

Specifies that all integer variables are printed in hexadecimal format (with a leading 0x).

Note that you cannot follow a **grinch** command with spaces. The following example fails because of the spaces before and after the equal (=) sign:

```
# grinch -p0 2.12.2.1.15.3.1 = 100000
grinch: No value to set.
Ignoring ''.
2.12.2.1.15.3.1: 10
100000: no such item
```

Without the space, the command executes properly:

```
# grinch -p0 2.12.2.1.15.3.1=100000
2.12.2.1.15.3.1: OK
```

ICMP message examples

These **grinch** commands control how many ICMP messages of each type a media card will generate each second. To begin message generation immediately without resetting the card, use the appropriate **grinch** command:

1. to set echo-reply, use:
   `grinch -p<slot> 2.12.2.card+1.4.port+1.15.3.1=<value>`
2. to set unreachable, use:
   `grinch -p<slot> 2.12.2.card+1.4.port+1.15.3.2=<value>`
3. to set redirect, use:
   `grinch -p<slot> 2.12.2.card+1.4.port+1.15.3.4=<value>`
4. to set TTL timeout, use:
   `grinch -p<slot> 2.12.2.card+1.4.port+1.15.3.9=<value>`
5. to set param-problem, use:
   `grinch -p<slot> 2.12.2.card+1.4.port+1.15.3.9=<value>`
6. to set time-stamp-reply, use
   `grinch -p<slot> 2.12.2.card+1.4.port+1.15.3.11=<value>`

HDLC parameter examples

1. to set debug level, use:
   `grinch -p<slot> 2.12.2.card+1.4.port+1.2.3.1=<value>`
2. to set keepalive-enabled, use:
   `grinch -p<slot> 2.12.2.card+1.4.port+1.2.3.2=on`
3. to set keepalive-interval, use:
   `grinch -p<slot> 2.12.2.card+1.4.port+1.2.3.3=<value>`
4. to set keepalive-error-threshold, use:
   `grinch -p<slot> 2.12.2.card+1.4.port+1.2.3.4=<value>`

Note: if you now reboot the box, you must rerun the command(s).

grinstall
(shell)

The **grinstall** command installs new system software only on a GRF system that uses an RMS node. A site typically uses this command when it has downloaded a new version of software via ftp.

Unless otherwise specified, **grinstall** searches the ***directory*** `/usr/netstar/Release` for an archive of software for the indicated ***release***. It unpacks and installs the software contained in the archive. After a successful installation, the script writes a confirmation to the file `/usr/netstar/release/log`.

Refer also to the **grinstall** man page.

Syntax

```
grinstall release_name [-nqtv] [-d directory] [-f file] [-s script]
                       [-r current_version] [root]
```

Options

`release_name`

Name of GRF software release, 1.4.12, for example.

`-n`

Suppresses log record of the installation.

`-q`

Suppresses installation status messages from **grinstall.**

`-t`

Prints a list of the files that will be extracted (equivalent to the **tar(1) -v** option).

`-v`

When used with **-t** option, prints a verbose list of the files that will be extracted (equivalent to the **tar(1) -v** option).

`-d directory`

Specifies the directory in which the **grinstall** script searches for the archived software. If a file `named.log` exists in ***directory*** and the ***-n*** or ***-t*** options are not used, a record of the installation is logged to the file.

`-f file`

File from which **grinstall** extracts software instead of default ***release_name***.

`-s script`

Specifies that **grinstall** extract the specified ***script*** from the archive and use it to unpack the software instead of the default script **EXTRACT.**

`-r current_version`

Specifies the currently running version of software. This option is normally used when there is no Log file in which **grinstall** can find the currently running version.

`root`

Not available.

grlamap
(CLI+shell)

The **grlamap** script is used to initialize and manipulate the configuration of logical addresses on HIPPI media cards. When a HIPPI media card boots, **grinchd** uses **grlamap** to configure the logical addresses for that card.

By default, **grlamap** reads logical address configuration information from the file `/etc/grlamap.conf` and uses **grinch** to download the configuration information to the appropriate media card. Please see Chapter 2 for a template of the `/etc/grlamap.conf` file.

Permission level: system

Syntax

```
grlamap  -p slot [-bdnq]  [-f config_filename]  [-i timeout_interval]
[-o output_filename]
```

Options

`-p` *`slot`*

Send only the logical address mappings intended for ***slot***.

If ***slot*** is “all”, send logical address mappings to all cards.

`-b`

Background, when set, all messages go to **syslogd** only. If not set, messages are printed to standard out.

`-d`

Print debug messages.

`-n`

No execution: prints **grinch** commands that would be used to configure the media card, but does not actually execute them.

`-q`

Query the media card for its logical address mappings and print the results.

`-f` *`config_filename`*

Get configuration from ***config_filename*** instead of the default, `/etc/grlamap.conf`.

`-i` *`timeout_interval`*

Indicates how long to wait for a reply from the media card.

`-o` *`output_filename`*

Sends all output to ***output_filename.***

Default configuration file

`/etc/grlamap.conf`

This is the default configuration file for the HIPPI logical address configuration. The name of another configuration file can be specified as the ***-f config_filename*** option. A file name of **–** *(hyphen)* specifies that the entries will be read from standard input.

grmaint
(CLI+shell)

Sends a hand-coded maintenance packet to a port.

Permission level: system

Syntax

```
grmaint [-B] [-c count] [-d] [-h hwtype] [-i interval] [-p portcard]
[-q]
        [-v] value [...]
```

Options

`-B`

Specifies that the value command-line arguments (following the options) will specify the values of individual bytes in the packet, not entire words. Arguments for each byte must be presented in network byte order.

`-c count`

Specifies that count copies of the packet will be sent to the destination media card. By default, the packets will be sent one second apart.

`-d`

Specifies the debug option, which currently does nothing.

`-h` *`hwtype`*

Specifies that the value of the hardware type field in the GRID header should be set to the value for *`hwtype`*.

`-i` *`interval`*

Specifies that each copy of the packet will be sent interval seconds apart.

`-p` *`portcard`*

Specifies that the packet or group of packets will be sent to the destination portcard.

`-q`

Specifies the quiet option, no information printed.

`-v`

Specifies the verbose option, contents of any unexpected packets received.

`value`

Specify the values of individual bytes in the packet, not entire words.

grmem
CLI+shell

This is a debug-only command.

It is used to access media card memory for debug purposes. Lucent recommends you use this command only under direction of Customer Support staff.

Permission level: system

Syntax

```
grmem [-B] [-c count] [-d] [-h hwtype] [-i interval] [-p portcard] [-v]
         [-q] mem_request [...]
       [portcard:]address[,count]
       [portcard:]address=value[[,value]...]
```

portcard

The media card for which this request is intended.

address

The (starting) address being set or examined, specified in standard C notation. A leading '0x' indicates a hexadecimal number; a leading '0' indicates an octal number; anything else is a decimal number.

count

Specifies that count consecutive words of memory should be examined and their contents displayed, **count** is separated from the address by a comma. (When no count is specified, only a single word of memory is examined and displayed.)

value

Specifies that the contents of address should be set to **value**, which is specified in standard C notation. A leading '0x' indicates a hexadecimal number; a leading '0' indicates an octal number; anything else is a decimal number. **value** is separated from the address by an equal sign (=). A comma-separated list of value specifies that each consecutive word in memory, starting at address, should be set to each consecutive value in the list.

Options

`-B`

Print the memory values as separate 8-bit bytes, not as complete 32-bit words.

`-c` *count*

Specifies that count groups of packets will be sent to the destination port card. By default, each group of packets will be sent one second apart.

`-d`

Specifies the debug option, which currently does nothing.

`-h` *hwtype*

Specifies that the value of the hardware type field in the GRID header should be set to the value for ***hwtype***.

`-i` *`interval`*

Specifies that each copy of the packet (or group of packets) will be sent ***interval*** seconds apart.

`-p` *`portcard`*

Specifies that the packet or group of packets will be sent to the destination portcard.

`-q`

Specifies the quiet option. **grmem** will not print any information about received packets.

`-v`

Specifies the verbose option. **grmem** will print the contents of any unexpected packets received (which are otherwise simply ignored).

grmrflash
(CLI+shell)

This command completely initializes a flash memory device prior to installing software on the GRF control board. **grmrflash** replaces **grflash**.

A workaround may be required to use this command due to the lack of space on RAM.

Warning: Dangerous, use only in an emergency recovery situation under direction of the Customer Support staff.

Permission level: system

Syntax

```
grmrflash
```

If there is not enough space in RAM to load new release files, use this procedure:

1 Log in as root. Put all media cards in hold:

```
# grreset -h all
```

2 Delete the media card binaries to make room in memory:

```
# rm -rf /usr/libexec/portcards
```

Now there will be enough room for the new release files and you can proceed with **grmrflash**.

grms
(CLI+shell)

Non-privileged users can use **grms** to shutdown, reboot, or halt a GRF system that uses an RMS node. **grms** can be configured with specific-purpose logins as described below. A man page for **grms** is available.

The primary requirement is that the **grms** command be entered directly from the RMS node or the VT-100. It will not function remotely if entered from another terminal networked to the node. If the site operates in this manner, the **shutdown** command and its corresponding options should be used.

grms -h corresponds to the **-h** option of the **shutdown** command. **grms -s** corresponds to **shutdown** with no options.

Enter **grms** without an option to view a short user guide.

Permission level: system

See also: **shutdown**

Syntax

```
grms   [-h]  [-r]  [-s]
```

Options

`-h`

Halts operating system indefinitely.

`-r`

Reboots system.

`-s`

Shuts down the system in an orderly manner.

Special logins

If you are working from the VT-100 or similar unit connected to the control board, specific-purpose logins enable non-privileged operators to halt, reboot, or shut down the system :

```
login: halt
login: reboot
login: shutdown
```

To configure these logins, run **adduser** and add these account entries in the password file:

```
halt:*:0:0::0:0:halt:/:/bin/grms-halt
reboot:*:0:0::0:0:reboot:/:/bin/grms-reboot
shutdown:*:0:0::0:0:shutdown:/:/bin/grms-shutdown
```

After running **adduser**, use the **passwd** command to assign passwords to these accounts.

grppp
(CLI+shell)

The **grppp** is a program that control PPP on the GRF. It manages a PPP interface and can be used to configure the interface as well as to display status. Please see Chapter 2 for a template of the `/etc/grppp.conf` file.

Warning: Configuration changes made with **grppp** interactively are not maintained across media card reboots. Permanent changes must be made in the `/etc/grppp.conf` file.

Permission level: system

Syntax

`grppp [-n] [-f` *`command_file`*`] [-i` *`interface_name`*`]`

`-n`

Enables the no execute option. Instead of executing the corresponding **grinch** commands, they are copied to `standard out`. Using this option is an excellent way to verify editing changes to a batch file, such as `/etc/grppp.conf`, without changing the system.

`-f` *`command_file`*

Starts batch mode instead of running interactively, **grppp** reads its input from ***command_file***. When end-of-file is reached, **grppp** exits.

`-i` *`interface_name`*

Execute commands only for the interface ***interface_name***. Commands for any other interface are silently ignored.

Options

`help`

Displays a synopsis of all the commands.

`interface` *`interface_name`*

Attaches **grppp** to a PPP interface. PPP interface names follow the `gx0yz` interface name conventions

`quit`

Terminates the program.

`show configuration`

Displays the current values of all configurable PPP objects.

Interactive commands

You can use a **grppp** command to display interface-specific protocol information. These are the **grppp** status commands:

```
show configuration
show negotiation trace status
show maximum configuration request count
show maximum failure count
show maximum terminate count
show restart timer interval
show lcp keepalive interval
show lcp keepalive packet threshold
show lcp mru
show lcp status
show lqr timer interval
show lqr status
show ipcp status
```

Enter commands in lower case, short forms of words can be used. First, you must declare a specific logical interface with an **interface gx0yz** statement, then the `gx0yz>` prompt is generated.

```
# grppp
> interface gs090
gs090>
```

Once you have set the target interface in the prompt, you can enter the desired **show** commands against this interface. Use another **interface** statement to change interfaces.

Here are two examples:

```
# grppp
> interface go060
go060> show config
  General Configuration:
    Maximum configure request count: 10
    Maximum request failure count: 5
    Maximum terminate request count: 2
    Negotiation tracing is enabled
    Restart timer interval: 3000 milliseconds
  LCP Configuration:
    Magic number is disabled
    Initial MRU: 1500
    Keepalive interval: 0 milleseconds, disabled
    Keepalive packet threshold: 5
  LQR Configuration:
    LQR is disabled
    Timer interval: 0 milleseconds
  IPCP Configuration:
    enable IPCP
  OSINLCP Configuration:
    disable OSINLCP
go060>
```

Here is the output from a **show lcp status** command:

```
go060> show lcp st
  LCP Status:
    Bad addresses: 0
    Bad controls: 0
    Packets too long: 0
    Bad FCSs: 0
    Local MRU: 1500
    Remote MRU: 1500
  LCP Configuration:
    Magic number is disabled
    Initial MRU: 1500
    Keepalive interval: 0 milleseconds, disabled
    Keepalive packet threshold: 5
go060>
```

grreset
(CLI+shell)

This command resets one or more specified media cards. The **grreset** command can be used on a media card without disturbing normal GRF system operations.

The **grreset -D *slot*** command is the suggested way to perform a media card dump.

Permission level: system

Syntax

```
grreset  [-Ddhv]  all
grreset  [-Ddhv]  port   [...]
```

`slot`
: Chassis slot number, from 0 to 3 or 0 to 15.

`all`
: All media cards are selected to reset.

Options

`-d`
: Turns debug mode on, debug messages are sent.

`-D`
: Dumps memory when media card comes back up.

`-h`
: Holds media card in reset until a second reset command is executed.

`-v`
: Specifies the verbose option. **grreset** prints the contents of any unexpected packets received (which are otherwise simply ignored).

Examples

You must log in as root. The **grreset** command requires that you specify the appropriate media card by its chassis slot number.

To reset all the media cards, enter:
```
# grreset all
```

To reset the media cards in slots 0 and 1, enter:
```
# grreset 0 1
```

To reset the card in slot 4 and dump its memory, enter:
```
# grreset -D 4
```

To reset the card in slot 4 and return debug information, enter:
```
# grreset -d 4
```

The hold reset option (-h) has numerous uses, here are three.

- To isolate a possible problem, set all cards to hold reset and bring them on-line one at a time.
- To determine whether a card is affecting other media cards, put that one in hold reset to verify what is happening. If certain cards cannot restart while under heavy load from the router switch, you can put all cards in hold reset and then bring up those cards having trouble first while the switch load is light.
- These commands hold either all or one media card in reset:

```
# grreset all -h
# grreset slot -h
```

grrmb
(CLI+shell)

The **grrmb** command sends one or more GRID packets containing ASCII strings to the destination media card or the GRF control board (router manager board).

Additionally, **grrmb** enables you to access a development version of the maintenance command set that runs on the GRF control board and a RMS node system's router manager board (RMB). Each media card has a set of maint commands that display statistics for the card and its switch and combus connections.

After **grrmb** is entered, the GR ##> prompt returns where ## is the number of a chassis slot. The default is 66, which specifies that the command will act on slot 66, the control board. To change the default slot, use the maintenance command **port *slot_number*** as shown in the example below. To exit **grrmb**, enter **quit** or **exit**.

Note: Do not enter a space after the GR ##> prompt. Leading spaces in **maint** commands prevent them from being parsed.

Permission level: system

Syntax

```
grrmb [-c string] [-d] [-f file] [-h hwtype] [-i interval] [-n]
             [-p portcard] [-v]
```

Options

`-c string`
Sends the specified string to the destination media card.

`-d` Specifies the debug option, which currently does nothing.

`-f` *`file`*
Reads text from the specified file instead of from standard input.

`-h hwtype`
Specifies that the value of the hardware type field in the GRID header should be set to the value for *`hwtype`*.

`-i` *`interval`*
Specifies the amount of time (in seconds) to wait before timing out a request sent via the **-c** option. Default is 1 second.

`-n`
No wait option, do not wait for a response when sending a command via the **-c** option.

`-p portcard`
Specifies that the packet or packets will be sent to the destination media card.

`-v` Specifies the verbose option, which currently does nothing.

maint example

The following scenario starts the GR##> prompt, displays a list of commands, and then exits **grrmb** to return to the CLI prompt:

```
super> grrmb
GR 66> ?
Ascend Commands
        ?
        bignore
        fan
        maint -[hi|fd] [<port>:]<data> [,<data>[...]]
        power
        port <port>
        temp
GR 66> quit
super>
```

maint-level commands

`?`

Lists **grrmb** command set.

`bignore`

Displays broadcast ignore status, on or off.

`fan`

Displays chassis fan RPMs.

Here is output for a GRF 400:

```
GR 02> fan
Fan 0 : Curr 75 Last 9 Diff 66
Fan 1 : Curr 229 Last 162 Diff 67
Fan 2 : Curr 179 Last 112 Diff 67
```

Here is output for a GRF 1600:

```
GR 15> fan
Fan 0 : Curr 1 Last 0 Diff 1
Fan 1 : Curr 1 Last 0 Diff 1
```

`maint`

Operates media-specific commands, described in media configuration chapters in GRF *Configuration and Management* manual.

`power`

Displays on/off status of GRF 1600 power supplies, not available on GRF 400:

```
GR 15> power
power {1|2} {on|off}
```

`port number`

Sets slot number, 0–3, 0–15.

`temp`

Returns current temperatures measured at the control board:

```
GR 66> temp
                     LT     HT    TEMP
                   ----------------------
 Measured Junction     55 C   60 C   26 C
Calc. Ext. Ambient     49 C   54 C   20 C
```

The **reset**, **echo**, **help**, and **ver** commands are listed but not available.

grroute
(CLI+shell)

The **grroute** script initializes and manipulates the *static* routes maintained in the `/etc/grroute.conf` configuration file.

Note: These remote routes are accessible via gateways on networks directly connected to media card interfaces.

By default, **grroute** reads a table of routes from `/etc/grroute.conf` and downloads the routes to all available media cards. When a new media card boots, **grroute** is used to configure the new remote route into the kernel and the kernel then distributes them to currently active media cards. Please see Chapter 2 for a template of the `/etc/grroute.conf` file.

Note: If GateD is configured and in use, set static routes in a GateD Static statement. There should be no entries in `/etc/grroute.conf`.

More information is available in the **grroute** man page.

Permission level: system

Syntax

```
grroute [-n]  [-f file]  [-p card [...]]
grroute -g gatenet  -m mask  [-n]  [-f file]   [destination [...]]
grroute -c  [-n]   destination  netmask  gateway
```

Arguments

The format of each entry in `/etc/grroute.conf` is:

`destination   netmask   gateway`

`destination`

The Internet address of the destination host or network (a default route can be specified with a ***destination*** value of `default` or `0.0.0.0`).

`netmask`

Netmask for the destination (must be `255.255.255.255` for destination hosts).

`gateway`

The Internet address of the gateway to which packets destined for ***destination*** will be forwarded. This gateway must be on a network directly attached to a media card interface.

Options

`-c`

Causes **grroute** to configure a single route from command line arguments instead of reading arguments from `/etc/grroute.conf.`

`-f` *`file`*

Causes **grroute** to read the table of routes from ***file*** instead of the default `/etc/grroute.conf`. A file name of "-" specifies the table will be read from standard input.

`-g` ***gatenet***

Specifies that **grroute** configure only those routes whose gateway is on a directly-attached network ***gatenet* .** Use with **-m *mask*** to specify the network mask for ***gatenett.***

`-m` ***mask***

Used with **-g *gatenet*** to specify the network mask for ***gatenet***.

`-n`

Prints the **route** commands that would be used to configure the kernel or media cards but does not execute them.

grrt
(CLI+shell)

The **grrt** command can be used to examine individual entries on a specific media card, or to examine the entire route table on a media card.

Note: During normal operations**, grrt** is not recommended for route table configuration because it does not ensure that route tables are synchronized among the various media cards. Without this coordination, the router is not likely to forward traffic correctly.

The recommended method for configuring routes is to use the `/etc/grifconfig.conf` and `/etc/grroute.conf` files. Before you look at the route table, use this command to display the number of entries:

```
# netstat -rn | wc -l
```

See also: **traceroute**, **netstat**

Permission level: system

Syntax

```
grrt [-c count] [-d] [-h hwtype] [-i interval] [-p portcard] [-v]
       address  netmask nexthop dest_if [metric]

grrt [-c count] [-d] [-h hwtype] [-i interval] [-p portcard] [-v]
       nexthop  dest_if

grrt -[s] [-c count] [-d] [-h hwtype] [-i interval] [-p portcard]
       [-v] address

grrt -[S] [-c count] [-d] [-h hwtype] [-i interval] [-p portcard] [-v]
```

Arguments

The **grrt** command interprets command line-arguments as follows:

`address`

IP address of the destination host or network. A destination host is indicated by specifying a netmask of all ones.

`netmask`

Netmask of the destination address. A destination host address is indicated by specifying a netmask of 255.255.255.255 (all ones).

`nexthop`

IP address of the system on a directly-attached network to which the packet should be forwarded

`dest_if`

GRIT (GR Internal Transmission) address of the media card/interface to which packets intended for the destination address should be routed. A GRIT address takes the form 0:<*slot*>:<*interface*>, for example 0:2:11.

`metric`

A flag used to indicate special cases:

- 0, normal host or network route.

- 16, internal route to the RMS (the value of *nexthop* is ignored).
- 32, route to a network directly attached to the router (value of *nexthop*, if any, is ignored).

grrt command options

The **grrt** command supports the following options that specify the configuration command to be performed and the command-line arguments expected:

`-s` *`address`*

Shows (displays) the route table entry in a media card routing table that would be used for a packet destined for ***address*** (does not show the default route).

`-S`

Shows (displays) a media card routing table (does not show the default route).

Functional options

The **grrt** command supports the following options:

`-c` *`count`*

Specifies that ***count*** copies of the command will be sent to the destination media card (by default, each copy will be sent one second apart).

`-d`

Specifies the debug option (not available).

`-h` *`hwtype`*

Specifies that the value of the hardware type field in the GRID header should be set to the value for ***hwtype***.

`- H`

Specifies that whenever it receives a packet from a card, **grrt** will precede its print out of the routes contained in that packet with a header describing each column in the output.

`-i` *`interval`*

Specifies that each copy of the command will be sent ***interval*** seconds apart.

`-n`

Specifies that **grrt** will number each route it prints.

`-p` *`slot`*

Specifies that the command will be sent to the destination media card.

`-q`

Specifies the quiet option in which **grrt** does not print any information about received packets.

`-v`

Specifies the verbose option.

If specified with the -S option, **grrt** prints the number of routes it expects to receive from the card and prints a line specifying when the traversal of the route table is complete. **grrt** also prints the contents of any unexpected packets received (which are otherwise simply ignored).

Deprecated options

`-a` *`address netmask nexthop dest_if`* `[`*`metric`*`]`

Adds a route to a media card route table but does not change the kernel, changes are lost at media card reset (similar to UNIX **route** command).

`-A` *`nexthop dest_if`*

Adds a default route to a media card routing table.

`-r` *`address`*

Deletes a route from a media card routing table.

`-R`

Deletes the default route from a media card routing table.

Example

Before you look at the route table, you can use this command to display the number of entries:

```
# netstat -rn | wc -l
    342
```

To view the routing table for the media card in slot 1, enter:

```
# grrt -p 1 -S
```

Here is an example of the type of information returned:

```
# grrt -S -p 1
default                            0    0.0.0.0         RMS    UNREACH
0.0.0.0          255.255.255.255  1    0.0.0.0         RMS       DROP
10.20.1.0        255.255.255.0    22   0.0.0.0         ge034      FWD
10.20.1.133      255.255.255.255  21   0.0.0.0         ge034    LOCAL
10.20.1.255      255.255.255.255  20   0.0.0.0         ge034    BCAST
10.20.2.0        255.255.255.0    10   10.205.1.150    ga00f0     FWD
10.205.1.0       255.255.255.0    9    0.0.0.0         ga00f0     FWD
10.205.1.133     255.255.255.255  8    0.0.0.0         ga00f0   LOCAL
10.205.1.255     255.255.255.255  7    0.0.0.0         ga00f0   BCAST
127.0.0.0        255.0.0.0        3    0.0.0.0         RMS       DROP
127.0.0.1        255.255.255.255  3    0.0.0.0         RMS       DROP
198.174.11.0     255.255.255.0    4   206.146.160.1    RMS        RMS
192.168.8.0      255.255.255.0    16   0.0.0.0         gf020      FWD
192.168.8.133    255.255.255.255  15   0.0.0.0         gf020    LOCAL
192.168.8.255    255.255.255.255  14   0.0.0.0         gf020    BCAST
192.168.10.0     255.255.255.0    24   10.205.3.150    ga00f1     FWD
192.168.10.0     255.255.255.0    24   10.205.3.150    ga00f1     FWD
192.168.1.1      255.255.255.255  25   204.101.9.150   ge031      FWD
192.168.1.2      255.255.255.255  23   205.1.1.2       inx 0      ATMP
224.0.0.0        240.0.0.0        2    0.0.0.0         RMS      MCAST
224.0.0.0        255.0.0.0        2    0.0.0.0         RMS      MCAST
224.0.0.0        255.255.255.255  2    0.0.0.0         RMS      MCAST
224.0.0.9        255.255.255.255  15   0.0.0.0         RMS      MCAST
255.255.255.255  255.255.255.255  7    0.0.0.0         RMS      BCAST
#
```

grsavecore
(CLI+shell)

The **grsavecore** command copies and formats information generated when a kernel panic occurs. The data is written to standard output. Normally it is used to copy data out of flash or a swap partition (similar to **savecore**), or to pretty-print that data.

Permission level: system

Syntax

```
grsavecore [-h] [-d] [-f] [-n] [-r] [-C] [-t] [-i inputfile]
```

Options

`-h`

Causes a brief usage message to be printed, the program then exits without performing any other action.

`-d`

Sets debug option enables the printing of a little extra diagnostic information to `stderr` as the program runs.

`-f`

Sets format option to cause the dump data to be formatted so that it human-readable for humans (default).

`-i` *`inputfile`*

Specifies the filename to read as raw input data. By default, **grsavecore** reads data from from the dump device (normally a partition, `/dev/wd0b`, on a flash disk). This is useful for formatting a file that was previously generated by running this program with the **-r** flag.

`-n`

Generate no output, sometimes useful with -C option.

`-r`

Sets raw option, causes dumped data to be output without any alteration or formatting. This is useful when for copying data from the dump device to a normal file system.

`-C`

Sets the clear flag to cause the header block on the data source (either default or specified with -i) to be overwritten. This is most useful for clearing out old dumps from the dump device. The block is only clobbered after the data has been successfully read and output.

`-t`

Sets stack trace residue for use with **gdb.**

grsite
(CLI+shell)

grsite saves specified files that are not located in the `/etc` directory. **grwrite** saves all the files in the `/etc` directory.

The **grsite** command manages special site-specific images, creating a site file that contains the special images. These images are extra files that are installed after the main release is moved to the RAM drive. The site file overlays the system and configuration files, and in this way allows special files to be easily incorporated into a configuration and just as easily removed.

grsite preserves the site file over a system boot. The **grsite --perm** option preserves the site file across a **grfins** installation or upgrade.

grsite can be useful for installing a single binary image, such as a `/var/portcards/fddi.run` binary, that is to be used for debugging or testing. The **grsite** command provides options to add, delete, and list files in either the current release set, the next boot release set, or an arbitrary release set. Note that the command works relative to the root of the file system unless told explicitly to do otherwise.

To add the `fddi.run` binary to the currently-running system:

```
# cd /var/portcards
# grsite fddi.run
```

Note that **grsite** provides the `/var/portcard` prefix when it copies the ***pattern*** (`fddi.run`) into the site file.

See also: **grwrite**, **grsnapshot**

Permission level: system

Syntax

```
grsite  options  pattern
```

Options

`--delete`

Indicates that the ***pattern***`(s)` should be removed from the given release.

`--list`

Lists the given ***pattern*** or, if no ***patterns*** are specified, displays all files in the site file archive. Note that duplicate entries may exist if a file has been added (that is, overwritten). The last entry is the one that will get used.

`--next`

Uses the next-boot version instead of the current version.

`--nopath`

Indicates that the command should not fill in the current directory name in front of the file patterns. The expectation is that the user would then fill in the complete relative path on each pattern. For example, to install `/home/me/var/portcards/fddi.run`, the user would **cd** to `/home/me` and issue a **grsite** command with the **--nopath** option and a pattern of /`var/portcards/fddi.run`.

`--perm`

Perfoms the action to the “permanent” site file. The permanent site file is carried forward during an upgrade or installation via **grfins**.

`--target`

Specifies the target device, the target can be one of the following:

P (internal primary flash), S (internal secondary flash), A (PCMCIA slot A), or B (PCMCIA slot B).

`--version`

Specifies an explicit version instead of using the next boot or the current boot.

`--help`

Provides a short help message.

`pattern`

A file name prefix or file name.

Examples

grsite can create two files (databases):

- one to save changes during a reboot
- one to save changes when upgrading

For example:

`1.4.x,default.site.gz`	- the database used to retain changes when rebooting
`1.4.x,default.sitep.gz`	- database used to retain changes when upgrading

To view contents in the `1.4.x,default.site.gz` file:

```
# grsite --list
```

To view contents in the `1.4.x,default.sitep.gz` file:

```
# grsite --list --perm
```

If you wish to **grsite** a directory and its contents, change to the parent directory and execute **grsite** on directory name. For example, enter:

```
# grsite foo
```

where `foo` is a subdirectory off `root` (/`foo`)

Note: If you delete a file or directory from the RAM file system before executing the **grsite --delete** command, you can **tar gunzip** it from `/flash`. This is to prevent rebooting the router to restore defaults.

For example, if you deleted the directory `/foo` with contents before executing **grsite**, execute the following command to view contents of the tar file:

```
# gzcat /flash/1.4.x,default.site.TAR.gz | tar tf -
```

results from `gzcat`:

```
/foo
/foo/file1
/foo/file2
/foo/dir
/foo/dir/file3
```

where `1.4.x,default.site.TAR.gz` is the site file to restore defaults.

Look for files or directory to extract, go to parent directory on the RAM file system and execute:

```
# gzcat /flash/1.4.x,default.site.TAR.gz | tar tf - /foo
```

Caution: Permanent site files are not carried over when switching from one version to another using **setver** command. Permanent site files are carried over when using **grfins**.

grsnapshot
(CLI+shell)

This command is for GRF systems with new control boards. The equivalent command for RMS node-based systems is **grc**.

The **grsnapshot** command saves configurations and release images to the internal or external flash device, or to another file system. It is used to copy or create another version of the configuration files or to make a complete backup of the flash device. This script will back up, copy, and save configuration files to either a new version name or the same version name on a different flash device.

To make an exact, bootable copy of the internal flash device to the external flash device, **grsnapshot** is used with the **--dup** option. This command initializes the external flash device, writes a file system to it, writes the boot blocks to the device, and copies all the files and directories from the internal flash drive to the external flash drive.

See also: **grwrite**, **grsite, grc**

Permission level: system

Syntax

```
grsnapshot -s[P|S|A|B|u|d]=release,version
           -d[P|S|A|B|u|d]=release,version
grsnapshot --dup=[PA|PB|SA|SB|AP|AS|BP|BS]   [--force]
```

`release,version`
Specifies the system software release and version name, as in `1.4.12,default`

Options

`P`
Primary internal flash (GRF has only a primary flash).

`S`
Secondary internal flash (not available).

`A`
External PCMCIA slot A.

`B`
External PCMCIA slot B.

`d`
Specifies the directory to copy from/to.

`u`
Specifies the URL to copy from/to (not available).

`--s`
Set the source release to copy from.

`--d`
Set the destination release to copy to.

`--dup=`

Duplicates the source to destination.

Therefore, **PA** is primary internal flash to device in slot A. **AP** is from device in slot A to primary internal flash.

`--force`

Used with **--dup**, requires a copy or overwrite to be done if a file system exists on the destination device.

Examples

The following command copies the `1.4.x,default` system on the internal flash to the external flash device in slot B as `1.4.x,default:`

```
# grsnapshot -sP=1.4.x,default -dB=1.4.x,default
```

The following command makes a copy of the `1.4.x,default` system on the internal flash to the file name `1.4.12,bob` on the internal flash:

```
# grsnapshot -sP=1.4.x,default -dP=1.4.12,bob
```

The following command copies or overwrites the configuration contained in the file `/tmp/mybackup.tar.gz` onto the internal flash device:

```
# grsnapshot -sd=/tmp/mybackup.tar.gz -dP
```

The following command creates an alternate configuration on the internal flash based upon the currently-running configuration on the internal flash device:

```
# grsnapshot -sP -dP=release,version
```

The following command backs up the internal flash device to the external device in slot B. It initializes the external device, writes a file system to it, writes the boot blocks to the device, and copies all the files and directories from the internal flash device to the external flash device:

```
# grsnapshot --dup=PB
```

The following sequence illustrates how the **--force** option is applied:

```
# grsnapshot --dup=PB
```

If a file system exists on the destination device, you will receive a message to that effect.
To enable the overwrite, use the **--force** option:

```
# grsnapshot --dup=PB --force
```

The following command copies the `1.4.x,default` system on the internal flash to a gzipped tar file with the name `/tmp/mybackup.tar.gz` (the **-dd** option not currently available):

```
# grsnapshot -sP=1.4.x,default -dd=/tmp/mybackup.tar.gz
```

grstat
(shell)

Layer 3 statistics

The **grstat** command reports layer 3 (IP and ICMP) forwarding statistics for all media card types except HIPPI. Error reporting includes the saved source and destination IP addresses of the packet that caused the last error of each type reported.

Layer 2 statistics

The **grstat** command reports many of the Layer 2 (data link layer) statistics currently reported by individual media card **maint** commands. Examples are at the end of this section. Layer 2 statistics are reported for ATM OC-3c (ATM/Q), HSSI, Ethernet, and SONET OC-3c media cards.

In addition, **grstat** now reports a number of ATMP-specific statistics.

grid

GRID manages internal command messaging. Commands such as route updates, interface adds/deletes, GRINCH, are some examples of GRID use. In **grstat**, the **grid** command returns statistics such as the number of **grid** packets received, echoes, and messages dropped. A **grstat grid** example follows the layer 2 examples.

Syntax

`grstat [-ahltvzZ] [-w` *`width`*`] [-c` *`count`*`] [-i` *`seconds`*`]` *`stat_type`* `[`*`card | interface`* `...]`

`stat_type` specifies the type of statistics to retrieve. Types include:

- `grid`, displays the combus and other messaging statistics.
- `ipstat`, lists IP statistics.
- `ipdrop`, lists IPDROP statistics.
- `ip`, lists IPSTAT and IPDROP statistics.
- `icmpin`, lists ICMPIN (input) statistics.
- `icmpout`, lists ICMPOUT (output) statistics.
- `icmperr`, lists ICMPERR (error) statistics.
- `icmp`, lists ICMPERR, ICMPIN, and ICMPOUT statistics.
- `l2`, displays the media I/O statistics:
  `grstat l2rx` , displays receive side statistics.
  `grstat l2tx`, displays transmit side statistics.
- `switch`, displays the switch statistics.

`grid`

List **grid** statistics.

`all`

Lists all of the above statistics.

Options

`card`

Specifies the slot number of the media card to query, 5, for example.

`interface`

Specifies the logical interface name in the format `gx0yz`.

The ***interface*** and ***card*** options can be combined. However, if a logical interface is specified twice as a result of being on a specified card and by being specified as an interface, that interface is counted twice when values are totalled.

`-a`

Enables statistics lines with a count of zero to be printed out, overriding the default case in which they are not. A second **`-a`** also prints zeroed IP addresses instead of blanks in output displays that have IP addresses.

`-h`

Help function, prints the option summary, including the complete list of supported ***stat_type*** values.

`-l`

Lists out data per individual interface. By default, if a card is specified, the data shown is a summary total for all interfaces found on the card.

`-t`

Displays totals for the statistics requested for specified interfaces after the interfaces are listed. This option is not necessary to obtain totals for an entire card. It can be used in conjunction with **-l** to view a total at the bottom, below the list of individual interfaces .

`-v`

Displays additional debugging or error information, generally useful only for debugging.

`-z`

Zeroes ***stat_type*** statistics on the specified card or interface.

`-Z`

Prints out statistics for the specified interface(s) and then zeroes them. This option can be used in conjunction with **-c**.

`-w` *`width`*

Specifies display in ***width*** number of characters. A line width can be set greater than 80 characters to improve viewing truncated statistics descriptions.

`-c` *`count`*

Displays the specified statistics ***count*** number of times at five-second intervals, the default interval is modified with **-i**.

`-i` *`seconds`*

Specifies an interval in seconds, the default is five seconds, used in conjunction with **-c.**

Beware that statistics gathering consumes communications bus bandwidth, avoid using a shorter interval than the default.

Return values

- **grstat** returns "`-1`" if invalid options or other serious errors occur, 0 otherwise:

```
card 1
error -1 reporting data
```

In general. **grstat** tries to keep going when it has trouble talking to what are otherwise supposed to be valid interfaces.

- If a slot is empty, you see an “`unable to gather data`“ message:

```
card 4
unable to gather data for card 4
```

- If there are no statistics available, you see the type of statistic requested and a “`count description`“ message with no data following:

```
card 3
  Switch statistics
               count description
card 4
```

- By default, only the statistics with non-zero counts are displayed unless the user specifies **-a** (print all) with any **grstat** command.

Layer 3 examples

grstat IP commands

- List all IP statistics on the GRF:

```
# grstat ip
all cards (19 interfaces found)
  ipstat totals
       count description
  1022766098 packets received
     1035601 packets dropped
  1021730497 packets forwarded
  ipdrop totals
       count description
     1035601 no route to destination address
```

- List ICMPERR statistics for the system:

```
# grstat icmperr
all cards (19 interfaces found)
  icmperr totals
       count error
     1265253 too much ICMP; type throttled
        7284 no route back to originator
```

- List ICMPERR statistics totals on card 5 and individually for interfaces `gf0b0` and `gf0b1`:

```
# grstat icmperr 5 gf0b0 gf0b1
```

- List IPSTAT statistics totals for all interfaces on the GRF:

```
# grstat ipstat
```

- List IPSTAT statistics individually for all interfaces on the GRF:

```
# grstat -l ipstat
```

- List IPSTAT statistics individually for all interfaces on the GRF and then list the totals for all interfaces found (ignore the `“0 interfaces found “` message at the beginning of the display, the actual total is displayed at the end):

```
# grstat -l -t ipstat
(0 interfaces found)
  ipstat totals
       count description
gs020
  ipstat
       count description
    •
    •
    •
ge035
  ipstat
       count description
       12167 total packets received
           1 packets dropped
        1128 packets forwarded normally
       11038 packets forwarded to the RMS
        7548 multicast packets received
        7548 multicast packets sent to RMS
ge036
  ipstat
       count description
ge037
  ipstat
       count description
all cards (17 interfaces found)
  ipstat totals
       count description
       13782 total packets received
           1 packets dropped
        2665 packets forwarded normally
       11116 packets forwarded to the RMS
        7614 multicast packets received
        7614 multicast packets sent to RMS
```

- List IPSTAT statistics on `gf0b3`, clear them, wait 60 seconds, and repeat a total of 100 times:

```
# grstat -Z -c 100 -i 60 ipstat gf0b3
```

grstat arpserver

The **grstat arpserver** command can be used with an `ga0yz` interface or `ga0y` slot name.

```
# grstat arpserver ga017f
ga017f
  atmarpstat
       count description
           1 InverseARP requests received
         928 InverseARP replies received
          74 ARP requests received
          64 ARP replies transmitted
          10 ARP naks transmitted
```

grstat switch

```
# grstat switch
card 0
  Switch statistics
                count description
                26055 RX packets
              2062084 RX bytes
                22332 TX packets
              1125905 TX bytes
```

grstat -a ip - display all IP statistics available

This is a sample of all the IP counts and statistics that can be covered by **grstat**. When you use the **-a** option, all counts, even if 0, are displayed:

```
# grstat -a ip
all cards (17 interfaces found)
  ipstat totals
       count description
       13751 total packets received
           1 packets dropped
        2665 packets forwarded normally
           0 packets redirected out receiving interface
           0 packets fragmented
           0 fragments created
           0 packets forwarded locally to card
           0 packets handled by the card
       11085 packets forwarded to the RMS
           0 packets segmented to the RMS
        7593 multicast packets received
           0 multicast packets attempted to route
        7593 multicast packets sent to RMS
           0 multicast packets received on rincorrect interface
           0 multicast packets for forwarding
           0 multicast packets (copies) transmitted
           0 packets ATMP encapsulated
           0 packets ATMP decapsulated
           0 packets forwarded to resolved bridge destinations
           0 packets ATMP encapsulated with pre-fragmentation
           0 packets ATMP encapsulated with pre-fragmentation and
                          mtu-limit override
          0 packets ATMP encapsulated with pre-fragmentation,clear-
ing DF
ipdrop totals
       count description
           1 packet received on down interface
           0 received data below IP header minimum
           0 not version 4 IP
           0 header length below minimum
           0 bad header checksum
           0 header length longer than packet length
           0 received data less than packet length
           0 source address from 127.*.*.*
           0 source address from 192.0.2.*
           0 packet filtered on input from media
           0 no route to destination address
```

```
                                          0 destination interface is down
                                          0 TTL expired
                                          0 needed to frag packet but DF set
                                          0 multicast routing not yet supported
                                          0 routing to bridge groups not supported
                                          0 route table says to drop dest address
                                          0 unknown special processing code
                                          0 IP option with length <= 0
                                          0 IP option length past header length
                                          0 Record Route offset less than minimum
                                          0 couln't find interface address for dest
                                          0 TimeStamp offset less than minimum
                                          0 bad TimeStamp option flag
                                          0 TimeStamp option overflow beyond maximum
                                          0 Source Route offset less than minimum
                                          0 next Strict Source Route address not ours
                                          0 bad ICMP checksum
                                          0 local interface is down for ICMP ECHO
                                          0 no route back for reply to ICMP ECHO
                                          0 no next route for Strict Source Route
                                          0 no next route for Loose Source Route
                                          0 no interface addr for Source Route
                                          0 can't forward link-layer broadcast
                                          0 can't forward link-layer multicast
                                          0 loose source routing disabled
                                          0 strict source routing disabled
                                          0 no buffer to generate packet
                                          0 Rate overflow on ICMP generation
                                          0 packet filtered into-me
                                          0 ATMP err: can't find Home Network entry
                                          0 ATMP err: bad GRE header
                                        355 ATMP err: can't find mobile node entry
                                          0 ATMP err: Invalid IP header
                                          0 multicast replicated, original dropped
                                          0 multicast TTL expired
                                          0 multicast packet on wrong interface
                                          0 can't forward to/through control board
```

Layer 2 options and examples

Layer 2 statistics are reported for ATM OC-3c (ATM/Q), HSSI, Ethernet, and SONET OC-3c media cards.

grstat l2

Layer 2 statistics are reported for the entire card. If slots are empty or cards are not responding, error messages are generated.

```
# grstat l2
card 0
  Layer 2 statistics
    physical port 0
               count description
               22316 RX packets
              321724 RX bytes
               32446 TX packets
              625624 TX bytes
               10130 Odd length packets
    physical port 1
               count description
card 1
  Layer 2 statistics
    physical port 0
               count description
               11684 RX packets
              259215 RX bytes
               11467 TX packets
              219288 TX bytes
                   1 Odd length packets
    physical port 1
               count description
card 2
  Layer 2 statistics
    physical port 0
               count description
                 392 RX packets
               37632 RX bytes
                3586 TX packets
             1371312 TX bytes
    physical port 1
               count description
card 3
  Layer 2 statistics
    physical port 0
               count description
                   1 Runt errors
                   1 Alignment errors
    physical port 1
               count description
    physical port 2
               count description
    physical port 3
               count description
    physical port 4
               count description
    physical port 5
               count description
               15667 RX packets
             2551160 RX bytes
                   1 RX discard
                1858 Unknown protocol
                5809 TX packets
             1218274 TX bytes
    physical port 6
               count description
                3723 TX packets
              692058 TX bytes
    physical port 7
               count description
```

Layer 2 statistics are not gathered for FDDI, HIPPI, and ATM OC-12c media cards.

grstat grid

```
# grcard
 0       SONET_V1   running
 2        FDDI_V2   running
# grstat grid
card 0
  GRID statistics
               count description
              141153 COMBUS messages received
              140892 COMBUS GRID messages received
                   2 COMBUS receive long messages
                 261 COMBUS GRIT messages for TX-CPU
             5255261 COMBUS last status register
                   1 COMBUS time in
                 194 GRIDAX packets received
                   4 GRIDAX restart
                  24 GRIDAX acks received
                   5 GRIDAX requested acks received
                  29 GRIDAX control packets received
                 165 GRIDAX output queued
                 187 GRIDAX packets sent
                  22 GRIDAX acks sent
                  22 GRIDAX control packets sent
card 1
unable to gather data for card 1
card 2
  GRID statistics
               count description
               28041 COMBUS messages received
               28041 COMBUS GRID messages received
               27797 COMBUS GRID echo requests
                   4 COMBUS receive long messages
                 207 GRIDAX packets received
                   5 GRIDAX restart
                  27 GRIDAX acks received
                   2 GRIDAX requested acks received
                  29 GRIDAX control packets received
                 178 GRIDAX output queued
                 204 GRIDAX packets sent
                  26 GRIDAX acks sent
                  26 GRIDAX control packets sent
card 3
unable to gather data for card 3
```

By default, only the statistics with non-zero counts are displayed unless the user specifies -**a** (print all).

grwrite
(CLI+shell)

The **grwrite** command saves changes made to the `/etc` directory on the RAM file system over to the internal flash device so the changes are available during the next boot.

Specifically, the **grwrite** command saves the `/etc` directory to flash so the `/etc` configuration files and configuration changes are made permanent, and also preserves symbolic links created within the `/etc` directory. It is critical to save configuration file changes intended to be permanent. Changes made to files other than those in `/etc` must be saved using **grsite**.

When installing a new software version, make sure all changes to files in `/etc` that you intend to be carried forward to the new release have been saved to flash. Execute **grwrite** -before- you begin a software upgrade. If you execute the **grwrite** command after doing a **grfins**, the changes are only saved to the currently-running version. The currently-running version is assigned by **grwrite** to be `default`. **grwrite** uses the **-r** ***revision*** option as the revision structure to copy the files to. If **-r** ***revision*** is not specified, the revision information is retrieved by the **getver** command.

Note: Any changes you make to the `/etc/services` file are overwritten when you install a new software release. Record these changes and add them back after the upgrade.

See also: **grsnapshot**, **grsite**

Permission level: system

Syntax

```
grwrite [-d [device]] [filename] [-n] [-r revision] [-v]
```

Options

The **grwrite** command supports the following options:

`d` *`device_descriptor`*

Uses the specified device descriptor, this option enables another device to be acted upon

`filename`

grwrite saves the specified ***filename*** to flash. Specify a file in the `/etc` directory

`-n`

Specifies the no execute mode. In this mode, **grwrite** checks to see if there are files that need updating but does not save them. If you use the **-vn** option, **grwrite** lists files currently unsaved in the `/etc` directory.

`-r` *`revision`*

Specifies the software release and version against which to perform the **grwrite**. This option is in the form ***release,version***. For example, `1.4.x,default` or `1.4.x,atmtestB`.

`-v`

Specifies verbose mode so that **grwrite** lists the files that are being updated in, added to, and/or removed from flash memeory. If you use the **-vn** option, **grwrite** lists files currently unsaved in the `/etc` directory.

gsm
(CLI + shell)

The GateD State Monitor (GSM) is an interactive interface used to query a running GateD daemon about internal GateD variables. Commands include protocol-specific (IP, OSPF, BGP, RIP) queries for memory, route table, interface list, and other internal parameters. You can start and use the **gsm** command in the CLI or at the UNIX prompt, both methods are described below. The difference between the two is how you access and start the GSM interface. After you start GSM, the commands are the same for both.

Permission level: system

CLI access and syntax

```
gsm [ hostname ]
```

When you enter GSM from the CLI prompt, **gsm** returns information about the GateD running on that local machine. To obtain GateD statistics for a different GRF system, you must access that GateD through the UNIX shell, shell usage is described below.

Shell usage and syntax

```
gsm [help] [option]
```

From a UNIX shell you open a telnet connection on TCP port 616 to the machine running GateD. You can telnet from the administrative LAN or from the GateD machine itself.

Refer to the *GRF GateD* manual for **gsm** usage information.

help or ?
(CLI)

help and its alias **?** return a list of all registered commands authorized by user's security profile. Use **help** or **?** followed by a specific command name to obtain description or usage information.

Permission level: user

Syntax

`help, ?`

`help` *`command-name`*`, ?` *`command-name`*

Example

```
super> help ls

ls profile-type [ profile-index ] [ field-name [ field-index ]
    [ sub-profile-name ] ... ]
Usage:  view a field or group of fields from the specified profile

ls . [ field-name [ field-index ] [ sub-profile-name ] ... ]
Usage:  view a field or group of fields from the working profile,
same as get command
```

ifconfig
(CLI+shell)

This command is modified for GRF use to assign addresses, mask, and other network parameters to a logical interface. Also, when no optional parameters are specified, **ifconfig** displays the current configuration for a network interface. If a protocol family is specified, **ifconfig** reports only the details specific to that protocol family. To use **ifconfig** to modify the configuration of a network interface, you must be logged in as super user.

If you log in to a GRF as `root`, you automatically get the CLI shell. In the CLI, `root` is super user, hence the `super>` prompt.

Permission level: system

Syntax

`ifconfig` *`interface address_family`* `[` *`address`* `[` *`dest_addr`* `]]` `[`*`parameters`*`]`

`ifconfig` *`interface`* `[`*`protocol_family`* `]`

`interface`

GRF interface name in format `gx0yz`

- ATM interface names look like: `ga037f, ga0281, ga01ff`
- FDDI names: `gf030, gf021, gf012, gf003`
- HIPPI names: `gh030` (only the slot # changes)
- HSSI names: `gs030, gs021, gs0180`
- SONET OC-3c: `go030` (only the slot # changes)
- Ethernet names: `ge030, ge026, ge017`

`address_family`

Specifies the address family which affects interpretation of the remaining parameters. Since an interface can receive transmissions in differing protocols with different naming schemes, specifying the address family is recommended. The address families currently supported are INET, ISO, NS, and GRIT. A GRIT address is specified `cage:card:interface`

`address`

IP address, ISO address (for PPP).

`dest-addr`

Specifies the address of the correspondent on the other end of a point-to-point link.

`protocol_family`

Specifies protocol, protocol families currently supported are **inet**, **iso**, **ns**, and **grit**.

`parameters`

The following parameters can be set with **ifconfig**:
```
alias, -alias, arp, broadcast, debug, -debug, delete, dest_address,
down, ipdst, linktype, metric, mtu, netmask, nsellength, ptp, -ptp,
trailers, -trailers, link[0-2], -link[0-2], up, proxy, -proxy
```

`alias`

Specifies an additional or alias network address for the specified interface.

`-alias`

Deletes the specified alias.

`arp`

Enables ARP between network-level addresses and link level addresses (default), currently implemented for mapping between DARPA Internet addresses and 10Mb/s Ethernet addresses.

`-arp`

Disables ARP.

`broadcast`

Specifies the address to use to represent broadcasts to the network, the default broadcast address is the address with a host part of all 1s (inet only).

`debug`

Enables driver-dependent debugging code, usually turns on extra console error logging.

`-debug`

Disables driver-dependent debugging code.

`delete`

Removes specified ***address***.

`dest-addr`

Specifies the address of the correspondent on the other end of a point-to-point link.

`down`

Specifies an interface as down, the system will not attempt to transmit messages through that interface.

`ipdst`

Specify an Internet host which is willing to receive IP packets encapsulating NS packets bound for a remote network. An apparent point-to-point link is constructed, and the address specified will be taken as the NS address and network of the destination.

`linktype` *`type`*

Specifies the type of link to be **type**. More common types are PPP, CHDLC, and Frame Relay. Only point-to-point interfaces support setting the link type. Some types understood by **ifconfig** may not be compiled into or understood by the kernel.

`link[0-2]`

Enables special processing of the link level of the interface. The three link options (0, 1, 2) are interface specific in actual effect; however, they are in general used to select special modes of operation.

`-link[0-2]`

Disable special processing at the link level with the specified interface.

`metric` *`n`*

Sets the routing metric of the interface to ***n***, default is 0. The routing metric is used by the routing protocol **routed**. Higher metrics have the effect of making a route less favorable; metrics are counted as addition hops to the destination network or host

`mtu` *`n`*

Sets the maximum transmission unit of the GRF interface to ***n***.

`netmask` *`mask`*

Specifies how much of the address to reserve for subdividing networks into sub-networks. The mask includes the network part of the local address and the subnet part, which is taken from the host field of the address.

`nsellength` *n*

Specifies a trailing number of bytes for a received NSAP used for local identification, the remaining leading part of which is taken to be the NET (Network Entity Title). (ISO only) The default value is 1, which is conformant to US GOSIP. When an iso address is set in an **ifconfig** command, it is really the NSAP which is being specified.

`ptp`

Sets the point-to-point flag for the interface (only on GRF HIPPI or ATM interfaces).

`-ptp`

Clears the point-to-point flag.

`trailers`

Requests the use of a “trailer” link level encapsulation when sending (default). If a network interface supports trailers, the system will, when possible, encapsulate outgoing messages in a manner which minimizes the number of memory to memory copy operations performed by the receiver.

On networks that support the Address Resolution Protocol, this flag indicates that the system should request that other systems use trailers when sending to this host. Similarly, trailer encapsulations will be sent to other hosts that have made such requests. Currently used by Internet protocols only.

`-trailers`

Disables the use of a “trailer” link level encapsulation.

`up`

Specifies an interface as up, may be used to enable an interface after an **ifconfig down.** If the interface was reset when previously marked down, the hardware is re-initialized.

`proxy`

Enables Proxy ARP on an interface, allowing an interface to respond to ARP requests destined for a host to which the interface has a route.

`-proxy`

Disables Proxy ARP on an interface.

iflash
(CLI)

Warning: Perform a **mountf** before you do an **iflash** command.
The system always (and onlys) boots from the boot files resident on internal flash. **iflash** erases boot files on the internal flash device if so directed.

The **iflash** command initializes (formats) specified flash memory.

Permission level: system

Syntax

```
iflash -P|S|A|B [-f]
iflash -r [raw_device] [-f]
```

Options

`-P`

Use the internal primary flash disk, `/dev/rwd0a`.
Warning: Formatting the internal flash disk can leave the system in an unbootable state. Only use this option if you are instructed to do so by Customer Support.

`-S`

Use the internal secondary flash disk, `/dev/rwd1a`.
Warning: Formatting the internal flash disk can leave the system in an unbootable state. Only use this option if you are instructed to do so by Customer Support.

`-A`

Use the external flash disk in PCMCIA slot A, `/dev/rwd2a`.

`-B`

Use the external flash disk in PCMCIA slot B, `/dev/rwd3a`.

`-f`

Will force the initialization even if device is formatted already. **iflash** will fail if it detects a flash is already initialized. Use this option to override the check.
Caution: The **-f** option wipes the device clean. Use with care.

`-r` *`raw_device`*

Can be used to specify any other device.

Examples

Initialize the external flash device in PCMCIA slot A:

```
iflash -r raw_device    - or -    iflash /dev/rwd2a
```

Initialize the primary internal flash device:

```
iflash -r raw_device    - or -    iflash /dev/rwd0a
```

If you run **iflash** against a corrupted external flash, **iflash** should report file system errors and then format the flash device with a “`Formatting...`” message.

If you run **iflash** against an external flash that is not corrupted, **iflash** should tell you that there already is a file system on the device, and that you need to use the **-f** (force) option to reformat a flash device that contains a valid file system.

list
(CLI)

Lists the field in the current profile. After a profile is read, you can view the contents of that profile by using the **list** command (same as **cd** command).

A second use is to move back (or up) in a profile. The **list ..** command moves you back to the level you were viewing prior to the last **list** or **cd** entered.

See also: **cd**, **get**, **ls**

Permission level: user

Syntax

```
list [field-index] [field-index] [...]
```

```
list .. [ .. ]
```

Example

This example shows how you use **list** to see the contents after you read a profile:

```
super> read dump
DUMP read
super> list
hw-table = <{hippi 20 /var/portcards/grdump 0}{rmb 20 /var/portcar+
dump-vector-table = < {2 fddi"FDDI default dump vectors"<{1"fddi c+
config-spontaneous = off
keep-count = 0

super> list hw-table
hippi = { hippi 20 /var/portcards/grdump 0 }
rmb = { rmb 20 /var/portcards/grdump 3 }
hssi = { hssi 20 /var/portcards/grdump 7 }
dev1 = { dev1 20 /var/portcards/grdump 9 }
atm-oc3+ = { atm-oc3-v2 20 /var/portcards/grdump 5 }
fddi-v2 = { fddi-v2 20 /var/portcards/grdump 6 }
atm-oc1+ = { atm-oc12-v1 20 /var/portcards/grdump 10 }
etherne+ = { ethernet-v1 20 /var/portcards/grdump 8 }
sonet-v1 = { sonet-v1 20 /var/portcards/grdump 11 }
atm-oc12-v2 = { atm-oc12-v2 20 /var/portcards/grdump 14 }

super> list atm-oc12-v2
media = atm-oc12-v2
config = 20
path = /var/portcards/grdump
vector-index = 14
```

load
(CLI)

Restores (loads) previous configuration profile into current use. These files are always located in the `/etc/prof` directory.

Permission level: update

See also: **save**

Syntax

`load` *`filename`*

where *`filename`* is the name of the file in which the configuration script is saved.

Example

Load the previously-saved `system.conf` profile:

```
super> load system.conf
SYSTEM read
SYSTEM written
super>
```

If you try to load a profile that does not exist, or enter a typing error, you see an error message:

```
super> load superisp.conf
error:  superisp.conf does not exist
super>
```

ls
(CLI)

Displays the contents of the current working profile read into local memory. The **ls** command retrieves the names and contents of fields within profiles without changing the user's location within the tree. **get** is an alias of **ls**.

See also: **cd**, **get**, **list**

Permission level: user

Syntax

ls [. | *profile-type* [*profile-index*]] [*field-name field-index* ...]

Example

When you use **ls** to look at fields in the Dump profile, specify each level of profile:

```
super> read dump
DUMP read

super> ls .
hw-table = <{hippi 20 /var/portcards/grdump 0}{rmb 20 /var/portcar+
dump-vector-table = < { 3 rmb "RMB default dump vectors" <{1 SRAM +
config-spontaneous = off
keep-count = 0
```

Look at the hardware table fields:

```
super> ls . hw
hippi = { hippi 20 /var/portcards/grdump 0 }
rmb = { rmb 20 /var/portcards/grdump 3 }
hssi = { hssi 20 /var/portcards/grdump 7 }
dev1 = { dev1 20 /var/portcards/grdump 9 }
atm-oc3+ = { atm-oc3-v2 20 /var/portcards/grdump 5 }
fddi-v2 = { fddi-v2 20 /var/portcards/grdump 6 }
atm-oc1+ = { atm-oc12-v1 20 /var/portcards/grdump 10 }
etherne+ = { ethernet-v1 20 /var/portcards/grdump 8 }
sonet-v1 = { sonet-v1 20 /var/portcards/grdump 11 }
atm-oc12-v2 = { atm-oc12-v2 20 /var/portcards/grdump 14 }
super>
```

Look at the Ethernet dump fields, specify each level of the profile:

```
super> ls . hw eth
media = ethernet-v1
config = 20
path = /var/portcards/grdump
vector-index = 8
super>
```

man
(CLI+shell)

Returns a man page for the specified command

Permission level: system

Syntax

```
man  title [-achw] [-C file] [-M path] [-m path] [[-s] section] ...
```

where *title* is the name of a command.

Options

`-a`
Display all of the manual pages for a specified section and name combination.

`-C`
Use the specified file instead of the default configuration file.

`-c`
Copy the manual page to the standard output instead of using **more** to paginate it.

`-h`
Display only the ``SYNOPSIS" lines of the requested manual pages.

`-M`
Override the list of standard directories **man** searches.

`-m`
Augment the list of standard directories **man.**

`-s`
Restrict the directories to section that **man** will search.

`-w`
List the pathnames of the manual pages which **man** would display for the specified section and name combination.

Example

Retrieve the **gratm** man page:

```
super> man gratm


GRATM(8)                  BSD System Manager's Manual
GRATM(8)

NAME

     gratm - configure GigaRouter ATM cards
                       •
                       •
                       •
```

mem
(CLI)

mem displays how much system RAM is installed on a GRF control board or on an RMS node system's router management board.

```
    super> mem
    System memory 256 Mbytes
```

Permission level: user

Syntax

```
mem ?
```

```
mem
```

mountf
(CLI+shell)

The command mounts a flash device using locks and counts so that multiple processes can take advantage of the mounted device. **mountf** verifies (**fsck**) the flash device before doing the mount. By default, **mountf** mounts a flash device on `/flash`. **mountf** normally mounts the flash device as read-only at the mount point, `/flash.`

Permission level: system

Syntax

```
mountf [-d device_descriptor]  [-P]  [-S]  [-A]  [-B]  [-m mount_point]
[-w]
```

Options

The **mountf** command supports the following options:

`-P`

Primary internal flash (GRF has only a primary flash).

`-S`

Secondary internal flash (not available).

`-A`

External PCMCIA slot A.

`-B`

External PCMCIA slot B.

`-d` *`device_descriptor`*

Use the specified device descriptor, this option enables another device to be acted upon.

`-w`

Mount the flash device as writeable, the default is to mount the device as read-only.

`-m` *`mount_point`*

Loads the device at the specified mount point.

netstat (CLI+shell)

The **netstat** command displays interface routing, protocol, and connection statistics. Information about ATMP and bridging interfaces are returned in some **netstat** displays, refer to the specific protocol chapter in the GRF *Configuration and Management* manual for examples.

See also: **traceroute**, **grrt**

Permission level: system

Syntax

```
netstat [-Aan] [-f address_family] [-M core] [-N system]
netstat [-dgimnrs] [-f address_family] [-M core] [-N system]
netstat [-dn] [-I interface] [-M core] [-N system] [-w wait]
netstat [-M core] [-N system] [-p protocol]
```

Options

`-A`

With the default display, show the address of any protocol control blocks associated with sockets; used for debugging.

`-a`

With the default display, 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.

`-f` *`address_family`*

Limit statistics or address control block reports to those of the specified address family. The following address families are recognized: **inet**, for AF_INET, **ns**, for AF_NS, **iso**, for AF_ISO, **unix**, for AF_UNIX, and **grit**, for AF_GRIT.

`-g`

Show information related to multicast (group address) routing. By default, show the IP Multicast virtual-interface and routing tables. If the **-s** option is also present, show multicast routing statistics.

`-I` *`interface`*

Show information about the specified interface; used with a wait interval as described below.

`-i`

Show the state of interfaces which have been auto-configured (interfaces statically configured into a system, but not located at boot time are not shown). If the **-s** option is present more information is specified, but only for a single interface. If the **-a** option is also present, multicast addresses currently in use are shown for each Ethernet interface and for each IP interface address.

`-M`

Extract values associated with the name list from the specified core instead of the default `/dev/kmem`.

`-m`

Show statistics recorded by the memory management routines.

`-N`

Extract the name list from the specified system instead of the default `/bsd`.

`-n`

Show network addresses as numbers (normally **netstat** interprets addresses and attempts to display them symbolically).

`-p` *`protocol`*

Show statistics about protocol, which is either a well-known name for a protocol or an alias for it. Some protocol names and aliases are listed in the file `/etc/protocols`.

`-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.

Examples

These **netstat** commands are useful in the GRF environment, examples are in the GRF *Configuration and Management* manual:

– **netstat -r -s** prints routing statistics

– **netstat -i -n** shows all configured interfaces

– **netstat -a -n** prints a list of all active connections

– **netstat -r -n** prints the current table of installed routes

– **netstat -s** prints comprehensive statistics for protocols, IP, ICMP, TCP, UDP...

Two flags displayed by **netstat -rn** that support ATMP; `s` (sacred) and `a` (not advertised).The `s` flag indicates a route owned by the kernel. The user or **gated** are not allowed to delete it. The `a` flag is a kernel-managed route that GateD does not advertise. GateD does not delete or advertise a route that is flagged `sa`. The **netstat** man pages include information about these flags. Here is an example of output showing these flags:

```
Destination        Gateway            Flags     Refs     Use  Interface
15.15.3.1           15.15.3.1          UHs         0        0  atmp0
15.15.4.1           15.15.4.1          UHs         0        0  atmp0
15.15.5.1           15.15.5.1          UHs         0        0  atmp0
127                 127.0.0.1          UR          0        0  lo0
127.0.0.1           127.0.0.1          UH          0        0  lo0
192.132.27.33       192.132.27.33      UHIsa       0        0  ge070
192.132.27.255      192.132.27.0       UBsa        0        0  ge070
```

new (CLI)

The **new** command is used to create a new instance of a profile in local memory. That new instance is not permanent until the profile is written. The profile instance can be a main-level profile or a member of a profile list in a list field.

Permission level: system

Syntax

```
new profile-type [profile-index]
```

Example

Create a new main-level User profile for Fred, notice that the default profile is read:

```
super> new user
USER/default read

super> new user bob
USER/fred bob
```

Create a new member of the ports profile list, port 8:

```
super> read card 1
CARD/1 read
super> new ports
ports/0 created

super> new port 8
ports/8 created
super> write
ports/8 written
```

You cannot create a member of a profile list that already exists:

```
super> new port 8
error: profile already exists
```

If you try to create a main-level profile instance that already exists, it will just read the existing profile:

```
super> new user admin
USER/admin read
super>
```

ping
(CLI+shell)

The **ping** tool generates and receives ICMP/IP echo request and reply messages from a host or gateway. It is used to test connectivity to a specific interface or host.

When **ping** is directed from the kernel out to a system external to the GRF, the command behaves in the standard manner. When **ping** is directed from the kernel to one of the router's interface addresses, the echo request is sent to the appropriate media card and the status of the network interconnection is checked.

When **ping** is directed from an external system to any GRF address, the echo request is sent to the appropriate media card and the status of the network interconnection is checked.

Do not perform a flood ping from the operating system to a media card interface. This causes excess traffic on the internal communications bus that can degrade the reliability of the system.

Permission level: system.

Syntax

```
ping host [-dfLnqRrv] [-b sockbuf] [-c count] [-g groupsize] [-I wait]
        [-i interface] [-l preload] [-p pattern] [-s packetsize] [-t
ttl]
```

where *host* is the network host address.

Options

`-b` *sockbuf*

Set the kernel socket buffer sizes for both send and receive to size bytes. Use this option if an attempt to send large pings with the **-s** option results in error messages like ``sendto: Message too long."

`-c` *count*

After sending ***count*** echo request packets, **ping** waits a short time—twice the longest response received so far—for outstanding responses, or until all responses are received, whichever is quicker.

`-d`

Set the SO_DEBUG option on the socket being used.

`-f`

Flood ping, outputs packets as fast as they come back or one hundred times per second, whichever is more.

Note: Do not perform a flood ping from the operating system to a media card interface. This causes excess traffic on the internal communications bus that can degrade the reliability of the system.

For every echo request sent, a period "." is printed, while for every echo reply received, a backspace is printed. This provides a rapid display of how many packets are being dropped. Only the super-user may use this option as it is very hard on a network and should be used with caution.

`-g` *`groupsize`*

Send ***groupsize*** number of packets at a time, the default is to send one echo request each interval.

`-I` *`wait`*

Wait ***wait*** seconds between sending each packet or group of packets. The default is to wait one second between each packet or group of packets.

Note: This option is incompatible with the **-f** option.

`-i` *`interface`*

Specify the outgoing ***interface*** to use for multicast packets.

`-L`

Turn off loopback of multicast packets. Normally, if there are members in the host group on the outgoing interface, a copy of the multicast packets will be delivered to the local machine.

`-l` *`preload`*

If ***preload*** is specified, **ping** sends that many packets as fast as possible before falling into its normal mode of behavior.

`-n`

Numeric output only, no attempt is made to lookup symbolic names for host addresses.

`-p` *`pattern`*

You may specify up to 16 “pad” bytes to fill out the packet you send, useful for diagnosing data-dependent problems in a network. For example, “**-p ff**” will cause the sent packet to be filled with all ones.

`-q`

Quiet output, nothing is displayed except the summary lines at startup time and when finished.

`-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. This option can be used to ping a local host through an interface that has no route through it.

`-s` *`packetsize`*

Specifies the number of data bytes to be sent, the default is 56. This translates into 64 ICMP data bytes when combined with the 8 bytes of ICMP header data.

`-t` *`ttl`*

Specify the IP time to live ***ttl*** for multicast packets, the default time to live for multicast is one hop.

`-v`

Verbose output, received ICMP packets other than ECHO_RESPONSE are listed.

pinglog
(shell)

The **pinglog** command sends an SNMP trap when a GRF interface or device goes down.

pinglog reads the ping logging file and when a device down message is detected, the message is converted to an SNMP trap and sent to the gateways specified in the SNMP configuration file. The traps generated have the enterprise set to `netstar` (1080) and the general trap-type set to `SNMP_TRAP_ENTERPRISESPECIFIC`. The specific trap-type is set to `grInterfaceDown` (`netstar` enterprise trap 18).

Syntax

```
pinglog [-x instance] [-g filename] [-i timing interval]
start_pinglog [-x instance]
stop_pinglog [-x instance]
```

Options

`-x instance`
Identifies the instance number of the process since this module supports multiple instantiations.

`-g filename`
Names a gateway configurations file which defines the specific trap to be sent. The program allows for on-the-fly change of the gateway configuration file.

`-i timing interval`
The timing interval in seconds on how often to check for pinglog and gateway configuration file updates. The default time is set to five minutes.

Refer to the **pinglog** man page for more information. The “Management Commands and Tools” chapter in the *GRF Configuration and Management* manual has examples and usage information.

See also **threshpoll**.

port
(grrmb)

This **grrmb** command sets a specified slot as the default slot to be acted upon by **grrmb** commands.

When set, the default slot number remains in effect until changed by the **port** command. If a slot number is specified in any command, the default setting is overridden.

When the **grrmb** interface first starts, the default slot is the control board (or router manager board), 66. After that, the interface retains the last default slot set, and opens the next session with that slot set as default.

Syntax

```
port slot number
```

Example

To set slot 1 as the default slot, enter:

```
GR 66> port 1
The current port is 1.
GR 01>
```

The prompt is changed to contain the newly specified slot number.

power
(grrmb)

This **grrmb** command displays the on/off status of GRF 1600 power supplies.

Syntax

```
power
```

Example

```
GR 15> power
power {1|2} {on|off}
GR 15>
```

pwd
(CLI)

Shows the current user location (context) in the profile tree.

The **pwd** command determines the user's current location in the profile tree. The output is similar to a path in a file system. Each level in the tree is separated by forward slashes (/). If a profile is indexed, that is, if it is a member of a list, the index will follow the profile name. **whereami** is an alias for **pwd**.

Permission level: user.

Syntax

```
pwd
```

Example

whereami and **pwd** are used interchangeably:

```
super> read card 2
CARD/2 read
super> whereami
CARD 2

super>cd ports 1
port_num = 1
hippi = { 1 32 0 999999 4 3 5 300 0 10 10 03:00:0f:c0 disabled 0 0 +
fddi = { single }
sonet = { "" "" 1 sonet internal-oscillator 0 }
hssi = { False 16-bit }
ether = { autonegotiate disabled }

super>pwd
CARD 2/ports 1
super>
```

pwrfaild
(shell)

This command supports an optional uninterruptible power source (UPS) to which the RMS node can be connected. The software is designed to work with a specific cable that connects the RMS node to a specific APC UPS model.

When the UPS signals that main power has been lost**, pwrfaild** shuts the system software down gracefully. The RMS node must be connected to the UPS via an APC SU700RM Control Cable (part number CABLE0053).

Warning: A standard RS-232 cable cannot be used to connect the RMS node to the UPS.

This type of cable will not be able to read power state signals from the UPS and, further, such a cable could damage serial ports on both the UPS and the RMS node.

pwrfaild has been tested with the APC Smart-UPS 700 Uninterruptible Power Source. Other APC models with serial port signalling identical to this model should work with this software.

If you already have another type of UPS, compare the serial port wiring information on the next page to decide if that UPS would be compatible with this software.

To configure **pwrfaild** so that it is started when the RMS boots, uncomment or add this code at the end of `/etc/grstart`:

```
if [-x /usr/nbin/pwrfaild ] ; then
     /usr/nbin/pwrfaild
fi
```

Default tty port

The default tty port used by **pwrfaild** is `/dev/tty01`. This port is labelled “COM 2” and is on the back of the RMS node. The RS-232 (`/dev/tty00`) port on the front of the RMS node is reserved for the system console. This port can be the control connection to the UPS.

Older RMS systems

A previous generation of RMS systems are based on PCs that have only a single serial port (`/dev/tty00`). To invoke **pwrfaild** with this type of system, modify `/etc/grstart` with the `-f /dev/tty00` argument:

```
if [ -x /usr/nbin/pwrfaild ] ; then
        /usr/nbin/pwrfaild -f /dev/tty00
fi
```

Customers upgrading from earlier systems who use **pwrfaild** must take one of two actions depending on their type of RMS hardware:

Customers with Compaq Presario RMS systems must change `/etc/rc.local`, and use the **-f** option to specify that **pwrfaild** should continue to use serial port tty00, as follows:

```
       /usr/nbin/pwrfaild -f /dev/tty00
```

Syntax

```
pwrfaild [-D] [-d] [-f /dev/ttyXX] [-t seconds]
```

Options

`-d`

Turns on debugging.

`-D`

Prevents **pwrfaild** from detaching from its controlling tty (so it can be run under a debugger).

`-f /dev/ttyXX`

Specifies the ***XX*** tty device to use for the connection to the UPS
(the default tty used by **pwrfaild** is `/dev/tty01`).

`-t seconds`

Sets the length of time in ***seconds*** that the RMS can draw power from the UPS before the RMS begins the shutdown sequence.

The default for this parameter is 30 seconds. If main power is lost, but not restored after the number of seconds specified in this parameter, the RMS will shut down.

If main power is restored within this time, the RMS does not shut down.
The shutdown timer is reset, so that the next time power fails, the full value of this parameter must elapse before the RMS shutdown sequence commences.

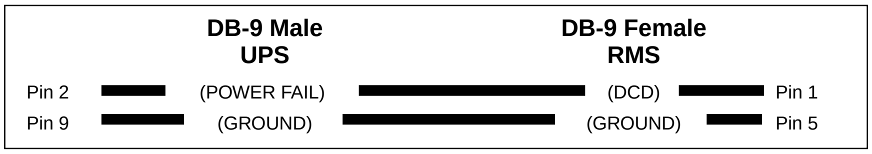
Cable signalling specifications

The following DB-9 cable design is used to connect an APC Back-UPS 700 to the GRF RMS system. Customers should obtain this cable directly from Lucent to ensure proper wiring and to avoid damage to serial ports on either the RMS or the APC UPS.

This wiring diagram is provided for informational purposes only.

– Pin 2 on the UPS (normally low) should be wired to DCD (CAR) on the RMS serial port (pin 1).
– Pin 9 on the UPS should be connected to ground on the RMS (pin 5).
– The rest of the wires, if any, should be cut. Only 24-gauge solid copper wire should be used.

The resulting cable should look like this diagram:



quit
(CLI)

Terminates current CLI session. If the session originated from a remote device, an associated connection is terminated (telnet or modem connection). If the session is on a local console and system-wide authentication is in use, the login prompt is issued.

Permission level: user

Syntax

```
quit
```

read
(CLI)

Before you are able to look at and/or change any data, you must first 'read' the profile in which the data is stored into local memory.

You can only work with one profile at a time. Once you read another profile, the profile that was previously in local memory is now replaced by the new profile. Once the profile is in local memory, you may look at and/or change the data in the profile. If you do make a change, you need to save that change to make it permanent. Some profiles are read-only. This is noted in the response to the read command. Once a profile is read, you will receive a response indicating that the read was successful.

Permission level: user

Syntax

read *profile-type* [*profile-index*]

Example

Read a one-of-a-kind profile:

```
super> read system
SYSTEM read
```

Read a many-of-a-kind profile:

```
super> read user default
USER/default read
```

route
(CLI+shell)

Adds or deletes static routes manually if dynamic routing is not running.

Permission level: system

Syntax

```
route [ -nqv ] command [ [ modifiers]  args ]
```

`command`

The **route** utility provides six commands:

- `add`, add a route.
- `flush`, remove all routes.
The **flush** command has the syntax: `route [-n] flush [family]`
If the **flush** command is specified, **route** will ``flush" the routing tables of all gateway entries. When the address family is specified by any of the -osi, -xns, or -inet modifiers, only routes having destinations with addresses in the delineated family will be deleted.
- `delete`, delete a specific route.
- `change`, change aspects of a route (such as its gateway).
- `get`, lookup and display the route for a destination.
- `monitor`, continuously report any changes to the routing information base, routing lookup misses, or suspected network partitionings.
The **monitor** command has the syntax: `route [-n] monitor`

The other commands have the following syntax:

```
route [-n] command [-net | -host] destination gateway
```

where ***destination*** is the destination host or network, ***gateway*** is the next-hop intermediary via which packets should be routed. Routes to a particular host may be distinguished from those to a network by interpreting the Internet address specified as the destination argument. The optional modifiers **-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, 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.

Options

`-n`

Bypasses attempts to print host and network names symbolically when reporting actions.

`-v`

Enable verbose mode, print additional details.

`-q`

Suppress all output.

save
(CLI)

The **save** command saves the current profile configuration in a script form to permanent storage. This file can then be loaded at a later date to restore the previous configuration.

Permission level: update

Syntax

Syntax to save the configuration to a file:

```
save [ -a ] [ -m ] filename [ profile-type [ profile-index ] ]
```

Syntax to write the configuration only to the screen, not to a file:

```
save  [-a]  [-m ]  console  [ profile-type [ profile-index ] ]
```

If a ***profile-type*** is not specified, all savable profiles will be saved. If a ***profile-type*** is specified, but a ***profile-index*** is not specified (and it is a multiple-instance profile), all profiles of that type will be saved.

If the **`-a`** option is specified, all fields will be explicitly saved. Otherwise, only those fields whose contents differ from the default values will be saved.

If **`-m`** is specified, all fields will be saved by their field numbers rather than their field names.

If the current user does not have password accessibility, a message will appear warning the user not to save any profiles that contain passwords. This is done because all passwords will be written as strings of stars.

All files will be saved in the `/etc/prof` directory. If the specified profile already exists, a message will appear warning the user that this file already exists and asking the user if s/he wants to overwrite this file.

Examples

Save all savable profiles to a specified file

```
super> save all.conf
super>
```

Save all user profiles to a specified file:

```
super> save user.conf user
super>
```

Save the User admin profile to a specified file:

```
super> save admin.conf user admin
super>
```

Saving to a file that already exists, resulting warning:

```
super> save all.conf card

WARNING: all.conf already exists. If you choose to save to this
file, all configuration information that now exists in all.conf
will be overwritten. Continue? [y/n] n
```

```
save aborted
super>
```

Save a user profile when you do not have password access:

```
super> save default.conf user default

WARNING: the current user has insufficient rights to view password
fields.  A configuration saved under this circumstance should not
be used to restore profiles containing passwords.

Save anyway? [y/n] n
super>
```

set
(CLI)

The **set** command is used to modify fields in the last profile read. Modifications do not take effect until the profile is written using the CLI **write** command. The **?** function returns help text about a field.

Permission level: system

Syntax

The first format is used to change a field, the ***field-name*** must match a name in the last profile read. When changing a field, the ***field-value*** is everything between the white space following the = and the end of the line.

```
set field-name = field-value
```

This usage returns help on the type of values to which the field can be set.

```
set field-name ?
```

Example

Read the User profile to create a new copy of the default:

```
super> read user default
USER/default read
super> list
```

Now set the fields to be changed:

```
super> set name = operator
super> set password = mypasswd
super> set active-enabled = yes
super> set allow-system = yes
super> set allow-update = yes
super> set prompt = operator
```

Check the new values?

```
super> list
name* = operator
password = mypasswd
auth-method = { PASSWD { "" 1645 udp "" } { 5500 udp 5510 tcp +
active-enabled = yes
allow-system = yes
allow-update = yes
allow-password = no
allow-debug = no
prompt = operator=>
log-display-level = none
super>
```

The ***field-value*** is everything between the white space following the = and the end of the line. In this example, `set prompt` is the ***field-name***, `operator` is the ***field-value***:

```
set prompt = operator
```

setver
(CLI+shell)

The **setver** command specifies which version of a software release will be loaded and run after the next system boot. **setver** mounts the flash device specified and verifies that the software version on the flash device has the correct pieces needed to boot.

During verification, **setver** performs preliminary checks on the release tar files and the boot, bsd, and startup scripts. Then it verifies that the memory file system and the configuration files exist. If verification is successful, **setver** configures the specified release and version to be loaded during the next boot. It then unmounts the flash device.

Permission level: system

See also: **getver**, **grfins**

Syntax

```
setver  revision,version   [-P | -S | -A | -B]  [-d device_descriptor
]  [-p path ]
```

where ***revision,version***

Specifies the software release and version against which to perform the command, for example `1.4.10,default` or `1.4.x`,`atmtestB`.

The currently-running version is assigned by **flashcmd** to be `default`.

Options

The **setver** command supports the following options:

`-P`

Specifies the internal flash device: `/dev/wd0a.`

`-S`

Specifies a secondary internal flash device: `/dev/wd1a.`

`-A`

Specifies the PCMCIA device in slot A: `/dev/wd2a.`

`-B`

Specifies the PCMCIA device in slot B: `/dev/wd3a.`

`-d` *`device_descriptor`*

An alternative method of specifying a flash device.

`-p` *`path`*

Do not mount device, just use specified path, used when flash device is already mounted.

This option is in the form ***revision,version***. For example, `1.4.x,default` or `1.4.x,atmtestB`. The currently-running version is assigned by **flashcmd** to be `default`.

sh
(CLI)

The **sh** command creates a UNIX shell in the CLI. Always use **exit** or **quit** to return to the CLI prompt.

You can create one UNIX shell per CLI session. Multiple shells and sessions do not nest within each other. You see the GRF copyright and revision notice whenever you invoke the shell.

Permission level: user

Syntax

```
sh
```

Example

```
super> sh

Copyright 1992, 1993, 1994, 1995, 1996 Berkeley Software Design,
Inc.
Copyright (c) 1980, 1983, 1986, 1988, 1990, 1991, 1993, 1994
        The Regents of the University of California.  All rights
reserved.

Ascend Embedded/OS GR TA1.4.20R.8 Kernel #0 (nit): Wed Aug 23
02:23:18 CDT 1999

                Ascend Embedded/OS 1.4.20R.8

Copyright 1992,1993,1994,1995,1996,1997,1998,1999 Ascend Communi-
cations.

IMPORTANT:  By use of this software you become subject to the terms
and conditions of the license agreement on file /etc/license and
any other license agreements previously provided to you by Ascend
Communications.
#
```

shutdown
(CLI+shell)

Halts, reboots, shuts down the operating system. When the GRF is shut down, media card operations stop abruptly. Connections are broken and enroute packets are lost.

Refer to the **shutdown** man page for more information.

Permission level: system

Syntax

```
shutdown [ - ] [ -fhikrn ] time
```

Options

`-f`

The **shutdown** command arranges, in the manner of **fastboot**, for the file systems not to be checked on reboot.

`-h`

The system is halted at the specified ***time*** when **shutdown** executes **halt.**

`-i`

The **-i** option is only available on the GRF system.

This option is an override flag to ignore file system check. The GRF system has a built-in safety feature where it prevents the administrator from accidentally rebooting the system without saving the configuration files in `/etc`. It determines this by running the **grwrite** command.

If the **grwrite** command exits with a non-zero exit status, then a warning message is printed and the shutdown operation is aborted. The **-i** option is used to override this safety feature. This option is not applicable on a non GRF system.

`-r`

The command will reboot at the specified ***time***.

`time`

The ***time*** at which **shutdown** will bring the system down, use **now** to indicate an immediate action.

Example

Log in as root and prepare to power off the GRF system:

```
super> shutdown -h now
```

Log in as root and prepare to reboot the GRF system:

```
# shutdown -r now
```

temp
(grrmb)

The **temp** command reads temperature sensors mounted on the control board and returns the chassis's internal temperature.

Syntax

```
temp
```

Example

To print its contents and then clear the trace buffer, enter:

```
GR 66> temp
GR 66>  temp

                      LT     HT    TEMP
                   ----------------------
Measured Junction     53 C   58 C   39 C
Calc. Ext. Ambient    47 C   52 C   33 C

GR 66>
```

These are the reported fields:

`LT`

Lower temperature threshold (start of software shutdown begins).

`HT`

High temperature threshold (hardware shutdown - immediate).

`TEMP`

Current operating temperature.

`Measured Junction`

Temperature on board surface.

`Calculated External Ambient`

Estimated temperature of air surrounding the chassis.

threshpoll
(shell)

The **threshpoll** command sends an SNMP trap if a threshold condition is detected for a media card operation.

Currently, thresholding is only supported for the following objects:

- ifInOctets
- ifOutOctets
- ifInUcastPkts
- ifOutUcastPkts
- ifInErrors
- ifOutErrors
- ifInDiscards
- ifOutDiscards

For all of these objects, thresholding is done on the delta of the current value and the previous value.

A poll group file determines the object identifiers to poll and the trap-type to send when the threshold condition is met. A thresholding file determines the thresholding operation and the value associated with each poll's object identifiers. A device setup file which determines the list of devices to poll and the poll groups associated with each device. In addition, this file also indicates which poll group to threshold on a per device, per poll group basis.

The specified values are thresholded against the values in the thresholding configuration file. If the thresholding condition is met, an SNMP trap is output to the gateway specified in the SNMP configuration file (`/etc/snmpd.conf`). If the log file environment variable is set, then values will be logged to a `/var/log` file.

Syntax

```
threshpoll -x instance   [ -i timing_interval  ]
start_threshpoll -x instance
stop_threshpoll -x instance
```

Option

`-x` *`instance`*

Identifies the ***instance*** number of the process since this module supports multiple instantiations.

`-i` *`timing_interval`*

The ***timing_interval*** number of seconds on how often to poll the defined interfaces. The default time is set to 5 minutes.

Refer to the **threshpoll** man page for more information. The “Management Commands and Tools” chapter in the *GRF Configuration and Management* manual has examples and usage information.

See also: **pinglog**

traceroute
(CLI+shell)

Prints the packet route to a specified destination host/network.

See also: **netstat**, **grrt**

Permission level: system

Syntax

```
traceroute host [-m max_ttl] [-n] [-p port] [-q nqueries] [-r] [-s
src_addr]
[-t tos] [-w waittime]  [packetsize]
```

where ***host*** is a destination host name or IP number.

Options

`-m max_ttl`

Set the max time-to-live (max number of hops) used in outgoing probe packets. The default is 30 hops (the same default used for TCP connections).

`-n`

Print hop addresses numerically rather than symbolically and numerically. Saves a nameserver address-to-name lookup for each gateway found on the path.

`-p port`

Set the base UDP port number used in probes (default is 33434).

`-q nqueries`

Set the number of probes per ***ttl*** to ***nqueries*** (default is three probes).

`-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. This option can be used to ping a local host through an interface that has no route through it (e.g., after the interface was dropped by routed).

`-s src_addr`

Use the following IP address (which must be given as an IP number, not a hostname) as the source address in outgoing probe packets.

On hosts with more than one IP address, this option can be used to force the source address to be something other than the IP address of the interface the probe packet is sent on. If the IP address is not one of this machine's interface addresses, an error is returned and nothing is sent.

`-t tos`

Set the type-of-service in probe packets to the ***tos*** value, the default is zero.

The value must be a decimal integer in the range 0 to 255. This option can be used to see if different types-of-service result in different paths.

`-v`

Verbose output. Received ICMP packets other than TIME_EXCEDED and UNREACHABLEs are listed.

`-w` *`waittime`*

Set the ***waittime*** (in seconds) to wait for a response to a probe, default is 3 seconds.

`packetsize`

Specify the packet size in bytes after the destination host name ***host.*** The default probe datagram length is 38 bytes.

umountf
(CLI+shell)

The **umountf** command unmounts a flash device that was previously mounted using the **mountf** command.

Permission level: system

Syntax

```
umountf [-d device_descriptor ] [-e] [-i] [-A] [-B]
```

Options

The **umountf** command supports the following options:

`-P`

Primary internal flash (GRF has only a primary flash).

`-S`

Secondary internal flash (not available).

`-A`

External PCMCIA slot A.

`-B`

External PCMCIA slot B.

`-d` *`device_descriptor`*

Use the specified device descriptor, this option enables an alternate device to be acted upon.

vpurge (CLI+shell)

The **vpurge** command removes a configuration from a flash device. **vpurge** removes the files **snapshot** puts on a flash drive. This includes the main `TAR.gz`, `root.dd.gz`, and the `/etc` release directory.

Caution: Use the **--force** option with care, it directs the command to remove files that are in use or set to run at the next boot.

Permission level: system

Syntax

```
vpurge --release=release,version  options
```

Options

`--release`

Sets the ***release,version*** in the form *x.x.x,<version>*, for example, `1.4.x,default.`

`--force`

Forces **vpurge** to remove the files even when they are in use or are set to run at next boot.

Warning messages are returned when **--force** is specified:

```
"fatal: is the next boot release"
"fatal: is the running release"
```

`--target`

Specifies the target device. The target can be one of the following:

P (internal primary flash)

S (internal secondary flash)

A (PCMCIA slot A)

B (PCMCIA slot B)

If this flag is not set, then it assumes the target device to use is the device it loaded from, `/etc/load_device`.

`--verbose`

Prints out more information about which actions are taking place.

`--help`

Shows a short help message.

whatami
(CLI)

Tells you if the system is RMS node-based (`IRMS`) or has the new control board (`CB`). Permission level: system

Syntax

```
whatami
```

Example

```
super> whatami
CB
super> whatami
IRMS
```

whereami
(CLI)

Shows the current user location (context) in the profile tree.

The **whereami** command determines the user's current location in the profile tree. The output is similar to a path in a file system. Each level in the tree is separated by forward slashes (/). If a profile is indexed, that is, if it is a member of a list, the index will follow the profile name. **pwd** is an alias for **whereami**.

Permission level: user.

Syntax

```
whereami
```

Example

whereami and **pwd** are used interchangeably:

```
super> read card 2

CARD/2 read
super> whereami
CARD 2

super> cd ports 1
port_num = 1
hippi = { 1 32 0 999999 4 3 5 300 0 10 10 03:00:0f:c0 disabled 0 0 +
fddi = { single }
sonet = { "" "" 1 sonet internal-oscillator 0 }
hssi = { False 16-bit }
ether = { autonegotiate disabled }

super> pwd
CARD 2/ports 1
super>
```

whoami
(CLI)

The **whoami** command returns the user profile name associated with a session no matter which profile the user is currently in.

Permission level: user

Syntax

```
whoami
```

Example

```
super> read card 3
CARD/3 read
card-num* = 3
media-type = fddi-v2
debug-level = 0
hssi-frame-protocol = Frame-Relay
sonet-frame-protocol = PPP
ether-verbose = 0
ports = < { 0{dual off}{ "" "" 1 sonet internal-oscillator 0}{0
16-bit }+
load = { 0 < > 1 0 0 }
dump = { 0 < > 0 }
config = { 0 1 1 4 0 0 }
icmp-throttling = { 10 10 2147483647 10 10 10 }

super> list . du
config = 0
hw-table = < >
config-spontaneous = 0

super> who
bob
super>
```

write
(CLI)

The **write** command validates the profile and stores a local profile into memory. **write** is used to apply changes you have made to a field value and is part of the process to create a new instance of a User or Card profile.

A profile-type or profile-index is not required for the **write** command as the current profile is always the one written.

Note: The CLI **write** command differs from the **grwrite** command. **grwrite** saves the `/etc` directory in the operating system. **write** saves the profile data to the `/etc/profs` directory.

Permission level: update

Syntax

```
write
```

Example

Use **write** to store a new version of a User profile:

```
super> dir user
92  01/31/97 10:16:08  default
103  01/31/97 10:16:09  admin
106  01/31/97 10:16:10  super
99  02/03/97 15:27:00  bob

super> read user bob
USER/bob read
super> set name = bob2
super> write
USER/bob2 written

super> dir user
 92  01/31/97 10:16:08  default
103  01/31/97 10:16:09  admin
106  01/31/97 10:16:10  super
 99  02/03/97 15:27:00  bob
100  02/03/97 16:00:00  bob2
```

Configuration file templates

2

Chapter 2 contains copies of the `/etc/*.conf` configuration files for user reference. These are template versions, and have not been edited.

Many configuration files have examples and extensive notes about the parameters and options available in each. It is helpful to read through the contents of the files before you begin making entries. These files are accessed in the UNIX shell, and require a UNIX editor such as **vi** to write and save.

Note: The GateD configuration file is in the *GRF GateD Manual*.

These files are included here:

Archiving and saving configuration files

On systems using an RMS node, the **grc** archiving utility saves copies of each of the `/etc` files. Use it to create backups and manage configuration sets.

On the GRF 400, the `/etc` directory resides and is maintained in system RAM. To save `/etc` files between system boots, use the **grwrite** command to save the configuration files to the internal flash disk. **grsnapshot** can be used to archive and name special configurations. **grsite** installs and saves special configuration files for site experimentation and debugging.

GRF configuration files and their uses

This is an alphabetized list of `/etc` GRF configuration files and their application:

`aitmd.conf`	- defines ATMP home network and foreign agent parameters
`bridged.conf`	- defines bridge groups and other bridging parameters
`filterd.conf`	- defines system filtering services
`gated.conf`	- enables dynamic routing functions (see the *GRF GateD Manual* for this file)
`grarp.conf`	- maps IP addresses to physical hardware addresses
`gratm.conf`	- configures ATM media cards, including PVCs, logical interfaces for SVCs, traffic shaping, QoS, peak and sustainable cell rates, UNI signalling, SONET and SDH mode, ATMP links
`grfr.conf`	- configures Frame Relay on HSSI and SONET cards
`grifconfig.conf`	- identifies each logical interface on a media card
`grlamap.conf`	- maps HIPPI logical addresses to media cards
`grppp.conf`	- assigns Point to Point Protocol interfaces on HSSI and SONET cards
`grroute.conf`	- sets static routes
`snmpd.conf`	- enables SNMP components
`syslog.conf`	- configures remote logging of log files via **syslogd**

/etc/aitmd.conf

This configuration file contains the foreign agent, home network, and interface parameters required for the Ascend Tunnel Management Protocol, ATMP.

Please refer to the “ATMP Configuration Guide” chapter in the *GRF Configuration and Management* manual for a description of ATMP as implemented on the GRF. Note that this configuration file has a man page available, **man aitmd.conf**.

```
#  file:  /etc/aitmd.conf
#
#  This is a sample configuration file for the ATMP home agent server on
#  the GRF. It is read by the daemon "aitmd".  See the man page for aitmd
#  and for aitmd.conf for additional information. See the "GRF Configura-
# tion and Management" manual for complete information. By default,the
#  aitmd daemon expects to find its configuration file at /etc/aitmd.conf
#
#  This file contains three kinds of records.
#
#  First, and simplest, are the foreign agent records.  These give the IP
#  addresses of ATMP foreign agents that are permitted to initiate connections
#  to the home agents. Also, they include the password that will be used to
#  authenticate the foreign agents to us.  The foreign agent must be configured
#  to use this password.
#
#  Second is the default foreign agent record. Only one default foreign agent
#  record may be specified. A default foreign agent record allows many foreign
#  agents to be declared using a single record. This is done by using a 'wild-
#  card IP address' that matches multiple ip addresses. When a tunnel request
#  is initiated, if the IP address from the request does not match any foreign
#  agent record, then the default foriegn agent record is checked for a match.
#  In this case, the foreign agent must be configured to use the password in the
#  default foreign agent record.
#
#  Third is the home network record. Each home network is represented with a
#  different home agent IP address on the GRF.  For each home network you must
#  configure a name, which is what the foreign agent uses to identify the home
#  network. Each home network maps to a different local IP address on the GRF.
#  This is the home agent address.  This is the address to which the
#  foreign agents send to for tunnel negotiation and encapsulated
#  traffic.  Every home network must have a unique IP address.
#
#  Presently, connections from the GRF back to the home network are only
#  supported on the HSSI and ATM-OC3 cards.  The home network record
#  contains an interface record which describes the logical interface
#  that is connected back the named home network.  The interface record
#  lists the logical interface name, and optionally, the IP address and
#  netmask size for this interface.  (Note that this address is on the
#  Virtual Private Network, not the public network, and must not be
#  specified in grifconfig.conf.)
#
#  The hssi/frame relay logical interfaces described in the aitmd
#  configuration file must each have a single ATMP virtual circuit
#  configured with the normal frame relay tools.  See the man pages for
#  grfr, grfr.conf, and fred for more information on frame relay circuit
#  configuration.
#
#  ATM logical interfaces described in the aitmd configuration file must
#  each have a single ATMP virtual circuit defined in gratm.conf.  See
#  the man pages for gratm and gratm.conf for more information.
#
#  Interfaces
#  ----------
#
#  Each logical interface used as an ATMP home network link must be defined
#  in grifconfig.conf.  These interfaces will be used in /etc/grfr.conf for
#  the pvcatmp statement, and in /etc/gratm.conf with the PVC where
#  proto=vc_atmp.  The interfaces in grifconfig.conf for ATMP home networks
#  should resemble the following.  Note that the first three options are
#  dashes.  Do NOT configure addresses for ATMP interfaces in
#  /etc/grifconfig.conf.
#
#  gs031   -  -  -  up     # sample interface for HSSI pvcatmp circuit
#  ga0288  -  -  -  up     # sample interface for ATM vc_atmp circuit
```

```
#
#
#  Syntax
#  ------
#  Comments begin with the pound sign (#) and continue until the end of
#  the line.  Most fields are represented with a keyword followed by
#  its value.  The value  must be followed with a semi-colon.  Some fields
#  are grouped into records.  Records begin with a keyword followed
#  by a list enclosed withing curly braces {}.
#
#  Addresses may be given in dotted decimal notation, such 10.11.12.192,
#  or with host names.  It is recommended that IP addresses be used.
#  Names are more convenient to use and easier to remember, but if the
#  DNS name server is unreachable or down, then the aitmd server will not
#  be able to convert the name to an address and the configuration will
#  fail.  Using IP addresses directly makes the system immune to DNS
#  failures.
#
#  Numbers in this sample file are all in normal decimal notation.
#  Numeric values like the netmask size, card number, and port number may
#  also be entered in hexidecimal by using the '0x' prefix.  For example
#  0x4c would be the decimal value 76.  If you prefer octal, prefix the
#  number with a '0', like 0377 for decimal 255.
#
#  Below is a detailed syntax description for the three records,
#
#     foreign_agent {
#          addr addr_parameter;
#          password password_parameter;
#     }
#
#     default_foreign_agent {
#          [ addr wildcard_addr_parameter; ]
#          password password_parameter;
#     }
#
#     home_network {
#          name name_parameter;
#          home_agent_addr home_agent_addr_parameter;
#          interface {
#                name interface_name_parameter;
#                [ vpn_addr vpn_addr_parameter; ]
#                [ vpn_netmask_size vpn_netmask_size_parameter; ]
#                [ ripv2 { [ enabled enabled_parameter; ]
#                          [ metric metric_parameter; ]
#                  } ]
#          }
#          circuit {
#                card card_parameter;
#                port port_parameter;
#                s0 s0_paramater;
#                s1 s1_parameter;
#                [ vpn_addr vpn_addr_parameter; ]
#                [ vpn_netmask_size vpn_netmask_size_parameter; ]
#                [ ripv2 { [ enabled enabled_parameter; ]
#                          [ metric metric_parameter; ]
#                  } ]
#          }
#          [ mtu_limit mtu_limit_parameter; ]
#          [ force_fragmentation force_fragmentation_parameter; ]
#          [ bad_source_notification bad_source_notification_parameter; ]
#          [ max_tunnels max_tunnels_parameter; ]
#          [ inactivity_timeout inactivity_timeout_parameter; ]
#     }
#
#     where [] indicates an optional item. Up to two occurences of circuit
#     or interface is permitted, however both are not permitted in the same
#     home_network declaration.
#     If "interface { ... }" is present, then interface_name_parameter must
#     be valid interface name. A valid interface name is of the form gx0yz
#     where x = 'a' or 's', y is a hex digit (0..f) and z is a hex number
#     whose range is 0..ff.
#
#     If  "vpn_addr vpn_addr_parameter;" is present then "vpn_netmask_size
#     vpn_netmask_size_parameter;" must be present. vpn_addr_parameter must be
#     a Class A, B, or C IP address and vpn_netmask_size_parameter must be a
#     number in the range 1..30.
#
```

```
#     If ripv2 is present, then "enabled enabled_parameter;".
#     enabled_parameter must be either yes or no.
#
#     If 'enabled yes' is specified for ripv2, then the vpn_addr must be
#     defined and must be valid.
#
#     If  "metric metric_parameter;" is not present, then the default value
#     for metric will be 0. Otherwise, metric_parameter must be a number in
#     value for mtu_limit will be 0. Otherwise, mtu_limit_parameter must be
#     either a the range 0..15.
#
#     If  "mtu_limit mtu_limit_parameter;" is not present, then the default
#     value for mtu_limit will be 0. Otherwise, mtu_limit_parameter must be
#     either a number in the range 40..65507 or the word auto.
#
#     If  "force_fragmentation force_fragmentation_parameter ;" is not
#     present, then the default value for force_fragmentation will be no.
#     Otherwise, force_fragmentation_parameter must be either yes or no.
#
#     If  "bad_source_notification bad_source_notification_parameter;" is not
#     present, then the default value for bad_source_notification will be yes.
#     Otherwise, bad_source_notification_parameter must be either yes or no.
#
#     If  "max_tunnels max_tunnels_parameter ;" is not present, then the
#     default value for max_tunnel will be NO TUNNEL LIMIT. Otherwise,
#     max_tunnels_parameter must be a  positive number.
#
#     If  "inactivity_timeout inactivity_timeout_parameter ;" is not present,
#     then the default value for inactivity_timeout will be NO TIMEOUT.
#     Otherwise, inactivity_timeout_parameter must be a  number in the range
#     0..20. If inactivity_timeout_parameter is 0, then this will indicate NO
#     TIMEOUT.
#
#  Example
#  -------
#
#  foreign_agent {
#     addr 172.17.1.1;       # IP address of the foreign agent
#     password OurSecret117; # shared secret.
#  }
#
#  home_network  {
#     name Kansas;      # text string name.  no more than 31 characters
#
#     home_agent_addr 10.2.2.2;  # IP address of the home agent on the grf
#                                # Only one home_network may use this address
#     interface {
#        name ga0c83;            # ATM card in slot 12, port 1
#        vpn_addr 192.222.1.1;   # GRF's address on the VPN
#        vpn_netmask_size 24;    # size of VPN netmask
#     }
#  }
#
#
#  foreign_agent {
#     addr 172.20.5.5;        # IP address of another foreign agent
#     password AnotherSecret; # shared secret.
#  }
#
#
#  foreign_agent {
#     addr 172.20.5.100;    # IP address of the foreign agent
#     password TrustNoOne;  # shared secret.
#  }
#
#  default_foreign_agent {
#     addr 172.20.*.*;
#     password DefaultSecret;
#  }
#
#
#  home_network  {
#     name Iowa;        # text string name.  no more than 31 characters
#
#     home_agent_addr 10.200.7.6; # IP address of the home agent on the grf
#                                 # Only one home_network may use this address
#     interface {
#        name gs0a1;              # HSSI card in slot 10, port 0
#        vpn_addr 192.200.3.4;    # GRF's address on the VPN
#        vpn_netmask_size 26;     # size of VPN netmask
#        ripv2 {
#           enabled yes;          # Send RIPv2 multicasts on this interface
#           metric 2;             # metric for advertised routes
#        }
#     }
#
#     # The following entries are optional.
#
#     mtu_limit 1200;            # pre-fragment packets before they enter the
#                                #  tunnel. (Default is 0 which means that no
#                                #  pre-fragmentation is performed)
#     force_fragmentation yes;   # ignore the IP DF bit when pre-fragmenting,
#                                #  if necessary. (Default is no)
#     max_tunnels 1000;          # Indicates the maximum number of tunnels this
#                                #  home network will allow to be created.
#                                #  (Default is NO_LIMIT)
#     inactivity_timeout 15;  # The time in hours to wait before tearing
#                             #  down inactive tunnels. (Default is NO_TIMEOUT)
#     bad_source_notification no;
#                             # Determines whether an ATMP Error Notification
#                             #  is sent to a Foreign Agent when a packet is
#                             #  received which contains a source address which
#                             #  is not registered for the tunnel on which it
#                             #  arrived. (Default is yes)
#  }
```

/etc/bridged.conf

The `/etc/bridged.conf` configuration file is used to configure bridge groups and other bridging parameters.

```
# Configuration file for Bridge Daemon (bridged).
#
#       Note: bridged will not start if it finds an error while
#       trying to parse this file. Use the "-d" option on the
#       command line with bridged to find proximity of the offending
#       line.
#

#bridge_group bg0 {
      #
      # The main reason we need to configure this stuff is to
      # specify which ports are part of a bridge group.

      # Declare all the ports in this group on a single
      # line if you don't need to set anything special
      # for the port. This is the normal case.
      #
      # multiple lines are ok.
      #
      # eg. port gf070;
      #     port gf041 gf072;
      #     port gf080;

      #port gf090 gf091;

      #
      # If you need to set specific values for a port in this
      # bridge group, then use the structure below..
      #
      # port gf040 {
        #
        # priority : port priority allows the network manager to
        #       influence the choice of port when a bridge has
        #       two ports connected in a loop.
        #
        #       priority 5;
        #
        # state disabled : Used to explicitly disable a port.
        #
        #       state disabled;
        #
        # path_cost : This is the cost to be added to the root
        #       path cost field in a configuration message received
        #       on this port in order to determine the cost of the
        #       path to the root through this port. This value is
        #       individually settable on each port.
        #
        #       Setting this value to be large on a particular port
        #       makes the LAN reached through that port more likely
        #       to be a lead or at least low in the spanning tree.
        #       The closer a LAN is to being a leaf in the tree, the
        #       less through traffic it will be asked to carry. A
        #       LAN would be a candidate for having a large path
        #       cost if it has a lower bandwidth ot if someone wants
        #       to minimize unnecessary traffic on it. A better
        #       description is possibly link cost or port cost.
        #
        #       path_cost 5;
        #
        # Forward packets with the following destination addresses
        # through this port.
        #
        #       forward 00:a0:24:2a:50:e6 00:a0:24:2a:50:e7;
        #
        # ipx_translate_to_ethernet : Used to set up explicit
        #       translation of IPX FDDI packets when they are bridged
        #       to Ethernet.  This parameter is specified for the
        #       outbound Ethernet or ATM interface. If this parameter
        #       is NOT specified, then FDDI IPX packets will be
        #       translated to Ethernet by the usual rules:
        #         From:                 To:
        #          FDDI 802.2             Ethernet 802.2
        #          FDDI SNAP              Ethernet II
        #
        #       ipx_translate_to_ethernet ethernet_802.2;
        #       ipx_translate_to_ethernet ethernet_ii;
        #       ipx_translate_to_ethernet ethernet_snap;
        #       ipx_translate_to_ethernet ethernet_802.3_raw;
        #
        # ipx_translate_to_fddi : Used to set up explicit translation
        #       of IPX Ethernet packets when they are bridged to FDDI.
        #       This parameter is specified for the outbound FDDI or
        #       ATM interface.  If this parameter is NOT specified,
        #       then Ethernet IPX packets will be translated by the
        #       usual rules:
        #        From:                 To:
        #         Ethernet 802.2         FDDI 802.2
        #         Ethernet II            FDDI SNAP
        #         Ethernet SNAP          FDDI SNAP
        #         Ethernet 802.3 (Raw)   Invalid FDDI LLC frame
        #
        #       ipx_translate_to_fddi fddi_802.2;
        #       ipx_translate_to_fddi fddi_snap;
        #
        # };
        # port gf041 {
        #       state disabled;
        #       path_cost 6;
        #       priority 5;
        # };
        #
        # Configuration and tuning parameters that
        # govern how a bridge group functions.
        #
        # priority: This is used to create the bridge ID for
        #       this bridge. this along with the MAC address of one
        #       of the ports in the group is used to set the bridge ID.
        #       This value allows the network manager to influence the
```

```
  #       choice of root bridge and the designated bridge. It is
  #       appended as the most significant portion of a bridge ID.
  #       A lower numerical value for bridge priority makes the
  #       bridge more likely to become the root.
  #
  #       priority 128;
  #
  #
  # hello_time: Interval between the transmission of configuration
  #       BPDUs by a bridge that is attempting to become the root
  #       bridge or is root bridge.
  #
  #       It is the timer that elapses
  #       between generation of configuration messages by a bridge
  #       that assumes itself to be the root.
  #       Shortening this time makes the protocol more robust, in
  #       case the probability of loss of configuration messages
  #       is high. Lengthening the timer lowers overhead of the
  #       alogorithm (because the interval between transmission
  #       of configuration messages will be larger).
  #
  #       The recommended time is 2 sec.
  #
  #       hello_time 2 seconds;
  #
  # forward_delay: The time value advertised by this
  #      bridge for deciding the time delay that a port must
  #      spend in the listening and learning states.
  #
  #     This parameter temporarily prevents a bridge from starting
  #     to forward data packets to and from a link until news of
  #     a topology change has spread to all parts of a bridged
  #     network.This should give all links that need to be turned
  #     off in the new topology time to do so before new links are
  #     turned on.
  #
  #     Setting forward delay too small would result in temporary
  #     loops as the spannign tree algorithm converges. Setting
  #     this value too large results in longer partitions after
  #     the spanning tree reconfigures.
  #
  #      The recommended value is 15 sec.

#forward_delay 15 seconds;

#
# maximum_age: This is the time value advertised by this bridge
#     for deciding whether to discard spanning tree frames
#     based on message age.
#
 #    If the selected max_age value is too small, then occasionally,
#    the spanning tree will reconfigure unnecessarily, possibly
#    causing temporary loss of connectivity in the network. If the
 #     selected value is too large, the network will take longer then
 #     necessary to adjust to a new spanning tree after a topological
#    event such as restarting or crashing of a bridge or link.
#
```

```
        #       The recommended value is 20 sec.
        #
        #       maximum_age  20 seconds;
        #
        # route_maximum_age: This parameter determines how often routes
        #       will be aged out of the learnt route table.
        #
        #       Default : 300 seconds
        #
        #       route_maximum_age 300 seconds;
        #
        # And last, hardwiring the forwarding table with
        # specific MAC addresses to discard.
        #
        # Sink packets with the following destination
        # addresses.
        #
        #       discard 00:40:0b:0c:95:60 00:40:0b:0c:95:6a;
        #
        # Disable Spanning Tree for the group
        #
        #       spanning_tree disabled;
#} ;

#
# To create a bridge group with no ports in it, use the
# following NULL declaration:
#
#       bridge_group bg0 {
#                       ;
#       };
#

#debug_level           5; # traces all events of level NOTICE and above
```

/etc/filterd.conf

In `filterd.conf`, you name a filter and specify the filter’s rules, and assign (bind) the filter to a particular media card. The filter function is enabled after you copy the template file, `/etc/filterd.conf.template`, to `/etc/filterd.conf` and then edit and save the file.

```
# Template file to give an example of how filtering is setup.

# The first step is to define filters that you want to have.  These are
# general items that can be applied to more than one media card/interface
# or don't have to be applied at all.  If you are not using a filter,
# however, comment it out using /* filter.... */ to allow for faster
# load time.
#
#
# The following is a fairly complex filter example that denies access to
# the world but lets in some "things". A definition of each of these
# "things" is included in a comment by each rule.
#
# In GR filtering lingo, the following is a "filter":
#
/* ------
filter example_in_1 {
        implicit deny;          # if no rules matched, filter DENIES

       #
       # Make our first match rule.  In this case allow packets from
       # ports > 1023 and returning packets in established TCP connections.
       #
       # In GR filtering lingo, this is a "rule".
       permit {
               ipv4protocol tcp {
                       established;
                       port gt 1023;
               }
               ipv4protocol udp {
                       port gt 1023;
               }
       }

       #
       # Ok, now lets do something to allow people to talk to our
       # mail server.
       #
       permit {
               to 222.222.222.1 0.0.0.0;
               ipv4protocol tcp {
                       port 25;
               }
       }

       #
       # Let the consultant from somehost.somewhere.com into one of our
       # networks.
       #
       permit {
               from 221.222.222.32 0.0.0.0;
               to 222.222.222.0 0.0.0.255;
       }

}

#
# The following is a second filter, which we will stick on transmit side
# of our "internal" network. The filter prevents someone from dropping
# packets from the outside that look like they come from the inside.
#
# Note that it may make more sense to place this on the upstream side
# of all the "unsafe" interfaces instead of on the downstream side.This
# is especially true if there are more than one interface on the "safe"
# side that move traffic amongst themselves.  Putting the filter on the
# downstream side in this case causes safe traffic to take the performance
# hit of blocking traffic on the upstream side.
#

filter spoof_block {
       # By default, let everything through.  Our upstream side will catch
       # the stuff it wants to weed out.
       implicit permit;

       # our "internal" network is 222.222.222.*.
       deny {
               from 222.222.222.0 0.0.0.255;
               to 222.222.222.0 0.0.0.255;
       }
}

#
# And now, an unusual filter to show how some other things work, but
# would never actually be created.
#
filter weird_filter {
       implicit deny;

       permit {
               from 128.101.101.101 0.0.0.0;   # from this one
               from 128.101.101.102 0.0.0.0;   # OR from this one
               to 220.220.220.0 0.0.0.255;     # AND to this one
               to 220.220.220.0 0.0.0.255;     # AND to this one
               ipv4protocol icmp;              # ICMPs allowed
               ipv4protocol tcp {
                       # note that allow port 0, even though it is not
                       # a valid port.  By including it, range checking
                       # becomes more efficient in many cases.
                       port 0..3, 10..20, gt 1023;
               }
               ipv4protocol udp;               # all udp through
       }
}

#
#
# Once all filters are declared, we can declare what is called a
# “binding”. This means attaching the above filters to interfaces.
#

media fddi 11 {
        # ok, what filter?
        bind example_in_1 {
                vlif 0;                 # do the first interface
                direction in;           # inbound traffic only
                action filter;          # route/don't route
        }

        #
        # and we have another interface that we are going to pretend
        # routes to our internal net.
        bind spoof_block {
                vlif 1;                 # the other interface (DAS)
                direction out;          # outbound traffic only
                action filter;
        }
}

#
# And another card for our _other_ "internal" network.  We attach the
# weird filter to it.
media atm 3 {
        bind weird_filter {
                vlif 0..255;            # all interfaces
                direction in;           # in
                direction out;          # AND outbound
                action filter;
        }

}
-----*/
```

/etc/grarp.conf

The `/etc/grarp.conf` file is set up to configure general address resolution for HIPPI, ATM, and FDDI media cards. It maps IP addresses to physical hardware addresses in support of ARP functions.

`/etc/grarp.conf` can also be used to manually manipulate the ARP cache to resolve a temporary network irregularity. This type of problem occurs when, for example, a destination's file does not properly respond to ARP requests, or when its hardware address has recently changed. When an ATM destination does not respond to Inverse ARP requests, `grarp.conf` supplies the IP address information that should have come from the InARP reply.

Here is the `/etc/grarp.conf` file format with sample entries:

```
#ifname  hostname  phys_addr    [temp]   [pub]   [trail]

gh010   192.0.2.1      0x          temp
gf023   192.0.99.1     0x          pub trail
ga0364  192.0.130.1    0x
```

ifname

`ifname` is the interface name in the form "`gx0yz`."

The interface name defines the physical location in the chassis of each configured logical interface. It is used in the `grifconfig.conf` file to identify each logical interface.

hostname

`hostname` is the IP address or name of the host to which the address is to be mapped.

phys_addr

The third entry, `phys_addr`, is the remote system's hardware address. The format of this entry is specific to each type of media.

For Ethernet or FDDI, use the 48-bit MAC address (in hex):
```
xx:xx:xx:xx:xx:xx
```

For HIPPI, use the 32-bit I-field, an unsigned integer in C language convention (prefixed with "0x" for hexadecimal, "0" for octal, or a non-zero digit for decimal).

For ATM, there are three options:

– the NSAP address in 20 hexadecimal bytes, two digits each, optionally separated by periods:

```
xx.xx.xx ... xx.xx.xx
```

Multiple NSAP addresses are separated by plus signs (+).

– for PVC connections, the VPI/VCI values are in decimal integers separated by a slash:

```
vp/vc
```

– in CCITT E.164-style address

To configure ARP on the ATM media card, you must specify addresses of local ARP servers in the `/etc/gratm.conf` file in addition to setting addresses in `/etc/grarp.conf`.

See the **grarp** command description or the **grarp** man page for more information.

This is the template for `grarp.conf`:

```
#  NetStar $Id: grarp.conf,v 1.2.22.1 1997/05/09 17:35:00 jim Exp $
#
# Template grarp.conf file.
#
#    This file contains any hardwired IP-address-to-hardware-address
#    mappings you may want for GigaRouter interfaces.  It is especially
#    important for HIPPI, whose ARP tables are manually configured.
#    For HIPPI, this file replaces the "gria" command.
#
# File syntax:
#
# [ifname]  host    hwaddr   [temp]  [pub]  [trail]
#
# [ifname] If given, this is the interface name as you would find it
#    in column 1 of the netstat -i output.
#
# host    Hostname or IP address of the remote system
#
# hwaddr   Hardware address of the remobe system in one of the
#    following formats:
#
#      48-bit MAC address for
#      Ethernet or FDDI:  xx:xx:xx:xx:xx:xx
#            where 'xx' are hexadecimal digits.
#
#      32-bit I-field for
#      HIPPI:  C-language syntax for a 32-bit constant
#            Example:  0x03000555 for logical address x'555.
#
#      20-byte NSAP address or VPI/VCI for
#      ATM:   vp/vc        VPI/VCI for PVCs
#             where 'vp' and 'vc' are decimal integers
#            xx.xx.xx. . .xx.xx.xx.xx     NSAP address
#             where 'xx' are hexadecimal digits.
#
#
###########   Add your configuration following this line   ##########
```

/etc/gratm.conf

The `/etc/gratm.conf` file defines ATM parameters and permenent virtual circuits (PVCs). Parameters to configure ATMP tunneling and encapsulated bridging are also provided.

```
#
# NetStar $Id$
#
# gratm.conf - GigaRouter ATM Configuration File
#

#
# This file is used to configure GigaRouter ATM interfaces.
# Statements in this file are used to configure ATM PVC's,
# signalling protocols, arp services, and traffic shapes.
#
# gratm(8) uses this file as input when it is run by grinchd(8)
# whenever an ATM media card boots to configure the card.

# gratm.conf is divided into five sections:
#
# The Service section is where ATM ARP services are defined (entries
#   defined in this section are referenced from the Interface section
#   of this file to define which ARP service an interface should use).
#
# The Traffic Shaping section is where traffic shapes are defined
#   entries defined in this section are referenced from the Interface
#   and PVC sections of this file to define which traffic shapes
#   interfaces and PVC's should use).
#
# The Signalling section is where the signalling protocol to be used
#   by a physical interface to establish Switched Virtual Circuits
#   is specified.
#
# The Interfaces section is where per-logical-interface parameters
#   such as ARP services and Traffic shapes are bound to specific
#   logical interfaces.
#
# The PVC section is where Permanent Virtual Circuits are defined,
#   using traffic shapes defined in the Traffic Shaping section,
#   along with other parameters specific to PVC configuration.


#
# Notes on the format of this file:
#
# Comments follow the Bourne Shell style (all characters following a #
# on a line are ignored).
#
# Statements in this file are separated by newlines.  A statement may
# span multiple lines by ending each incomplete line of the statement
# with a '\' character.  Example:
#
# Traffic_Shape name=high_speed peak=15000   # this is a statement
#
# Service name=bc0 type=bcast addr=198.174.11.1 \
#  addr=198.176.11.1                       # this is also a statement
#
# User-defined names for Traffic_Shape's and Service's must be defined
# before they are used, ie., the definition of a traffic shape must
# precede its use in an Interface or PVC specification, and the
# definition of a Service must precede the use of the name of that
# service in defining any Interfaces.



#
# ARP Service info
#
# Lines beginning with the keyword "Service" define virtual "services"
# which may or may not be present on an ATM network attached to a
# GigaRouter.
# Each Service entry in the ATM configuration file has the format:
#
# Service name=value type=arp|bcast|arpserver addr=value
# [addr=value ...]
#
# The "name" field is a unique name to identify this ATM service
#
# The "type" field specifies the type of ATM service being configured.
# and how the address argument(s) which follow are interpreted.
#
#type=arpserver  Defines a local arpserver. The NSAP address can be
# determined automatically by specifying addr=auto or manually by
# specifying addr=yyyyyyyyyy where yyyyyyyyyy is the NSAP address.
#
# type=arp   Indicates an ARP service.  One to three "addr" field(s)
#      must follow, defining NSAP addresses of ARP
#      services on the attached ATM network.
#
#      A logical interface using this "Service" entry
#      will connect to one of the addresses defined for
#      this service to get ATM address information for
#      any IP addresses it does not already know the
#      ATM address of.
#
#type=bcast     Defines a "broadcast service".  The "addr" field(s)
#      contain IP addresses of hosts on a given logical
#      logical ATM network to which copies of any broadcast
#      packets will be sent, allowing the GigaRouter, when
#      so configured, to simulate broadcast over a logical
#      IP network.
#
```

```
#Service name=arp0 type=arp \
addr=47000580ffe1000000f21513eb0020481513eb00

#Service name=arpserver0 type=arpserver \
addr=47000580ffe1000000f21513eb0020481513eb00
#Service name=arpserver1 type=arpserver addr=auto

#Service name=bc0 type=bcast addr=198.174.20.1 addr=198.174.22.1 \
#addr=198.174.21.1
#



#
#
# Traffic shaping parameters
#
# Lines beginning with the keyword "Traffic_Shape" define
# traffic shapes which may be used to configure the performance
# characteristics of ATM Virtual Circuits.
#
# The Traffic_Shape's defined here are to be referenced by name when
# to assign traffic shapes to PVC's or Interfaces later in this
# configuration file.  (See Examples in the PVC or Interface section
# of this file for examples on how to reference traffic shapes defined
# here.)
#
# Each Traffic_Shape entry the ATM configuration file has the format:
#
# Traffic_Shape name=value peak=bps [sustain=bps burst=cells]
# [qos=high|low]
#
# The "name" field is a unique name to identify this ATM service, so we
# can refer to a collection of peak,[sustain, burst],[qos] parameters
# as a group when configuring PVC’s or Interfaces later in the file.
#
# The 'peak', 'sustain', and 'burst' fields specify, respectively,
# the peak cell rate, the sustained cell rate, and the burst rate.
# The values for 'peak' and 'sustain' are in kilobits per second (max
# of 155000), and the value for 'burst' is in cells (maximum of 2048).
#
# The 'qos' (Quality of Service) field specifies which rate queues to
# use.  A value of 'high' corresponds to high priority service which
# uses the high-priority rate queues, and a value of 'low' corresponds
# to low priority service which uses the low-priority rate queues.
#
# The peak rate is the only parameter which is mandatory.  If ommitted,
# the sustain and burst rates are set to match the peak rate.  If qos
# is not specified, it defaults to "high".
#
```

```
#Traffic_Shape name=high_speed_high_quality \
#    peak=155000 sustain=155000 burst=2048 qos=high

#Traffic_Shape name=medium_speed_low_quality \
#    peak=75000 qos=low

#Traffic_Shape name=low_speed_high_quality \
#    peak=15000 qos=high
#
#



#
# Signalling parameters
#
# Lines beginning with the keyword "Signalling" define
# the signalling protocol which will be used on a physical
# ATM interface to establish Switched Virtual Circuits for
# any logical interfaces on the named physical interface.
#
# Physical interfaces on GigaRouter ATM cards are identified by
# the slot number of the interface card in the GigaRouter chassis
# in hex notation (0-f) plus the location of the physical interface on
# the card (either the top or the bottom connector on the card).
#
# Each Signalling entry the ATM configuration file has the format:
#
# Signalling card=hex connector=top|bottom \
# [protocol=UNI3.0|UNI3.1|NONE] [mode=SDH|SONET] [clock=Ext|Int]
#
# The 'card' and 'connector' specification are mandatory.
#
# The card should be identified by a hexidecimal digit representing
# the slot number of the card in the GigaRouter chassis.
#
# The connector should be either 'top' or 'bottom'.
#
# The 'protocol' parameter defines the signalling protocol to be used
# in the setup of Switched Virtual Circuits (SVC's) on this physical
# interface.  This parameter is optional.  If left unspecified, the
# ATM card uses UNI3.0 signalling by default.
#
# Valid values for the signalling parameter include:
#
#          UNI3.0    - for the UNI 3.0 signalling protocol.
#          UNI3.1    - for the UNI 3.1 signalling protocol.
#          NONE      - for no signalling protocol.
#
# The 'mode' specification is optional. It can be either SDH or SONET.
# By default, it uses SONET.
#
#
```

```
# The 'clock' specification is also optional.  It can be either
# Ext(ernal) or Int(ernal). The default setting is Internal clocking.
#
# The buf_limit specification is optional (it's default value is 15).
# This parameter is used to control the depth of the queue of buffers
# on a given vc.  It allows one to keep a given vc from consuming all
# the buffers on a board if it is oversubscribed with respect to its
# traffic shape.
# As the circuit becomes oversubscribed, the number of buffers queued
# to that circuit is evaluated if the depth exceeds buf_limit, then the
# packet is discarded rather than tx'd as the threshold has been
# crossed.  This limit is set once for the whole board.  You may set it
# on either port, the last setting for the board will take effect.  To
# achieve behavior similar to older
# releses where all buffers could queue on a particular circuit set
# this value to 256.  This value is only used on atm-oc3 boards.

# Signalling card=3 connector=top protocol=none mode=sonet clock=Int\
# buf_limit=15

# Signalling card=9 connector=top protocol=UNI3.1
# Signalling card=9 connector=bottom protocol=NONE

# Signalling card=a connector=top protocol=UNI3.1
# Signalling card=a connector=bottom protocol=NONE
#



# Interfaces
#
# Lines beginning with the keyword "Interface" define
# GigaRouter logical ATM interfaces.
#
# The format of a logical interface definition is:
#
# Interface ifname [service=service_name] [traffic_shape=shape_name] \
#   [bridge_method=method[,restriction]]
#
# The optional 'service' parameter allows an ATM service to be
# be defined for this logical interface (the 'service_name' must be
# a name defined in the Services section above).
#
# The optional 'traffic_shape' parameter allows a traffic shape
# to be defined for this logical interface (the 'shape_name' must be
# a named defined as a Traffic_Shape above).
#
# If no traffic shape is specified, a default shape of 155Kbps, high
# quality of service is used.
#
# The optional 'bridge_method' parameter allows an interface to be used
# for RFC 1483 bridging.  The valid values for the bridge_method
# parameter are:
#
#   llc_encapsulated   - Use a single PVC, with each frame
#                        encapsulated with an LLC header to identify
#                        the protocol
#
#   vc_multiplexed     - Use a separate PVC for each protocol
#
#LLC encapsulated bridging allows any LAN frame type to be transmitted,
# and also allows IP datagrams to be sent directly on the VC.  The
# optional 'restriction' parameter can limit how IP datagrams are
# routed to the interface, and on what kind of LAN frames are trans
# -mitted on it. The valid values for the restrction parameter are:
#
#   broute_to_ether   - Transmit all routed IP datagrams as
#                         Ethernet frames
#
#   ether_only        - Transmit all frames (routed IP datagrams and
#                        all bridged LAN frames) as Ethernet frames
#
#   broute_to_fddi    - Transmit all routed IP datagrams as
#                        FDDI frames
#
#   fddi_only          - Transmit all frames (routed IP datagrams and
#                       all bridged LAN frames) as FDDI frames
#
#
# Note that unless a restriction or an ARP service is specified, an
# LLC-encapsulated bridging interface will only be able to route to the
# host at the other end of the ATM PVC.

#Interface ga090  service=arp0 traffic_shape=high_speed_high_quality
#Interface ga0980 service=net20 traffic_shape=low_speed_high_quality

#Interface ga0a0  service=arp0 traffic_shape=high_speed_high_quality
#Interface ga0a80 service=net20 traffic_shape=low_speed_high_quality

#Interface ga091  traffic_shape=high_speed_high_quality \
#   bridge_method=llc_encapsulated,broute_to_ether


#
#
# PVC's
#
# Lines beginning with the keyword "PVC" define
# Permanent virtual circuits.
#
# The format of a PVC definition is:
#
#  PVC ifname VPI/VCI \
#  proto=ip|raw|vc|ipnllc|isis|llc[,bridging]|vcmux_bridge,bpro|
#    vc_atmp|llc_atmp \
#  [input_aal=3|5|NONE] [traffic_shape=shape] \
#  [dest_if=logical_if [dest_vc=VPI/VCI]]
#
# The first three parameters (ifname, VPI/VCI, proto) are mandatory.
#
# 'ifname' specifies the GigaRouter ATM logical interface in the usual
# format (e.g., ga030, ga0e80).
#
# 'VPI/VCI' specifies the (decimal) Virtual Path Identifier and
# Virtual Circuit Identifier of the PVC, separated by a slash (/).
#
# 'proto' specifies the protocol to be supported on this PVC.  Legal
# values are:
#
#  'ip'  Internet Protocol (with LLC/SNAP headers)
#  'raw'  raw adaptation layer (AAL-5 or AAL-3/4) packets
#  'isis'  IS-IS packets
#  'llc'   any LLC-encapsulated protocol supported by the GRF
#         (except for RFC 1483 bridging)
#  'llc,bridging'   RFC 1483 bridging [This is the PVC type for an
#      interface using bridge_method=llc_encapsulated.]
#  'vcmux_bridge'   bridged packets [A PVC type for an interface
#          using bridge_method=vc_multiplexed.] An additional
#          parameter specifies the protocol carried on the VC:
#    ether_fcs      Ethernet frames, with Frame Check Sequence
#    ether_nofcs    Ethernet frames, without Frame Check Sequence
#    fddi_fcs       FDDI frames, with Frame Check Sequence
#    fddi_nofcs     FDDI frames, without Frame Check Sequence
#    bpdu           802.1D Bridging Protocol Data Units
#    'vc'           IP datagrams (NULL encapsulation)
#    'ipnllc'       same as 'vc'
#    'vc_atmp'      atmp home network connection using VC-based
#                     multiplexing (NULL encapsulation of IP packets)
#    'llc_atmp'  atmp home network connection using LLC encapsulation
#
# If the 'proto' specified is 'raw', 'dest_if' parameter specifies the
# GigaRouter interface of the destination for this raw adaptation layer
# connection (specified in the same GigaRouter interface format
# described above), and (optionally) the dest_vc parameter specifies
# the destination VPI/VCI.
#
# the 'input_aal' parameter specifies the adaptation layer,
#  input_aal=3    specifies AAL-3/4
#  input_aal=5    specifies AAL-5
#
# The optional 'traffic_shape' parameter allows a traffic shape
# to be defined for this logical interface (the 'shape_name' must be
# a named defined as a Traffic_Shape above).
#
#
#
```

```
# If no traffic shape is specified, a default shape of 155Kbps, high
# quality of service is used.

#PVC ga090  0/32 proto=ip traffic_shape=high_speed_high_quality
#PVC ga0980 0/32 proto=ip traffic_shape=high_speed_high_quality

#PVC ga0a0  1/33 proto=raw traffic_shape=high_speed_high_quality \
#   dest_if=ga090 dest_vc=5/50  input_aal=5

#PVC ga0a80 1/33 proto=ip traffic_shape=high_speed_high_quality

#PVC ga0b0  0/40 proto=isis traffic_shape=high_speed_high_quality
#PVC ga0c0  0/41 proto=isis_ip traffic_shape=high_speed_high_quality
#
#PVC ga030  0/32 proto=llc,bridging
#
#PVC ga031  0/32 proto=vcmux_bridge,ether_nofcs
#PVC ga031  0/33 proto=vcmux_bridge,fddi_nofcs
#PVC ga031  0/34 proto=vcmux_bridge,bpdu
```

/etc/grfr.conf

The `grfr.conf` file configures the Frame Relay protocol on HSSI and SONET cards.

Note: The space before or after equal signs in `grfr.conf` can be there or not, it does not matter. This is true for all of the parameters that use the equal sign. That is why `Enabled = Y` works, and so does `CIR=56000`.

The `grfr.conf` file has eight sections:

– Link section:	set Frame Relay link (physical port) parameters
– PVC section:	set Frame Relay route circuit parameters
– PVCS section:	define Frame Relay switch circuits parameters
– PVCM1 section:	define Frame Relay 1-way multicast group circuits
– PVCM2 section:	define Frame Relay 2-way multicast group circuits
– PVCMN section:	define Frame Relay N-way multicast group circuits
– PVCEP section:	define an individual endpoint's parameters
– PVCATMP section:	define GRF home agent connection to home network

```
########################################################################
#   grfr.conf: Frame Relay Configuration
########################################################################
#   Link Section: Set Frame Relay Link (physical port) Parameters.
#
#    The first two parameters are mandatory (Slot and Port). Parameters
#    with * prefixed are newly defined parameters.
#
#    Optional Parameters:
#
#     Name:              String: Link Description.  Deafult = "".
#     Enabled:           Y|N: Enable/Disable Link.  Default = Y.
#     LMIType:           None|AnnexA|AnnexD.
#                        Default = None
#
#     N391:    1..255: Polling Intervals per Full Status Message.
#                          Default = 6.
#     N392:   1..10: Error Reporting Threshold.  Default = 3.
#     N392:    1..10: Measurement Interval for mN2. Default = 4.
#     T391:    5|10|20|25|30: Heartbeat Poll Interval. Default = 10.
#  *  T392:   5|10|15||20|25|30: Poll Verification Timer. Default =15.
#  *  Linktype:         UNI-DTE|UNI-DCE|NNI. Default is UNI-DTE.
#     AutoAddGrif:      gs0xx: | Auto | not specified
#                       gs0xx: If LMI reports a new PVC, they will be
#                        added automatically to this interface, using
#                        default values, and Inverse Arp.
#
#              Auto: If LMI reports a new PVC the system will attempt
#              to determine which interface the newly added circuit
#              belongs and attach the circuit to that interface.
#
```

```
#              If the parameter is not specified, such PVCs will not
#              be added (the default behavior).
#    NOTES:
#              Only Routed and ATMP circuits are affected by this
#              parameter, switch and multicast circuits are not.
#
#              Currently, AutoAddGrif=auto is not supprted yet.
#
#
#   Slot Port Optional Parameters
#   ==== ==== ==================
#link 6  0   name ="Upper Port 6" LMIType = AnnexD  AutoAddGrif = gs060
#link 6  1   LMIType = AnnexD

######################################################################
#  PVC Section: Set Frame Relay Route Circuit Parameters.
#
#   Keyword: pvc or pvcr. pvc for backward compatibility
#
#   The first three parameters (Logical Interface, DLCI, and Peer IP
#   address) are mandatory. Parameters with * prefixed are new defined
#   parameters.
#
#  lif:        gsXXX:            One of the logical Interfaces defined
#                                  in grifconfig.conf
#  DLCI:       16..991:          Channel ID.
#  IP Address: 255.255.255.255:  IP Address of station at other end of
#                                this PVC. Use 0.0.0.0 to resolve the
#                                address using Inverse Arp.
#
#   Optional parameters:
#
#   Name:      Quoted String:    PVC name.  Default = "".
#   Enabled:   Y|N:              Enable/Disable PVC.  Default = Y.
#  *CIR:       int:          Committed Information Rate. Default = 55M
#  *Bc:        int:          Committed Burst Size. Default = 55M
#  *Be:        int:          Excess Burst Size. Default = 0
#   isis:       Y|N          Y to support ISIS traffic. Default is N.
#
#  Notes: Together, CIR, Be, and Bc are referred as "Traffic Enforcing
#         parameters". Values are in bits.
#
#    lif    DLCI Peer IP Address Optional Parameters
#    ===    ==== ==============  ===================
#pvc  gs060  405  0.0.0.0       Name="Router1" isis=Y
#pvcr gs060  600  192.0.2.6     Enabled = N  Name="Rotuer2"
#pvc  gs060  505  192.0.2.50    Enabled = Y CIR=56000 Bc=56000 Be=2400

#####################################################################
#   PVCS Section: To define Frame Relay Switch Circuit Parameters.
#
#   Keyword: pvcs
```

```
#
#   The first two parameters (Circuit Endpoint-A and Endpoint-B)
#   are mandatory.
#
#    EPA:      Slot:Port:DLCI   Slot: 0..15, Port: 0..1, DLCI:16..991
#    EPB:      Slot:Port:DLCI   Slot: 0..15, Port: 0..1, DLCI:16..991
#
#   Optional parameters:
#
#    Name:      Quoted String:    PVC name.  Default = ASCII string
#                                 of EPA. Example: EPA = 1:1:459 then
#                                 name = "1:1:459". Name could be used
#                                 to retrieve status of the circuit.
#    Enabled:    Y|N:       Enable/Disable switch PVC.  Default = Y.
#    CIR:        int:       Committed Information Rate. Default = 55M
#    Bc:         int:       Committed Burst Size. Default = 55M
#    Be:         int:       Excess Burst Size. Default = 0
#
#  Notes: Both endpoints, A and B, will have the same CIR, Bc, and
#        Be. These parameters could be overridden using pcvep keyword.
#
#
#     EPA         EPB       Optional Parameters
#     ===         ====      ==================
#pvcs  0:1:199    12:0:98   Name="la_mpls" CIR=64000 Bc=64000 Be=2400

######################################################################
#  PVCM1 Section: To define Frame Relay 1-way Multicast Group Circuits.
#
#   Keyword: pvcm1
#
#   The first (Root Circuit Endpoint), and the next N (leaf Endpoints)
#   parameters are mandatory. Where 1 <= N <= X. X is limited by the
#   support hardware.
#
#     Theoretically, X could be infinite, but as the value of X gets
#     higher, the line gets longer, less manageable. Using multiple
#     pvcm1 lines is an alternative to define a 1-way circuit with
#     many leaf nodes.
#
#    EPR:      Slot:Port:DLCI   Slot: 0..15, Port: 0..1, DLCI:16..991
#    EP1:      Slot:Port:DLCI   Slot: 0..15, Port: 0..1, DLCI:16..991
#    ....      ....             .......
#    EPN:      Slot:Port:DLCI   Slot: 0..15, Port: 0..1, DLCI:16..991
#
#   Optional parameters:
#
#   ame:     Quoted String:  Group Name.  Default = ASCII string
#                            of EPR. Example: EPR = 1:1:459 then
#                             name = "1:1:459". Name could be used to
#                             retrieve status of the multicast group.
#    Enabled:    Y|N:       Enable/Disable Multicast group. Default = Y
#    CIR:       int:       Committed Information Rate. Default = 55M
#    Bc:        int:       Committed Burst Size. Default = 55M
#    Be:        int:       Excess Burst Size. Default = 0
#
#  Notes:  All the leaf endpoints could have been defined as part of
#          switch circuits with pcvs keyword. The switched circuits
#          that are part of the 1-way Mcast circuit must be defined
#          before the multicast circuit is defined.
#
#          The optional parameters specified here apply to the root
#          endpoint and those leaf nodes that have not been defined
#          as switch endpoints.
#
#         Multiple pcvm1 keywords/lines can be used to define a
#         multicast group. The parameters of the first line will be
#        used and the root endpoint must be the same. In the following
#         example, the first 3 lines define a multicast group of
#         1 root endpoint and 6 leave endpoints with the name of the
#         group is "netstar_g".
#
#     EPR       EP1       EP2       Optional Parameters
#     ===       ====      ===       ==================
#pvcm1 1:0:200   12:1:66   12:1:67   Name="netstar_g"
#pvcm1 1:0:200   12:1:68   12:1:69
#pvcm1 1:0:200   12:1:70   12:1:71

######################################################################
#  PVCM2 Section: To define Frame Relay 2-way Multicast Group Circuits.
#
#   Keyword: pvcm2
#
#   The first (Root Circuit Endpoint), and the next N (leave Endpoints)
#   parameters are mandatory. Where 1 <= N <= X. X is limited by the
#   support hardware.
#
#    EPR:       Slot:Port:DLCI   Slot: 0..15, Port: 0..1, DLCI:16..991
#    EP1:       Slot:Port:DLCI   Slot: 0..15, Port: 0..1, DLCI:16..991
#    ....        ....             .......
#    EPN:       Slot:Port:DLCI   Slot: 0..15, Port: 0..1, DLCI:16..991
#
#   Optional parameters:
#
#   Name:    Quoted String:   Group Name.  Default = ASCII string
#                             of EPR. Example: EPR = 1:1:459 then
#                             name = "1:1:459". Name could be used to
#                             retrieve status of the multicast group.
.    Enabled:    Y|N:       Enable/Disable Multicast group. Default = Y
#    CIR:        int:       Committed Information Rate. Default = 55M
#    Bc:         int:       Committed Burst Size. Default = 55M
#    Be:         int:       Excess Burst Size. Default = 0
#
#          Multiple pcvm2 keywords/lines can be used to define a
#          2-way multicast group. The parameters of the first line
#          will be used and the root endpoint must be the same.
#
#     EPR       EP1       EP2       Optional Parameters
#     ===       ====      ===       ==================
#pvcm2 1:0:200   12:1:66   12:1:67   Name="ascend_g"
#pvcm2 1:0:201   12:1:68   12:1:69
#pvcm2 1:0:202   12:1:70   12:1:71

#pvcm2 0:1:199   12:0:98   11:1:49   Name="minn_g" CIR=64000 Bc=64000
Be=2400


######################################################################
# PVCMN Section: To define Frame Relay N-way Multicast Group Circuits.
#
#   Keyword: pvcmn
#
#   The first N (Endpoints) parameters are mandatory. Where
#         1 <= N <= X. X is limited by the support hardware.
#
#    EP1:      Slot:Port:DLCI   Slot: 0..15, Port: 0..1, DLCI:16..991
#    ....      ....             .......
#    EPN:      Slot:Port:DLCI   Slot: 0..15, Port: 0..1, DLCI:16..991
#
#   Optional parameters:
#
#    Name:    Quoted String:  Group Name.  Default = ASCII string
#                            of EP1. Example: EP1 = 1:1:459 then
#                            name = "1:1:459". Name could be used to
#                            retrieve status of the multicast group.
.    Enabled:    Y|N:     Enable/Disable Multicast group. Default = Y
#    CIR:        int:     Committed Information Rate. Default = 55M
#    Bc:         int:     Committed Burst Size. Default = 55M
#    Be:         int:     Excess Burst Size. Default = 0
#
#   Notes: The parameters specified here only apply to all endpoints.
#
#             Multiple pcvmn keywords/lines can be used to define a
#             n-way multicast group the parameters of the first line
#             will be used and the group name must be the same.
#
#     EP1       EP2       EP3       Optional Parameters
#     ===       ====      ===       ==================
#pvcmn 1:0:200   12:1:66   12:1:67   Name="ascend_1"
#pvcmn 1:0:200   12:1:65   12:1:64
#pvcmn 1:0:200   12:1:63   12:1:69

#pvcmn 1:0:201   12:1:68   12:1:69   Name="ascend_2"
#pvcmn 1:0:202   12:1:70   12:1:71   Name="ascend_3"

#pvcmn 0:1:199   12:0:98   11:1:49   Name="minn_g" CIR=64000 Bc=64000
Be=2400
```

```
######################################################################
#  PVCATMP Section: Set Frame Relay ATMP Circuit Parameters.
#
#   Keyword: pvcatmp  (atmp = Ascend Tunnel Management Protocols)
#
#   The first three parameters (Logical Interface, DLCI, and Peer IP
#   address) are mandatory. Parameters with * prefixed are new defined
#   parameters.
#
#    lif:       gsXXX:     One of the logical Interfaces defined in
#                          grifconfig.conf.  Do not set addresses on
#                          pvcatmp interfaces in grifconfig.conf.
#                          Use the form:    "gsXXX  -  -  -  up"
#    DLCI:       16..991:          Channel ID.
#    IP Address: 255.255.255.255:  IP Address of station at other end
#                                 of this PVC.  Use 0.0.0.0 to resolve
#                                 the address using Inverse Arp.
#
#   Optional parameters:
#
#   Name:      Quoted String:    PVC name.  Default = ASCII string
#                                 of EPA. Example: EPA = 1:1:459 then
#                                 name = 1:1:459". Name could be used
#                                 to retrieve status of the circuit.
#    Enabled:    Y|N:       Enable/Disable switch PVC.  Default = Y.
#    CIR:        int:       Committed Information Rate. Default = 55M
#    Bc:        int:        Committed Burst Size. Default = 55M
#    Be:        int:        Excess Burst Size. Default = 0
#
#        lif     DLCI Peer IP Address Optional Parameters
#        ===     ==== ============== ==================
#pvcatmp gs060   505  192.0.2.50    Enabled = Y CIR=56000 Bc=56000
Be=2400


######################################################################
#   PVCEP Section: To define an individual Endpoint's Parameters.
#
#   Keyword: pvcep
#
#   All parameters are mandatory.
#
#    EP:       Slot:Port:DLCI    Slot: 0..15, Port: 0..1, DLCI:16..991
#    CIR:      int:              Committed Information Rate.
#    Bc:       int:              Committed Burst Size.
#    Be:       int:              Excess Burst Size.
#
#    Notes: Parameters defined with pvcep keyword override parameters
#           defined by any other keywords. This is designed to provide
#           flexibility in configuring switch/multicast circuits and
#           Asymmetric PVCs.
```

```
#
#           For example, to configure an asymmetry circuit, use pvcep
#           to set a different value of one endpoint of the circuit
#           traffic shaping parameters.
#
#           pvcep entries must come after other entries which define
#           circuit endpoints.
#
#        EP         CIR        Bc         Be
#       ===        ====        ==         ==
#pvcs  0:1:199     12:0:98     Name="la_mpls" CIR=64000 Bc=64000
Be=2400
#pvcep 12:0:98   CIR=128000    Bc=128000   Be=9600


#######################   END OF GRFR.CONF  #########################
```

/etc/grifconfig.conf

The `/etc/grifconfig.conf` file defines the interface name, IP address, and netmask for each GRF logical interface. At minimum, the interface name must be established for every configured interface.

When a media card comes on-line, the **grifconfig** script consults the contents of the `/etc/grifconfig.conf` file to issue the proper commands that will configure the logical interfaces.

Here is the file format and sample entries for each type of media:

```
# name  address       netmask          broad_dest   arguments
#
gh010   192.0.2.1     255.255.255.0
gf023   192.0.99.1    255.255.255.0    192.0.99.255
ga0364  192.0.130.1   255.255.255.0    192.0.130.10
ga0382  192.0.131.1   255.255.255.0
```

In the example above, the first entry (`gh010`) is the HIPPI media card in slot 1, while the second entry (`gf093`) is logical interface 3 on the FDDI card in slot 2. The third entry refers to the ATM card in slot 3, and specifies a logical interface assigned as permanent virtual circuit (PVC) 64. ATM interface `ga0364` is an explicit point-to-point connection and includes the destination IP address of its point-to-point link in the destination address field.

Interface name

The `name` entry is the GRF interface name. The GRF interface name has five components that describe an individual interface in terms of its physical slot location in the chassis, and its specific and virtual locations on a media card.

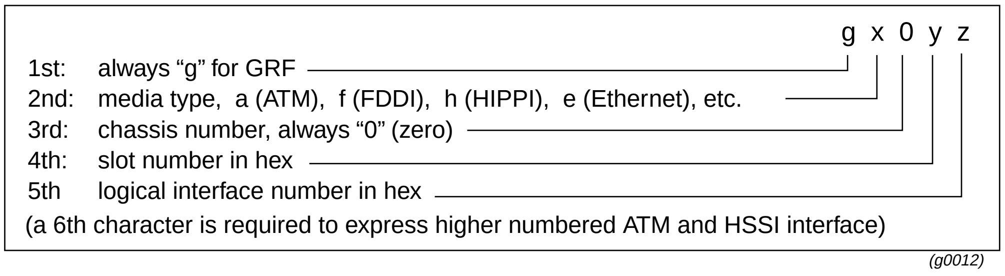

g x 0 y z
1st: always “g” for GRF
2nd: media type, a (ATM), f (FDDI), h (HIPPI), e (Ethernet), etc.
3rd: chassis number, always “0” (zero)
4th: slot number in hex
5th logical interface number in hex
(a 6th character is required to express higher numbered ATM and HSSI interface)
(g0012)

Figure 2-1. Components in the GRF interface name

Here are examples of interface names:

HIPPI interface names:	`gh030` (single interface per card, only the slot # changes)
ATM interface names:	`ga037f, ga0281, ga01ff`
FDDI interface names:	`gf030, gf021, gf012, gf003`
HSSI interface names:	`gs030, gs021, gs0180`
100Base-T interface names:	`ge030, ge026, ge017`
SONET interface names:	`go030` (single interface per card, only the slot # changes)

```
#NetStar $Id: grifconfig.conf,v 1.10.2.3 1997/08/01 17:24:04 pargal
#
# Configuration file for GigaRouter/GRF interfaces.
#
# The contents of this file specify the IP addressing information for
# the networks attached to the system's interfaces.  This includes
# interfaces on media cards as well as directly attached interfaces
# such as de0 or ef0 (maintenance Ethernet) or lo0 (software loopback).
#
# The addresses of directly attached interfaces are configured
# directly from this file by /etc/netstart calling the grifconfig(8)
# script.
#
# The addresses of the interface(s) on a given media card are
# configured into the BSD/OS kernel when the media card boots and
# comes on line.
#
# Each entry in this file has the following format:
#
# name  address               netmask        broad_dest     arguments
#
# The name of a GigaRouter interface encodes the hardware type,
# GigaRouter cage number, slot number, and interface number.
#
#   --  The first character must be 'g' (to specify a GigaRouter
#       interface).
#   --  The second character is the hardware type of the
#       interface:  'a' for ATM, 'e' for ETHERNET,
#       'f' for FDDI, 'h' for HIPPI, 'p' for PPP, 's' for HSSI.
#       ('l' is also used, for GigaRouter software loopback.)
#   --  The third character is the number of the GigaRouter cage.
#       (Currently this must be '0', as multiple GigaRouter cages
#       are not yet supported.)
#   --  The fourth character is the hex digit (0 through f) of
#       the slot number within the GigaRouter cage.
#   --  The fifth (and sixth) characters specify the number of the
#       LOGICAL interface on the card:
#
#               For ATM cards, the fifth and sixth characters are
#               the hex digits of the logical interface.  Logical
#               interfaces numbered from 0 to 7f are on the top
#               physical connector on the ATM card, and logical
#               interfaces numbered 80 to ff are on the bottom
#               physical connector.  NOTE:  These logical interface
#               numbers are NOT the same as the VPI/VCI numbers
#               of a PVC (see /etc/grpvc.conf for that).
#
#               For FDDI cards, the fifth character will be 0, 1,
#               2, or 3 to specify the logical interface on the
#               FDDI card.  NOTE:  The logical interface number
#               may be different from the physical interface on
#               the card, depending on the single- or dual-
#               attachedness of the various interfaces.  Examples:
#               "gf073" specifies the bottom-most connector
#               on the FDDI card in slot 7; "gf020" specifies
#               top-most connector on the FDDI card in slot 2,
#               or the top TWO connectors on that card if they're
#               configured dual-attached.
#
#               For ETHERNET cards, the fifth character will be 0, 1,
#               2, 3, 4, 5, 6 or 7 to specify the physical interface
#               on the ETHERNET card.
#               Examples:
#               "ge067" specifies the 8th physical connector
#                       on the ETHERNET card in slot 6.
#               "ge000" specifies the first (top) connector on the
#                       ETHERNET card in slot 0.
#               "ge0f7" specifies the last (bottom) connector on the
#                       ETHERNET card in slot 15.
#
#               For HIPPI cards, which only have one interface,
#               the fifth character is always 0.  Example:
#               "gh0f0" specifies the interface for a HIPPI card
#               in slot 15.
#
# The IP "address", "netmask" (option), and "broad_dest" (option)
# address fields must be specified in canonical IP dotted-quad nota-
# tion. An entry of "-" (a single hyphen) may be specified for any of
# the ields as a place-holder. This may be useful, e.g.,if no netmask
# is desired but a broadcast or destination address must be specified
# in the next field.
#
# For this release, the "broad_dest" field specifies the broadcast
# IP address for Ethernet & FDDI interfaces, and the destination of
# a point-to-point ATM or HIPPI interface.
#
# The "arguments" field is for any additional arguments to be supplied
# to the underlying ifconfig(8) command that will be executed by
# grifconfig(8).  The most useful purpose would be to specify an
# MTU value for the interface using the "mtu" keyword of ifconfig(8).
# The keyword "iso" can also be specified here which designates the
# current line as an iso address entry.
# See the example entry below, and the man page for ifconfig(8).
#
# NOTE: All interface names are case sensitive ! Always use lower case
#       letters when defining interface names.
#
# name  address             netmask         broad_dest      arguments
#
#de0    192.0.2.1           255.255.255.0   192.0.2.255     mtu 1024
#lo0    127.0.0.1           255.0.0.0
#gl000  127.0.1.1
# configuration for iso addresses
#gf0xx  49.0000.80.3260.3260.3260.00 49.0000.80 -       iso
```

/etc/grlamap.conf

Configuration file format

The format of `/etc/grlamap.conf` is:

```
    # portcard  logical_addr  dest_portcard
```

where:

portcard is a comma-separated list of media card numbers to which the other fields apply. An * *(asterisk)* indicates all media cards.

Ranges can be used. If media cards 5, 6, 7 and 11 get the same mapping, you can specify "5-7,11".

logical_address is a comma-separated list of logical addresses to map.
An * *(asterisk)* indicates the default mapping, that is, all logical addresses get mapped.

Ranges can also be used. If logical addresses 0xfc0 through 0xfff get mapped to the same set of media cards, you can specify "0xfc0-0xfff".

destination_portcard is a comma-separated list of destination media card numbers to which the logical address gets mapped.

Only the first four values are used. Any additional values are flagged as invalid, an error message is printed to the `gr.console` log, and **grlamap** exits. When only three values are designated, the first value fills out the fourth position. Ranges are not permitted.

Examples of /etc/grlamap.conf entries

To change the default mapping of IP routing for all media cards, enter:

```
    # portcard     logical_addr     dest_portcard
        *              0xfc0              0x40
```

To set up that all logical addresses map to media card 5 by default, enter:

```
    # portcard     logical_addr     dest_portcard
        *              *                  5
```

To set up that all logical addresses received by media cards 0 through 7 will map to media card 5 by default, enter:

```
    # portcard     logical_addr     dest_portcard
        0-7              *                  5
```

This example establishes that for media cards 1, 3, 4, and 15, the logical addresses 17, 42, and 99 will be sent to one of media cards 6, 8, 13, or 14.
It also sets media card 14 to send these same logical addresses to media cards 6, 8, or 13:

```
    # portcard     logical_addr     dest_portcard
    1,3,4,15        0x11,42,99       6,8,13,14
    14              0x11,42,99       6,8,13
```

This is the `grlamap.conf` file template:

```
#   NetStar $Id: grlamap.conf,v 1.1 1995/02/09 20:29:27 scotth Exp $
#
####################################################################
#
#
# There are 3 fields to determine how logical addresses get
# mapped to destination portcards for each hippi portcard.
#
# The first field is a comma separated list which determines the
# portcard on which the other 2 fields will apply.
# Ranges are allowed. i.e., 1,4-9,15 == 1,4,5,6,7,8,9,15
# '*' signifies that all portcards get this mapping.
#
# The second field is a comma-separated list of logical addresses being
# mapped.
# Ranges are allowed.
# values: 0 - 4095 inclusive, '*' signifies the default (i.e.,all LAs).
#
# The third field is a comma separated list of destination port cards.
# Only the first 4 values will be taken as valid, any extra values will
# be flagged invalid, a message will be printed to gr.console & grlamap
# will exit.
# If only 3 values are designated, the first value will be repeated as
# the fourth value.
#
#
# No whitespace is allowed in any field.
#
#
# ie.
# 5,6   997,998         1,0x4,8,15
#
# When port cards 5 & 6 receive an IP packet with a logical address of
# 997 or 998, it will then attempt to randomly forward the packet to
# one of the mapped ports 1, 4, 8 or 15.
#
#
#####################################################################

#*      *      5      # default mapping for all LAs for all portcards.
#5      *      6      # default mapping for LAs for portcard 5.

####################################################################
# IP mapping.
####################################################################
*      0xfc0   0x40        # default mapping of IP for all portcards.
###################################################################
#5      1-100   9          # default mapping for LAs for portcard 5.
#5      100-200 4          # default mapping for LAs for portcard 5.
```

/etc/grppp.conf

You use `/etc/grppp.conf` to assign the PPP protocol to logical interfaces on HSSI and SONET OC-3c media cards. Refer to the HSSI and SONET configuration chapters in the *GRF Configuration and Management* manual for PPP configuration information.

```
# Netstar $Id: grppp.conf,v 1.4 1997/03/25 16:54:45 suseela Exp $
# Template grppp.conf file.
#
# This file is used to set the initial configuration of PPP interfaces.
#
# When a media card configured for PPP is reset, grinchd executes grppp
# to process this file.   The following subset of grppp commands may be
# used in the grppp.conf file.  Most of these commands are used to
# overide default values, and should be used with caution.  Refer to
# the grppp man page for a full explanation of these commands.
#
#       interface INTERFACE_NAME
#       enable negotiation trace
#       set maximum configuration request count = INTEGER
#       set maximum failure count = INTEGER
#       set maximum terminate count = INTEGER
#       set restart timer interval = INTEGER
#       enable lcp magic number
#       set lcp keepalive interval = INTEGER
#       set lcp keepalive packet threshold = INTEGER
#       set lcp mru = INTEGER
#       enable lqr
#       set lqr timer interval = INTEGER
#       enable ipcp
#       enable osinlcp
#
# The example below shows the most commonly used grppp commands used in
# a grppp.conf file.
#
# Example Gigarouter PPP initial configuration
#
# interface gs0b0    # Card 11, port 0
#   enable negotiation trace   #copy negotiaton traces to
                                # /var/log/gr.console
#   enable ipcp      # allow IP traffic over PPP
#
# interface gs0b1    # Card 11, port 1
#   enable ipcp      # allow IP traffic over PPP
#   enable osinlcp   # allow osi traffic over PPP
```

/etc/grroute.conf

The `/etc/grroute.conf` file is a table of static routes for GRFs that do not use dynamic routing. This file only needs to include routes to remote networks and hosts, and networks and hosts that do not directly connect to a GRF interface. Routes for directly-connected networks and hosts are created from information in the `/etc/grifconfig.conf` file.

When a media card comes on-line, the **grroute** script reads the contents of `/etc/grroute.conf` to issue the necessary **route** commands that will add the routes having next hops through that media card's interface(s).

```
#   NetStar $Id: grroute.conf,v 1.3 1995/03/15 22:09:07 knight Exp $
#
# grroute.conf -configuration file for GigaRouter static remote routes
#
# This file should only contain routes to remote networks and
# hosts--i.e., networks and hosts not directly attached to a
# GigaRouter interface.  Routes for networks directly attached
# to the GigaRouter are created as part of configuring the
# GigaRouter interfaces; see /etc/grifconfig.conf.
#
# NOTE:  THIS FILE SHOULD NOT BE USED ON SYSTEMS WITH DYNAMIC
# ROUTING (gated).  If you are running gated, then you should specify
# static routes using the "static" statement in /etc/gated.conf.
#
# Whenever a port card boots, comes on line and has its interface(s)
# configured, the routes specified in this file that are for gateways
# on the network(s) directly attached to those interface(s) are
# configured into the BSD/386 kernel.
#
# The format of each line is follows:
#   destination         netmask         gateway/next hop
#
# The destination is the IP address of the remote host or network.
# For the default route, specify a destination of "0.0.0.0" or the
# word "default".
#
# A netmask is required for all entries in this configuration file.
# The netmask is normally the mask of the remote network:
#   192.0.2.0   255.255.255.0  123.45.67.89
#
# For remote host routes, specify a netmask of 255.255.255.255:
#  192.0.2.1    255.255.255.255  123.45.67.89
#
# In the case of the default route (0.0.0.0), the netmask is ignored,
# but some value must be present for the file to parse correctly.
#
#  destination   netmask   gateway/next hop
#  0.0.0.0   0.0.0.0    192.0.2.2
```

/etc/snmpd.conf

```
# Default Agent Configuration File
#
#       This file allows MANAGERS to be specified.  This is used to
#       specify which managers will be receiving which traps.
#
#       Also, COMMUNITYs can be specified. This allows that agent to
#       be configured such that it will only except requests from
#       certain managers and with certain community strings.
#
# GRAMMAR:
#       INITIAL         <name> <String>
#
#       TRANSPORT       <name>
#                       [SNMP | SMUX]
#                       [OVER [UNIX | UDP | TCP] [SOCKET | TLI]]
#                       [AT <addr>]
#
#       MANAGER         <addr> [ON TRANSPORT <name>]
#                       [SEND [ALL | NO | traplist] TRAPS
#                               [TO PORT <#> ]
#                               [WITH COMMUNITY <name>]]
#
#       COMMUNITY       <name>
#                       ALLOW op [,op]* [OPERATIONS]
#                       [AS <name>]
#                       [USE encrypt ENCRYPTION]
#                       [MEMBERS                <addr> [,<addr>] ]
#
#
#       ALLOW <subagentId> [ON <hostSpec>]
#             WITH <passwordSpec> [<entitySpec>] [<timeout>]
#
#       DENY <subagentId> ON <hostSpec> WITH <passwordSpec>
#
#       ENTITY <EntityName> DESCRIPTION <String>
#
#       LOCAL CONTEXT <contextName> [USES] VIEW <viewName>
#             REFERS TO ENTITY <entityName> AS <oid>
#
#       PROXY CONTEXT <oid> [USES] SOURCE PARTY <oid>
#                           DESTINATION PARTY <oid>
#                              [AND] CONTEXT <oid>
#
#VIEW <viewName> [[INCLUDE | EXCLUDE] SUBTREE <oid> [MASK <bitmask>]]+
#
#       ALLOW op [,op]* [OPERATIONS] <sugar> SOURCE PARTY <partyName>
#                                     DESTINATION PARTY <partyName>
#                                       [AND] CONTEXT <contextName>
```

```
#
#  <partyDefinition> ::= [LOCAL] PARTY <name> ON TRANSPORT <transport>
#                              AT <addr> <AuthPriv>
#                              AS <oid>
#
#   <transport> ::= [ snmpUDPDomain | snmpCLNSDomain | snmpCONSDomain |
#                        snmpDDPDomain | snmpIPXDomain  | rfc1157Domain
#
#       <transport> ::=
#       <AuthPriv> ::= <noAuth> <noPriv> |
#                      <md5Auth> <noPriv> |
#                      <md5Auth> <desPriv>
#
#       <noAuth> ::= <sugar> NO AUTHENTICATION
#
#       <sugar> ::= [AND] [[WITH | USING]]
#
#       <noPriv> ::= <sugar> NO ENCRYPTION
#
#       <md5Auth> ::= <sugar> MD5 AUTHENTICATION <key>
#
#       <key> ::= <sugar> <string> AS KEY
#
#       <desPriv> ::= <sugar> DES ENCRYPTION <key>
#
#       <subagentId>  ::=  SUBAGENT <oid> |
#                          SMUX SUBAGENT <oid> |
#                          UNSPECIFIED SUBAGENTS
#
#       <hostSpec> ::= HOST <hostid> |
#                      UNSPECIFIED HOST[S]
#
#       <passwordSpec> ::= PASSWORD <string>
#                          UNSPECIFIED PASSWORDS
#
#       entitySpec ::= AS ENTITY <entityName>
#
#       <timeout> ::= USING <specificTimeout> TIMEOUT
#       <specificTimeout> ::= <number> SECOND[S] |
#                             NO
#
#       addr ::=        <ip-kind> | <rfc1449addr> | <full-ip>
#       ip-kind ::=     <hostid> |
#                       <hostid> <portid> |
#                       <portid>
#
#       hostid ::=      <hostname> | <ip>
#                            where: hostname is defined in /etc/host
#       portid ::=      [PORT | : ] <#>
#
#       full-ip ::=     <ip>:<#>
#       ip ::=          <#>.<#>.<#>.<#>
#       traplist ::=    trap [, trap]*
#       trap ::=        <trap_name>
#
#
#       op ::=          ALL | GET | SET | TRAP
#       encrypt ::=     NO | <name>
#
#       rfc1449addr ::= tcp_ip_addr | osi_addr
#       tcp_ip_addr ::= <ip>/<#>
#       osi_addr ::= <nsap>/<tsel>
#       nsap ::= hexes
#       tsel ::= hexes
#       hexes = hexbyte[: hexbyte]*
#
#
ALLOW           SUBAGENT 1.3.6.1.4.1.1080.1.1.1
                WITH OTHER PASSWORD
                USE 15 SECOND TIMEOUT

COMMUNITY       public
                ALLOW GET OPERATIONS
                USE NO ENCRYPTION
```

15-second time-out

The `snmpd.conf` file template includes an `ALLOW` entry (near the end of the file, as shown above). This entry should be left intact. It gives **snmpd** a 15-second time-out for responses coming from **mib2d** and keeps **snmpd** active should **mib2d** hang.

/etc/syslog.conf

Network logging requires setting up a remote host on the administrative LAN to act as a **syslogd** server. This is the GRF `/etc/syslog.conf` file template:

```
# NetStar $Id$  syslog.conf,v 1.6.4.3 1997/09/15 14:57:37 pargal Exp $
#
# To enable the logging of system messages on GRF systems, edit the
# entries in the "Log messages to Network" section below.
#
#    - uncomment the lines in the "Log messages to Network" section
#       of this config file (just below these instructions)
#
#    - change "server.domain.com" to the name of a syslog server on
#       your local network to which this router may send log messages
#
#    - add the IP address of the log server and its host name to
#       /etc/hosts on this GRF system.
#
#    - run "grwrite -v" command to save your changes to
#       /etc/syslog.conf and /etc/hosts
#
#    - add the following lines to the /etc/syslog.conf on your log
#      server: (do not include the # from the first column of this
#      file, ie., add "local5.* /var/log/mib2d.log" not
#      "#local5.* /var/...")
#
#     # Syslog configuration for syslog server systems
#     # GRF-specific log files (from GRF systems over the network)
#     #
#     local0.info    /var/log/gritd.packets
#     local1.info    /var/log/gr.console
#     local2.*       /var/log/gr.boot
#     local3.*       /var/log/grinchd.log
#     local4.*       /var/log/gr.conferrs
#     local5.*       /var/log/mib2d.log
#     local6.*       /var/log/fred.log
#
#    - kill and restart syslogd on your syslog server machine after
#      making sure that the log files added to the config file exist
#      on the log server machine exist (use the "touch" command
#      to create them if they do not exist, then restart syslogd
#      on the syslog server machine).
#
#    - kill and restart syslogd on the GRF router (or reboot the GRF).
#
# To uncomment, remove "#net# from each line in this section

# Log messages to Network
#
#net#*.err;kern.debug;auth.notice;mail.crit        @server.domain.com
#net#*.notice;kern.debug;lpr,auth.info;mail.crit   @server.domain.com
```

```
#net#cron.info                                     @server.domain.com
#net#local0.info                                   @server.domain.com
#net#local1.info                                   @server.domain.com
#net#local2.*                                      @server.domain.com
#net#local3.*                                      @server.domain.com
#net#local4.*                                      @server.domain.com
#net#local5.*                                      @server.domain.com
#net#local6.*                                      @server.domain.com

#
---------------------------------------------------------------------
# GRF users need not read further - the following configuration
# examples are for IRMS systems
#
---------------------------------------------------------------------
#
# On IRMS systems (which have sufficient disk space) system messages
# may be logged to disk by uncommenting the lines in the
# "Log messages to disk"
# section below, touching the log file names to make sure they exist
# and then restarting syslogd.
#
# Note: using the "Log messages to Disk" entries to log messages to
# the RAM-based file system on a GRF system is strongly discouraged
# because the log files can easily fill the RAM file system.
#
# To uncomment, remove "#disk# from each line in this section

# Log messages to Disk
#
#disk#*.err;*.notice;kern.debug;lpr,auth.info;mail.crit var/log/mes-
sages
#disk#cron.info                                         /var/log/cron
#disk#local0.info                               /var/log/gritd.packets
#disk#local1.info                                  /var/log/gr.console
#disk#local2.*                                        /var/log/gr.boot
#disk#local3.*                                    /var/log/grinchd.log
#disk#local4.*                                    /var/log/gr.conferrs
#disk#local5.*                                      /var/log/mib2d.log
#disk#local6.*                                       /var/log/fred.log

*.notice;auth.debug     root
*.emerg                  *
```

Index