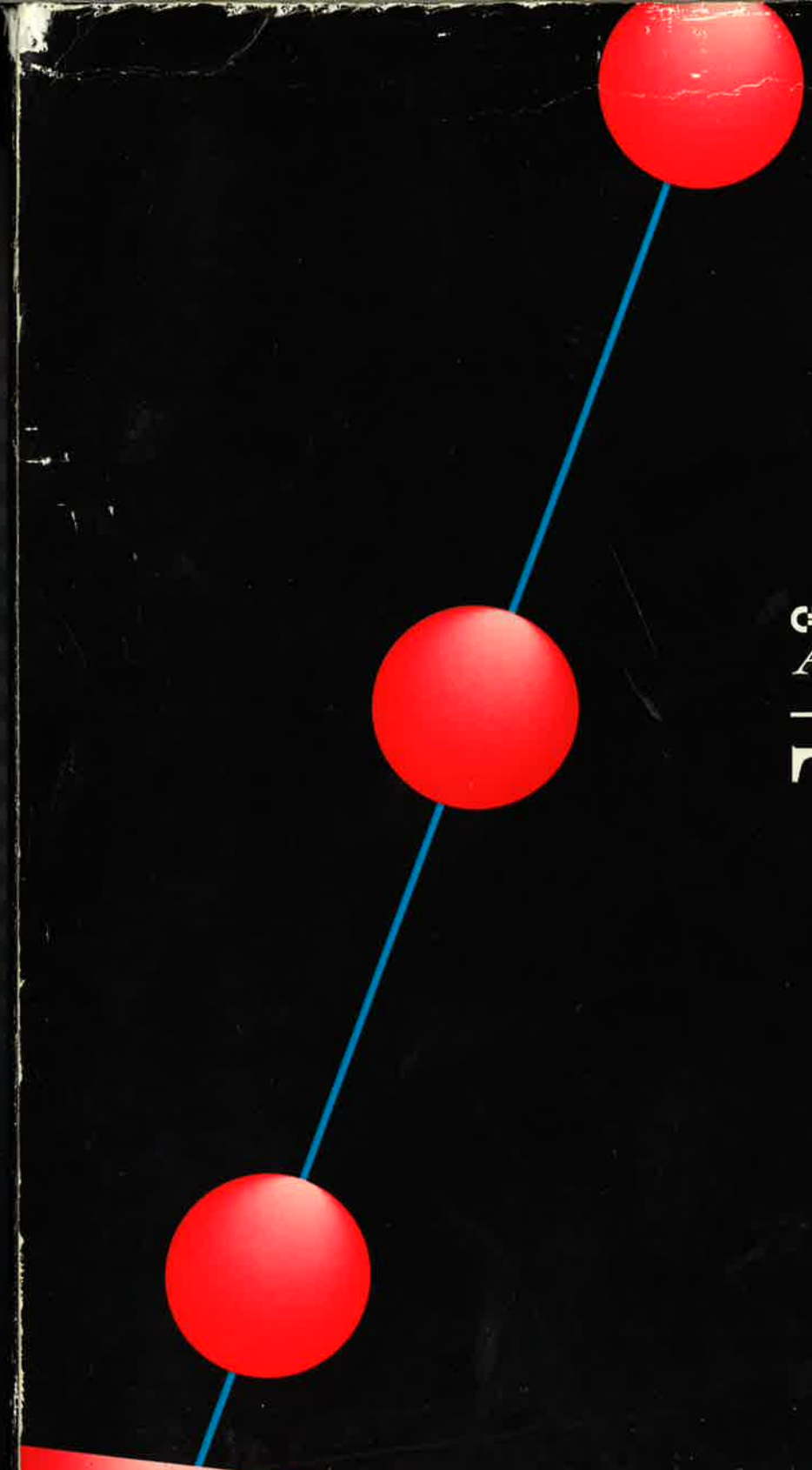


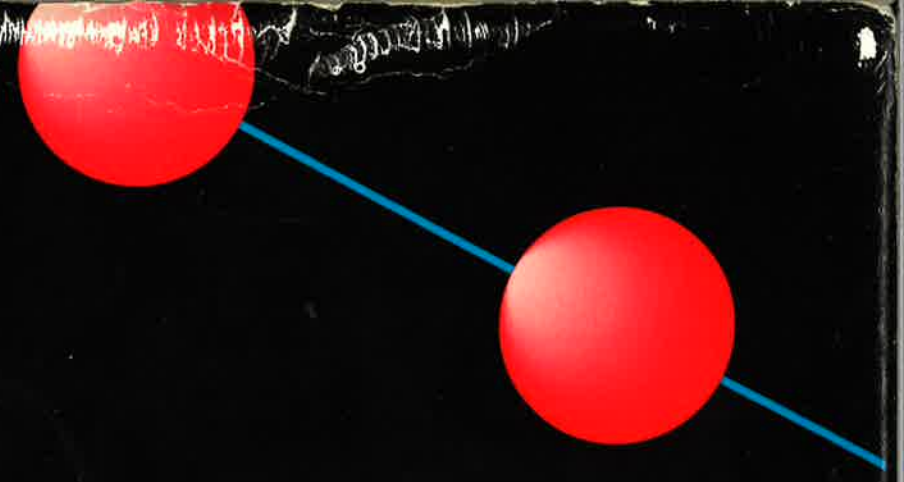
Commodore
AMIGA

TCP/IP

networking software

includes NFS® client software

requires:
Commodore A2065 Ethernet Card
AmigaDOS 1.3 or higher



TCP/IP

networking software

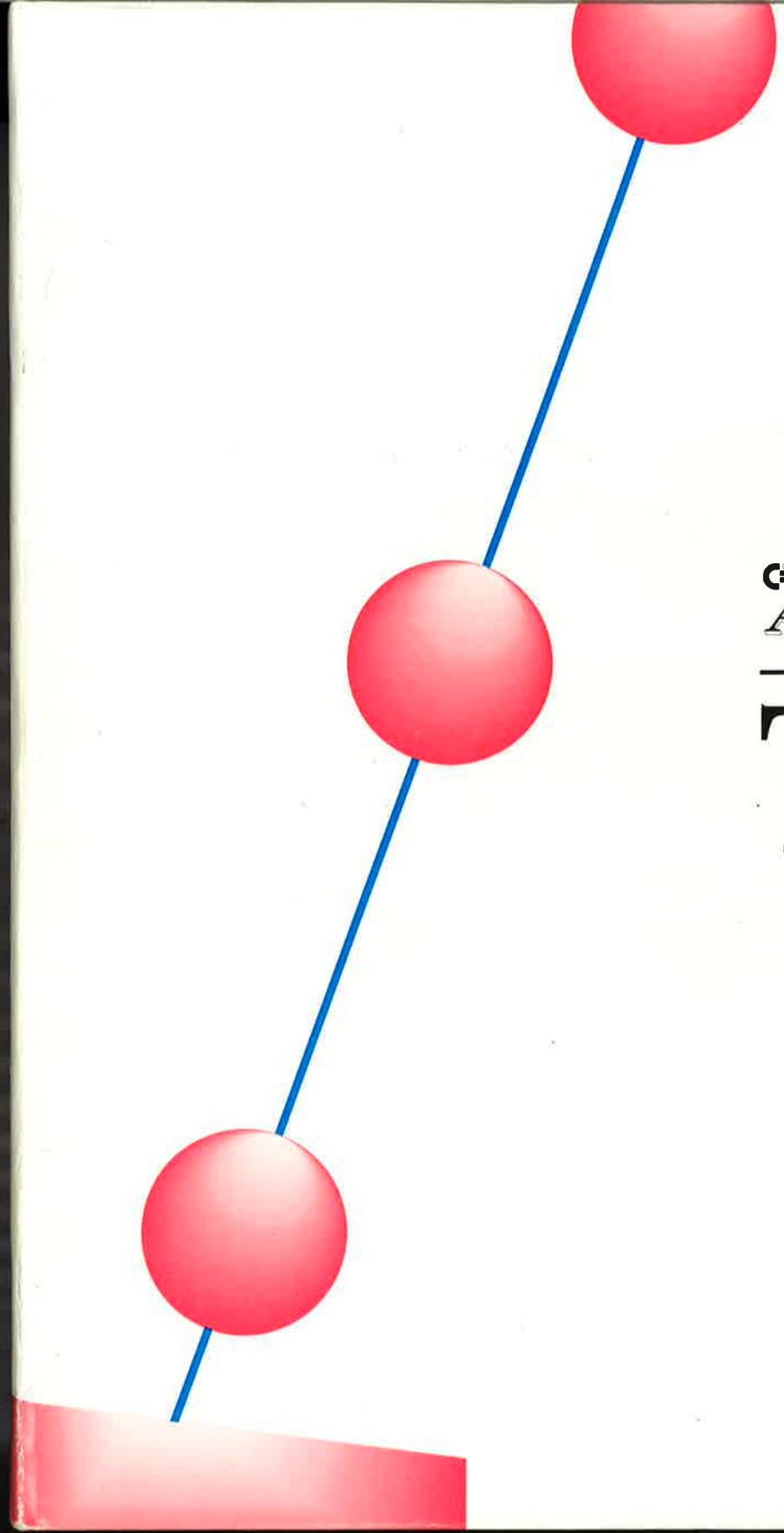
Features include:

- ★ *TCP/IP and UDP/IP protocol support for DoD and other TCP/IP network applications.*
- ★ *Network File System (NFS) client application support for Amiga client side file services from an NFS server.*
- ★ *Full host table support for symbolic lookup of internal address numbers.*
- ★ *VT100 network terminal emulation for remote access.*
- ★ *Supports Ethernet solutions based on both thick and thin Ethernet (Cheapernet).*
- ★ *Software utilities:*
  - *Filesystem automounting on system boot-up*
  - *Host table lookup file for symbolic ASCII hostname reference of internet address numbers.*
  - *Full local support for protection bit modification of NFS volume files and subdirectories.*

Commodore AMIGA

1200 Wilson Drive, West Chester, PA 19380

Part No. 318306-01



Commodore®
AMIGA®

TCP/IP

networking software

includes NFS® client software

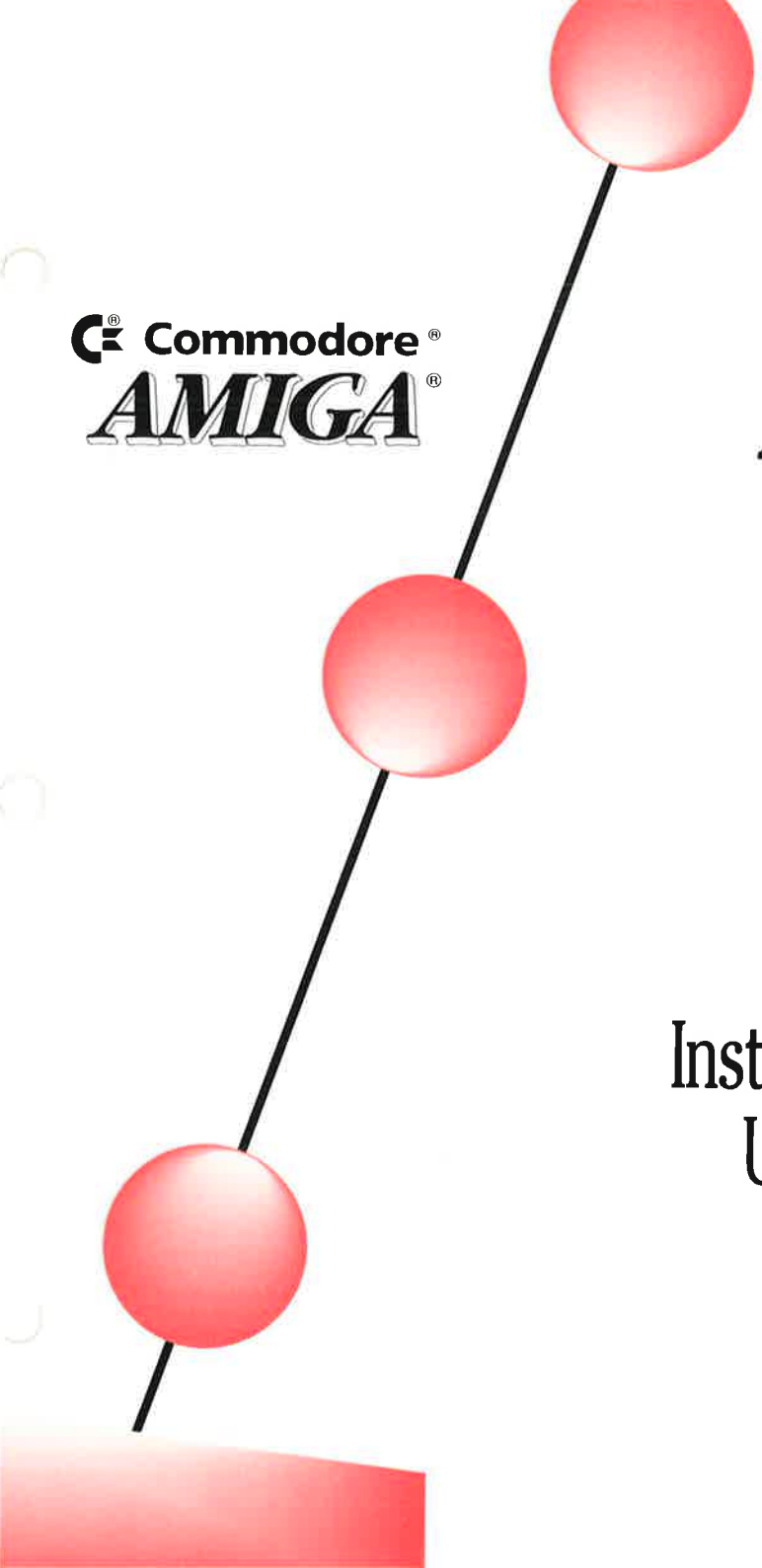
requires:
Commodore A2065 Ethernet Card
AmigaDOS 1.3 or higher

Commodore®
AMIGA®

AS225

TCP/IP Networking Software

Installation and User's Guide



First printing October 1990.

CONTENTS

INTRODUCTION

Networking makes it possible to share computing resources and services among different computing systems independent of hardware and operating systems. The Amiga TCP/IP Software allows you to implement TCP/IP (Transmission Control Protocol/Internet Protocol) and UDP/IP protocols, as well as a number of networking applications that use these protocols. This version of TCPIP is based in part on the Berkeley System Distribution (BSD), which provides enhanced internet support.

This document describes the installation and use of the Amiga Software. Installation of the software requires some knowledge of AmigaDOS at the CLI/SHELL level, and assistance from the local system administrator or site manager. For AmigaDOS information, consult the System Software manual that came with your Amiga.

Note

This manual assumes that you have experience with Ethernet networking and Unix.

Terms and Concepts

The following definitions lay the groundwork for the material presented throughout the remainder of this manual.

A *host* is a device which possesses an Ethernet interface. A *hostname* is an arbitrarily chosen character string which is used to refer to a host. A typical hostname might be athena.mit.edu or myamiga.sun.com or just amiga1.

An *Internet Address* (also known as an *Internet Number)* is a 32 bit binary number which is used by the Internet software to

uniquely identify every host interface. The Internet address 192.9.200.54 for instance represents a 32 bit binary integer whose first byte is 192, second byte is 9, third byte is 200 and fourth byte is 54. Every host interface is uniquely identified by a hostname and a corresponding Internet Address.

A *host table* is a file which contains a list of hostnames and their corresponding internet addresses. Many of the commands described in this manual require a host-address specification as a parameter. A host-address specification is simply a reference to a particular host by either its hostname or corresponding Internet Address.

A *gateway* is a host that has more than one Ethernet interface.

A *server* is a host which makes itself available to provide some form of service for other hosts. A host which obtains services from a server is known as a client.

Software Installation

After installing your network interface (hardware required to connect to the network), follow the procedures below to install the software.

Make sure you are running AmigaDOS version 1.3 or higher, and have the following information available from your system administrator:

1. Hostname and internet address of your Amiga.
2. Your username, userid and groupid.
3. User Mask. (umask)
4. Your time zone. (EST, PST, etc)
5. The hostname and internet address of at least one server.
6. If you are running in a class B network or have gateways, please have a site manager help you with information on subnet masks and broadcast addresses.

Note

Make backup copies of the disks supplied. For details on how to format or copy a disk see the documentation that came with your Amiga. After making backups, put the original disks away in a safe place and use the copy to install the package.

Using the Diagnostic Tools

The C directory on the distribution disk contains the programs *boards*, *lance-test*, and *vers* which may be executed from the CLI/Shell.

The *boards* program displays the manufacturer and serial numbers of any hardware which has been recognized by the Amiga auto-configuration routines. If the Ethernet controller has been installed, it should appear in the output list produced by the boards program.

The *lance-test* program performs hardware diagnostics on the card and displays its Ethernet address. It is invoked by typing **lance-test diags** at the prompt. Lance-test should be run before any network applications are executed (i.e. before start-inet), otherwise unpredictable results may occur.

The *vers* program accepts as an argument a filename and produces the creation date. This information is useful for bug reporting. There are more complex applications described in this manual for network debugging. See the sections on *ping* and *netstat* for more information.

Setting up an environment

To create a working environment you must copy the distribution software from both supplied disks onto corresponding directories on a hard disk or second floppy disk.

The following steps should be performed, assuming a system with a single floppy drive and a hard disk named WORK:

1. Make the directory **WORK:inet**. Do this by entering the command **MAKEDIR WORK:inet** at the CLI prompt. The process would be the same on a dual floppy system, but instead you would format a new disk on df1:, then enter **MAKEDIR df2:inet** at the CLI prompt.

2. Copy the software distribution, from both disks, from df0: into the corresponding directories on WORK: using the command df0:install, or run the "Install_Now" icon found on the first disk. The system will copy and create on WORK: the directories: **inet/c, inet/db, inet/devs, inet/log, inet/libs, inet/serv.**

   Run the "Read me" icon for more details.

3. In your startup-sequence you must then assign and add **inet:** to your path, and add a command to execute the script *start-inet*. This is done as follows:

   assign inet: WORK:inet
   path inet:c add
   execute inet:s/start-inet

4. Edit the **inet:db/hosts** file and add a hostname and internet address entry for any remote hosts and the Amiga. Note: you will only need an entry for a host which has the complete host table. You can then later copy the complete host table down to your Amiga.

5. Edit the **passwd** file and add an entry for each user. The fields are separated by |. You must have an entry for the userid and groupid for NFS to function properly. The passwd entry is also used by *ftp* and *finger*. The following is an example entry:

   bill||20|8|Bill McMan|WORK:bill|cli

In this case the user **bill** has a uid of **20** and gid of **8**. His home directory is **WORK:bill** and his shell is **CLI**. The username, uid and gid should be the same as his accounts on the remote hosts. Use the *passwd* program to set an encrypted password, or press return for the old password. The program will prompt the user for the new password.

To set up the environment on a two floppy-drive system, you will need the following files as a minimum:

all files in inet/s, inet/libs, and inet/log

from inet/c:
- **config, ifconfig** — to configure the system
- **nfsmgr, nfsc** — to use NFS
- **other commands** — as needed

from inet/serv:
- **inetd, portmapd** — to use servers
- **others** — as needed

> **Note**
>
> It is possible to install all the necessary files on a boot floppy and use this software on a single-floppy system.

What start-inet does

Start-inet is a script developed to remove some of the complexities involved in starting the various programs needed to use the network applications. Once you have completed setting up the environment, reboot the Amiga. The startup-sequence will execute start-inet and various initializations will take place. Start-inet may take the two optional arguments, *servers*, if you want to run the server daemons, or RIP if you are running in a gateway environment.

Start-inet gets its configuration information from the file **inet:s/inet.config.** This file may be edited with a text editor, or by typing **config -m.** If you edit it with a text editor, do not insert any spaces or comments.

A sample entry is:

```
uid=400
gid=10
umask=2
tz=300
tzname=EST
user=my-name
host=my-host
gateway=
configfile=
broadcast=192.1.200.255
subnetmask=255.255.255.0
gids=20
gids=30
gids=40
gids=60
```

uid	The userid of the Unix system to which you are connecting.
gid	Your primary group id; your files will have this gid.
umask	The default file protection mask. See chmod.
tz	The number of minutes your location is west of Greenwich Mean Time
tzname	The name of your time zone.
user	Your user name on the system to which you are connecting.
host	Your Amiga's host name. This must be in your **db/host** file.
gateway	The default gateway, if any.
configfile	Not used.
broadcast address	Consult your system administrator.
subnetmask	Consult your system administrator.
gids	Other groups of which you are a member. These must be listed with one each line.

Finishing the installation

You have now installed your software and hardware and should not have received any error messages. At this point it would be a good idea to download the complete host table from the server you included before. The host table is a file which contains the Internet address and corresponding hostname for every host on the network. The host table resides in the file **inet:db/hosts.** If the package is functioning properly, the host table may be downloaded using either the tftp or ftp applications described in the manual.

For example, to download from host 192.1.200.87 using the tftp application type **tftp 192.1.200.87** and when the tftp utility prompts for a command, type **get /etc/hosts inet: db/hosts.add.** If the tftp server is not available, it will be necessary to use the ftp application to download the host table. Once you have downloaded the new host table, add this to your original one by appending db/hosts.add to db/hosts with your favorite editor. (You don't want to overwrite the Amiga entry.)

The inet:db Directory

The db directory contains a number of configuration or database files which the network applications may require. The following list is broken down into two sections. The first section includes files you may have to modify, and the second section where files generally require no modification since they are generic.

inet:db/hosts

The file "hosts", also referred to as the "host table" is a file which contains the Internet address and corresponding hostname for every host on the network. A copy of this file should be on every host on the network. The file format is exactly the same as for UNIX. The format is as follows:

IP_number hostname nickname #comment

Where nickname is an alternate for hostname, and comment is optional.

The following line should always be included:
127.0.0.1 localhost

db/passwd

The *passwd* file is used by network applications like *ftp* and *finger* to validate users and provide a database of users on the host. This particular passwd file is only used when logging in to your Amiga. The passwd file format is slightly different than the Unix version, but the functionality is similar. See the passwd file for example users and their corresponding entry. Each user may add or modify an encrypted passwd by running the passwd program.

The format of the passwd file is as follows:
user|password|uid|gid|full name|home directory|cli

user = username

password = encrypted password. Leave blank when entering new users.
uid = user id
gid = group id
full name = the full name of the user
home directory = the directory this user will be placed in when he uses FTP to connect to this Amiga.
cli = not used. Just enter "cli"

A sample passwd file is

john|YMMjac[EJPX|246|10|John Doe|ram:|cli

guest||1|1|Guest|inet:|cli

In this example, user "guest" has no password. To create a password, use the passwd command.

inet:db/fstab

The *fstab* file contains a list of all filesystems which may be automatically mounted at boot time. You may wish to automate the mounting of nfs servers by placing an entry for each filesystem you wish to mount. During the execution of start-inet this file is automatically read. For information on nfs services, consult the nfsmgr section.

The contents of the fstab file is sent to the nfsmgr command. Look at start-inet and nfsmgr for details on how this is done. If nfsmgr encounters the end of file when it does not expect it, it will generate an error. This can cause misleading error messages, such as "syntax error in line 4". Just ignore these.

A sample fstab file is

```
mount bigunix:/users/softeng/john john
mount ghostwheel:/usr/commodore/amiga/D20 D20
mount smunix:/usr/tmp smunix
```

inet:db/hosts.equiv

This is a system file that lists the trusted hosts. If a machine is included in this list, then rlogin, rsh, rcp, etc. will be permitted freely from that machine to your machine.

Sample hosts.equiv file

```
amiga1
amiga2
vax
sun1
```

inet:db/exports

This system file lists which directories will be exported via NFS and which machines may access them. Not currently used.

inet:db/netrc

This is a user file that lists the name and passwords to automatically use when FTP'ing to a remote machine. This is always a security problem because account names and passwords are listed here in clear text.

Sample netrc file:

machine amiga1 login fred password derf
machine vax1 login fred password xyzzy
machine cray login fdd password foo*bar

inet:db/networks

The *networks* database maps a network name to its corresponding Internet address. You should add an entry for your network here.

System database and configuration files

WARNING: No changes should be made by user

inet:db/services

The services database is used to map between internet port numbers and well known symbolic ascii service names.

inet:db/inetd.conf

inet:Inetd.conf is a database used by *inetd*.

inet:db/protocols

The *protocols* database is used by *RPC* to map between symbolic ascii protocol names and well defined protocol numbers.

inet:db/rpc

The *rpc* file is a database to map between ascii rpc service names and well known program numbers. The *rpcinfo* command makes use of this file.

inet:log/utmp, log/wtmp, db/lastlog

These files collectively maintain login information. *Lastlog* contains a list of login events. It is used by some network applications such as ftp. The *wtmp* and *utmp* logs maintain information on all logins by all users. These files are generally updated and accessed by applications which generate or list information about users.

Network Applications

The Software Commands section describes the usage and options of the network applications. Generally, the applications may be divided into five types:

1. File sharing. The file sharing applications include *nfsmgr, rcp, ftp, and tftp.*
2. Network terminal. The remote terminal applications include *Rlogin* and *RLoginVT.*
3. Diagnostic. Diagnostics applications are *arp, route, portmapd, inetd, routed, fingerd, rshd, rpcinfo, ifconfig,* and *tftpd.*
4. System. The system related applications are *arp, route, portmapd, inetd, routed, fingerd, rshd, rpcinfo, ifconfig,* and *tftpd.*
5. Misc. Some other useful applications such as *chmod, ls, finger, showmount,* and *rsh* don't fit into any of the above categories.

COMMANDS

Introduction

> **Note**
>
> This guide assumes that you are familiar with Unix, Ethernet, AmigaDOS and the CLI/SHELL.

The following sections are organized like Unix manual pages. Each section can be used independently. The sections on the daemon processes (in system applications) are included for completeness but are not required reading for the average user.

Arp

Function

address resolution display and control

Usage arp hostname

arp -a
arp -d hostname
arp -s hostname ether_addr [temp] [pub]
arp -f filename

Description

The arp program displays and modifies the Internet-to-Ethernet address translation tables used by the address resolution protocol.

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

The -a flag displays all of the current ARP entries. If no entries exist, then the prompt will return.

With the -d flag, you may delete an entry for the host called hostname.

The -s flag creates an ARP entry for the host called hostname with the Ethernet address ether_addr. The Ethernet address is given as six hex bytes separated by colons. The entry will be permanent unless the word **temp** is given in the command. If the word **pub** is given, the entry will be published. This system will act as an ARP server, responding to requests for hostname even though the host address is not its own.

The -f flag 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 **hostname etheraddr [temp] [pub]** with argument meanings as given above.

Chmod

Function

Change NFS File Protections

Usage

chmod protection-specification object

Description

The CLI command **chmod protection-specification object** augments the functionality provided by the AmigaDOS PROTECT command. The chmod command can be used to modify any of the protection bits associated with a file or directory on an NFS volume.

The protection-specification may be a direct representation of the protection bits in the form of an octal number or may be specified symbolically. The following table defines the octal coding of the protection bits on a typical UNIX-based NFS server.

4000	set UID on execution
2000	set group-id on execution
1000	set sticky bit
400	read permission for owner
200	write permission for owner
100	execute permission for owner
070	read, write and execute permission bits for group
007	read, write and execute permission bits for others

The protection-specification 640, for instance, would grant the owner of the object read/write permission, members of the same group as the owner read-only permission and deny access completely to all others.

A symbolic protection-specification takes the form of one or more of the letters **u,g,o** or **a** followed by a **+**, **−** or **=** sign, followed by one or more of the letters **r, w** or **x.** The letters **u,g,o** and **a** specify the owner, group, others and all (owner group and others) respectively. A **+** sign is used to grant access, a - sign is used to disable access and an **=** sign is used to specify exact permissions. The letters **r, w** and **x** represent read, write and execute permission respectively.

For example, the command **chmod o-w nt0:myfile** would turn off write permission for others on the file nt0:myfile.

Finger

Function

user information lookup program

Usage

finger [-lp] [login1 [login2 . . .]]

Description

By default finger lists the login name, full name, terminal name and write status (as a * before the terminal name if write permission is denied), idle time, login time, and office location and phone number (if they are known) for each current user. (Idle time is minutes if it is a single integer, hours and minutes if a **:** is present, or days and hours if a **d** is present.)

A longer format also exists and is used by finger whenever a list of people's names is given. (Account names as well as first and last names of users are accepted.) This format is multi-line, and includes all the information described above as well as your home directory and login shell, any plan which the person has placed in the file **.plan** in their home directory, and the project on which they are working from the file .**project** also in the home directory.

Finger may be used to look up users on a remote machine. The format is to specify the user as userhost. If the user name is left off, the standard format listing is provided on the remote machine.

Options:

-l Force long output format.
-p Suppress printing of the .plan files

Fingerd

Function

remote user information server

Usage

Started from inetd

Description

Fingerd is a simple protocol based on RFC742 that provides an interface to the Name and Finger programs at several network sites. The program is supposed to return a friendly, human-oriented status report on either the system at the moment or a particular person in depth. There is no required format and the protocol consists mostly of specifying a single command line.

Fingerd listens for TCP requests at port 79. Once connected it reads a single command line terminated by a <CRLF> which is passed to finger. Fingerd closes its connections as soon as the output is finished.

If the line is null (just a <CRLF> is sent) then finger returns a default report that lists all people logged into the system at that moment. If a user name is specified (for example: eric<CRLF>) then the response lists more extended information for only that particular user, whether logged in or not. Allowable names in the command line include both login names and user names. If a name is ambiguous, all possible derivations are returned.

Ftp

Function

ARPANET file transfer program

Usage

ftp [-i] [-n] [-g] [host]

Description

Ftp is the user interface to the ARPANET standard File Transfer Protocol. The program allows a user to transfer files to and from a remote network site. The host with which ftp is to communicate may be specified on the command line. If this is done, ftp will immediately attempt to establish a connection to an FTP server on that host; otherwise, ftp will enter its command interpreter and await instructions from you. When ftp is awaiting commands the prompt **ftp>** is provided. The following options and commands are recognized by ftp:

Options

Options may be specified at the command line, or to the command interpreter.

The **-n** option restrains ftp from attempting auto-login upon initial connection. If auto-login is enabled, ftp will check the .netrc (see below) file in your home directory for an entry describing an account on the remote machine. If no entry exists, ftp will prompt for the remote machine login name (default is the user identity on the local machine), and, if necessary, prompt for a password and an account with which to login.

The **-i** option turns off interactive prompting during multiple file transfers.

The **-g** option disables file name globbing.

$ macro-name [args]

Execute the macro macro-name that was defined with the **macdef** command. Arguments are passed to the macro unglobbed.

account [passwd]

Supply a supplemental password required by a remote system for access to resources once a login has been successfully completed. If no argument is included, you will be prompted for an account password in a non-echoing input mode.

append local-file [remote-file]

Append a local file to a file on the remote machine. If remote-file is left unspecified, the local file name is used in naming the remote file after being altered by any **ntrans** or **nmap** setting. File transfer uses the current settings for type, format, mode, and structure.

ascii

Set the file transfer type to network ASCII. This is the default type.

bell

Arrange that a bell be sounded after each file transfer command is completed.

binary

Set the file transfer type to support binary image transfer.

bye

Terminate the FTP session with the remote server and exit ftp. An end of file will also terminate the session and exit.

case

Toggle remote computer file name case mapping during get commands. When case is on, remote computer file names with all letters in upper case are written in the local directory with the letters mapped to lower case. The default is case off.

cd remote-directory

Change the working directory on the remote machine to remote-directory.

cdup

Change the remote machine working directory to the parent of the current remote machine working directory.

close

Terminate the FTP session with the remote server, and return to the command interpreter. Any defined macros are erased.

cr

Toggle carriage return stripping during ascii type file retrieval. Records are denoted by a carriage return/linefeed sequence during ascii type file transfer. When cr is on, carriage returns are stripped from this sequence to conform with the UNIX single linefeed record delimiter. Records on non-UNIX remote systems may contain single linefeeds; when an ascii type transfer is made, these linefeeds may be distinguished from a record delimiter only when cr is off. The default is cr on.

delete remote-file

Delete the file **remote-file** on the remote machine.

debug

Toggle debugging mode. When debugging is on, ftp prints each command sent to the remote machine, preceded by the string —> .

dir [remote-directory] [local-file]

Print a listing of the directory contents in the directory **remote-directory,** and optionally place the output in the local-file. If interactive prompting is on, ftp will prompt you to verify that the last argument is indeed the target local file for receiving dir output. If no directory is specified, the current working directory on the remote machine is used. If no local file is specified, output is displayed in the current Shell window.

disconnect

A synonym for **close.**

form format

Set the file transfer form to format. The default form is file.

get remote-file [local-file]

Retrieve the remote-file and store it on the local machine. If the local file name is not specified, it is given the same name it has on the remote machine, subject to alteration by the current case, ntrans, and nmap settings. The current settings for type, form, mode, and structure are used while transferring the file.

glob

Toggle filename expansion for **mdelete, mget** and **mput.** If globbing is turned off with glob, the file name arguments are taken literally and not expanded. Globbing for mput is done as in csh. For mdelete and mget, each remote file name is expanded separately on the remote machine and the lists are not merged. Expansion of a directory name is likely to be different from expansion of the name of an ordinary file: the exact result depends on the foreign operating system and ftp server, and can be previewed by doing **mls remote-files -** .

> **Note**
>
> mget and mput are not meant to transfer entire directory subtrees of files.

hash

Toggle hash-sign **(#)** printing for each data block transferred. The size of a data block is 1024 bytes.

help [command]

Print an informative message about the meaning of command. If no argument is given, ftp prints a list of the known commands.

image [command]

A synonym for binary.

lcd [directory]

Change the working directory on the local machine. If no directory is specified, your home directory is used.

ls [remote-directory] [local-file]

Print a listing of the contents of a directory on the remote machine. The listing includes any system dependent information that the server chooses to include. For example, most UNIX systems will produce output from the command **ls -l.** (See also nlist.) If remote-directory is left unspecified, the current working directory is used. If interactive prompting is on, ftp will prompt you to verify that the last argument is indeed the target local file for receiving ls output. If no local file is specified, or if local-file is -, the output is displayed in the current Shell window.

macdef macro-name

Define a macro. Subsequent lines are stored as the macro macro-name; a null line (2 consecutive newline characters in a file or carriage returns from the terminal) terminates macro input mode. There is a limit of 16 macros and 4096 total characters in all defined macros. Macros remain defined until a close command is executed.

The macro processor interprets **$** and \ as special characters. A **$** followed by a number (or numbers) is replaced by the corresponding argument on the macro invocation command line. A **$** followed by an **i** signals that the executing macro is to be looped. On the first pass **$i** is replaced by the first argument on the macro invocation command line, on the second pass it is replaced by the second argument, and so on. A \ followed by any character is replaced by that character. Use the \ to prevent special treatment of the **$.**

mdelete [remote-files]

Delete the remote-files on the remote machine.

mdir remote-files [local-file]

Like dir, except multiple remote files may be specified. If interactive prompting is on, ftp will prompt you to verify that the last argument is indeed the target local file for receiving mdir output.

mget remote-files

Expand the remote-files on the remote machine and do a get for each file name thus produced. See **glob** for details on the filename expansion. Resulting file names will then be processed according to case, ntrans, and nmap settings. Files are transferred into the local working directory, which can be changed with lcd directory.

mkdir directory-name

Make a directory on the remote machine.

mls remote-files local-file

Like nlist, except multiple remote files may be specified, and the local-file must be specified. If interactive prompting is on, ftp will prompt you to verify that the last argument is indeed the target local file for receiving mls output.

mode [mode-name]

Set the file transfer mode to mode-name. The default mode is stream. No other modes are supported.

modtime file-name

Show the last modification time of the file on the remote machine. Inet ftp can give this command, but cannot reply to it.

mput local-files

Expand wild cards in the list of local files given as arguments and do a put for each file in the resulting list. See glob for details of filename expansion. Resulting file names will then be processed according to ntrans and nmap settings.

nlist [remote-directory] [local-file]

Print a list of the files of a directory on the remote machine. If remote-directory is left unspecified, the current working directory is used. If interactive prompting is on, ftp will prompt you to verify that the last argument is indeed the target local file for receiving nlist output. If no local file is specified, or if local-file is -, the output is displayed in the current Shell window.

open host [port]

Establish a connection to the specified host FTP server. An optional port number may be supplied; ftp will attempt to contact an FTP server at that port. If the auto-login option is on, ftp will also attempt to automatically log you in to the FTP server (see below). The default is auto-login.

prompt

Toggle interactive prompting. Interactive prompting occurs during multiple file transfers so you can selectively retrieve or store files. If prompting is turned off, any mget or mput will transfer all files, and any mdelete will delete all files. The default is prompt on.

put local-file [remote-file]

Store a local file on the remote machine. If remote-file is left unspecified, the local file name is used after processing according to any ntrans or nmap settings in naming the remote file. File transfer uses the current settings for type, format, mode, and structure.

pwd

Print the name of the current working directory on the remote machine.

quit

A synonym for **bye.**

quote arg1 arg2 . . .

The arguments specified are sent, verbatim, to the remote FTP server.

recv remote-file [local-file]

A synonym for **get.**

rhelp [command-name]

Request help from the remote FTP server. If a command-name is specified it is supplied to the server as well.

rstatus [file-name]

With no arguments, show status of remote machine. If file-name is specified, show status of file-name on remote machine.

rename from to

Rename the **from** file on the remote machine, to the **to** file.

reset

Clear reply queue. This command re-synchronizes command/reply sequencing with the remote ftp server. Re-synchronization may be necessary following a violation of the ftp protocol by the remote server.

rmdir directory-name

Delete a directory on the remote machine.

runique

Toggle storing of files on the local system with unique filenames. If a file already exists with a name equal to the target local filename for a **get** or **mget** command, **.1** is appended to the name. If the resulting name matches another existing file, **.2** is appended to the original name. If this process continues up to **.99,** an error message will be printed, and the transfer does not take place. The generated unique filename will be reported. Note that runique will not affect local files generated from a shell command (see below). The default value is off.

send local-file [remote-file]

A synonym for **put.**

sendport

Toggle the use of PORT commands. By default, ftp will attempt to use a PORT command when establishing a connection for each data transfer. The use of PORT commands can prevent delays when performing multiple file transfers. If the PORT command fails, ftp will use the default data port. When the use of PORT commands is disabled, no attempt will be made to use PORT commands for each data transfer. This is useful for certain FTP implementations which do ignore PORT commands but incorrectly indicate they've been accepted.

size file-name

Return the size of file-name on the remote machine.

status

Show the current status of ftp.

struct struct-name

Set the file transfer structure to struct-name. By default stream structure is used.

sunique

Toggle storing of files on remote machine under unique file names. The remote ftp server must support the ftp protocol STOU command for successful completion.

system

Show the type of operating system running on the remote machine.

tenex

Set the file transfer type to that needed to talk to TENEX machines.

trace

Toggle packet tracing.

type [type-name]

Set the file transfer type to type-name. If no type-name is specified, the current type is printed. The default type is network ASCII.

user user-name [password] [account]

Identify yourself to the remote FTP server. If the password is not specified and the server requires it, ftp will prompt you for it (after disabling local echo). If an account field is not specified, and the FTP server requires it, you will be prompted for it. If an account field is specified, an account command will be relayed to the remote server after the login sequence is completed if the remote server did not require it for logging in. Unless ftp is invoked with auto-login disabled, this process is done automatically on initial connection to the FTP server.

verbose

Toggle verbose mode. In verbose mode, all responses from the FTP server are displayed to you. In addition, if verbose is on, when a file transfer completes, statistics regarding the efficiency of the transfer are reported. The default is verbose on.

? [command]

A synonym for **help.** Command arguments which have embedded spaces may be quoted with quote (") marks.

Aborting a file transfer

To abort a file transfer, use the terminal interrupt key (usually Ctrl-C). Sending transfers will be immediately halted. Receiving transfers will be halted by sending an ftp protocol ABOR command to the remote server, and discarding any further data received. The speed at which this is accomplished depends upon the remote server's support for ABOR processing. If the remote server does not support the ABOR command, an ftp> prompt will not appear until the remote server has completed sending the requested file. The terminal interrupt key sequence will be ignored when ftp has completed any local processing and is awaiting a reply from the remote server. A long delay in this mode may result from

the ABOR processing described above, or from unexpected behavior by the remote server, including violations of the ftp protocol. If the delay results from unexpected remote server behavior, the local ftp program must be aborted manually.

File Transfer Parameters

The FTP specification defines many parameters which may affectf a file transfer. The type may be one of **ascii, image (binary), ebcdic,** and **local byte size** (for PDP- 10's and PDP-20's). Ftp supports the ascii and image types of file transfer, plus local byte size 8 for tenex mode transfers. Ftp supports only the default values for the remaining file transfer parameters: mode, from, and struct.

The .netrc File

The .netrc file contains login and initialization information used by the auto-login process. The following tokens are recognized (they may be separated by spaces, tabs, or newlines):

machine name

Identify a remote machine name. The auto-login process searches the .netrc file for a machine token matching the remote machine specified on the ftp command line or as an open command argument. Once a match is made, the subsequent .netrc tokens are processed, stopping when the end of file is reached or another machine or a default token is encountered.

default

This is the same as **machine name** except that **default** matches any name. There can be only one default token, and it must be after all machine tokens. This is normally used as: **default login anonymous password usersite** thereby giving you automatic anonymous ftp

login to machines not specified in .netrc. This can be overridden by using the **-n** flag to disable auto-login.

login name

Identify a user on the remote machine. If this token is present, the auto-login process will initiate a login using the specified name.

password string

Supply a password. If this token is present, the auto-login process will supply the specified string if the remote server requires a password as part of the login process.

account string

Supply an additional account password. If this token is present, the auto-login process will supply the specified string if the remote server requires an additional account password, or the auto-login process will initiate an ACCT command if it does not.

macdef name

Define a macro. This token functions like the ftp **macdef** command functions. A macro is defined with the specified name; its contents begin with the next .netrc line and continue until a null line (2 or more consecutive new-line characters) is encountered. If a macro named, **init** is defined, it is automatically executed as the last step in the auto-login process.

Ftpd

Function

DARPA Internet File Transfer Protocol server

Usage

Started from inetd

Description

Ftpd is the DARPA Internet File Transfer Protocol server process. The server uses the TCP protocol and listens at the port specified in the ftp service specification (see services). Ftpd should be run only after running **inetd** and **portmapd.**

The ftp server will abort an active file transfer only when the ABOR command is preceded by a Telnet Interrupt Process (IP) signal and a Telnet Synch signal in the command Telnet stream, as described in Internet RFC 959. If a STAT command is received during a data transfer, preceded by a Telnet IP and Synch, transfer status will be returned.

Ftpd authenticates users according to rules:

1. The user name must be in the password data base, passwd, and not have a null password. In this case a password must be provided by the client before any file operations may be performed.
2. If the user name is anonymous or ftp, an anonymous ftp account must be present in the password file (user ftp). In this case you are allowed to log in by specifying any password (by convention this is given as the client host's name).

The ftp server currently supports the following ftp requests (case is not distinguished).

Request	*Description*
ABOR	abort previous command
ACCT	not implemented
ALLO	allocate storage (vacuously)
APPE	append to a file
CDUP	change to parent of current working directory
CWD	change working directory
DELE	delete a file
HELP	give help information
LIST	give list files in a directory
MDTM	not implemented
MKD	make a directory
MODE	specify data transfer mode
NLST	give name list of files in directory
NOOP	do nothing
PASS	specify password
PASV	prepare for server-to-server transfer
PORT	specify data connection port
PWD	print the current working directory
QUIT	terminate session
RETR	retrieve a file
RMD	remove a directory
RNFR	specify rename-from file name
RNTO	specify rename-to file name
SITE	not implemented
SIZE	not implemented
STAT	not implemented
STOR	store a file
STOU	store a file with a unique name
STRU	specify data transfer structure
SYST	not implemented
TYPE	specify data transfer type
USER	specify user name

XCUP	change to parent of current working directory (deprecated)
XCWD	change working directory (deprecated)
XMKD	make a directory (deprecated)
XPWD	print the current working directory (deprecated)
XRMD	remove a directory (deprecated)

Ifconfig

Function

configure network interface parameters

Usage

ifconfig interface [address [destaddress][parameters]
ifconfig interface [protocol_family]

Description

Ifconfig is used to assign an address to a network interface and/or configure network interface parameters. Ifconfig must be used at boot time (see start-inet) to define the network address of each interface present on a machine; it may also be used at a later time to redefine an interface's address or other operating parameters. The interface parameter is a string of the form name unit, for example: ae0.

For the DARPA-Internet family, the address is either a hostname present in the hostname database **hosts,** or a DARPA Internet address expressed in the Internet standard dot notation. The host number may be omitted on 10Mb/s Ethernet interfaces, which use the hardware physical address, and on interfaces other than the first. Ifconfig displays the current configuration for a network interface when no optional parameters are supplied. If a protocol family is specified, ifconfig will report only the details specific to that protocol family.

Parameters:

up

Mark an interface up. This may be used to enable an interface after an ifconfig **down.** It happens automatically when setting the first address on an interface. If the interface was reset when previously marked down, the hardware will be re-initialized.

down

Mark an interface down. When an interface is marked down, the system will not attempt to transmit messages through that interface. If possible, the interface will be reset to disable reception as well. This action does not automatically disable routes using the interface.

trailers

Note
Trailers are not currently supported.

Request 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, (currently, only 10 Mb/s Ethernet), 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

Disable the use of a "trailer" link level encapsulation.

arp

Enable the use of the Address Resolution Protocol in mapping between network level addresses and link level addresses (default). This is currently implemented for mapping between DARPA Internet addresses and 10Mb/s Ethernet addresses.

-arp

Disable the use of the Address Resolution Protocol.

metric n

Set the routing metric of the interface to n, default 0. The routing metric is used by the routing protocol. Higher metrics have the effect of making a route less favorable; metrics are counted as addition hops to the destination network or host.

netmask mask

Specify 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. The mask can be specified as a single hexadecimal number with a leading **0x,** with a dot-notation Internet address, or with a pseudo-network name listed in the network table networks. The mask contains **1's** for the bit positions in the 32-bit address which are to be used for the network and subnet parts, and **0's** for the host part. The mask should contain at least the standard network portion, and the subnet field should be contiguous with the network portion.

dest_address

Specify the address of the correspondent on the other end of a point to point link.

broadcast

Specify the address to use to represent broadcasts to the network. The default broadcast address is the address with a host part of all **1's.**

Inetd

Function

internet "super-server"

Usage

Started from inet:c/start-inet servers
You must specify servers as argument to start-inet.

Description

Inetd should be run at boot time. It then listens for connections on certain internet sockets. When a connection is found on one of its sockets, it decides what service the socket corresponds to, and invokes a program to service the request. After the program finishes, it continues to listen on the socket (except in some cases which will be described below). Essentially, inetd allows running one daemon to invoke several others, reducing load on the system.

Upon execution, inetd reads its configuration information from a configuration file which, by default, is **inet:db/inetd.conf.** There must be an entry for each field of the configuration file, with entries for each field separated by a tab or a space. Comments are denoted by a **#** at the beginning of a line. There must be an entry for each field. The fields of the configuration file are as follows:

service name — The service name entry is the name of a valid service in the file **inet:db/services.** For internal services (discussed below), the service name must be the official name of the service (the first entry in inet:db/services).

socket type The socket type should be one of **stream, dgram, raw, rdm,** or **seq-packet,** depending on whether the socket is a stream, datagram, raw, reliably delivered message, or sequenced packet socket.

protocol The protocol must be a valid protocol as given in **inet:db/protocols.** Examples might be tcp or udp.

wait/nowait The wait/nowait entry is applicable to datagram sockets only (other sockets should have a nowait entry in this space). If a datagram server connects to its peer, freeing the socket so inetd can receive further messages on the socket, it is said to be a multi-threaded server, and should use the "nowait" entry. For datagram servers which process all incoming datagrams on a socket and eventually time out, the server is said to be single-threaded and should use a wait entry. **Comsat (biff)** and **talk** are both examples of the latter type of datagram server. **Tftpd** is an exception; it is a datagram server that establishes pseudo-connections. It must be listed as wait in order to avoid a race condition; the server reads the first packet, creates a new socket, and then forks and exits to allow inetd to check for new service requests to spawn new servers.

user The user entry should contain the name of the user as whom the server should run.

server program The server program entry should contain the pathname of the program which is to be executed by inetd when a request is found on its socket. If inetd provides this service internally, this entry should be **internal.** Inetd provides several trivial services internally by use of routines within itself. These services are **echo, discard, chargen (character generator), daytime (human readable time),** and **time (machine readable time, in the form of the number of seconds since midnight, January 1, 1900).** All of these services are tcp based. For details of these services, consult the appropriate RFC from the Network Information Center.

Ls

Function

List directory

Usage

ls [-ld] [name1 name2 name3..]

ls -? (help)

Description

The CLI command ls [-ld] [name1 name2 name3 . . .] is used to obtain more information about a file or directory on an NFS volume than that provided by the AmigaDOS LIST command. The command may be invoked with a combination of one letter options which control the output format and amount of information displayed. The remaining parameters represent the names of files or directories. In the absence of any options, the named files or directories are listed in a tabular format with the filenames sorted alphabetically. The contents of each directory are similarly listed.

The -d option causes only information about the named directories to be displayed. The contents of any named directories are not listed.

The -l option outputs complete information about each named file or directory. The -l option is generally useful when one wishes to obtain complete information about a file or directory residing on an NFS server.

Many other options are supported. Type the command "ls-?" to see a description.

Ls uses the Unix wildcards * and ?. Amiga pattern matching is not supported.

Netstat

Function

show network status

Usage

netstat [-Aan] [-f address_family] [system]
netstat [-himnrs] [-f address_family] [system]
netstat [-p protocol] [system]

Description

The netstat command symbolically displays the contents of various network-related data structures. There are a number of output formats, depending on the options for the information presented. The first form of the command displays a list of active sockets for each protocol. The second form presents the contents of one of the other network data structures according to the option selected. Using the third form, with an interval specified, netstat will continuously display the information regarding packet traffic on the configured network interfaces. The fourth form displays statistics about the named 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.

-i

Show the state of interfaces which have been autoconfigured (interfaces statically configured into a system, but not located at boot time are not shown).

-I interface

Show information only about this interface; used with an interval as described below.

-m

Show statistics recorded by the memory management routines (the network manages a private pool of memory buffers).

-n

Show network addresses as numbers (normally netstat interprets addresses and attempts to display them symbolically). This option may be used with any of the display formats.

-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 **inet:db/protocols.** A null response typically means that there are no interesting numbers to report. The program will complain if protocol is unknown or if there is no statistics routine for it.

-s

Show per-protocol statistics.

-r

Show the routing tables. When **-s** is also present, show routing statistics instead.

-f address_family

Limit statistics or address control block reports to those of the specified address family. The default display for active sockets shows the local and remote addresses, send and receive queue sizes (in bytes), protocol, and the internal state of the protocol. Address formats are of the form **host.port** or **network.port** if a socket's address specifies a network but no specific host address. When known, the host and network addresses are displayed symbolically according to the data bases **inet:db/hosts and inet:db/networks,** respectively. If a symbolic name for an address is unknown, or if the **-n** option is specified, the address is printed numerically, according to the address family. Unspecified, or wildcard, addresses and ports appear as *.

The **interface display** provides a table of cumulative statistics regarding packets transferred, errors, and collisions. The network addresses of the interface and the maximum transmission unit (mtu) are also displayed.

The **routing table display** indicates the available routes and their status. Each route consists of a destination host or network and a gateway to use in forwarding packets. The flags field shows the state of the route (U if up), whether the route is to a gateway (G), whether the route was created dynamically by a redirect (D), and whether the route has been modified by a redirect (M).

Direct routes are created for each interface attached to the local host; the gateway field for such entries shows the address of the outgoing interface. The **refcnt** field gives the current number of active uses of the route. Connection-oriented protocols normally hold on to a single route for the duration of a connection while protocols without a connection obtain a route while sending to the same destination. The use field provides a count of the number of packets sent using that route. The interface entry indicates the network interface utilized for the route.

Nfsmgr

Function

nfs client application

Usage

nfsmgr [cmd arguments]

Description

Nfsmgr is the nfs client interface application which provides the Amiga with client side file services from a nfs server. Nfsmgr is a front end to the program *nfsc* which is the actual nfs client. All nfsmgr commands are in the form **nfsmgr cmd arguments.** Each command has its specific form as follows:

nfsmgr stats client [volume]

The stats commands prints the usage statistics for all mounted client NFS partitions by default. If a volume is specified then information on that volume is displayed

nfsmgr mount [host:partition [volume] [timeout=n] [retry=n] [maxread=n] [port] [maxwrite=n] [-i filename] [case]

The mount command mounts the specified partition. If no partition is given then the file **inet:db/fstab** is read to determine which partitions to mount. When specified with the -i option the argument filename is read to determine cmds.

The mount command allows various NFS client parameters to be modified:

retry — time between successive retries
timeout — total timeout period for any operation

case to enable case-sensitivity. This is useful particularly when mounting volumes from a UNIX machine.

maxread — maximum NFS read size the daemon will use

maxwrite — maximum NFS write size the daemon will use

port — use specified port to send NFS packets

An example mount to host UNIX5 would be in the form:

1> nfsmgr mount UNIX5:/usr rdisk

This would mount the directory **/usr** on the remote host UNIX5 as the local Amiga volume name **rdisk.**

> **Note**
>
> Mount attempts may fail if the remote host fails to export the desired filesystem, or you cannot be authenticated on the remote host.

nfsmgr unmount [all | volname]

The **unmount** command unmounts the given volume or all volumes when the keyword all is given. For example:

2> nfsmgr unmount rdisk

would unmount the remotely mounted directory **/usr** and volume name **rdisk**. Note that you must **cd** out of the remote filesystem to unmount using this command.

nfsmgr quit client

The quit command instructs the NFS client to clean up and exit if possible. The client daemon will refuse unless all volumes are unmounted and any locks/files controlled by it are shut down.

nfsmgr kill client

The kill command tells the client to clean up and exit regardless of the state of any locks/files they control. This command is considered dangerous as application processes may send handler packets to the killed client daemon and thus cause a system failure.

Passwd

Function

Change a user's password

Usage

passwd user

Description

The passwd command prompts you for a new password. This password is then encrypted and placed in the inet:db/passwd file.

Ping

Function

send ICMP ECHO_REQUEST packets to network hosts

Usage

ping [-drv] host [datasize] [npackets]

Description

The DARPA Internet is a large and complex aggregation of network hardware, connected together by gateways. Tracking a single-point hardware or software failure can often be difficult. Ping utilizes the ICMP protocol's mandatory ECHO_REQUEST datagram to elicit an ICMP ECHO_RESPONSE from a host or gateway. ECHO_REQUEST datagrams (pings) have an IP and ICMP header, followed by a struct timeval, and then an arbitrary number of pad bytes used to fill out the packet. Default datagram length is 64 bytes, but this may be changed using the command-line option. Other options are:

-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).

-v

Verbose output. ICMP packets other than ECHO RESPONSE that are received are listed.

When using ping for fault isolation, it should first be run on the local host to verify that the local network interface is up and running. Hosts and gateways further away should then be tested. Ping sends one datagram per second, and prints one line of output for every ECHO_RESPONSE returned. No output is produced if there is no response. If an optional count is given, only that number of requests is sent. Round-trip times and packet loss statistics are computed. When all responses have been received or the program times out (with a count specified), or if the program is terminated with a control-c, a brief summary is displayed.

This program is intended for use in network testing, measurement and management. It should be used primarily for manual fault isolation. Because of the load it could impose on the network, it is unwise to use ping during normal operations or from automated scripts.

Rcp

Function

remote file copy

Usage

rcp [-p] file1 file2
rcp [-p] [-r] file . . directory

Description

Rcp copies files between machines. Each file or directory argument is either a remote file name of the form **rhost =path,** or a local file name.

If the **-r** option is specified and any of the source files are directories, rcp copies each subtree rooted at that name; in this case the destination must be a directory. By default, the mode and owner of file2 are preserved if it already exists; otherwise the mode of the source file modified by the umask on the destination host is used.

The **-p** option causes rcp to attempt to preserve (duplicate) in its copies the modification times and modes of the source files, ignoring the umask. A path on a remote host may be quoted (using \, ", or ') so that the metacharacters are interpreted remotely. Rcp does not prompt for passwords; your current local user name must exist on rhost and allow remote command execution via rsh. The following example illustrates the usage of rcp:

1>rcp UNIX5=/usr/junk WORK:

Would copy the file junk located on host UNIX5 to WORK:. The Amiga must be a trusted host in order to have access to files.

Note
rcp between two remote machines is not implemented.

Rlogin/RloginVT

Function

rlogin—remote login
rloginVT—remote login with VT100 emulation

Usage

switch options: -1cxwyhsr?

switches work in upper or lower case

rlogin host [-1 user_id] [-x width] [-y height] [-r] [-c]

switches can be used in any order Except the HOST field which must be first. A space MUST go between a switch and its respective parameter (if any).

Example command lines:

run rlogin cbmvax -1 myname -c

run rlogin cbmvax -x 400 -y 300 -1 myname
-or-
run rlogin cbmvax -w 400 -h 300 -1 myname

run rlogin?

The new switches allow you to set the X and Y window sizes of the rlogin window (in pixels) under the following restrictions:

1) You must be on the WorkBench screen (for now.)
2) The values given as parameters must fall into the range of

Width: (fontwidth * 81) <= parameter <= screenwidth

Height: 200 <= parameter <= screenheight

The minimum window WIDTH is calculated from your font width to always allow a MINIMUM of 80 columns to be displayed (Unless, of course, fontwidth * 80 is greater than your screen's width.) Of course, you can fool this by using a font with a width that can overrun the display. All you will get is ugliness. The lines will wrap and the host will think they didn't.

The font used for these calculations is the font in the Work-Bench screen.

The minimum window HEIGHT when you OPEN rlogin is simply 200 pixels. With an 8 pixel high font this gives 25 rows. If you use a taller font, you can set your window height by using the -y switch. Minimum window height as you SIZE it is 100.

If you want to guarantee that the window opens to maximum size just set the -x and -y flags to a value larger than you screen's x and/or y dimensions. (-x 2000 -y 2000).

Description

The rlogin command connects your terminal on the current local host system **lhost,** to the remote host system **rhost**.

RloginVT emulates a VT100 and uses the Amiga console. When using rlogin you must have an Amiga termcap entry.

Each host has a file **inet:/etc/hosts.equiv** which contains a list of rhosts with which it shares account names. The host names must be the standard names as described in **rsh(1c).** When you use the rlogin command to login as the same user on an equivalent host, you do not need to specify a password.

You can also have a private equivalence list in a file named **.rhosts** in your login directory. Each line in this file should contain the rhost name and a username separated by a space, giving additional cases where logins without passwords are permitted. If the originating user is not equivalent

to the remote user, then the remote system prompts for a login and a password.

To avoid security problems, the .rhosts file must be owned by either the remote user or root, and it may not be a symbolic link.

Options:

?	Displays the usage message.
-l id	Your login id
-c	Opens rlogin on a CUSTOM screen. The default is to open on the WorkBench screen to the minimum size required for a 80 col × 24 row display—based upon the WorkBench screen's current font. If the 'minimum' window size horizontal and/or vertical is greater than your WorkBench screens respective horizontal and/or vertical sizes, the screen size is used. So, if you are using a font that will not give you a full 80 characters on a full width window, well . . . The -c flag overrides the default.
-x XXX -w XXX	Sets the horizontal window size to XXX pixels with the restrictions listed above in '-c'.
-y XXX -h XXX	Sets the vertical window size to XXX pixels with the restrictions listed above in '-c'.
-r -s	Attaches a window RESIZE gadget to the window. If things are working correctly, the host should recognize window size changes (height changes, not width changes).

Logs you in as the specified user, not as your user login name.

Note

There are known problems with the VT100 emulation in rloginVT. This program will be replaced in a future version.

rloginvt

Function

network terminal program

Usage

rloginvt host [-l username]

Description

Rloginvt is the network terminal program which supports the rlogin protocol. You must specify the desired host that rloginvt will attempt to contact. The host specification may be in the form of an ascii hostname or internet address.

A username may be supplied with **-l** option followed by the username on the remote host. Within the Rloginvt utility you have control over the vt100 emulation session via three menus. The following describes the project menu and the setup menu:

Project Menu

Save Settings	Saves your rloginvt preferences in the files: net.term.setup.
Load Settings	Reloads previously saved settings.
Reset Terminal	This menu option resets the terminal to the default settings.
Send Break	This function outputs a standard break signal. It is generally used to get the attention of the other computer while it is sending.
About	Gives information about the current version.
Quit	Exit from rloginvt.

Terminal Menu

TTY Emulates a simple terminal.

VT52 Emulates a VT52 terminal. Note: many features are not supported.

VT100 Emulates a VT100 terminal. Note: many features are not supported.

Amiga Emulates an Amiga terminal. Note: many features are not supported.

Setup Menu

Linewrap At the end of each line, the text can either wraparound or continue to overwrite the single character at line 80. With line wrap on, the cursor automatically advances to the first column of the next row when it reaches the last column of a row. Line wrap off disables this.

CTRL Chars Selects whether control characters are executed or visible.

Tabs The way tabs are set can affect the spacing of lines in a file that is transmitted. Tabs should be set according to how the *host* computer expects to receive them. Tabs are treated like tab settings on a typewriter, and can be set individually, at every eight or ten spaces, or cleared completely. To set tabs, after selecting Tabs from setup, point and click to the number of the column you want to place the tab. There are 80 columns, and you can place as many tabs as you want in whatever column you prefer. The clear box removes all tabs. To set the tabs, click the box OK with your pointer to finalize your choices.

Bell Selects visible, audible or both on receipt of CTRL-g.

Del + Bksp Selects or disable swapping of DEL + BACKSPACE.

Columns This option allows you to set the number of columns you want for your screen. The number you choose must match that of the host computer. You can select 80 for an 80 column-wide screen, or 132 for a screen 132 columns wide. The 132 option corresponds to the 132 column mode of a VT100 terminal. When using this mode, you must notify the host computer that you are using 132 rather then 80 columns.

Rows This sets the number of rows on your monitor screen, either at 24 or 49. 24 is the default setting, while 49 is used for interlace mode.

Set Colors Sets foreground and background screen colors.

Func. Keys Assigns strings to send when function keys are pressed.

Note: Some editor functions will not work in 49 line mode.

Route

Function

manually manipulate the routing tables

Usage

route [-f] [-n] [command args]

Description

Route is a program used to manually manipulate the network routing tables. It normally is not needed, as the system routing table management daemon **routed** should tend to this task. Route accepts two commands: **add,** to add a route, and **delete,** to delete a route.

All commands have the following syntax:

route [options] command [net | host] destination gateway [metric]

where destination is the destination host or network, gateway is the next-hop gateway to which packets should be addressed, and metric is a count indicating the number of hops to the destination. The metric is required for add commands; it must be zero if the destination is on a directly-attached network, and nonzero if the route utilizes one or more gateways. If adding a route with metric 0, the gateway given is the address of this host on the common network, indicating the interface to be used for transmission. Routes to a particular host are distinguished from those to a network by interpreting the Internet address associated with destination. The optional keywords **net** and **host** force the destination to be interpreted as a network or a host, respectively. 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. If the route is to a destination

connected via a gateway, the metric should be greater than 0. All symbolic names specified for a destination or gateway are looked up first as a host name using gethostbyname. If this lookup fails, getnetbyname is then used to interpret the name as that of a network.

If the **-f** option is specified, route will flush the routing tables of all gateway entries. If this is used in conjunction with one of the commands described above, the tables are flushed before the command is executed.

The **-n** option prevents attempts to print host and network names symbolically when reporting actions.

Routed

Function

network routing daemon

Usage

routed [-g] [-s] [-q] [-t] [logfile]
routed is started by specifying RIP as an argument to start-inet

Description

Routed is invoked at boot time to manage the network routing tables. The routing daemon uses a variant of the Xerox NS Routing Information Protocol in maintaining up-to-date kernel routing table entries. It used a generalized protocol capable of use with multiple address types, but is currently used only for Internet routing within a cluster of networks.

In normal operation routed listens on the udp socket for the route service (see services) for routing information packets. If the host is an internetwork router, it periodically supplies copies of its routing tables to any directly connected hosts and networks.

When routed is started, it uses the SIOCGIFCONF ioctl to find those directly connected interfaces configured into the system and marked up (the software loopback interface is ignored). If multiple interfaces are present, it is assumed that the host will forward packets between networks. Routed then transmits a request packet on each interface (using a broadcast packet if the interface supports it) and enters a loop, listening for request and response packets from other hosts.

When a request packet is received, routed formulates a reply based on the information maintained in its internal tables. The response packet generated contains a list of known

routes, each marked with a hop count metric (a count of 16, or greater, is considered infinite). The metric associated with each route returned provides a metric relative to the sender.

Response packets received by routed are used to update the routing tables if one of the following conditions is satisfied:

1. No routing table entry exists for the destination network or host, and the metric indicates the destination is reachable (the hop count is not infinite).
2. The source host of the packet is the same as the router in the existing routing table entry. That is, updated information is being received from the very internetwork router through which packets for the destination are being routed.
3. The existing entry in the routing table has not been updated for some time (defined to be 90 seconds) and the route is at least as cost-effective as the current route.
4. The new route describes a shorter route to the destination than the one currently stored in the routing tables; the metric of the new route is compared against the one stored in the table.

When an update is applied, routed records the change in its internal tables and updates the kernel routing table. The change is reflected in the next response packet sent.

In addition to processing incoming packets, routed also periodically checks the routing table entries. If an entry has not been updated for 3 minutes, the entry's metric is set to infinity and marked for deletion. Deletions are delayed an additional 60 seconds to insure the invalidation is propagated throughout the local Internet.

Hosts acting as internetwork routers gratuitously supply their routing tables every 30 seconds to all directly connected hosts and networks. The response is sent to the broadcast address on nets capable of that function, to the destination

address on point-to-point links, and to the router's own address on other networks. The normal routing tables are bypassed when sending gratuitous responses. The reception of responses on each network is used to determine that the network and interface are functioning correctly. If no response is received on an interface, another route may be chosen to route around the interface, or the route may be dropped if no alternative is available.

Options:

-g

This flag is used on internetwork routers to offer a route to the default destination. This is typically used on a gateway to the Internet, or on a gateway that uses another routing protocol whose routes are not reported to other local routers.

-s

Supplying this option forces routed to supply routing information whether or not it is acting as an internetwork router. This is the default if multiple network interfaces are present, or if a point-to-point link is in use.

-q

This is the opposite of the -s option.

-t

If the -t option is specified, all packets sent or received are printed on the standard output. In addition, routed will not divorce itself from the controlling terminal so that interrupts from the keyboard will kill the process.

Any other argument supplied is interpreted as the name of the file in which routed's actions should be logged. This log contains information about any changes to the routing tables

and, if not tracing all packets, a history of recent messages sent and received which are related to the changed route.

In addition to the facilities described above, routed supports the notion of distant passive and active gateways. When routed is started it reads the file **inet:db/gateways** to find gateways which may not be located using only information from the SIOGIFCONF ioctl. Gateways specified in this manner should be marked passive if they are not expected to exchange routing information, while gateways marked active should be willing to exchange routing information (they should have a routed process running on the machine). Routes through passive gateways are installed in the kernel's routing tables once upon startup. Such routes are not included in any routing information transmitted.

Active gateways are treated equally to network interfaces. Routing information is distributed to the gateway and if no routing information is received for a period of the time, the associated route is deleted.

Gateways marked external are also passive, but are not placed in the kernel routing table nor are they included in routing updates. The function of external entries is to inform routed that another routing process will install such a route, and that alternate routes to that destination should not be installed. Such entries are only required when both routers may learn of routes to the same destination.

The **inet:db/gateways** file is comprised of a series of lines, each in the following format:

[net | host] name1 gateway name2 metric value [passive | active | external]

The **net** or **host** keyword indicates if the route is to a network or specific host. **Name1** is the name of the destination network or host. This may be a symbolic name located in **inet:db/networks** or **inet:db/hosts,** or an Internet address specified in dot notation. **Name2** is the name or address of the gateway to which messages should be forwarded. Value is a metric indicating the hop count to the destination host or network. One of the keywords **passive, active** or **external** indicates if the gateway should be treated as passive or active (as described above), or whether the gateway is external to the scope of the routed protocol.

Rpcinfo

Function

rpc server information

Usage

rpcinfo -u host prognum [versionnum]
rpcinfo -t host prognum [versionnum]
rpcinfo -p [host]

Description

Rpcinfo generates information of the remote rpc server activity.

Options:

-u

Make an RPC call to procedure 0 of programnum on the specified host using UDP and report if any response was received.

-t

Make an RPC call to procedure 0 of programnum on the specified host using TCP and report if any response was received.

-p

Probe the portmapper on host, and print a list of all registered RPC programs.

Rsh

Function

remote shell

Usage

rsh host [-l username] command

Description

Rsh connects to the specified host, and executes the specified command. Rsh copies its standard input to the remote command, the standard output of the remote command to its standard output, and the standard error of the remote command to its standard error. Interrupt, quit and terminate signals are propagated to the remote command; rsh normally terminates at the same time as the remote command does.

The remote username **used** is the same as your local username, unless you specify a different remote name with the **-1** option. This remote name must be equivalent to the originating account; no provision is made for specifying a password with a command.

Host names are given in the file **inet:db/hosts.** The hostname of your Amiga should be placed in the file **/etc/hosts.equiv** on the remote Unix host. Otherwise rsh may not be granted execution permission.

The execution of a remote command which requires **stdin** is currently not supported, i.e., rsh remotehost cat > junk cannot accept console data from the Amiga (stdin).

Rshd

Function

remote shell server

Usage

Started from inetd

Description

Rshd is the server for the rcmd routine and, consequently, for the rsh program. The server provides remote execution facilities with authentication based on privileged port numbers from trusted hosts.

Rshd listens for service requests at the port indicated in the cmd service specification (see services). When a service request is received the following protocol is initiated:

1. The server checks the client's source port. If the port is not in the range 0-1023, the server aborts the connection.
2. The server reads characters from the socket up to a null ('\0') byte. The resultant string is interpreted as an ASCII number, base 10.
3. If the number received in step 2 is non-zero, it is interpreted as the port number of a secondary stream to be used for the stderr. A second connection is then created to the specified port on the client's machine. The source port of this second connection is also in the range 0-1023.
4. The server checks the client's source address and requests the corresponding host name. If the hostname cannot be determined, the dot-notation representation of the host address is used.

5. A null-terminated user name of 16 characters maximum is retrieved on the initial socket. This user name is interpreted as the user identity on the client's machine.

6. A null-terminated user name of 16 characters maximum is retrieved on the initial socket. This user name is interpreted as a user identity to use on the server's machine.

7. A null-terminated command to be passed to a shell is retrieved on the initial socket. The length of the command is limited by the upper bound on the size of the system's argument list.

8. Rshd then validates you according to the following steps. The local (server-end) user name is looked up in the password file and a chdir is performed to your home directory. If either the lookup or chdir fail, the connection is terminated. If you are not the super-user, (user id 0), the file **hosts.equiv** is consulted for a list of hosts considered equivalent. If the client's host name is present in this file, the authentication is considered successful. If the lookup fails, or you are the super-user, then the file **.rhosts** in the home directory of the remote user is checked for the machine name and identity of the user on the client's machine. If this lookup fails, the connection is terminated.

9. A null byte is returned on the initial socket and the command line is passed to the normal login shell of the user. The shell inherits the network connections established by rshd.

Showmount

Function

Show all remote mounts

Usage

showmount [-e] host

Description

Showmount lists all clients that remotely mount a filesystem from a remote host. If the **-e** option is used, showmount will print the list of all exported filesystems.

Telnet

Function

user interface to the TELNET protocol

Usage

telnet [host] [port]

Description

The telnet command is used to communicate with another host, using the TELNET protocol. If telnet is invoked without arguments, it enters command mode, which is indicated by the prompt: **telnet>**. In this mode, telnet accepts and executes the commands listed below. If it is invoked with arguments, it performs an open command with those arguments.

Once a connection is opened, telnet enters input mode. The input mode is either character-at-a-time or line-by-line, depending on what the remote system supports. In character-at-a-time mode, text is sent to the remote host as it is typed. In line-by-line mode only completed lines are sent to the remote host. **Ctrl-E,** the initial local-echo-character, turns the local echo on and off. This can be used to enter passwords without echoing them to the screen.

In either mode, if the localchars toggle is set as TRUE (the default in line mode), then your quit, intr, and flush characters are trapped locally and sent as TELNET protocol sequences to the remote side. Options such as **toggle autoflush** and **toggle autosynch** flush previous terminal input, as in quit and intr, in addition to flushing subsequent output to the terminal until the remote host acknowledges the TELNET sequences.

To issue telnet commands when in input mode, the escape character must precede the command; initially the control

character followed by a right bracket (Ctrl-]). When in command mode, use the normal terminal editing conventions.

The following commands are available:

open host [port]

Opens a connection to the named host. If no port number is specified, telnet attempts to contact a TELNET server at the default port. The host specification may be either a host name or an Internet address specified in the dot notation.

close

Closes a TELNET session and returns to command mode.

status

Shows the current status of telnet.

mode [type]

The [type] is either line, for line-by-line mode, or character, for character-at-a-time mode. The remote host will enter the requested mode if possible.

display [argument...]

Displays the set and toggle values explained below.

? [command]

On-line help. Telnet prints a help summary. If a command is specified, telnet prints the help information for that command.

quit

Closes any open TELNET session and exits telnet.

send argument(s)

Sends one or more special character sequences to the remote host. One or more of the following arguments can be specified:

escape
Sends the current escape character (initially control followed by a right bracket, Ctrl-]).

synch
Sends the **SYNCH** sequence. This sequence uses the remote system to discard input that was previously entered but has not yet been read. It is sent as TCP urgent data and may not work if the remote system is a 4.2 BSD system. If it does not work, a lower case r may be echoed on the terminal screen.

ao
Sends the Abort Output sequence, which causes the remote system to flush all output from the remote system to your terminal.

ayt
Sends the **Are You There?** sequence. The remote system may or may not respond.

brk
Sends the Break sequence. This may not work on the remote system.

ec
Sends the Erase Character sequence, which causes the remote system to erase the last character entered.

el
Sends the Erase Line sequence, which causes the remote system to erase the line currently being entered.

ga
Sends the Go Ahead sequence. This command may not have any significance for the remote system.

ip
Sends the Interrupt Process sequence, which causes the remote system to abort the currently running process.

nop
Sends the No Operation sequence.

?
Prints out help information for the send command.

set argument value

Sets a variable to a specific value. The **off** value turns off the function associated with the variable. The current value of variables can be displayed with the display command.

The following variables can be specified:

echo
Toggles between local echoing of entered characters, and suppressing echoing of entered characters when in line-by-line mode. The value is initially Ctrl-E.

escape
Enters the telnet command mode when you are connected to a remote system. The value is initially Ctrl-[.

flushoutput
Sends a AO sequence (see **send ao** above) to the remote host if telnet is in localchars mode (see **toggle localchars** below) and the flushoutput character is typed. The initial value for the flush character is the terminal's flush character.

interrupt
Sends an IP sequence to the remote host if telnet is in localchars mode (see **toggle localchars** below) and the interrupt character is typed. The initial value for the interrupt character is the terminal's intr character.

quit
Sends a BRK sequence (see **send brk** above) to the remote host if telnet is in localchars mode (see **toggle localchars** below) and the quit character is typed. The initial value for the quit character is the terminal's quit character.

eof
Sends this character to the remote system if telnet is operating in line-by-line mode and this character is entered as the first character on a line. The initial value of the eof character is the terminal's eof character.

erase
Sends a EC sequence (see **send ec** above) to the remote system if telnet is in localchars mode (see **toggle localchars** below), and if telnet is operating in character-at-time mode. The initial value for the erase character is the terminal's erase character.

kill
Sends a EL sequence (see **send el** above) to the remote system if telnet is in localchars mode (see toggle **localchars** below) and if telnet is operating in character-at-a-time mode. The initial value for the kill character is the terminal's kill character.

toggle arguments

Toggles TRUE and FALSE flags that control how telnet responds. More than one argument may be specified and the current value of these flags can be displayed with the display command. Valid arguments for the toggle command are the following:

autoflush
Causes the telnet command to not display any data on your terminal until the remote system acknowledges (via a Timing Mark option) that it recognized and processed the following sequences: **ao, intr,** or **quit.** Both autoflush and localchars must be TRUE for autoflush to work in this manner. The initial value for this toggle is TRUE if the terminal user did not specify **stty noflsh.** Otherwise it is FALSE. For further information, see **stty(1).**

autosynch
Causes the SYNCH sequence to follow the sequence that is initiated when either the **intr** or **quit** character is typed. The autosynch flag works in this manner when both the autosynch and localchars are TRUE. This procedure should cause the remote system to begin throwing away all previously typed input until both of the TELNET sequences have been read and acted upon. The initial value of this toggle is FALSE.

crmod
Toggles carriage return mode. When this mode is enabled, most carriage return characters received from the remote host are mapped into a carriage return followed by a line feed. It is useful only when the remote host sends carriage returns without line feeds. The initial value for this toggle is FALSE.

localchars
Causes the **flush, interrupt, quit, erase,** and **kill** characters to be recognized locally and transformed into appropriate control sequences if this flag is set to TRUE. (See

set above). The appropriate control sequences are: **ao, ip, brk, ec,** and **el,** respectively. For more information see the send command. The initial value for this toggle is TRUE in line-by-line mode, and FALSE in character-at-a-time mode.

debug
Toggles socket level debugging which is useful only to the superuser. The initial value for this toggle is FALSE.

options
Toggles the display of internal telnet protocol processing that deals with TELNET options. The initial value for this toggle is FALSE.

netdata
Toggles the display of all network data (in hexadecimal format). The initial value for this toggle is FALSE.

?
Displays the legal toggle commands.

Restrictions

In line-by-line mode, the terminal's EOF character is only recognized and sent to the remote system when it is the first character on a line.

Tftp

Function

trivial file transfer program

Usage

tftp [host]

Description

Tftp is the user interface to the Internet TFTP (Trivial File Transfer Protocol), which allows users to transfer files to and from a remote machine. The remote host may be specified on the command line, in which case tftp uses host as the default host for future transfer (see the **connect** command below).

Commands:

Once tftp is running, it issues the prompt **tftp>** and recognizes the following commands:

connect host-name [port]

Set the host (and optionally port) for transfers. Note that the TFTP protocol, unlike the FTP protocol, does not maintain connections between transfers. Thus, the connect command does not actually create a connection, but merely remembers what host is to be used for transfers. You do not have to use the connect command; the remote host can be specified as part of the **get** or **put** commands.

mode transfer-mode

Set the mode for transfers; transfer-mode may be acsii or binary. The default is ascii.

put file
put localfile remotefile
put file1 file2 . . . fileN remote-directory

Put a file or set of files to the specified remote file or directory. The destination can be in one of two forms: a filename on the remote host, if the host has already been specified, or a string of the form **host:filename** to specify both a host and filename at the same time. If the latter form is used, the hostname specified becomes the default for future transfers. If the remote-directory form is used, the remote host is assumed to be a UNIX machine.

get filename
get remotename localname
get file1 file2 . . . fileN

Get a file or set of files from the specified sources. Source can be in one of two forms: a filename on the remote host, if the host has already been specified, or a string of the form **host:filename** to specify both a host and filename at the same time. If the latter form is used, the last hostname specified becomes the default for future transfers.

quit

Exit tftp. An end of file also exits.

verbose

Toggle verbose mode.

trace

Toggle packet tracing.

status

Show current status.

rexmt retransmission-timeout

Set the per-packet retransmission timeout, in seconds.

timeout total-transmission-timeout

Set the total transmission timeout, in seconds.

ascii

Shorthand for mode ascii.

binary

Shorthand for mode binary.

? [command-name . . .]

Print help information.

Tftpd

Function

DARPA Trivial File Transfer Protocol server

Usage

Started from inetd

Description

Tftpd is a server which supports the DARPA Trivial File Transfer Protocol. The TFTP server operates at the port indicated in the tftp service description. The server is normally started by inetd.

The use of tftp does not require an account or password on the remote system. Due to the lack of authentication information, tftpd will allow only publicly readable files to be accessed. Files may be written only if they already exist and are publicly writable. Note that this extends the concept of public to include all users on all hosts that can be reached through the network. This may not be appropriate on all systems, and its implications should be considered before enabling tftp service.

THIN ETHERNET

A popular type of ethernet cable is often called "thin ethernet". It involves 50 ohm cable that is about 5 mm in diameter, with BNC connectors. Note: There are several types of cable that meet this description. Have your system manager verify that it is the correct type. If you are using the common "thin ethernet", the following applies:

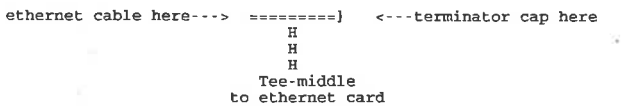
CABLES NOT CONNECTED PROPERLY

Is the tee firmly connected to your ethernet card?

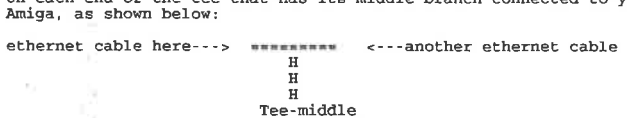
The tee should swivel but not pull off when pulled gently. If it pulls off, twist the collar a quarter turn clockwise until it pops into place. The cables and terminator caps also have collars that pop into place after a quarter turn clockwise.

You CANNOT connect the cable directly to your ethernet card. You must plug the middle-branch of a TEE into the Amiga, and the cable to one end-branch. The other end-branch must be connected to a 50 ohm TERMINATOR cap, or to another cable.

If your Amiga is the last machine on the end of the cable, you MUST connect a TERMINTOR CAP on the other end of the tee, as shown below:



If your Amiga is in the middle of a chain, there will be a cable on each end of the tee that has its middle branch connected to your Amiga, as shown below:



COMMUNICATION FAILURE OF ALL MACHINES ON THE CHAIN WILL RESULT FROM ANY OF:

1. Cable connected to ethernet card. (Card connects to tee-middle only.)
2. Cable end open. (Cable end must connect to tee-end.)
3. Tee with one tee-end open. (Tee-ends connect to cap or cables.)

4. Improper terminator. (Must use 50 ohm ethernet terminator)

If only the middle of the tee is dis[illegible]cted from an ethernet card, it affects only that card; the rest of [illegible]etwork will be unaffected.

Note: For devices other than computer ethernet cards, see system manager.

---------- End of "Thin Ethernet" Hardware Section ----------

THICK ETHERNET

The cable that connects the thick ethernet wire to your Amiga is about 12 mm in diameter, with 15 pin D-shell connectors.
Note: There are several types of cable that meet this description, so have your system manager verify that it is the right type.

This type of cable connects directly to the female 15 pin connector in the end of the ethernet card in the Amiga. For connections at the other end of the thick ethernet cable, see your system manager.

Is the cable firmly connected to your ethernet card?

CARD MUST BE CONFIGURED FOR THICK ETHERNET

The Commodore Ethernet card is shipped a jumper in position for thin ethernet. To use thick ethernet, you must:

1. Disconnect power and other cables from the Amiga.
2. Open the Amiga's cover.
3. Remove ethernet card from the Amiga.
4. Locate the 18-pin jumper block on the ethernet card.
   It is next to the 15-pin female D-shell connector.
5. Remove jumpers from pin rows A&B (THIN ethernet position)
   Replace jumpers on pin rows B&C (THICK ethernet position)
6. Replace ethernet card in the Amiga.
7. Replace the Amiga's cover.
8. Reconnect power and other cables.

SEE THE MANUAL THAT IS SHIPPED WITH THE ETHERNET CARD FOR MORE DETAILS.

There are often changes and additions in a software product after the manual is printed. This file contains a listing of these differences, as well as known problems which are intended to be fixed in a future release.

----- General -----

Many commands do not respond to control-c or breaks. However most will time out in 30 seconds or less and give a message if a needed resource is not available.

At present there is no way to terminate a server process except to reboot.

----- Utilities -----

ftp

The current version of ftp does not use the .netrc file. Ignore the message "netrc: Permission to access object is denied" that is displayed after an ftp connection is made.

In the current release, ftp command is not robust and does not recover well if an error does occur. Exit from ftp with the quit command and retry if an error occurs.

lance-test

When an ethernet address is reported, ignore a series of F's that may be displayed. Example: 00:FFFFFF23:00:10:44:A4 should be read as 00:23:00:10:44:A4.

To abort lance-test, type control-c first, then press return. If the control-c is omitted, it may be necessary to reboot.

ls
--
The current "ls" command is based on the public domain ls written by Justin V. McCormick. Type "ls -?" for more information.

rcp

Unreliable system operation or crashes may occur when several recursive rcp commands are run at the same time to or from the same Amiga.

rlogin

See the file "rlogin.doc" for more information on rlogin. Many improvements and additions have been made since the manual was printed. New features include resizable windows as well as full custom screen mode, cut & paste, and support for different fonts.

Rlogin -c now calculates screen height from the PAL/NTSC jumper setting. If the PAL/NTSC setting in Prefs does not match, screen

height is incorrect in rlogin -c. Rlogin -c incorrectly calculates screen width in Superhires modes.

The rlogin command has a correspondi rmcap file in the inet:Docs directory. A termcap file is a standardized description of a terminal's capabilities that allows all applications that use termcap to run on any type of terminal. The termcap file can be used directly with remote hosts, such as BSD UNIX, that use termcap for terminal handling.

Terminfo is another terminal description method. System V UNIX machines, including Amiga UNIX, provide support for both terminfo and termcap, although most newer applications use only terminfo. The captoinfo and tic commands can be used to convert the termcap to an equivalent terminfo. A terminfo.src file is also included for those who do not have the captoinfo command. See the captoinfo and tic man pages on your UNIX machine for more detail.

rloginvt

Rloginvt may be used instead of rlogin for applications that understand only vt100 type terminals. Rloginvt uses a custom screen and lacks resizable windows, cut & paste, and support for different fonts.

Rloginvt is most usable in 24 line, 80 column, vt100 mode. Text may be misplaced in 49 line mode when an application uses certain screen commands. This misplacement can insert, delete or change the wrong information when editors and some other screen applications are used. Emulations other than vt100 are incomplete.

Rloginvt will be replaced in a future version.

tftp

Tftp and tftpd were found to be unreliable and are not included in this release.

----- Configuration -----

The umask value must be converted from the usual octal form into decimal. For example, to set umask to 022, enter umask=18 in s:inet.config. Octal will be supported in a future release.

Make sure internet numbers in the inet:db/hosts file are entered correctly. If any of the four segments are greater than 255, a modulo 256 of the segment value is used without any error message. For example, if 190.227.1.1 is accidentally entered as 190.277.1.1, the 277 will be stored as 21. This will go undetected until communications fail to and from machines 190.227.1.1 or 190.21.1.1.

----- NFS File System -----

AmigaDOS exclusive file locks are not supported on NFS files. Applications that use exclusive locks cannot safely share NFS files.

Amiga TCP/IP NFS does not recognize symbolic links. A symbolic link appears as an ordinary file containing the name of the linked file.

The AmigaDOS R, W, and E protection bits are mapped to NFS r, w, and x, respectively. The AmigaDOS bit is indicated as set when the corresponding NFS bit is set for any of user, group, or other.

The AmigaDOS S,P,A and D file protection bits are not supported by NFS. On the AmigaDOS side, the S,P and A bits are treated as always unset, and the D bit as always set. A side effect is that files copied to NFS and back will have D set and S, P and A unset.

The Amiga TCP/IP "chmod" command can set or unset user, group and other protections separately.

AmigaDOS "protect" always sets or unsets user, group, and other simultaneously.

NFS file protections cannot be changed by the Workbench Information (Info) command.

Do NOT use ".." to refer to a parent directory in an NFS volume. Using ".." can extend above the NFS mount point and may crash the Amiga. USE the normal "/" AmigaDOS equivalent, it is safe. The ".." will work normally when typed directly to a UNIX host through rlogin or rloginvt.

nfsmgr

Nfsmgr has a new option "case". This keyword tells the NFS software to stop doing any case manipulations and use the same case sensitivity as the remote volume. Currently the Amiga NFS software does a case-insensitive search of a remote directory every time you create a file. This prevents you from creating a "Foo" and a "foo" file. Case is preserved, just like normal for the Amiga.

If you mount a volume using the "case" keyword, this check is eliminated. This speeds up some operations considerably and may be useful for people who desire case-sensitivity.

example: nfsmgr mount unixhost:/usr/tmp tmp: case

Amiga TCP/IP Software

FEATURES

The Amiga TCP/IP software supports the A2065 Ethernet card or Ameristar Ethernet cards. The software can support as many cards as you can connect to a machine (normally up to 5). It works on any Amiga under 1.3 or 2.0. Requires 1Mbyte RAM.

The A2065 is a Zorro II card. It provides 15 pin AUI connector for use with thick Ethernet (10BASE5) and a BNC connector for use with thin Ethernet (10BASE2). It has been tested with Amiga 2000, 2500 and 3000s.

The Amiga TCP/IP software supports the following basic protocols:

- ARP
- ICMP
- IP
- TCP
- UDP

The following applications are included:

Telnet	(client only)
FTP	(client and server)
rlogin	(client only)
rloginVT	(rlogin with VT100 emulation, client only)
ping	(client and server)
finger	(client and server)
rsh	(client and server, but no rlogin mode)
rcp	(client and server)
route	(client and server)

The following commands and diagnostic programs are included:

arp	netstat
chmod	passwd
lance-test	rpcinfo
ls	showmount

Network FileSystem (NFS)

The networking software also supports Sun XDR, RPC, and NFS (client only). NFS client software gives you the ability to mount disks served by an NFS server. Once mounted, access to remote NFS volumes is completely transparent. That is, you access remote files just like they were on a local partition of your hard drive.

For example, if I have an account on my_vax in directory /usr/graphics In my startup-sequence I can execute

nfsmgr mount my_vax:/usr/graphics graphics:

This creates a remote partition graphics: that is functionally equivalent to my local work: partition. If I'm running workbench, an icon comes up for graphics: and I can open it, move icons into it, etc. Unless I watch the hard disk light, I don't even realize that the files are being stored across the network, not locally.

NFS server software is available on most UNIX machines, as well as
many other operating systems.

Usage examples
---

ping

Ping is used to see if another machine is alive or to check
to see if a connection exists to the other machine.

Usage: ping [-drv] host [data size] [npackets]

>ping my_vax

PING my_vax (123.4.567.4): 56 data bytes
64 bytes from 123.4.567.4: icmp_seq=0. time=16. ms
64 bytes from 123.4.567.4: icmp_seq=1. time=0. ms
64 bytes from 123.4.567.4: icmp_seq=2. time=0. ms

----my_vax PING Statistics----
3 packets transmitted, 3 packets received, 0% packet loss
round-trip (ms) min/avg/max = 0/5/16

finger

Finger gives information about remote users

Usage: finger [user][@host]

>finger @wheel

[wheel]
Login   Name   TTY Idle   When   Where
joe   Joe Blank   p0   9d   Mon 16:04 toaster
cindy   Cindy Lane   p1   Thu 12:08 blender

>finger joe@wheel

[wheel]
Login name: joe   In real life: Joe Blank
Directory: /usr/software/joe   Shell: /bin/tcsh
On since Jun 21 12:08:24 on ttyp1 from toaster
47 seconds Idle Time
No unread mail
No Plan.

rcp
---
rcp is the UNIX remote copy command. In UNIX,
> rcp my_file host2:
would copy my_file to host2 (in your home directory, by default)
Because the colon ":" is used for volume names on the Amiga, Amiga
rcp uses an equals sign instead. So,
> rcp startup-sequence my_vax=
would copy your startup-sequence to my_vax. You can also use
> rcp startup-sequence my_vax=start
to copy "startup-sequence" to "start" in your home dir on my_vax.
The -r option copies recursively, so

```
> rcp -r dh0: my_vax=backup
would copy your entire dh0: partition to the backup directory
in your home dir on my_vax
```

rsh

rsh executes remote commands. On the Amiga, rsh works just like UNIX except an rsh into an Amiga cannot start up an interactive shell.

Usage: rsh host commands...

The following command execute the "status" command on the remote Amiga named "toaster"

```
> rsh toaster status
Process  1: Loaded as command: inet:c/NFSc
Process  2: Loaded as command: dh0:bin/fixlace
Process  3: Loaded as command: dh0:bin/dlineart
Process  4: No command loaded.
Process  5: Loaded as command: inet:serv/portmapd
Process  6: Loaded as command: inet:serv/inetd
Process  7: No command loaded.
Process  9: Loaded as command: inet:serv/rshd
Process 10: Loaded as command: status
```

For another example of using rsh, see the sample script on the last page.

rpcinfo

Gives RPC information on a remote server

```
>rpcinfo -p wheel

   program vers proto   port
    100004    2   udp   1027  ypserv
    100004    2   tcp   1024  ypserv
    100004    1   udp   1027  ypserv
    100004    1   tcp   1024  ypserv
    100007    2   tcp   1025  ypbind
    100007    2   udp   1035  ypbind
    100007    1   tcp   1025  ypbind
    100007    1   udp   1035  ypbind
    100009    1   udp   1023  yppasswdd
    100003    2   udp   2049  nfs
    100024    1   udp   1087  status
    100024    1   tcp   1031  status
    100021    1   tcp   1032  nlockmgr
    100021    1   udp   1092  nlockmgr
    100020    1   udp   1095  llockmgr
    100020    1   tcp   1033  llockmgr
    100021    2   tcp   1034  nlockmgr
    100012    1   udp   1115  sprayd
    100011    1   udp   1117  rquotad
    100005    1   udp   1119  mountd
    100008    1   udp   1121  walld
    100002    1   udp   1123  rusersd
    100002    2   udp   1123  rusersd
    100001    1   udp   1126  rstat_svc
    100001    2   udp   1126  rstat_svc
    100001    3   udp   1126  rstat_svc
```

```
    100015    6   udp   8769  selection_svc

showmount
---------
Shows which remote volumes may be mounted

>showmount wheel

Filesystem      Groups
/usr    gold, allsun, allsoft,
/usr/wheel      gold, allsun, allsoft,
/usr.MC68010/wheel      gold, baby, allsun, allsoft,
/usr.MC68010    gold, allsun, allsoft,

lance-test
----------
Tests A2065 cards
lance-test must be run before you start the networking software.

>lance-test diags
Ethernet address of board is 00:80:10:00:00:01


            Ethernet Controller Diagnostics

                Buffer memory test.............. PASS
                LANCE configuration test........ PASS
                Interrupt test.................. PASS
                LANCE collision logic test...... PASS
                Internal loopback test.......... PASS

            Controller passed diagnostics.


passwd
------
Updates the local password file.  Used for remote access from FTP.

arp
---
Get internet to ethernet address mappings.
>arp
usage: arp hostname
       arp -a
       arp -d hostname
       arp -s hostname ether_addr [temp] [pub] [trail]
       arp -f filename

>arp -a
my_vax (123.4.567.4) at aa:0:4:0:14:8
wheel (123.4.567.50) at 8:0:20:1:e:1f

netstat
-------
Print network statistics.

usage: netstat [ -Aaihmnrst ] [-p proto] [-I interface]

>netstat -p tcp
```

```
tcp:
        1619 packets sent
                699 data packets (153    es)
                0 data packets (0 byt    retransmitted
                865 ack-only packets (788 delayed)
                0 URG only packets
                0 window probe packets
                0 window update packets
                55 control packets
        1257 packets received
                741 acks (for 1581 bytes)
                29 duplicate acks
                0 acks for unsent data
                1047 packets (115880 bytes) received in-sequence
                15 completely duplicate packets (15 bytes)
                0 packets with some dup. data (0 bytes duped)
                15 out-of-order packets (0 bytes)
                0 packets (0 bytes) of data after window
                0 window probes
                5 window update packets
                0 packets received after close
                0 discarded for bad checksums
                0 discarded for bad header offset fields
                0 discarded because packet too short
        22 connection requests
        11 connection accepts
        31 connections established (including accepts)
        44 connections closed (including 0 drops)
        2 embryonic connections dropped
        741 segments updated rtt (of 763 attempts)
        2 retransmit timeouts
                0 connections dropped by rexmit timeout
        0 persist timeouts
        0 keepalive timeouts
                0 keepalive probes sent
                0 connections dropped by keepalive

>netstat -I ae0

Name  Mtu   Network     Address         Ipkts Ierrs    Opkts Oerrs  Coll
ae0   1500  xyz         amiga3           8390     0     2378     0     0

ls
--
UNIX-type ls command for the Amiga.  Can show owners and protection
bits on NFS volumes.

For example,

>list marvel:tmp

Directory "marvel:tmp" on Wednesday 20-Jun-90
.info                          81 ----rwed Thursday  11:23:16
readme                       7128 ----rwed Friday    21:02:37
rfc-index                  136342 ----rw-d Friday    14:48:33
work                          Dir --p-rwed Today     15:31:06
3 files - 1 directory - 308 blocks used

>ls -l marvel:tmp
```

```
drwxrwxrwx 406     14      90-06-20 15:31:06     0       Dir work
-rwxrw-rwx 406     14      90-06-14 11:23:16     2        81 .info
-rwxrw-rwx 406     14      90-06-15 21:  :7     14      7128 readme
-rw-rw-r-- 406     14      90-06-15 14     3   288    136342 rfc-index
Dirs:1     Files:3     Blocks:304    Byt..143551
```

chmod

UNIX-type chmod function for the Amiga. Can modify NFS volume protection bits.

```
>chmod a+w marvel:tmp/rfc-index
>ls -l marvel:tmp
drwxrwxrwx 406     14      90-06-20 15:31:06     0       Dir work
-rwxrw-rwx 406     14      90-06-14 11:23:16     2        81 .info
-rwxrw-rwx 406     14      90-06-15 21:02:37    14      7128 readme
-rw-rw-rw- 406     14      90-06-15 14:48:33   288    136342 rfc-index
Dirs:1     Files:3     Blocks:304    Bytes:143551
```

SECURITY

Currently the Amiga networking software, like many PC-based networking products, is a potential security problem. Every PC is configured with a machine name and internet number. Every user has a username, user ID (UID), and group ID (GID). The problem is that these are not secure and may be easily changed by a knowledgeable user.

Some of the important security files you should be aware of are:

hosts.equiv

This is a system file that lists the trusted hosts. If a machine is included in this list, then rlogin, rsh, rcp, etc. will be permitted freely from that machine to your machine.

exports

This system file lists which directories will be exported via NFS and which machines may access them.

.rhosts

This is a user file is the user's home directory. It permits rlogins from specific hosts without prompting for a password.

.netrc

This is a user file that lists the name and passwords to automatically use when FTP'ing to a remote machine. This is always a security problem because account names and passwords are listed here in clear text.

The interim solution to the security problem is to tell your users to not use .netrc and .rhosts files. You should also make sure that the hosts.equiv file contains no unsecure machines.

Preventing illegal access via NFS is difficult, because PCs can change their names easily. You can make thi[illegible]re difficult for potential troublemakers by making permanent ar[illegible]ries for each PC ethernet address. Of course this will make th[illegible]s more difficult in case of ethernet board trouble, but in some cases the additional security may be worth it.

The best solution to the security problem is to use some kind of authentication service. This means that before you use your PC on the network, you will first have to provide a valid username and password to some central server which maintains a secure master password file. We are looking at several different methods to do this.

Sample Script for remote printing

This is an example of the power of the rsh command combined with either NFS or rcp. You could easily set your amiga up to send mail, automatically move files between different machines, etc.

This script assumes the use of NFS. It could be written to use rcp if NFS is not available.

```
. rprint
.key file/a,homevol/k,username/k,printer/k,number/k,sides/k,type/k
.bra {
.ket }
.def number 2
.def sides 1
.def type "ascii"
.def printer "lp1"
.def homevol "marvel:"
.def username "marvel"


IF NOT EXISTS {homevol}.prspool
   makedir {homevol}.prspool
ENDIF

delete >NIL: {homevol}.prspool/#?

copy {file} {homevol}.prspool

run rsh my_vax -l {username} "cd ~{username}/.prspool;lpr -P{printer}
        -N{number} -K{sides} -D{type} *"

echo "Done printing"
```

RLOGIN.DOC for version 36.14 Revi[illegible]ec 7, 1990 -bj

** NOTE: Users of versions previous to this one will need to change
their termcap or terminfo!!!

Usage:

```
rlogin host [-l user][-x #][-y #][-w #][-h #][-r 0][-c][-t t/b][-u #]
```

where:	-l user	login name on the remote machine. The default is the entry given by "user=" in inet:s/inet.config
	-x #	left edge of the window
	-y #	top edge of the window
	-h #	height of the window
	-w #	width of the window
	-r 0	no resize gadget
	-c	gives a custom screen (and NO windows)
	-t t/b	changes default terminal type and baud rate
	-u #	toggle use of the 2.0 console device options

NOTE: All options and their arguments MUST be separated by at least one space

legal: rlogin -x 400 -y 200 -u 1

illegal: rlogin -x400 - y 200 -u1
^^^ ^^^

Resizing: -r

Use an argument of '0' (zero) to turn OFF the resize gadget.
Any other numeric argument turns it on. No argument is an error.

USAGE: -r <#>

EXAMPLE: -r 0

DEFAULT: On

Console: -u

Note: The -u option is functional only under the 2.x Operating System.

Under the 2.0 operating system the console device has the options to have a character mapped display and cut-and-paste between console windows. If you run rlogin under the 2.x operating system you will get these options be default. If you desire, you can alter the default by use of the -u flag ('u' for 'unit'.)

Under 2.0 you have the options of '0', '1' and '3'. Zero ('0') gives

you the same behavior as the console device under 1.3. Unit number one ('1') gives you a character mapped display. This means that if you resize the window, the text will be r wn to match the new window dimensions. The use of option '3' g you the character mapped display as well as the ability to do cut-and-paste operations between console windows (the Amiga's CLI for example.) The default under 2.0 is '3'. See the AmigaDOS 2.0 manuals for more information on these features.

A side benefit of this option is that it allows you to resize the rlogin window down to a very small size. Such resizing is NOT available under earlier (< 2.0) operating systems or if you don't set the -u flag to either '1' or '3'. The default is option 3 which gives you both character mapping and cut & paste.

Under the 1.3 operating system you have only '0' (the default)

```
USAGE:          -u <option>

EXAMPLE:        -u 0
                -u 1

DEFAULT:         Under OS 2.0 the default is   '-u 3'
                 Under OS 1.3 the default is   '-u 0'
```

```
TermType:       -t
```

The termtype flag ('-t') allows the user to tell the host what what his terminal type is. This is handy for using different termcap or terminfo files. There is a direct relationship between what Rlogin passes as it's termtype and which termcap or terminfo the host will use for you.

The termtype also passes a requested baud rate for your session. The host may or may not honor this, however.

The default for Rlogin (what it will tell the host if you don't use the -t flag) is:

```
        RLamiga/9600
```

The termcap that accompanies this release has it's identifiers matching the default termtype of Rlogin.

The included termcap file can be converted to a terminfo entry for use with applications that use terminfo, as is common on System V Unix machines. The System V "captoinfo" and "tic" commands can be used to convert termcap files into terminfo form. The file terminfo.src is included for those who do not have the captoinfo command. See the captoinfo and tic man pages.

```
USAGE:          -t <termtype/baud>

EXAMPLE:        -t myterm/9600

DEFAULT:        -t RLamiga/9600
```

LeftEdge: -x

This flag allows you to specify the horizontal position of the left edge of the window (if you are using a window and not a screen.)

USAGE: -x <position>

EXAMPLE: -x 100

DEFAULT: -x 0 (zero)

TopEdge: -y

This flag allows you to specify the vertical position of the top edge of the window (if you are using a window and not a screen.)

USAGE: -y <position>

EXAMPLE: -y 100

DEFAULT: -y 0 (zero)

Width: -w

This flag allows you to specify the width of the window. (if you are using a window and not a screen.)

USAGE: -w <width>

EXAMPLE: -w 700

DEFAULT: If you open RLogin in a window, it will attempt to open the window to a width that will give you a full 80 columns of text. This value is calculated from the current text that you are using. If the width value that it calculates turns out to be larger than the width of the screen, the window will open to the screen's width.

Height: -h

This flag allows you to specify the height of the window. (if you are using a window and not a screen.)

USAGE: -h <height>

EXAMPLE: -h 500

DEFAULT: If you open RLogin in a window, it will attempt to open the window to a height that will give you a full 24 lines of text. This value is calculated from the current text that you are using. If the height value that it calculates turns out to be larger than the height of the screen, the window will open to the screen's height.

Screen: -c

This flag tells Rlogin to open as an Amiga custom screen instead of as a window on the Amiga Workbench. This flag overrides any and all position and size flags.

USAGE: -c

DEFAULT: Off