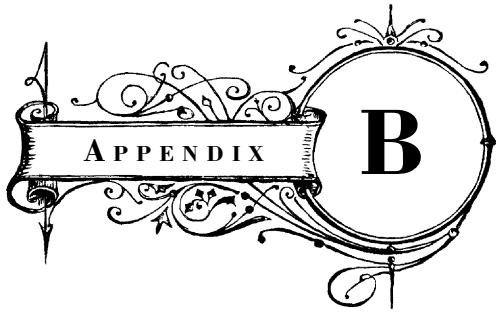
PART III

APPENDIXES

LIST OF WORKSHEETS

This book contains several worksheets designed to assist you in determining your own firewall and security needs. The list below is included to aid in quickly locating them.



SOURCES OF INFORMATION

A VAST COLLECTION OF tools and vendors offer software and security products and services. Even though the term and use of firewalls is relatively new, it is quickly becoming a major part of the network security business. The author recommends that you carefully examine any publicly available code before trusting it to protect your networks, computer systems, and data. This caution is not meant to insinuate that the code itself might be questionable, but to ensure that what you think you are getting is actually what you want. Furthermore, the tools and vendors listed here are not necessarily any better than those not mentioned.

> The path information supplied for each of these tools is the location of one source of the software. There might be other sources, some of which are more convenient to access from your location. Conduct a search with Archie or your favorite World Wide Web search tool to find other sources of these tools.

Most of the sites listed in this appendix also are a good source of other security-related information, so be sure to have a look around.

TOOLS

The list of publicly available tools is long, and the network administrator who is prepared to spend some time investigating each of them is wise. A list of some popular tools and where to get them is identified here.

Keep in mind that neither the authors of this book nor the authors of the software make any claims as to the software's usefulness. As the network administrator, you are responsible for verifying the usefulness and risks associated with any of the software listed here.

TCPWRAPPER AND PORTMAPPER

This is probably one of the best known tools for adding logging and filtering to most standard services. The tcpwrapper program supports only services that are invoked through inetd, whereas portmapper is used for RPC services that are invoked through the standard portmapper. The TCP Wrapper software also is included on the CD-ROM that accompanies this book.

Host: FTP.WIN.TUE.NL

Path: /pub/security/tcp_wrapper

Path: /pub/security/portmap

FIREWALL KIT

The TIS Firewall Toolkit is a collection of software components and system configuration practices that enable you to put together an Internet firewall. Included in the kit are application proxies for ftp, telnet, and rlogin as well as tools for securing SMTP mail.

Host: FTP.TIS.COM

Path: /pub/firewalls/toolkit

BELLCORE S/KEY

S/Key is a one-time password scheme that requires no extra hardware. The user can print a list of challenges and responses prior to traveling. S/Key is included on this book's CD-ROM.

Host: THUMPER.BELLCORE.COM

Path: /pub/nmh/skey

ONE-TIME PASSWORDS IN EVERYTHING (OPIE)

OPIE is an implementation of one-time passwords that was developed by the United States Naval Research Labs. The OPIE software is included on the CD-ROM that accompanies this book.

Host: FTP.NRL.NAVY.MIL

Path: /pub/security/nrl-opie

SWATCH LOGFILE MONITOR

Swatch is a tool that lets you associate actions with logfile entries. When logfile entries are found, the administrator can arrange for a command to be executed such as mail, finger, etc.

Host: FTP.STANFORD.EDU

Path: /general/security-tools/swatch

TCPDUMP

Tcpdump is a collection of tools that can be used to capture and examine the TCP/IP packets flowing on a network. It is considered by many to be one of the best network analysis tools available on the Internet.

Host: FTP.EE.LBL.GOV

Path: tcpdump-3.0.tar.Z

TAMU TIGER

The TAMU Tiger system is a collection of tools you can use to build a firewall or detect attack signatures. A collection of scripts are included that can be used to assess the security of the machines in your own network. Be advised that the anonymous FTP server at this site has some very tight restrictions governing the number of anonymous FTP users. It may take a while for you to get this code.

Host: NET.TAMU.EDU

Path: /pub/security/TAMU

COPS

COPS (Computer Oracle and Password Program) is another popular system auditing package. The COPS package is a collection of scripts and programs that can be used to evaluate the status of a system's security. It includes features to check passwords, SUID and SGID files, protected programs, and more. The COPS software also is included on the CD-ROM that accompanies this book.

Host: FTP.CERT.ORG

Path: /pub/tools/cops

CRACK

Chapter 2, "Security," discussed the use of a password-cracking program. This is one of the best known password-cracking programs, and it can be customized to use your own dictionaries.

Host: FTP.CERT.ORG

Path: /pub/tools/crack

A good set of alternative dictionaries can be found at the following:

Host: FTP.OX.AC.UK

Path: /pub/wordlists

SATAN

SATAN, also known as the Security Administrator's Tool for Analyzing Networks, was written by Dan Farmer and Wietse Venema. It is a collection of tools that run on Perl 5 and your HTML browser to analyze a machine or group of machines for common security problems. A copy of SATAN is included on the accompanying CD-ROM.

Host: FTP.WIN.TUE.NL

Path: /pub/security/satan-1.0.tar.Z

PASSWD+

The passwd+ program performs an analysis of the entered passwords based on the specific rules that your site specifies.

Host: FTP.DARTMOUTH.EDU

Path: /pub/security/passwd+.tar

NPASSWD

The npasswd program is a drop-in replacement for the passwd program. It can be extended to allow changes to other fields in the password file.

Host: FTP.UGA.EDU

Path: /pub/security/npasswd.tar.gz

TRIPWIRE

The Tripwire tools check the files stored on the system to determine whether any have been changed. It checks for damaged or tampered files in an effort to limit the damage.

Host: FTP.NORDU.NET

Path: /networking/security/tools/tripwire

FINDING COMMERCIAL FIREWALL VENDORS

Many commercial operations offer firewall software products and other security related services. Some of these vendors include the following:

- Advanced Network and Services (ANS) of Reston, Virginia

  URL: http://www.ans.net/InterLock

- Border Network Technologies of Mississaugua, Ontario, Canada

  URL: http://www. borderware.com

- Digital Equipment Corporation in Stow, Massachusetts

  URL: http://www. dec.com/info/security/products.htm

- Raptor Systems of Wilmington, Delaware

  URL: http://www.raptor.com

- Trusted Information Systems in Glenwood, Maryland

  URL: http://www.tis.com

- Milkyway Corporation in Ottawa, Canada

  URL: http://www.milkyway.com

EXPLORING FIREWALL AND SECURITY MAILING LISTS

A number of mailing lists and forums are available on the topics of firewalls and security in general. Some are distributed through e-mail, whereas others are part of the USENET News System.

FIREWALL MAILING LISTS

There are two major mailing lists for firewalls. One is hosted by GREATCIRCLE.COM, and the other is hosted by TIS.COM. To subscribe to the Great Circle mailing list, send a message to majordomo@greatcircle.com, with the body of the message reading as follows:

```
subscribe firewalls your-email-address
```

The TIS.COM firewall list, which focuses primarily on using the TIS firewall toolkit, can be subscribed to by sending a message to fwall-user-request@tis.com, with the body of the message reading as follows:

```
subscribe fwall-users your-email-address
```

In either case, the messages from each list will start flowing to your mailbox within a day or two.

Other discussion groups for security-related topics exist. For S/Key and one-time passwords, a mailing list is maintained by Bellcore. To be added to the list, send a request to skey-users-request@bellcore.com. This list is not automated, so it takes a day or two for requests to be completed.

SECURITY FORMS

Other forums are available for the discussion of security in general, not necessarily just firewalls. These forums are generally part of the USENET News system and include the following newsgroups:

comp.security.announce

comp.security.misc

comp.security.unix

alt.security

This is not meant to be an all-inclusive list, as security-related discussions arise in many other newsgroups as well.

VENDOR LIST

The following vendors and their products represent security packages or firewall implementations:

3COM *LAN Security Architecture* is a randomly generated mask for Ethernet networks. 5400 Bayfront Plaza, Santa Clara, CA 95052; (800)638-3266. URL: http://www.3com.com.

ANS *Interlock* provides gateway protection, encryption, and firewall implementation. 1875 Campus Commons Drive, Suite 200, Reston, VA 22091; (800)456-8267. URL: http://www.ans.net.

BASELINE SOFTWARE Several products include *Password Coach*, *Information Security Policies Made Easy*, and *Password Genie*. P.O. Box 1219, Sausalito, CA 94966; (800)829-9955. URL: http://pomo.nbn.com/people/infosec/.

BAY NETWORKS *Lattis Secure* is access control hardware. 4401 Great America Parkway, Santa Clara, CA 95052; (800)776-6895. URL: http://www.baynetworks.com.

BLUE LANCE LT *Auditor* provides access control for DOS, Macs, and Windows. 1700 W. Loop S., Suite 1100, Houston, TX 77027; (713)680-1187.

BORDER NETWORK TECHNOLOGIES INC. *Borderware* is a firewall implementation. 20 Toronto Street, Suite 400, Toronto, Canada M5C 2B8; (416)368-7157. URL: http://www.borderware.com.

COMPUTER ASSOCIATES *CA-Unicenter* offers encryption capabilities for a wide range of operating systems. 1 Computer Associates Plaza, Islandia, NY 11788; (800)225-5224. URL: http://www.cai.com.

CYBERSAFE *Challenger/Kerberos* offers access control encryption. 2443 152nd Avenue N.E., Redmond, WA 98052; (206)883-8721.

IC ENGINEERING *Modem Security Enforcer* provides call-back modem protection, as well as access encryption and several other features. P.O. Box 321, Owings Mills, MD 21117; (410)363-8748.

LIVINGSTON ENTERPRISES *Portmaster*, *IRX*, and *Firewall* packages. 6920 Koll Center Parkway, Suite 220, Pleasanton, CA 94566; (800)458-9966. URL: http://www.livingston.com/.

M&T TECHNOLOGIES *Microsafe* offers operating system independent virus protection. 1435 N. Hayden Road, Scottsdale, AZ 85257; (602)994-5131.

MILKYWAY NETWORKS *Black Hole* and other network security services. 255-2650 Queensview Drive, Ottawa, ON Canada K2B 8H6; (613)596-5549. URL: http://www.milkyway.com.

PARALON TECHNOLOGIES *Path Key* provides encryption, access control, and authentication. 3650 131st Avenue S.E., Suite 210, Bellevue, WA 98006; (206)641-8338.

RAPTOR SYSTEMS *Eagle* firewalls for Unix and Windows NT systems. 69 Hickory Drive, Waltham, MA 02154; (617)487-7700. URL: http://www.raptor.com.

THE ROOT GROUP *Sudo* provides access control with on-screen and email event notification. 4700 Walnut Street, Suite 110, Boulder, CO 80301; (303)447-3938. URL: http://www.rootgroup.com.

SEMAPHORE COMMUNICATIONS *Semaphore Network Security System* adds access control, encryption, and data integrity features. 2040 Martin Avenue, Santa Clara, CA 95050; (408)980-7750.

THE OPIE AND LOG DAEMON MANUAL PAGES

THIS APPENDIX CONTAINS the manual pages for the OPIE and Log Daemon packages. The manual pages are presented here to complement the text in Chapter 4, "The One-Time Password Authentication System."

THE OPIE MAN PAGES

This section presents the online man pages for the NRL OPIE distribution.

OPIEFTPD

opieftpd—File Transfer Protocol server that uses OPIE authentication

SYNOPSIS

opieftpd [-d] [-l] [-t *timeout*] [-T *maxtimeout*]

DESCRIPTION

opieftpd is the 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(5)*.

OPTIONS

-d	Debugging information is written to the system logs.
-l	Each ftp(1) session is logged in the system logs.
-t	The inactivity timeout period is set to *timeout* seconds (the default is 15 minutes).
-T	A client may also request a different timeout period; the maximum period allowed may be set to *maxtimeout* seconds with the **-T** option. The default limit is 2 hours.

COMMANDS

The ftp server currently supports the following ftp requests; case is not distinguished:

Request	*Description*
ABOR	Abort previous command
ACCT	Specify account (ignored)
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

Request	*Description*
LIST	Give a list of files in a directory
MKD	Make a directory
MDTM	Show last modification time of file
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
REST	Restart incomplete transfer
RETR	Retrieve a file
RMD	Remove a directory
RNFR	Specify rename-from file name
RNTO	Specify rename-to file name
SITE	Non-standard commands (see next section)
SIZE	Return size of file
STAT	Return status of server
STOR	Store a file
STOU	Store a file with a unique name
STRU	Specify data transfer structure
SYST	Show operating system type of server system
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)

The following non-standard or Unix-specific commands are supported by the SITE request:

Request	*Description*
UMASK	Change umask (e.g., SITE UMASK 002)
IDLE	Set idle-timer (e.g., SITE IDLE 60)
CHMOD	Change mode of a file (e.g., SITE CHMOD 755 file)
HELP	Give help information (e.g., SITE HELP)

The remaining ftp requests specified in Internet RFC-959 are recognized, but not implemented. MDTM and SIZE are not specified in RFC-959, but will appear in the next updated FTP RFC.

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. *Opieftpd* interprets file names according to the globbing conventions used by *csh(1)*. This allows users to utilize the metacharacters \&*?[]{}~.

Opieftpd authenticates users according to three rules:

- The user name must be in the password database, */etc/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.
- The user name must not appear in the file */etc/ftpusers*.
- The user must have a standard shell returned by *getusershell(3)*.

If the user name is *anonymous* or *ftp*, an anonymous ftp account must be present in the password file (user ftp). In this case, the user is allowed to log in by specifying any password (by convention, this is given as the client host's name). In the last case, *opieftpd* takes special measures to restrict the client's access privileges. The server performs a *chroot(2)* command to the home directory of the *ftp* user.

In order that system security is not breached, it is recommended that the *ftp* subtree be constructed with care; the following rules are recommended:

~ftp	Make the home directory owned by ftp and unwritable by anyone.
~ftp/bin	Make this directory owned by the super-user and unwritable by anyone. The program *ls(1)* must be present to support the LIST command. This program should have mode 111.

~ftp/etc Make this directory owned by the super-user and unwritable by anyone. The files *passwd(5)* and *group(5)* must be present for the *ls(1)* command to be able to produce owner names rather than numbers. The password field in *passwd* is not used, and should not contain real encrypted passwords. These files should be mode 444.

~ftp/pub Make this directory mode 777 and owned by *ftp*. Users should then place files which are to be accessible via the anonymous account in this directory.

SEE ALSO

ftpd(8), ftp(1), opie(4), opiekey(1), opiepasswd(1), opieinfo(1), opiesu(1), opieftpd(8), opiekeys(5), opieaccess(5)

BUGS

The anonymous account is inherently dangerous and should be avoided when possible. In *opieftpd*, it is a compile-time option that should be disabled if it is not being used. The server must run as the super-user to create sockets with privileged port numbers. It maintains an effective user id of the logged in user, reverting to the super-user only when binding addresses to sockets. The possible security holes have been scrutinized, but are possibly incomplete.

AUTHOR

Originally written for BSD, *ftpd* was modified at NRL by Randall Atkinson, Dan McDonald, and Craig Metz to support OTP authentication.

OPIEKEY

SYNOPSIS

opiekey | opie-des | opie-md4 | opie-md5 | otp-md4 | otp-md5 [-v] [-h] [-4|-5]

[-d] [-a] [-n *count*] *sequence_number seed*

.sp 0

DESCRIPTION

opiekey takes the optional count of the number of responses to print along with a (maximum) sequence number and seed as command-line arguments. It prompts for the user's secret password twice and produces an OPIE response as six words. The second password entry can be circumvented by entering only an end of line.

opiekey is downward compatible with the *key(1)* program from the Bellcore S/Key Version 1 distribution and several of its variants.

OPTIONS

-v	Displays the version number and compile-time options, then exit.
-h	Displays a brief help message and exit.
-4, -5	Selects MD4 or MD5, respectively, as the response generation algorithm. The default for opie-md4 and otp-md4 is MD4, and the default for opie-md5 and opie-md5 is MD5. The default for opie-des and opiekey depends on compile-time configuration, but should be MD5. MD4 is compatible with the Bellcore S/Key Version 1 distribution.
-d	Selects DES-based key processing, if opiekey was built with this optional support. The default is not to use DES key munging.
-a	Allows you to input an arbitrary secret pass phrase, instead of running checks against it. Arbitrary currently does not include '\0' or '\n' characters. This can be used for backwards compatibility with key generators that do not check passwords.
-n <count>	The number of one-time access passwords to print. The default is one.

EXAMPLE

```
wintermute$ opiekey -5 -n 5 495 wi01309
Using MD5 algorithm to compute response.
Reminder: Don't use opiekey from telnet or dial-in sessions.
Enter secret password:
Again secret password:
491: HOST VET FOWL SEEK IOWA YAP
492: JOB ARTS WERE FEAT TILE IBIS
493: TRUE BRED JOEL USER HALT EBEN
494: HOOD WED MOLT PAN FED RUBY
495: SUB YAW BILE GLEE OWE NOR
wintermute$
```

BUGS

opiekey(1) can lull a user into revealing his/her password when remotely logged in, thus defeating the purpose of OPIE. This is especially a problem with xterm. opiekey(1) implements simple checks to reduce the risk of a user making this mistake. Better checks are needed.

SEE ALSO

opie(4), opiepasswd(1), opieinfo(1), opiesu(1), opielogin(1), opieftpd(8), opiekeys(5), opieaccess(5)

AUTHOR

Bellcore's S/Key was written by Phil Karn, Neil M. Haller, and John S. Walden of Bellcore. DES key crunching contributed by Marcus J. Ranum of TIS. OPIE was created at NRL by Randall Atkinson, Dan McDonald, and Craig Metz.

S/Key is a trademark of Bell Communications Research (Bellcore).

OPIEPASSWD

opiepasswd—Change or set a user's password for the OPIE authentication system

SYNOPSIS

opiepasswd [-v] [-h] [-c] [-n *inital_sequence_number*][-s *seed*] [*user_name*]

DESCRIPTION

opiepasswd will initialize the system information to allow one to use OPIE to login. *opiepasswd* is downward compatible with the keyinit(1) program from the Bellcore S/Key Version 1 distribution.

OPTIONS

- **-v** Display the version number and compile-time options, then exit.
- **-h** Display a brief help message and exit.
- **-c** Set console mode where the user is expected to have secure access to the system. In console mode, you will be asked to input your password directly instead of having to use an OPIE calculator. If you do not have secure access to the system (i.e., you are not on the system's console), you are volunteering your password to attackers by using this mode.
- **-n** Manually specify the initial sequence number. The default is 499.
- **-s** Specify a non-random seed. The default is to generate a "random" seed using the first two characters of the host name and five pseudo-random digits.

EXAMPLE

Using *opiepasswd* from the console:

```
wintermute$ opiepasswd -c
Updating kebe:
Reminder: Only use this method from the console; NEVER from remote. If you
are using telnet, xterm, or a dial-in, type ^C now or exit with no password.
Then run opiepasswd without the -c parameter.
Using MD5 to compute responses.
```

```
Old secret password:
New secret password:
New secret password (again):
ID kebe OPIE key is 499 be93564
CITE JAN GORY BELA GET ABED
wintermute$
```

Using *opiepasswd* from remote:

```
wintermute$ opiepasswd
Updating kebe:
Reminder: You need the response from your OPIE calculator.
Old secret password:
otp-md5 482 wi93563
Response: FIRM BERN THEE DUCK MANN AWAY
New secret password:
otp-md5 499 wi93564
Response: SKY FAN BUG HUFF GUS BEAT
ID kebe OPIE key is 499 wi93564
SKY FAN BUG HUFF GUS BEAT
wintermute$
```

FILES

/etc/opiekeys database of key information for the OPIE system.

SEE ALSO

passwd(1), opie(4), opiekey(1), opieinfo(1), opiesu(1), opielogin(1), opieftpd(8), opiekeys(5), opieaccess(5)

AUTHOR

Bellcore's S/Key was written by Phil Karn, Neil M. Haller, and John S. Walden of Bellcore. DES key crunching contributed by Marcus J. Ranum of TIS. OPIE was created at NRL by Randall Atkinson, Dan McDonald, and Craig Metz.

S/Key is a trademark of Bell Communications Research (Bellcore).

OPEINFO

opieinfo—Extract sequence number and seed for future OPIE challenges

SYNOPSIS

opieinfo [-v] [-h] [*user_name*]

DESCRIPTION

opieinfo takes an optional user name and writes the current sequence number and seed found in the OPIE key database for either the current user or the user specified. opiekey is

compatible with the *keyinfo(1)* program from Bellcore's S/Key Version 1 except that specification of a remote system name is not permitted.

opieinfo can be used to generate a listing of your future OPIE responses if you are going to be without an OPIE calculator and still need to log into the system. To do so, you would run something like:

opiekey -n 42 `opieinfo`

OPTIONS

-v	Display the version number and compile-time options, then exit.
-h	Display a brief help message and exit.
<user_name>	The name of a user whose key information you wish to display. The default is the user running opieinfo.

EXAMPLE

```
wintermute$ opieinfo
495 wi01309
wintermute$
```

FILES

/etc/opiekeys	Database of key information for the OPIE system.

SEE ALSO

opie(4), opiekey(1), opiepasswd(1), opiesu(1), opielogin(1), opieftpd(8), opiekeys(5) opieaccess(5)

AUTHOR

Bellcore's S/Key was written by Phil Karn, Neil M. Haller, and John S. Walden of Bellcore. DES key crunching contributed by Marcus J. Ranum of TIS. OPIE was created at NRL by Randall Atkinson, Dan McDonald, and Craig Metz.

S/Key is a trademark of Bell Communications Research (Bellcore).

OPIELOGIN

opielogin—Replacement for login(1) that issues OPIE challenges

SYNOPSIS

opielogin [-p] [-r *hostname* | -h *hostname* | -f *username* | *username*]

DESCRIPTION

opielogin provides a replacement for the *login(1)* program that provides OPIE challenges to users and accepts OPIE responses. It is downward compatible with the *keylogin(1)* program from the Bellcore S/Key Version 1 distribution, which, in turn, is downward compatible with the *login(1)* program from the 4.3BSD Net/2 distribution.

OPTIONS

-p	By default, login discards any previous environment. The -p option disables this behavior.
-r	Process remote login from hostname.
-h	The -h option specifies the host from which the connection was received. It is used by various daemons such as telnetd(8). This option may only be used by the super-user.
-f	The -f option is used when a user name is specified to indicate that proper authentication has already been done and that no password need be requested. This option may only be used by the super-user or when an already logged in user is logging in as themselves.
Username	The user name to log in as.

EXAMPLE

```
wintermute$ opielogin
login: kebe
otp-md5 499 wi43143
Password: (echo on)
Password: SLY BLOB TOUR POP BRED EDDY
              Welcome to wintermute.
wintermute$
```

FILES

/etc/opiekeys	Database of information for the OPIE system.
/etc/opieaccess	List of safe and unsafe networks and masks to go with them.
$HOME/.opiealways	Presence makes OPIE for logins mandatory for the user.

SEE ALSO

login(1), opie(4), opiekey(1), opiepasswd(1), opieinfo(1), opiesu(1), opieftpd(8), opiekeys(5), opieaccess(5)

AUTHOR

Bellcore's S/Key was written by Phil Karn, Neil M. Haller, and John S. Walden of Bellcore. DES key crunching contributed by Marcus J. Ranum of TIS. OPIE was created at NRL by Randall Atkinson, Dan McDonald, and Craig Metz.

S/Key is a trademark of Bell Communications Research (Bellcore).

OPIESU

opiesu—Replacement su(1) program that uses OPIE challenges

SYNOPSIS

opiesu [-f] [-c] [*user_name*]

DESCRIPTION

opiesu is a replacement for the su(1) program that issues OPIE challenges and uses OPIE responses. It is downward compatible with keysu(1) from the Bellcore S/Key Version 1 distribution and the su(1) program from the 4.3BSD Net/2 distribution.

Unlike other OPIE programs, *opiesu* always requires an OPIE response and will not accept a normal password.

OPTIONS

-f	If the invoked shell is csh(1), this option prevents it from reading the ".cshrc" file. (The [f] option may be passed as a shell argument after the login name, so this option is redundant and obsolescent.)
-c	Set console mode where the user is expected to have secure access to the system. In console mode, you will be asked to input your password directly instead of having to use an OPIE calculator. If you do not have secure access to the system (i.e., you are not on the system's console), you are volunteering your password to attackers by using this mode.
user_name	The name of the user to become. The default is root.

EXAMPLE

```
wintermute$ opiesu kebe
otp-md5 498 wi910502
(OTP response required)
kebe's password: (echo on)
kebe's password: RARE GLEN HUGH BOYD NECK MOLL
wintermute#
```

FILES

/etc/opiekeys Database of information for OPIE system

SEE ALSO

su(1), opie(4), opiekey(1), opieinfo(1), opiesu(1), opielogin(1), opieftpd(8), opiekeys(5), opieaccess(5)

AUTHOR

Bellcore's S/Key was written by Phil Karn, Neil M. Haller, and John S. Walden of Bellcore. DES key crunching contributed by Marcus J. Ranum of TIS. OPIE was created at NRL by Randall Atkinson, Dan McDonald, and Craig Metz.

S/Key is a trademark of Bell Communications Research (Bellcore).

THE LOG DAEMON MANUAL PAGES

The following section provides manual pages for the current implementation of the Log Daemon tools.

FTPD

ftpd—DARPA Internet File Transfer Protocol server

SYNOPSIS

/etc/ftpd [**-d**] [**-S**] [-t timeout] [-T maxtimeout]

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(5).

If the **-d** option is specified, debugging information is written to the syslog.

If the **-S** option is specified, each anonymous file transfer is logged in /var/adm/ftp-log.

The ftp server will timeout an inactive session after 15 minutes. If the **-t** option is specified, the inactivity timeout period will be set to timeout seconds. A client may also request a different timeout period; the maximum period allowed may be set to timeout seconds with the **-T** option. The default limit is 2 hours.

The ftp server currently supports the following ftp requests; case is not distinguished:

Request	*Description*
ABOR	Abort previous command
ACCT	Specify account (ignored)
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 (“ls -lgA”)
MKD	Make a directory
MDTM	Show last modification time of file
MODE	Specify data transfer \fImode\fP
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	Non-standard commands (see next section)
SIZE	Return size of file
STAT	Return status of server
STOR	Store a file
STOU	Store a file with a unique name
STRU	Specify data transfer \fIstructure\fP
SYST	Show operating system type of server system
TYPE	Specify data transfer \fItype\fP

continues

Request	*Description*
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)

The following non-standard or Unix specific commands are supported by the SITE request:

Request	*Description*
UMASK	Change umask. \fIE.g.\fP SITE UMASK 002
IDLE	Set idle-timer. \fIE.g.\fP SITE IDLE 60
CHMOD	Change mode of a file. \fIE.g.\fP SITE CHMOD 755 filename
HELP	Give help information. \fIE.g.\fP SITE HELP

Note: SITE requests are disabled in case of anonymous logins.

The remaining ftp requests specified in Internet RFC 959 are recognized, but not implemented.

MDTM and SIZE are not specified in RFC 959, but will appear in the next updated FTP RFC.

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 interprets file names according to the "globbing" conventions used by csh(1). This allows users to utilize the metacharacters ``*?[]{}~''.

Ftpd authenticates users according to four rules.

1. The user name must be in the password database, /etc/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. The user name must not appear in the file/etc/ftpusers.
3. The user must have a standard shell returned by getusershell(3).

4. If the user name is "anonymous" or "ftp," an anonymous ftp account must be present in the password file (user "ftp"). In this case the user is allowed to log in by specifying any password (by convention this is given as the client host's name).

In the last case, ftpd takes special measures to restrict the client's access privileges. The server performs a chroot(2) command to the home directory of the "ftp" user. The process umask is changed so that files delivered by an anonymous user cannot be retrieved by an anonymous user. In order that system security is not breached, it is recommended that the "ftp" subtree be constructed with care; the following rules are recommended.

~ftp)	Make the home directory owned by the super-user (not "ftp") and unwritable by anyone. In fact, there is no need for any file or directory below ~ftp to be owned by "ftp."
~ftp/bin)	Make this directory owned by the super-user and unwritable by anyone. The program ls(1) must be present to support the list command. This program should have mode 111.
~ftp/etc)	Make this directory owned by the super-user and unwritable by anyone. The file motd is printed after successful login. The files passwd(5) and group(5) must be present for the ls command to be able to produce owner names rather than numbers. The password field in passwd is not used, and should not contain real encrypted passwords. These files should be mode 444, and should be owned by the super-user.
~ftp/pub)	If you make this directory world writable (mode 1777), it is recommended that you run a "cron" job to wipe files delivered after anonymous login.

SEE ALSO

ftp(1), getusershell(3), syslogd(8)

BUGS

The anonymous account is inherently dangerous and should be avoided when possible.

The server must run as the super-user to create sockets with privileged port numbers. It maintains an effective user id of the logged-in user, reverting to the super-user only when binding addresses to sockets. The possible security holes have been extensively scrutinized, but are possibly incomplete.

KEY

key—Compute responses to S/Key challenges

SYNOPSIS

key [-n count] sequence seed

DESCRIPTION

key takes an S/Key sequence number and seed as command-line arguments, prompts for the user's secret password, and formats the response as English words.

OPTIONS

-n Count the number of one-time access passwords to print. The default is one.

EXAMPLE

```
>key -n 5 99 th91334
Enter password: <your secret password is entered here>
OMEN US HORN OMIT BACK AHOY
.... 4 more passwords.
```

SEE ALSO

skey(1), keyinit(1), keysu(1), keyinfo(1)

AUTHOR

Command by Phil Karn, Neil M. Haller, John S. Walden

KEYINFO

keyinfo—Display current S/Key sequence number and seed

SYNOPSIS

keyinfo [username]

DESCRIPTION

keyinfo takes an optional user name and displays the user's current sequence number and seed found in the S/Key database /etc/skeykeys.

The command can be useful when generating a list of passwords for use on a field trip, by combining with the command key in the form

>key -n <number of passwords to print> `ke yinfo`|lpr

EXAMPLE

```
Usage example:
>keyinfo
0098 ws91340
```

ARGUMENTS

username The S/key user to display the information for. The default is to display S/Key information on the user who invokes the command.

SEE ALSO

keyinit(1), key(1)

AUTHOR

Command by Phil Karn, Neil M. Haller, John S. Walden

KEYINIT

keyinit—Change password or add user to S/Key authentication system

SYNOPSIS

keyinit [-s] [<user ID >]

DESCRIPTION

keyinit initializes the system so you can use S/Key one-time passwords to log in. The program will ask you to enter a secret pass phrase; enter a phrase of several words in response. After the S/Key database has been updated you can log in using either your regular Unix password or using S/Key one-time passwords.

fIkeyinit requires you to type a secret password, so it should be used only on a secure terminal. For example, on the console of a workstation. If you are using \fIkeyinit while logged in over an untrusted network, follow the instructions given below with the -s option.

REMOTE LOGIN PROCEDURE

When logging in from another machine you can avoid typing a real password over the network by typing your S/Key pass phrase to the key command on the local machine: the program will respond with the one-time password that you should use to log into the remote machine. This is most conveniently done with cut-and-paste operations using a mouse. Alternatively, you can pre-compute one-time passwords using the key command and carry them with you on a piece of paper.

KEYINIT OPTIONS

-s Set secure mode where the user is expected to have used a secure machine to generate the first one-time password. Without the -s the system will assume you are directly connected over secure communications and prompt you for your secret password. The -s option also allows one to set the seed and count for complete control of the parameters. You can use keyinit -s in combination with the key command to set the seed and count if you do not like the defaults. To do this run keyinit in one window and put in your count and seed, then run key in another window to generate the correct 6 English words for that count and seed. You can then "cut" and "paste" them or copy them into the keyinit window.

<user ID> The ID for the user to be changed/added

FILES

/etc/skeykeys, database of information for S/Key system

SEE ALSO

skey(1), key(1), keysu(1), keyinfo(1)

AUTHOR

Command by Phil Karn, Neil M. Haller, John S. Walden

REXECD

rexecd—Remote execution server

SYNOPSIS

/etc/rexecd

DESCRIPTION

Rexecd is the server for the rexec(3) routine. The server provides remote execution facilities with authentication based on user names and passwords.

Rexecd listens for service requests at the port indicated in the "exec" service specification; see services(5). When a service request is received the following protocol is initiated:

1. The server reads characters from the socket up to a null ('\e0') byte. The resultant string is interpreted as an ASCII number, base 10.
2. If the number received in step 1 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.

3. A null terminated user name of at most 16 characters is retrieved on the initial socket.
4. A null terminated, unencrypted password of at most 16 characters is retrieved on the initial socket.
5. 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.
6. Rexecd then validates the user as is done at login time and, if the authentication was successful, changes to the user's home directory, and establishes the user and group protections of the user. If any of these steps fail the connection is aborted with a diagnostic message returned.
7. 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 rexecd.

DIAGNOSTICS

Except for the last one listed below, all diagnostic messages are returned on the initial socket, after which any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 7 above upon successful completion of all the steps prior to the command execution).

"username too long"	The name is longer than 16 characters.
"password too long"	The password is longer than 16 characters.
"command too long "	The command line passed exceeds the size of the argument list (as configured into the system).
"Login incorrect."	No password file entry for the user name existed.
"Password incorrect."	The wrong password was supplied.
"No remote directory."	The chdir command to the home directory failed.
"Try again."	A fork by the server failed.
"<shellname>: ..."	The user's login shell could not be started. This message is returned on the connection associated with the stderr, and is not preceded by a flag byte.

SEE ALSO

rexec(3)

BUGS

A facility to allow all data and password exchanges to be encrypted should be present.

RLOGIND

rlogind—Remote login server

SYNOPSIS

/etc/rlogind [-ln]

DESCRIPTION

Rlogind is the server for the rlogin(1) program. The server provides a remote login facility with authentication based on privileged port numbers from trusted hosts.

Rlogind listens for service requests at the port indicated in the "login" service specification; see services(5). 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 512–1023, the server aborts the connection.
2. The server checks the client's source address and requests the corresponding host name (see gethostbyaddr(3), hosts(5), and named(8)). If the hostname cannot be determined, the dot-notation representation of the host address is used.

Once the source port and address have been checked, rlogind proceeds with the authentication process described in rshd(8C). It then allocates a pseudo terminal (see pty(4)) and manipulates file descriptors so that the slave half of the pseudo terminal becomes the stdin, stdout, and stderr for a login process. The login process is an instance of the login(1) program, invoked with the -f option if authentication has succeeded. If automatic authentication fails, the user is prompted to log in as if on a standard terminal line. The -l option prevents any authentication based on the user's ".rhosts" file, unless the user is logging in as the super-user.

The parent of the login process manipulates the master side of the pseudo terminal, operating as an intermediary between the login process and the client instance of the rlogin program. In normal operation, the packet protocol described in pty(4) is invoked to provide ^S/^Q type facilities and propagate interrupt signals to the remote programs. The login process propagates the client terminal's baud rate and terminal type, as found in the environment variable, "TERM"; see environ(7).

The screen or window size of the terminal is requested from the client, and window size changes from the client are propagated to the pseudo terminal.

Transport-level keepalive messages are enabled unless the -n option is present. The use of keepalive messages allows sessions to be timed out if the client crashes or becomes unreachable.

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the stderr, after which any network connections are closed. An error is indicated by a leading byte with a value of 1.

"Try again."	A fork by the server failed.
"/bin/sh: ..."	The user's login shell could not be started.

SEE ALSO

ruserok(3), rshd(8)

BUGS

The authentication procedure used here assumes the integrity of each client machine and the connecting medium. This is insecure, but is useful in an "open" environment.

A facility to allow all data exchanges to be encrypted should be present.

A more extensible protocol should be used.

RSHD

rshd—Remote shell server

SYNOPSIS

/etc/rshd

DESCRIPTION

Rshd is the server for the rcmd(3X) routine and, consequently, for the rsh(1C) 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(5). 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 ('\e0') 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 (see gethostbyaddr(3N), hosts(5), and named(8)). If the hostname cannot be determined, the dot-notation representation of the host address is used.

5. A null terminated user name of at most 16 characters 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 at most 16 characters 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 the user according to the following steps. The local (server-end) user name is looked up in the password file and a chdir is performed to the user's home directory. If either the lookup or chdir fail, the connection is terminated. If the user is not the super-user, (user id 0), the file /etc/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 the user is 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.

DIAGNOSTICS

Except for the last one listed below, all diagnostic messages are returned on the initial socket, after which any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 9 above upon successful completion of all the steps prior to the execution of the login shell).

"locuser too long"	The name of the user on the client's machine is longer than 16 characters.
"remuser too long"	The name of the user on the remote machine is longer than 16 characters.
"command too long "	The command line passed exceeds the size of the argument list (as configured into the system).
"Login incorrect."	No password file entry for the user name existed.
"No remote directory."	The chdir command to the home directory failed.
"Permission denied."	The authentication procedure described above failed.

"Can't make pipe."	The pipe needed for the stderr wasn't created.
"Try again."	A fork by the server failed.
"<shellname>: ..."	The user's login shell could not be started. This message is returned on the connection associated with the stderr and is not preceded by a flag byte.

SEE ALSO

rsh(1), rcmd(3)

BUGS

The authentication procedure used here assumes the integrity of each client machine and the connecting medium. This is insecure, but is useful in an "open" environment.

A facility to allow all data exchanges to be encrypted should be present.

A more extensible protocol should be used.

SKEY.ACCESS

skey.access—S/Key password control table

DESCRIPTION

The S/Key password control table (/etc/skey.access) is used by login-like programs to determine when Unix passwords may be used to access the system. When the table does not exist, there are no password restrictions. The user may enter the Unix password or the S/Key one. When the table does exist, Unix passwords are permitted only when explicitly specified.

For the sake of sanity, Unix passwords are always permitted on the systems console.

TABLE FORMAT

The format of the table is one rule per line. Rules are matched in order. The search terminates when the first matching rule is found, or when the end of the table is reached.

Rules have the form:

permit condition condition...

deny condition condition...

where permit and deny may be followed by zero or more conditions. Comments begin with a '#\' character, and extend through the end of the line. Empty lines or lines with only comments are ignored. A rule is matched when all conditions are satisfied. A rule without conditions is always satisfied. For example, the last entry could be a line with just the word deny on it.

CONDITIONS

"hostname wzv.win.tue.nl"	True when the login comes from host wzv.win.tue.nl. See the WARNINGS section below.
"internet 131.155.210.0 255.255.255.0"	True when the remote host has an internet address in network 131.155.210. The general form of a net/mask rule is: internet net mask The expression is true when the host has an Internet address for which the bitwise and of address and mask equals net. See the WARNINGS section below.
"port ttya"	True when the login terminal is equal to /dev/ttya. Remember that Unix passwords are always permitted with logins on the system console.
"user uucp"	True when the user attempts to log in as uucp.
"group wheel"	True when the user's primary group is wheel, or when the user is explicitly listed in the group file under the wheel group.

COMPATIBILITY

For the sake of backwards compatibility, the Internet keyword may be omitted from net/mask patterns.

WARNINGS

Several rule types depend on host name or address information obtained through the network. What follows is a list of conceivable attacks to force the system to permit Unix passwords.

"Host address spoofing (source routing)"	An intruder configures a local interface to an address in a trusted network and connects to the victim using that source address. Given the wrong client address, the victim draws the wrong conclusion from rules based on host addresses or from rules based on host names derived from addresses.

	Remedies: (1) do not permit Unix passwords with network logins; (2) use network software that discards source routing information (e.g., a tcp wrapper).
	Almost every network server must look up the client host name using the client network address. The next obvious attack therefore is:
"Host name spoofing (bad PTR record)"	An intruder manipulates the name server system so that the client network address resolves to the name of a trusted host. Given the wrong host name, the victim draws the wrong conclusion from rules based on host names, or from rules based on addresses derived from host names.
	Remedies: (1) do not permit Unix passwords with network logins; (2) use network software that verifies that the hostname resolves to the client network address (e.g., a tcp wrapper).
	Some applications, such as the Unix login program, must look up the client network address using the client host name. In addition to the previous two attacks, this opens up yet another possibility:
"Host address spoofing (extra A record)"	An intruder manipulates the name server system so that the client host name (also) resolves to a trusted address.
	Remedies: (1) do not permit Unix passwords with network logins; (2) the skeyaccess() routines ignore network addresses that appear to belong to someone else.

DIAGNOSTICS

Syntax errors are reported to the syslogd. When an error is found the rule is skipped.

FILES

/etc/skey.access, password control table

AUTHOR

Wietse Venema
Eindhoven University of Technology
The Netherlands

SKEYSH

skeysh—S/Key login shell

SYNOPSIS

/usr/etc/skeysh [username]

DESCRIPTION

skeysh is a solution for sites that cannot replace the login program. Instead, one sets up an unprivileged dummy account with skeysh as its login shell.

A user first logs into the S/Key dummy account. Skeysh then prompts for the user's real account name and presents the corresponding S/Key challenge. When the user produces the correct responses, the program invokes the user's login shell after performing user-specific activities: updating of environment variables; updating of the last login, wtmp, and utmp files; looking at the motd and mail files.

AUTHOR

Command by Wietse Venema, idea from *Hobbit*.

SU

su—substitute user identity

SYNOPSIS

su [-**Kflm**] [login]

DESCRIPTION

Su requests the Kerberos password for login (or for login.root, if no login is provided), and switches to that user and group ID after obtaining a Kerberos ticket-granting ticket. A shell is then executed. Su will resort to the local password file to find the password for login if there is a Kerberos error. If su is executed by root, no password is requested, and a shell with the appropriate user ID is executed; no additional Kerberos tickets are obtained.

By default, the environment is unmodified with the exception of USER, HOME, and SHELL. HOME and SHELL are set to the target login's default values. USER is set to the target login, unless the target login has a user ID of 0, in which case it is unmodified. The invoked shell is the target login's. This is the traditional behavior of su.

The options are as follows:

-K Do not attempt to use Kerberos to authenticate the user.

-f If the invoked shell is csh(1), this option prevents it from reading the '.cshrc' file.

-l Simulate a full login.

The environment is discarded except for HOME, SHELL, PATH, TERM, and USER. HOME and SHELL are modified as above. USER is set to the target login. PATH is set to '/bin:/usr/bin' . TERM is imported from your current environment. The invoked shell is the target login's, and su will change the directory to the target login's home directory.

-m Leave the environment unmodified.

The invoked shell is your login shell, and no directory changes are made. As a security precaution, if the target user's shell is a non-standard shell (as defined by getusershell(3)) and the caller's real uid is non-zero, su will fail.

The -l and -m options are mutually exclusive; the last one specified overrides any previous ones.

Only users in group 0 (normally 'wheel') can su to 'root' .

By default (unless the prompt is reset by a startup file) the super-user prompt is set to '#' to remind one of its awesome power.

SEE ALSO

csh(1), login(1), sh(1), kinit(1), kerberos(1), passwd(5), group(5), environ(7)

ENVIRONMENT

Environment variables used by su:

HOME Default home directory of real user ID unless modified as specified above.

PATH Default search path of real user ID unless modified as specified above.

TERM Provides terminal type, which may be retained for the substituted user ID.

USER The user ID is always the effective ID (the target user ID) after a su unless the user ID is 0 (root).

HISTORY

A su command appeared in Version 7 AT&T Unix. The version described here is an adaptation of the MIT Athena Kerberos command.

TELNETD

telnetd—DARPA TELNET protocol server

SYNOPSIS

/etc/telnetd

DESCRIPTION

Telnetd is a server which supports the DARPA standard TELNET virtual terminal protocol. Telnetd is invoked by the Internet server (see inetd(8)), normally for requests to connect to the TELNET port as indicated by the /etc/services file (see services(5)).

Telnetd operates by allocating a pseudo-terminal device (see pty(4)) for a client, then creating a login process which has the slave side of the pseudo-terminal as stdin, stdout, and stderr. Telnetd manipulates the master side of the pseudo-terminal, implementing the TELNET protocol and passing characters between the remote client and the login process.

When a TELNET session is started up, telnetd sends TELNET options to the client side indicating a willingness to do remote echo of characters, to suppress go ahead, and to receive terminal type information from the remote client. If the remote client is willing, the remote terminal type is propagated in the environment of the created login process. The pseudo-terminal allocated to the client is configured to operate in "cooked" mode, and with XTABS and CRMOD enabled (see tty(4)).

Telnetd is willing to do: echo, binary, suppress go ahead, window size, and timing mark. Telnetd is willing to have the remote client do: binary, terminal type, and suppress go ahead.

SEE ALSO

telnet(1)

BUGS

Some TELNET commands are only partially implemented.

Because of bugs in the original 4.2 BSD telnet(1), telnetd performs some dubious protocol exchanges to try to discover if the remote client is, in fact, a 4.2 BSD telnet(1).

Binary mode has no common interpretation except between similar operating systems (Unix in this case).

The terminal type name received from the remote client is converted to lowercase.

The packet interface to the pseudo-terminal (see pty(4)) should be used for more intelligent flushing of input and output queues.

Telnetd never sends TELNET go ahead commands.