NAME

tnftp — Internet file transfer program

SYNOPSIS

tnftp [**-46AadefginpRtVv?**] [**-N** *netrc*] [**-o** *output*] [**-P** *port*] [**-q** *quittime*]
[**-r** *retry*] [**-s** *srcaddr*] [**-T** *dir*,*max*[,*inc*]] [**-x** *xfersize*]
[[*user*@]*host* [*port*]] [[*user*@]*host*:[*path*][/]] [file:///*path*]
[ftp://[*user*[:*password*]@]*host*[:*port*]/*path*[/][;type=*type*]]
[http://[*user*[:*password*]@]*host*[:*port*]/*path*]
[https://[*user*[:*password*]@]*host*[:*port*]/*path*] ...
tnftp -u *url file ...*

DESCRIPTION

tnftp is the user interface to the Internet standard File Transfer Protocol. The program allows a user to transfer files to and from a remote network site.

The last five arguments will fetch a file using the FTP or HTTP protocols, or by direct copying, into the current directory. This is ideal for scripts. Refer to**A UTO-FETCHING FILES** below for more information.

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

-4 Forces **tnftp** to only use IPv4 addresses.

-6 Forces **tnftp** to only use IPv6 addresses.

-A Force active mode FTP. By default, **tnftp** will try to use passive mode FTP and fall back to active mode if passive is not supported by the server. This option causes**tnftp** to al ways use an active connection. It is only useful for connecting to very old servers that do not implement passive mode properly.

-a Causes **tnftp** to bypass normal login procedure, and use an anonymous login instead.

-d Enables debugging.

-e Disables command line editing. This is useful for Emacs ange-ftp mode.

-f Forces a cache reload for transfers that go through the FTP or HTTP proxies.

-g Disables file name globbing.

-i Turns off interactive prompting during multiple file transfers.

-N *netrc* Use *netrc* instead of ~/.netrc. Refer to**THE .netr c FILE** for more information.

-n Restrains **tnftp** from attempting “auto-login” upon initial connection for non auto-fetch transfers. If auto-login is enabled,**tnftp** will check the .netrc (see belo w) file in the user’s home directory for an entry describing an account on the remote machine. If no entry exists, **tnftp** 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. To override the auto-login for auto-fetch transfers, specify the username (and optionally, password) as appropriate.

-o *output*
When auto-fetching files, save the contents in *output*. *output* is parsed according to the **FILE NAMING CONVENTIONS** below. If*output* is not ‘-’ or doesn’ t start with ‘|’, then only the first file specified will be retrieved into *output*; all other files will be retrieved into the basename of their remote name.

-P *port* Sets the port number to *port*.

-p Enable passive mode operation for use behind connection filtering firewalls. This option has been deprecated as **tnftp** now tries to use passive mode by default, falling back to active mode if the server does not support passive connections.

-q *quittime*
Quit if the connection has stalled for *quittime* seconds.

-R Restart all non-proxied auto-fetches.

-r *wait* Retry the connection attempt if it failed, pausing for *wait* seconds.

-s *srcaddr*
Uses *srcaddr* as the local IP address for all connections.

-t Enables packet tracing.

-T *direction*,*maximum*[,*increment*]
Set the maximum transfer rate for *direction* to *maximum* bytes/second, and if specified, the increment to *increment* bytes/second. Refer to **rate** for more information.

-u *url file ...*
Upload files on the command line to *url* where *url* is one of the '`ftp://`' URL types as supported by auto-fetch (with an optional target filename for single file uploads), and *file* is one or more local files to be uploaded.

-V Disable **verbose** and **progress**, overriding the default of enabled when output is to a terminal.

-v Enable **verbose** and **progress**. This is the default if output is to a terminal (and in the case of **progress**, **tnftp** is the foreground process). Forces **tnftp** to show all responses from the remote server, as well as report on data transfer statistics.

-x *xfersize*
Set the size of the socket send and receive buffers to *xfersize*. Refer to**xferbuf** for more information.

-? Display help to stdout, and exit.

The client host with which **tnftp** is to communicate may be specified on the command line. If this is done, **tnftp** will immediately attempt to establish a connection to an FTP server on that host; otherwise, **tnftp** will enter its command interpreter and await instructions from the user. When**tnftp** is a waiting commands from the user the prompt `ftp>` is provided to the user. The following commands are recognized by **tnftp**:

! [*command* [*args*]]
Invoke an interactive shell on the local machine. If there are arguments, the first is taken to be a command to execute directly, with the rest of the arguments as its arguments.

$ *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, the user 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 **tnftp**. An end of file will also terminate the session and exit.

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

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.

chmod *mode remote-file*
Change the permission modes of the file *remote-file* on the remote system to *mode*.

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 (the def ault), 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.

debug [*debug-value*]
Toggle debugging mode. If an optional *debug-value* is specified it is used to set the debugging level. When debugging is on, **tnftp** prints each command sent to the remote machine, preceded by the string `-->`.

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

dir [*remote-path* [*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`. If *remote-path* is left unspecified, the current working directory is used. If interactive prompting is on, **tnftp** will prompt the user to verify that the last argument is indeed the target local file for receiving **dir** output. If no local file is specified, or if *local-file* is '**-**', the output is sent to the terminal.

disconnect
A synonym for **close**.

edit Toggle command line editing, and context sensitive command and file completion. This is automatically enabled if input is from a terminal, and disabled otherwise.

epsv, **epsv4**, **epsv6**
Toggle the use of the extended EPSV and EPRT commands on all IP, IPv4, and IPv6 connections respectively. First tryEPSV/EPRT, and then PASV/PORT. This is enabled by def ault. If an extended command fails then this option will be temporarily disabled for the duration of the current connection, or until **epsv**, **epsv4**, or **epsv6** is executed again.

exit A synonym for **bye**.

features Display what features the remote server supports (using the FEAT command).

fget *localfile*
Retrieve the files listed in *localfile*, which has one line per filename.

form *format*
Set the file transfer **form** to *format*. The default (and only supported) format is "non-print".

ftp *host* [*port*]
A synonym for **open**.

gate [*host* [*port*]]
Toggle gate-ftp mode, which used to connect through the TIS FWTK and Gauntlet FTP proxies. This will not be permitted if the gate-ftp server hasn't been set (either explicitly by the user, or from the FTPSERVER environment variable). If*host* is gi ven, then gate-ftp mode will be enabled, and the gate-ftp server will be set to *host*. If*port* is also gi ven, that will be used as the port to connect to on the gate-ftp server.

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**, **mput**, and **mreget**. 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(1). For **mdelete**, **mget**, and **mreget**, 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**, **mput** and **mreget** are not meant to transfer entire directory subtrees of files. That can be done by transferring a tar(1) archive of the subtree (in binary mode).

hash [*size*]
Toggle hash-sign ('#') printing for each data block transferred. The size of a data block defaults to 1024 bytes. This can be changed by specifying *size* in bytes. Enabling **hash** disables **progress**.

help [*command*]
Print an informative message about the meaning of *command*. If no argument is given, **tnftp** prints a list of the known commands.

idle [*seconds*]
Set the inactivity timer on the remote server to *seconds* seconds. If *seconds* is omitted, the current inactivity timer is printed.

image A synonym for **binary**.

lcd [*directory*]
Change the working directory on the local machine. If no *directory* is specified, the user's home directory is used.

less *file*
A synonym for **page**.

lpage *local-file*
Display *local-file* with the program specified by the **set pager** option.

lpwd Print the working directory on the local machine.

ls [*remote-path* [*local-file*]]
A synonym for **dir**.

macdef *macro-name*
Define a macro. Subsequent lines are stored as the macro *macro-name*; a null line (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. Macro names can be a maximum of 8 characters. Macros are only applicable to the current session they are defined within (or if defined outside a session, to the session invoked with the next **open** command), and remain defined until a **close** command is executed. To invoke a macro, use the **$** command (see above).

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 '$' followed by an 'i' signals the macro processor that the executing macro is to be looped. On the first pass '$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, **tnftp** will prompt the user 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 e xpansion. 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`; new local directories can be created with '`! mkdir directory`'.

mkdir *directory-name*
Make a directory on the remote machine.

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

mlsd [*remote-path*]
Display the contents of *remote-path* (which should default to the current directory if not given) in a machine-parsable form, using `MLSD`. The format of display can be changed with

‘`remopts mlst ...`’.

mlst [*remote-path*]
Display the details about *remote-path* (which should default to the current directory if not given) in a machine-parsable form, using `MLST`. The format of display can be changed with ‘`remopts mlst ...`’.

mode *mode-name*
Set the file transfer **mode** to *mode-name*. The default (and only supported) mode is “stream”.

modtime *remote-file*
Show the last modification time of the file on the remote machine, in `RFC 2822` format.

more *file*
A synonym for **page**.

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.

mreget *remote-files*
As per **mget**, but performs a **reget** instead of **get**.

msend *local-files*
A synonym for **mput**.

newer *remote-file* [*local-file*]
Get the file only if the modification time of the remote file is more recent that the file on the current system. If the file does not exist on the current system, the remote file is considered **newer**. Otherwise, this command is identical to **get**.

nlist [*remote-path* [*local-file*]]
A synonym for **ls**.

nmap [*inpattern outpattern*]
Set or unset the filename mapping mechanism. If no arguments are specified, the filename mapping mechanism is unset. If arguments are specified, remote filenames are mapped during **mput** commands and **put** commands issued without a specified remote target filename. If arguments are specified, local filenames are mapped during **mget** commands and **get** commands issued without a specified local target filename. This command is useful when connecting to a non−UNIX remote computer with different file naming conventions or practices. The mapping follows the pattern set by *inpattern* and *outpattern*.

inpattern is a template for incoming filenames (which may have already been processed according to the **ntrans** and **case** settings). Variable templating is accomplished by including the sequences ‘`$1`’, ‘`$2`’, ..., ‘`$9`’ in *inpattern*. Use ‘`\`’ to pre vent this special treatment of the ‘`$`’ character. All other characters are treated literally, and are used to determine the **nmap** [*inpattern*] variable values. For example, given *inpattern* ‘`$1.$2`’ and the remote file name ‘`mydata.data`’, ‘`$1`’ would have the value ‘`mydata`’, and ‘`$2`’ would have the value ‘`data`’.

The *outpattern* determines the resulting mapped filename. The sequences ‘`$1`’, ‘`$2`’, ..., ‘`$9`’ are replaced by any value resulting from the *inpattern* template. The sequence ‘`$0`’ is replaced by the original filename. Additionally, the sequence “[*seq1*, *seq2*]” is replaced by *seq1* if *seq1* is not a null string; otherwise it is replaced by *seq2*. For example, the command

```
nmap $1.$2.$3 [$1,$2].[$2,file]
```

would yield the output filename '`myfile.data`' for input filenames '`myfile.data`' and '`myfile.data.old`', '`myfile.file`' for the input filename '`myfile`', and '`myfile.myfile`' for the input filename '`.myfile`'. Spaces may be included in *outpattern*, as in the example:

```
nmap $1 sed s/  *$// > $1
```

Use the '\' character to prevent special treatment of the '`$`', '`[`', '`]`', and '`,`' characters.

ntrans [*inchars* [*outchars*]]
Set or unset the filename character translation mechanism. If no arguments are specified, the filename character translation mechanism is unset. If arguments are specified, characters in remote filenames are translated during **mput** commands and **put** commands issued without a specified remote target filename. If arguments are specified, characters in local filenames are translated during **mget** commands and **get** commands issued without a specified local target filename. This command is useful when connecting to a non− UNIX remote computer with different file naming conventions or practices. Characters in a filename matching a character in *inchars* are replaced with the corresponding character in *outchars*. If the character's position in *inchars* is longer than the length of *outchars*, the character is deleted from the file name.

open *host* [*port*]
Establish a connection to the specified *host* FTP server. An optional port number may be supplied, in which case, **tnftp** will attempt to contact an FTP server at that port. If the **set auto-login** option is on (default), **tnftp** will also attempt to automatically log the user in to the FTP server (see below).

page *file*
Retrieve **file** and display with the program specified by the **set pager** option.

passive [**auto**]
Toggle passive mode (if no arguments are given). If **auto** is gi ven, act as if FTPMODE is set to 'auto'. If passive mode is turned on (default), **tnftp** will send a PASV command for all data connections instead of a PORT command. The PASV command requests that the remote server open a port for the data connection and return the address of that port. The remote server listens on that port and the client connects to it. When using the more traditional PORT command, the client listens on a port and sends that address to the remote server, who connects back to it. Passive mode is useful when using **tnftp** through a gateway router or host that controls the directionality of traffic. (Note that though FTP servers are required to support the PASV command by RFC 1123, some do not.)

pdir [*remote-path*]
Perform **dir** [*remote-path*], and display the result with the program specified by the **set pager** option.

pls [*remote-path*]
Perform **ls** [*remote-path*], and display the result with the program specified by the **set pager** option.

pmlsd [*remote-path*]
Perform **mlsd** [*remote-path*], and display the result with the program specified by the **set pager**option.

preserve Toggle preservation of modification times on retrieved files.

progress Toggle display of transfer progress bar. The progress bar will be disabled for a transfer that has *local-file* as '**-**' or a command that starts with '**|**'. Refer to**FILE N AMING CONVENTIONS** for more information. Enabling **progress** disables **hash**.

prompt Toggle interactive prompting. Interactive prompting occurs during multiple file transfers to allow the user to selectively retrieve or store files. If prompting is turned off (default is on), any **mget** or **mput** will transfer all files, and any **mdelete** will delete all files.

When prompting is on, the following commands are available at a prompt:

- **a** Answer 'yes' to the current file, and automatically answer 'yes' to any remaining files for the current command.
- **n** Answer 'no', and do not transfer the file.
- **p** Answer 'yes' to the current file, and turn off prompt mode (as is "prompt off" had been given).
- **q** Terminate the current operation.
- **y** Answer 'yes', and transfer the file.
- **?** Display a help message.

Any other response will answer 'yes' to the current file.

proxy *ftp-command*

Execute an FTP command on a secondary control connection. This command allows simultaneous connection to two remote FTP servers for transferring files between the two servers. The first **proxy** command should be an **open**, to establish the secondary control connection. Enter the command '`proxy ?`' to see other FTP commands executable on the secondary connection. The following commands behave differently when prefaced by **proxy**: **open** will not define new macros during the auto-login process, **close** will not erase existing macro definitions, **get** and **mget** transfer files from the host on the primary control connection to the host on the secondary control connection, and **put**, **mput**, and **append** transfer files from the host on the secondary control connection to the host on the primary control connection. Third party file transfers depend upon support of the FTP protocol PASV command by the server on the secondary control connection.

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 [*arg ...*]

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

rate *direction* [*maximum* [*increment*]]

Throttle the maximum transfer rate to *maximum* bytes/second. If *maximum* is 0, disable the throttle.

direction may be one of:

all Both directions.
get Incoming transfers.
put Outgoing transfers.

maximum can be modified on the fly by *increment* bytes (default: 1024) each time a given signal is received:

SIGUSR1 Increment *maximum* by *increment* bytes.

SIGUSR2 Decrement *maximum* by *increment* bytes. The result must be a positive number.

If *maximum* is not supplied, the current throttle rates are displayed.

Note: **rate** is not yet implemented for ascii mode transfers.

rcvbuf *size*
Set the size of the socket receive buffer to *size*.

recv *remote-file* [*local-file*]
A synonym for **get**.

reget *remote-file* [*local-file*]
reget acts like **get**, except that if *local-file* exists and is smaller than *remote-file*, *local-file* is presumed to be a partially transferred copy of *remote-file* and the transfer is continued from the apparent point of failure. This command is useful when transferring very large files over networks that are prone to dropping connections.

remopts *command* [*command-options*]
Set options on the remote FTP server for *command* to *command-options* (whose absence is handled on a command-specific basis). Remote FTP commands known to support options include: MLST (used for MLSD and MLST).

rename [*from* [*to*]]
Rename the file *from* on the remote machine, to the file *to*.

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

restart *marker*
Restart the immediately following **get** or **put** at the indicated *marker*. On UNIX systems, marker is usually a byte offset into the file.

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

rmdir *directory-name*
Delete a directory on the remote machine.

rstatus [*remote-file*]
With no arguments, show status of remote machine. If *remote-file* is specified, show status of *remote-file* on 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, a '.1' is appended to the name. If the resulting name matches another existing file, a '.2' is appended to the original name. If this process continues up to .99, an error message is 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, **tnftp** will attempt to use a PORT command when establishing a connection for each data transfer. The use ofPORT commands can pre vent delays when performing multiple file transfers. If the PORT command fails, **tnftp** 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.

set [*option value*]
Set *option* to *value*. If*option* and *value* are not gi ven, display all of the options and their values. The currently supported options are:

anonpass Defaults to $FTPANONPASS

ftp_proxy Defaults to $ftp_proxy.

http_proxy Defaults to $http_proxy.

https_proxy Defaults to $https_proxy.

no_proxy Defaults to $no_proxy.

pager Defaults to $PAGER.

prompt Defaults to $FTPPROMPT.

rprompt Defaults to $FTPRPROMPT.

site [*arg ...*]
The arguments specified are sent, verbatim, to the remote FTP server as a SITE command.

size *remote-file*
Return size of *remote-file* on remote machine.

sndbuf *size*
Set the size of the socket send buffer to *size*.

status Show the current status of **tnftp**.

struct *struct-name*
Set the file transfer *structure* to *struct-name*. The default (and only supported) structure is "file".

sunique Toggle storing of files on remote machine under unique file names. The remote FTP server must support FTP protocol STOU command for successful completion. The remote server will report unique name. Default value is off.

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.

throttle A synonym for **rate**.

trace Toggle packet tracing.

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

umask [*newmask*]
Set the default umask on the remote server to *newmask*. If*newmask* is omitted, the current umask is printed.

unset *option*
Unset *option*. Refer to**set** for more information.

usage *command*
Print the usage message for *command*.

user *user-name* [*password* [*account*]]
Identify yourself to the remote FTP server. If the*password* is not specified and the serv er requires it, **tnftp** will prompt the user for it (after disabling local echo). If an *account* field is not specified, and the FTP server requires it, the user 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 **tnftp** 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 the user. In addition, if verbose is on, when a file transfer completes, statistics regarding the efficiency of the transfer are reported. By default, verbose is on.

xferbuf *size*
Set the size of the socket send and receive buffers to *size*.

? [*command*]
A synonym for **help**.

Command arguments which have embedded spaces may be quoted with quote '"' marks.

Commands which toggle settings can take an explicit **on** or **off** argument to force the setting appropriately.

Commands which take a byte count as an argument (e.g., **hash**, **rate**, and **xferbuf**) support an optional suffix on the argument which changes the interpretation of the argument. Supported suffixes are:

- b Causes no modification. (Optional)
- k Kilo; multiply the argument by 1024
- m Mega; multiply the argument by 1048576
- g Giga; multiply the argument by 1073741824

If **tnftp** receives a SIGINFO (see the **status** argument of stty(1)) or SIGQUIT signal whilst a transfer is in progress, the current transfer rate statistics will be written to the standard error output, in the same format as the standard completion message.

AUTO-FETCHING FILES

In addition to standard commands, this version of **tnftp** supports an auto-fetch feature. To enable auto-fetch, simply pass the list of hostnames/files on the command line.

The following formats are valid syntax for an auto-fetch element:

[*user*@]*host*:[*path*][/]
"Classic" FTP format.

If *path* contains a glob character and globbing is enabled, (see **glob**), then the equivalent of 'mget path' is performed.

If the directory component of *path* contains no globbing characters, it is stored locally with the name basename (see basename(1)) of **path**, in the current directory. Otherwise, the full remote name is used as the local name, relative to the local root directory.

`ftp://`[*user*[:*password*]@]*host*[:*port*]/*path*[/][;type=*type*]

An FTP URL, retrieved using the FTP protocol if **set ftp_proxy** isn't defined. Otherwise, transfer the URL using HTTP via the proxy defined in **set ftp_proxy**. If **set ftp_proxy** isn' t defined and *user* is given, login as *user*. In this case, use *password* if supplied, otherwise prompt the user for one.

If a suffix of '`;type=A`' or '`;type=I`' is supplied, then the transfer type will take place as ascii or binary (respectively). The default transfer type is binary.

In order to be compliant with RFC 3986, **tnftp** interprets the *path* part of an '`ftp://`' autofetch URL as follows:

- The '`/`' immediately after the *host*[:*port*] is interpreted as a separator before the *path*, and not as part of the *path* itself.
- The *path* is interpreted as a '`/`'-separated list of name components. For all but the last such component, **tnftp** performs the equivalent of a **cd** command. For the last path component, **tnftp** performs the equivalent of a **get** command.
- Empty name components, which result from '`//`' within the *path*, or from an e xtra '`/`' at the beginning of the *path*, will cause the equivalent of a **cd** command without a directory name. This is unlikely to be useful.
- Any '`%`*XX*' codes (per RFC 3986) within the path components are decoded, with *XX* representing a character code in hexadecimal. This decoding takes place after the *path* has been split into components, but before each component is used in the equivalent of a **cd** or **get** command. Some often-used codes are '`%2F`' (which represents '`/`') and '`%7E`' (which represents '`~`').

The above interpretation has the following consequences:

- The path is interpreted relative to the default login directory of the specified user or of the 'anonymous' user. If the/ directory is required, use a leading path of '`%2F`'. If a user' s home directory is required (and the remote server supports the syntax), use a leading path of '`%7E`*user*`/`'. For example, to retrieve `/etc/motd` from '`localhost`' as the user '`myname`' with the password '`mypass`', use '`ftp://myname:mypass@localhost/%2fetc/motd`'
- The exact **cd** and **get** commands can be controlled by careful choice of where to use '`/`' and where to use '`%2F`' (or '`%2f`'). For example, the following URLs correspond to the equivalents of the indicated commands:

ftp://host/dir1/dir2/file	"cd dir1", "cd dir2", "get file".
ftp://host/%2Fdir1/dir2/file	"cd /dir1", "cd dir2", "get file".
ftp://host/dir1%2Fdir2/file	"cd dir1/dir2", "get file".
ftp://host/%2Fdir1%2Fdir2/file	"cd /dir1/dir2", "get file".
ftp://host/dir1%2Fdir2%2Ffile	"get dir1/dir2/file".
ftp://host/%2Fdir1%2Fdir2%2Ffile	"get /dir1/dir2/file".

- You must have appropriate access permission for each of the intermediate directories that is used in the equivalent of a **cd** command.

`http://`[*user*[:*password*]@]*host*[:*port*]/*path*

An HTTP URL, retrieved using the HTTP protocol. If **set http_proxy** is defined, it is used as a URL to an HTTP proxy server. If HTTP authorization is required to retrieve *path*, and *user* (and optionally *password*) is in the URL, use them for the first attempt to authenticate.

`https://`[*user*[`:`*password*]`@`]*host*[`:`*port*]`/`*path*
An HTTPS URL, retrieved using the HTTPS protocol. If **`set https_proxy`** is defined, it is used as a URL to an HTTPS proxy server. If HTTPS authorization is required to retrieve *path*, and *user* (and optionally *password*) is in the URL, use them for the first attempt to authenticate. There is currently no certificate validation and verification.

`file:///`*path*
A local URL, copied from */path* on the local host.

`about:`*topic*
Display information regarding *topic*; no file is retrieved for this auto-fetched element. Supported values include:

`about:ftp` Information about **tnftp**.

`about:version` The version of **tnftp**. Useful to provide when reporting problems.

Unless noted otherwise above, and **`-o`** *output* is not given, the file is stored in the current directory as the basename(1) of *path*. Note that if a HTTP redirect is received, the fetch is retried using the new target URL supplied by the server, with a corresponding new *path*. Using an explicit **`-o`** *output* is recommended, to avoid writing to unexpected file names.

If a classic format or an FTP URL format has a trailing '`/`' or an empty*path* component, then **tnftp** will connect to the site and **`cd`** to the directory given as the path, and leave the user in interactive mode ready for further input. This will not work if **`set ftp_proxy`** is being used.

Direct HTTP transfers use HTTP 1.1. Proxied FTP and HTTP transfers use HTTP 1.0.

If **`-R`** is given, all auto-fetches that don't go via the FTP or HTTP proxies will be restarted. For FTP, this is implemented by using **`reget`** instead of **`get`**. For HTTP, this is implemented by using the 'Range: bytes=' HTTP/1.1 directive.

If WWW or proxy WWW authentication is required, you will be prompted to enter a username and password to authenticate with.

When specifying IPv6 numeric addresses in a URL, you need to surround the address in square brackets. E.g.: '`ftp://[::1]:21/`'. This is because colons are used in IPv6 numeric address as well as being the separator for the port number.

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, the prompt will not appear until the remote server has completed sending the requested file.

If the terminal interrupt key sequence is used whilst **tnftp** is awaiting a reply from the remote server for the ABOR processing, then the connection will be closed. This is different from the traditional behaviour (which ignores the terminal interrupt during this phase), but is considered more useful.

FILE NAMING CONVENTIONS

Files specified as arguments to **tnftp** commands are processed according to the following rules.

1. If the file name '**`-`**' is specified, the*stdin* (for reading) or *stdout* (for writing) is used.
2. If the first character of the file name is '`|`', the remainder of the argument is interpreted as a shell command. **tnftp** then forks a shell, using popen(3) with the argument supplied, and reads (writes) from the stdout (stdin). If the shell command includes spaces, the argument must be quoted; e.g.

‘"| ls −lt"’. A particularly useful example of this mechanism is: ‘dir "" |more’.

3. Failing the above checks, if globbing is enabled, local file names are expanded according to the rules used in the csh(1); see the **glob** command. If the **tnftp** command expects a single local file (e.g. **put**), only the first filename generated by the globbing operation is used.

4. For **mget** commands and **get** commands with unspecified local file names, the local filename is the remote filename, which may be altered by a **case**, **ntrans**, or **nmap** setting. The resulting filename may then be altered if **runique** is on.

5. For **mput** commands and **put** commands with unspecified remote file names, the remote filename is the local filename, which may be altered by a **ntrans** or **nmap** setting. The resulting filename may then be altered by the remote server if **sunique** is on.

FILE TRANSFER PARAMETERS

The FTP specification specifies many parameters which may affect 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 mostly). **tnftp** supports the ascii and image types of file transfer, plus local byte size 8 for **tenex** mode transfers.

tnftp supports only the default values for the remaining file transfer parameters: **mode**, **form**, and **struct**.

THE .netrc FILE

The .netrc file contains login and initialization information used by the auto-login process. It resides in the user’s home directory, unless overridden with the **−N** *netrc* option, or specified in the NETRC environment variable. The following tokens are recognized; they may be separated by spaces, tabs, or new-lines:

machine *name*
: Identify a remote machine *name*. The auto-login process searches the .netrc file for a **machine** token that matches the remote machine specified on the **tnftp** command line or as an **open** command argument. Once a match is made, the subsequent .netrc tok ens 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 user@site
```

thereby giving the user an 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. Note that if this token is present in the .netrc file for any user other than *anonymous*, **tnftp** will abort the auto-login process if the .netrc is readable by anyone besides the user.

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 **tnftp macdef** command functions. A macro is defined with the specified name; its contents begin with the next `.netrc` line and continue until a blank line (consecutive new-line characters) is encountered. Like the other tokens in the `.netrc` file, a **macdef** is applicable only to the **machine** definition preceding it. A **macdef** entry cannot be used by multiple **machine** definitions; rather, it must be defined following each **machine** it is intended to be used with. If a macro named **init** is defined, it is automatically executed as the last step in the auto-login process. For example,

```
default
macdef init
epsv4 off
```

followed by a blank line.

COMMAND LINE EDITING

tnftp supports interactive command line editing, via the `editline`(3) library. It is enabled with the **edit** command, and is enabled by default if input is from a tty. Previous lines can be recalled and edited with the arrow keys, and other GNU Emacs-style editing keys may be used as well.

The `editline`(3) library is configured with a `.editrc` file — refer to `editrc`(5) for more information.

An extra key binding is available to **tnftp** to provide context sensitive command and filename completion (including remote file completion). To use this, bind a key to the `editline`(3) command **ftp-complete**. By default, this is bound to the TAB key.

COMMAND LINE PROMPT

By default, **tnftp** displays a command line prompt of '`ftp> `' to the user. This can be changed with the **set prompt**command.

A prompt can be displayed on the right side of the screen (after the command input) with the **set rprompt** command.

The following formatting sequences are replaced by the given information:

`%/` The current remote working directory.

`%c`[[`0`]*n*], `%.`[[`0`]*n*]
The trailing component of the current remote working directory, or *n* trailing components if a digit *n* is given. If*n* be gins with '`0`', the number of skipped components precede the trailing component(s) in the format "*/<number>trailing*" (for '`%c`') or "*...trailing*" (for '`%.`').

`%M` The remote host name.

`%m` The remote host name, up to the first dot '`.`'.

`%n` The remote user name.

`%%` A single percent character '`%`'.

ENVIRONMENT

tnftp uses the following environment variables.

`FTPANONPASS` Password to send in an anonymous FTP transfer. Defaults to "`` `whoami`@ ``".

`FTPMODE` Overrides the default operation mode. Support values are:

active active mode FTP only

auto automatic determination of passive or active (this is the default)

gate gate-ftp mode

passive passive mode FTP only

`FTPPROMPT` Command-line prompt to use. Defaults to '`ftp> `'. Refer to**COMMAND LINE PROMPT** for more information.

`FTPRPROMPT` Command-line right side prompt to use. Defaults to empty string. Refer to **COMMAND LINE PROMPT** for more information.

`FTPSERVER` Host to use as gate-ftp server when **gate** is enabled.

`FTPSERVERPORT` Port to use when connecting to gate-ftp server when **gate** is enabled. Default is port returned by a `getservbyname`(3) lookup of "`ftpgate/tcp`".

`FTPUSERAGENT` The value to send for the HTTP User-Agent header.

`HOME` For default location of a `.netrc` file, if one exists.

`NETRC` An alternate location of the `.netrc` file.

`PAGER` Used by various commands to display files. Defaults to `more`(1) if empty or not set.

`SHELL` For default shell.

`ftp_proxy` URL of FTP proxy to use when making FTP URL requests (if not defined, use the standard FTP protocol).

See `http_proxy` for further notes about proxy use.

`http_proxy` URL of HTTP proxy to use when making HTTP URL requests. If proxy authentication is required and there is a username and password in this URL, they will automatically be used in the first attempt to authenticate to the proxy.

If "unsafe" URL characters are required in the username or password (for example '`@`' or '`/`'), encode them with `RFC 3986` '`%`*XX*' encoding.

Note that the use of a username and password in `ftp_proxy` and `http_proxy` may be incompatible with other programs that use it (such as `lynx`(1)).

NOTE: this is not used for interactive sessions, only for command-line fetches.

`https_proxy` URL of HTTPS proxy to use when making HTTPS URL requests.

See `http_proxy` for further notes about proxy use.

`no_proxy` A space or comma separated list of hosts (or domains) for which proxying is not to be used. Each entry may have an optional trailing '`:`*port*', which restricts the matching to connections to that port.

EXTENDED PASSIVE MODE AND FIREWALLS

Some firewall configurations do not allow **tnftp** to use extended passive mode. If you find that even a simple **ls** appears to hang after printing a message such as this:

```
229 Entering Extended Passive Mode (|||58551|)
```

then you will need to disable extended passive mode with **epsv4 off**. See the above section **The .netrc File** for an example of how to make this automatic.

SEE ALSO

`getservbyname`(3), `editrc`(5), `services`(5), `ftpd`(8)

STANDARDS

`tnftp` attempts to be compliant with:

`RFC 959`
File Transfer Protocol

`RFC 1123`
Requirements for Internet Hosts - Application and Support

`RFC 1635`
How to Use Anonymous FTP

`RFC 2389`
Feature negotiation mechanism for the File Transfer Protocol

`RFC 2428`
FTP Extensions for IPv6 and NATs

`RFC 2616`
Hypertext Transfer Protocol -- HTTP/1.1

`RFC 2822`
Internet Message Format

`RFC 3659`
Extensions to FTP

`RFC 3986`
Uniform Resource Identifier (URI)

HISTORY

The **`tnftp`** command appeared in 4.2BSD.

Various features such as command line editing, context sensitive command and file completion, dynamic progress bar, automatic fetching of files and URLs, modification time preservation, transfer rate throttling, configurable command line prompt, and other enhancements over the standard BSD **`tnftp`** were implemented in NetBSD 1.3 and later releases by Luke Mewburn ⟨lukem@NetBSD.org⟩.

IPv6 support was added by the WIDE/KAME project (but may not be present in all non-NetBSD versions of this program, depending if the operating system supports IPv6 in a similar manner to KAME).

BUGS

Correct execution of many commands depends upon proper behavior by the remote server.

An error in the treatment of carriage returns in the 4.2BSD ascii-mode transfer code has been corrected. This correction may result in incorrect transfers of binary files to and from 4.2BSD servers using the ascii type. Avoid this problem by using the binary image type.

`tnftp` assumes that all IPv4 mapped addresses (IPv6 addresses with a form like `::ffff:10.1.1.1`) indicate IPv4 destinations which can be handled by `AF_INET` sockets. However, in certain IPv6 network configurations, this assumption is not true. In such an environment, IPv4 mapped addresses must be passed to `AF_INET6` sockets directly. For example, if your site uses a SIIT translator for IPv6-to-IPv4 translation, **`tnftp`** is unable to support your configuration.