wu-ftpd

COLLABORATORS			

	TITLE : wu-ftpd		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		December 8, 2024	

REVISION HISTORY			

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

wu-ftpd

1.1   wu-ftpd 2.4, Amiga Version 37.21

```
W A S H I N G T O N   U N I V E R S I T Y   F T P   D A E M O N
              Version 2.4, Amiga Version 37.21
----------------------------------------------------------------

 Introduction
 Software Requirements
 Installation Guide
 Configuration Files
 Using The Server
 MultiUser Support  - Read this if you have MultiUser installed!!!
 Important Notes
 Common Problems
 Author
 Reporting Bugs
 Copyrights and Credits
 Future Plans
 Revision History
```

1.2   Introduction

Welcome to the Amiga port of the Washington University FTP server. If you
are upgrading from previous releases please read the Revision History
section for a list of new features in this version. Please read the
 Installation Guide even if you think you know everything about the
installation!

I would also highly recommend you to read the Important Notes and
 Common Problems sections.

If you have any problems you can find my address here .

wu-ftpd for the Amiga supports allmost all the Unix wu-ftpd features. The
only exception is the "group access" which is not possible to implement
under current MultiUser versions (hopefully it will be under MultiUser 2.0)
and is useless with AmiTCP's usergroup.library.

1.3 Software Requirements

For proper operation of wu-ftpd the following software is needed:

 – Kickstart and Workbench 2.0 or higher
 – AmiTCP/IP 3.0 beta 2 or higher

MultiUser File System (MuFS) is supported.

You will also need some kind of network link (SLIP, Ethernet, ...) for
obvious reasons (although I did almost all testing of this server with the
local loopback device :-).

To recompile the server you need SAS C 6.51, the AmiTCP network include
files and libraries (supplied in a separate package with AmiTCP 3.0b2) and
the MultiUser include files. You also need GNU Bison to create ftpcmd.c
from ftpcmd.y.

1.4 Installation Guide

After you have successfully installed AmiTCP copy bin/ftpd to AmiTCP:serv.
Copy bin/ftpshut and bin/ftpcount to AmiTCP:bin. Copy examples to
AmiTCP:db (only if you are not upgrading from a previous version). All
this steps are also performed by the supplied Installer script, thus if you
have used the Installer script you don't need to go through this steps.

Edit the file AmiTCP:db/ftpaccess (for instructions on what/how to change
click here). If not already present add the following line
to AmiTCP:db/inetd.conf:

ftp   stream   tcp nowait root amitcp:serv/ftpd in.ftpd

Then start the network as you usually do (startnet) and ftp is ready for
use. If you want to start anonymous ftp you have to add a line like this
to your AmiTCP:db/passwd file:

ftp|*|3|99|Anonymous FTP User|Work:Archive|nologin

The newly created user 'ftp' must be a member of group 'guest' (in this
example group 'guest' has a group ID of 99, user 'ftp' has a user ID of 3).
If you don't already have a guest group in your AmiTCP:db/group file then
add the following line to the file:

guest|*|99|root

IMPORTANT: If you have MultiUser installed read the MultiUser section.

IMPORTANT: If you are upgrading from version 37.11 then you should
replace your ftpconversions file with the supplied one (in the examples
directory) or change all colons in the file to semicolons.

Now you are basically ready to start your own anonymous ftp server. Please
read the Important Notes section on bugs/features/omissions
in this version.

1.5 Using The Server

```
The FTP Daemon
Shutting Down the Server
Counting Users
Site Specific Commands
```

1.6 The FTP Daemon

```
NAME
   ftpd - DARPA Internet File Transfer Protocol server

SYNOPSIS
   ftpd [ -d ] [ -l ] [ -ttimeout ] [ -Tmaxtimeout ] [ -a ] [ -A ] [ -L ] [
   -i ] [ -I ] [ -o ] [ -O ] [ -k ] [ -bbuffer ]

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 AmiTCP documentation.

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

   If the -l option is specified, each ftp session is logged in the syslog.

   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 -b options specifies the number of buffers that wu-ftpd uses
   for sending/receiving files. A higher setting speeds up file transfers.
   Buffers are 512 bytes big, thus the option -b200 would create a 100K
   buffer (this is the default). Remember that each ftp session eats that
   much memory, so if you have 10 sessions running with a buffer size of
   1M you would need 10M of memory!

   The -k option enables a kludge that fixes a problem when trying to
   use Frank Munkerts AmiCDROM with wu-ftpd. If you get the error "not a
   plain file" when trying to get a file from the CD-ROM then add this
   flag. This flag is no longer needed with AmiCDROM 1.13 and up because
   those version fix the problem (they implement the ACTION_EXAMINE_FH
   packet). I will probably remove this switch in a future version, but I
   have left it for now if somebody is still using some older AmiCDROM.
   Don't add this flag unless its absolutely neccesary!

   If the -a option is specified, the use of the  ftpaccess  configuration
   file is enabled.

   If the -A option is specified, use of the  ftpaccess  configuration
   file is disabled.
```

If the -L option is specified, commands sent to the ftpd server will
be logged to the syslog. The -L option is overridden by the ftpaccess
file. If the -L flag is used, command logging will be on by default
as soon as the ftp server is invoked. This will cause the server to
log all USER commands, which if a user accidentally enters a password
for that command instead of the username, will cause passwords to be
logged via syslog.

If the -i option is specified, files received by the ftpd server will
be logged to the xferlog (AmiTCP:log/xferlog). The -i option is
overridden by the ftpaccess file.

If the -o option is specified, files transmitted by the ftpd server
will be logged to the syslog. The -o option is overridden by the
 ftpaccess file.

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 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 filename
    HELP            give help information. E.g. SITE HELP
    NEWER           list files newer than a particular date
    MINFO           like SITE NEWER, but gives extra information
    GROUP           request special group access. E.g. SITE GROUP foo
    GPASS           give special group access password. E.g. SITE GPASS bar
    EXEC            execute a program.  E.g. SITE EXEC program params
```

The GROUP and GPASS requests are currently not functional in the
Amiga version of wu-ftpd.

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. This allows users to utilize the metacharacters
''*?[]{}~''.

Ftpd authenticates users according to four rules.

1)   The user name must be in the password data base, AmiTCP:db/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 AmiTCP:db/ftpusers.

3)   The user must have a standard shell returned by getusershell.

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 command to the home
directory of the ''ftp'' user.

GENERAL FTP EXTENSIONS

There are some extensions to the FTP server such that if the user speci-
fies a filename (when using a RETRIEVE command) such that:

```
   True Filename    Specified Filename   Action
   -------------    ------------------   ----------------------------------
   <filename>.Z     <filename>           Decompress file before transmitting
   <filename>       <filename>.Z         Compress <filename> before
                                              transmitting
   <filename>       <filename>.tar       Tar <filename> before transmitting
   <filename>       <filename>.tar.Z     Tar and compress <filename> before
                                              transmitting
```

You can disable the automatic archiving if you put a file with the
name '.notar' in the directory that you want to exclude from automatic
archiving.

The FTP server will attempt to check for valid e-mail addresses and chide
the user if he doesn't pass the test. For users whose FTP client will hang
on "long replies" (i.e. multiline responses), using a dash as the first
character of the password will disable the server's lreply() function.

The FTP server can also log all file transmission and reception, keeping
the following information for each file transmission that takes place.

Mon Dec 3 18:52:41 1990 1 wuarchive.wustl.edu 568881 /files.lst.Z a _ o
a chris@wugate.wustl.edu ftp 0 *

  %.24s %d %s %d %s %c %s %c %c %s %s %d %s
    1   2 3 4 5 6 7 8 9 10 11 12 13

  1 current time in the form DDD MMM dd hh:mm:ss YYYY
  2 transfer time in seconds
  3 remote host name
  4 file size in bytes
  5 name of file
  6 transfer type (a>scii, b>inary)
  7 special action flags (concatenated as needed):
        C   file was compressed
        U   file was uncompressed
        T   file was tar'ed
        _   no action taken
  8 file was sent to user (o>utgoing) or received from
    user (i>ncoming)
  9 accessed anonymously (r>eal, a>nonymous) -- mostly for FTP
 10 local username or, if guest, ID string given
    (anonymous FTP password)
 11 service name ('ftp', other)
 12 authentication method (bitmask)
        0   none
        1   RFC931 Authentication
 13 authenticated user id (if available, '*' otherwise)

SEE ALSO
  ftpaccess

BUGS
  When running under MultiUser the server must run as the super-user
  in order to be able to access all the files and set its user ID. See
  the MultiUser notes for more information.
```

## 1.7 Shutting Down The Server

```
Name
  ftpshut - close down the ftp servers at a given time

Syntax
  ftpshut [ -l min] [ -d min] time [ warning-message ... ]

Description
  The ftpshut command provides an automated shutdown procedure that a
  superuser can use to notify ftp users when the ftp server is shutting
  down.

  The time is the time at which ftpshut will bring the ftp servers down.
  It may be the word 'now', indicating an immediate shutdown, or specify a
  future time in one of two formats: + number or HHMM. The first form
  brings the ftp servers down in number minutes.  The second brings the
  ftp servers down at the time of day indicated, using a 24-hour clock
  format.

  Ten minutes before shutdown, or immediately if ftpshut is timed for less
  than ten minutes, new ftp access will be disabled.  This time may be
  adjusted through the -l flag.

  Five minutes before shutdown, or immediately if ftpshut is timed for
  less than five minutes, all current ftp connections will be discon-
  nected.  This time may be adjusted through the -d flag.

  The [ warning-message ... ] will be formatted to be 75 characters wide.
  ftpshut knows about the actual string length of the magic cookies.

  The following magic cookies are available:
        %s      time system is going to shut down
        %r      time new connections will be denied
        %d      time current connections will be dropped
        %C      current working directory
        %E      the maintainer's email address as defined in ftpaccess.
        %F      free space in partition of CWD (kbytes)
                [not currently supported on all systems]
        %L      local host name
        %M      maximum allowed number of users in this class
        %N      current number of users in this class
        %R      remote host name
        %T      local time (form Thu Nov 15 17:12:42 1990)
        %U      username given at login time

  To bring the system up again you can use the keyword 'off'. Thus,
  'ftpshut off' will bring the server up again.

Restrictions
  You can kill the servers only between now and 23:59, if you use the
  absolute time for ftpshut.

See Also
  ftpaccess
```

1.8 Counting Users

```
Name
  ftpcount - show current number of users for each class

Syntax
  ftpcount

Description
  The ftpcount command shows the current number of users (and the limit)
  for each class defined in the  ftpaccess  file.
```

1.9 Site Specific Commands

The server supports the "SITE EXEC" command which enables one to add site specific commands to the server. F.e. if you want to enable your users to view the memory status of your machine do this:

  - create the directory AmiTCP:bin/ftp-exec
  - copy the avail command from your C: directory to AmiTCP:bin/ftp-exec

Now every user of your ftp server can type "SITE EXEC AVAIL" and he will see how much memory is available on your machine.

WARNING: Carefully think about which commands you will put into the ftp-exec directory. Generally it is not a wise idea to give users access to any other commands but system status commands (like avail, info, etc.) or some custom scripts that just output some information (and don't enable the user to change things on your system).

A nice example would be if you put the FORMAT command there everybody will be able to format your harddisks :-) This can also be dangerous with less dangerous commands, f.e. with the DATE command everbody could change the date on your computer. If you just want your users to be able to see the date and not change it create a simple script with only the date command in it:

  echo >AmiTCP:bin/ftp-exec/date date

And don't forget to set the script bit:

  protect AmiTCP:bin/ftp-exec/date +s

This also works for ARexx scripts (you will also have to set the script bit on arexx scripts).

Generally spoken it is not a good idea to enable the use of SITE EXEC because a hacker could find some strange bug in wu-ftpd which would enable him to do nasty things. So if you really need good security then don't enable SITE EXEC commands.

NOTE: The output length of the site specific commands is limited to 20 lines. If the output of your command is more than 20 lines long it will be truncated (and the message "*** Truncated ***" will be displayed to

the FTP user).

1.10 Configuration Files

1. ftpaccess
The main configuration file is AmiTCP:db/ftpaccess . Also see example
ftpaccess file in the 'examples' directory of this distribution.

2. ftpconversions
The file AmiTCP:db/ftpconversions is used for configuring things
like automatic archiving of directories. Example:

Suppose we have a file called 'file.txt' which is uncompressed. If you add
the following line to your ftpconversions file

 ;.Z; ; ;compress -d -c %s;T_REG|T_ASCII;O_UNCOMPRESS;UNCOMPRESS

then the user will be able to retrieve the file with 'GET file.txt.Z' and
receive a compressed file (the command 'compress' must be present in your
DOS path) even if it is not compressed (it will be compressed before
sending to the user). If the user tries to fetch the file in ASCII mode he
will be warned. There is an example ftpconversions file provided in the
'examples' directory.

I currently don't know of any Amiga version of tar that would support the
-Z and -z option (for automatic compressing/gziping of the tar archive).
If you have a version of tar that supports the -Z and -z options you can
add the following two lines to your ftpconversions file:

 ;   ; ;.tar.Z;tar -c -Z -f - %s;T_REG|T_DIR;O_COMPRESS|O_TAR;TAR+COMPRESS
 ;   ; ;.tar.gz;tar -c -z -f - %s;T_REG|T_DIR;O_COMPRESS|O_TAR;TAR+GZIP

3. shells
The file AmiTCP:db/shells contains names of shells that users can have as
their login shell in order to be able to use the FTP server. The shells
are listed one after another, each in a new line.

Here is an example. You have following line in your AmiTCP:db/passwd file:

test|PASSWORD HERE|2|100|Blaz Zupan|Work:|testshell

The user will currently not be able to login, because he does not have a
valid shell entry ("testshell" is not valid). If we add the line
"testshell" to AmiTCP:db/shells he will be able to login. There are two
shell entries that are valid by default: "cli" and "shell". If all your
users have one of those two then you don't need the file AmiTCP:db/shells.
If you have the file AmiTCP:db/shells then you also must add "cli" and
"shell" otherwise they won't be valid shell entries anymore. Only
anonymous users are able to login without a valid shell entry.

4. ftphosts
This file contains names of hosts that are allowed or denied access to your
FTP site. Template:

allow <user> <host>

```
deny  <user> <host>
```

See example file in 'examples' directory.

1.11 Configuration Files: ftpaccess

Description
  The ftpaccess file is used to configure the operation of ftpd.

Access Capabilities

  autogroup <groupname> <class> [<class> ...]
        This command is present in the Amiga version but currently has
        no real function. Here is what it should do (but does not):

        If an ANONYMOUS user is a member of any of <class>, the ftp server
        will perform a setegid() to <groupname>. This allows access to
        group-and-owner-read-only files and directories to a particular
        class of anonymous users. <groupname> is a valid group from
        AmiTCP:db/group (or wherever your getgrent() call looks).

  class <class> <typelist> <addrglob> [<addrglob> ...]
        Define <class> of users, with source addresses of the form
        <addrglob>. Multiple members of <class> may be defined. There may
        be multiple "class" commands listing additional members of the
        class. If multiple "class" commands can apply to the current ses-
        sion, the first one listed in the access file is used. Failing to
        define a valid class for a host will cause access to be denied.
        <typelist> is a comma-separated list of any of the keywords
        "anonymous", "guest" and "real". If the "real" keyword is
        included, the class can match users using FTP to access real
        accounts, and if the "anonymous" keyword is included the class can
        match users using anonymous FTP. The "guest" keyword matches guest
        access accounts (see "guestgroup" for more information)

        <addrglob> may be a globbed domain name or a globbed numeric
        address.

  deny <addrglob> <message_file>
        Always deny access to host(s) matching <addrglob>. <message_file>
        is displayed. <addrglob> may be "!nameserved" to deny access to
        sites without a working nameserver.

  guestgroup <groupname> [<groupname> ...]
        If a REAL user is a member of any of <groupname>, the session is
        set up exactly as with anonymous FTP. In other words, a chroot()
        is done, and the user is no longer permitted to issue the USER and
        PASS commands. <groupname> is a valid group from AmiTCP:db/group
        (or wherever your getgrent() call looks).

  limit <class> <n> <times> <message_file>
        Limit <class> to <n> users at times <times>, displaying
        <message_file> if user is denied access. Limit check is performed
        at login time only. If multiple "limit" commands can apply to the
        current session, the first applicable one is used. Failing to
```

        define a valid limit, or a limit of -1, is equivalent to unlimited.
        <times> is in same format as the times in the UUCP L.sys file.

    loginfails <number>
        After <number> login failures, log a "repeated login failures" mes-
        sage and terminate the FTP connection. Default value is 5.

    private <yes|no>
        Currently not supported in the Amiga version. This is what
        it will do in a future version (hopefully):

        After user logs in, the SITE GROUP and SITE GPASS commands may be
        used to specify an enhanced access group and associated password.
        If the group name and password are valid, the user becomes (via
        setegid()) a member of the group specified in the group access file
        AmiTCP:db/ftpgroups.

        The format of the group access file is:

        access_group_name:encrypted_password:real_group_name

        where access_group_name is an arbitrary (alphanumeric + punctua-
        tion) string. encrypted_password is the password encrypted via
        crypt, exactly like in AmiTCP:db/passwd. real_group_name is
        the name of a valid group listed in AmiTCP:db/group.

        NOTE: For this option to work for anonymous FTP users, the ftp
        server must keep AmiTCP:db/group permanently open and the group
        access file is loaded into memory. This means that (1) the ftp
        server now has an additional file descriptor open, and (2) the
        necessary passwords and access privileges granted to users via
        SITE GROUP will be static for the duration of an FTP session.
        If you have an urgent need to change the access groups and/or
        passwords *NOW*, you just kill all of the running FTP servers.

  Informational Capabilities

    banner <path>
        Works similarly to the message command, except that the banner is
        displayed before the user enters the username/password. The <path>
        is relative to the real system root, not the base of the anonymous
        FTP directory.

        WARNING: use of this command can completely prevent non-compliant
        FTP clients from making use of the FTP server. Not all clients can
        handle multi-line responses (which is how the banner is displayed).

    email <name>
        Defines the email address of the ftp archive maintainer. This
        string will be printed every time the %E magic cookie is used.

    message <path> {<when> {<class> ...}}
        Define a file with <path> such that ftpd will display the contents
        of the file to the user login time or upon using the change working
        directory command. The <when> parameter may be "LOGIN" or
        "CWD=<dir>". If <when> is "CWD=<dir>", <dir> specifies the new
        default directory which will trigger the notification.

The optional <class> specification allows the message to be
displayed only to members of a particular class. More than one
class may be specified.

There can be "magic cookies" in the readme file which cause the ftp
server to replace the cookie with a specified text string:

        %T   local time (form Thu Nov 15 17:12:42 1990)
        %F   free space in partition of CWD (kbytes)
                [not supported on all systems]
        %C   current working directory
        %E   the maintainer's email address as defined in ftpaccess
        %R   remote host name
        %L   local host name
        %U   username given at login time
        %M   maximum allowed number of users in this class
        %N   current number of users in this class

The message will only be displayed once to avoid annoying the user.
Remember that when MESSAGEs are triggered by an anonymous FTP user,
the <path> must be relative to the base of the anonymous FTP direc-
tory tree.

readme <path> {<when> {<class>}}
    Define a file with <path> such that ftpd will notify user at login
    time or upon using the change working directory command that the
    file exists and was modified on such-and-such date. The <when>
    parameter may be "LOGIN" or "CWD=<dir>". If <when> is "CWD=<dir>",
    <dir> specifies the new default directory which will trigger the
    notification. The message will only be displayed once, to avoid
    bothering users.

    The optional <class> specification allows the message to be
    displayed only to members of a particular class. More than one
    class may be specified.

Logging Capabilities

logpath path
    Specifies where the xferlog will be put. If not specified
    'AmiTCP:log/xferlog' will be used.

log commands <typelist>
    Enables logging of individual commands by users. <typelist> is a
    comma-separated list of any of the keywords "anonymous", "guest"
    and "real". If the "real" keyword is included, logging will be
    done for users using FTP to access real accounts, and if the
    "anonymous" keyword is included logging will done for users using
    anonymous FTP. The "guest" keyword matches guest access accounts
    (see "guestgroup" for more information).

log transfers <typelist> <directions>
    Enables logging of file transfers for either real or anonymous FTP
    users. Logging of transfers TO the server (incoming) can be
    enabled separately from transfers FROM the server (outbound).
    <typelist> is a comma-separated list of any of the keywords

"anonymous", "guest" and "real". If the "real" keyword is
included, logging will be done for users using FTP to access real
accounts, and if the "anonymous" keyword is included logging will
done for users using anonymous FTP. The "guest" keyword matches
guest access accounts (see "guestgroup" for more information).
<directions> is a comma-separated list of any of the two keywords
"inbound" and "outbound", and will respectively cause transfers to
be logged for files sent to the server and sent from the server.

Miscellaneous Capabilities

alias <string> <dir>
     Defines an alias, <string>, for a directory. Can be used to add
     the concept of logical directories.

     For example:
       alias   rfc: pub:doc/rfc

     would allow the user to access pub:doc/rfc from any directory by
     the command "cd rfc:". Aliases only apply to the cd command.
     Not very useful on Amiga (because you can use assigns), but still
     present if someone finds it useful.

cdpath <dir>
     Defines an entry in the cdpath. This defines a search path that is
     used when changing directories.

     For example:

       cdpath pub:packages
       cdpath aliases

     would allow the user to cd into any directory directly under
     pub:packages or aliases directories. The search path is defined
     by the order the lines appear in the ftpaccess file.

     If the user were to give the command:

       cd foo

     The directory will be searched for in the following order:

       foo
       an alias called "foo"
       /pub/packages/foo
       aliases/foo

     The cd path is only available with the cd command. If you have a
     large number of aliases you might want to set up an aliases direc-
     tory with links to all of the areas you wish to make available to
     users.

compress <yes|no> <classglob> [<classglob> ...]

tar <yes|no> <classglob> [<classglob> ...]
     Enables compress or tar capabilities for any class matching any of
     <classglob>. The actual conversions are defined in the external

```
              file  AmiTCP:db/ftpconversions .
```

shutdown <path>
      If the file pointed to by <path> exists, the server will check the
      file regularly to see if the server is going to be shut down. If a
      shutdown is planned, the user is notified, new connections are
      denied after a specified time before shutdown and current connec-
      tions are dropped at a specified time before shutdown. <path>
      points to a file structured as follows:

       <year> <month> <day> <hour> <minute> <deny_offset> <disc_offset>
       <text>

       <year> any year > 1970
       <month> 0-11 <---- LOOK!
       <hour> 0-23
       <minute> 0-59

      <deny_offset> and <disc_offset> are the offsets in HHMM format
      before the shutdown time that new connections will be denied and
      existing connections will be disconnected.

      <text> follows the normal rules for any message (see "message"),
      with the following additional magic cookies available:

              %s   time system is going to shut down
              %r   time new connections will be denied
              %d   time current connections will be dropped

      all times are in the form: ddd MMM DD hh:mm:ss YYYY. There can be
      only one "shutdown" command in the configuration file.

      The external program ftpshut can be used to automate the process
      of generating this file.

   lslong <command>

      Specifies which directory lister should be used for listing
      directories. The default is 'AmiTCP:bin/ls -lgA'.

   lsshort <command>

      Same as lslong, but used if user is logged in anonymously
      and if he writes a '-' as the first character of the password.

 Permission Capabilities

   allowdir <path> <user>
      Explicitly allow access to specified directory. This is a new
      keyword in the Amiga version of wu-ftpd which allows one to mount
      CD-ROMs under the anonymous FTP directory structure. F.e. if you
      have a softlink "Work:Archive/cdrom" which points to "CD0:" (a CD-ROM
      device) you would have to add "allowdir CD0:" to your ftpaccess file
      or your guest users would not be able to access the CD-ROM (because
      the softlink points outside of the users home directory). <user> can
      be a standard pattern (like ~(ftp|b*)). Warning: the standard "#?"
      convention for patterns doesn't work, because the character "#" is

interpreted as the start of a comment. Use "*" instead of "#?".

denydir <path> <user>
      Deny access to specified directory even if it is normally possible
      to access it (i.e. it is under the anonymous users home directory).
      This can be useful if you have some restricted directory which only
      certain users are allowed to access. <path> and <user> are specified
      exactly as with "allowdir".

chmod <yes|no> <typelist>

delete <yes|no> <typelist>

overwrite <yes|no> <typelist>

rename <yes|no> <typelist>

umask <yes|no> <typelist>
      Allows or disallows the ability to perform the specified function.
      By default, all users are allowed.

      <typelist> is a comma-separated list of any of the keywords
      "anonymous", "guest" and "real".

passwd-check <none|trivial|rfc822> (<enforce|warn>)
      Define the level and enforcement of password checking done by the
      server for anonymous ftp.

        none   no password checking performed.
        trivial   password must contain an '@'.
        rfc822   password must be an rfc822 compliant address.

        warn   warn the user, but allow them to log in.
        enforce   warn the user, and then log them out.

path-filter <typelist> <mesg> <allowed_charset> {<disallowed regexp>
      For users in <typelist>, path-filter defines regular expressions
      that control what a filename can or can not be. There may be mul-
      tiple disallowed regexps. If a filename is invalid due to failure
      to match the regexp criteria, <mesg> will be displayed to the user.
      For example:

        path-filter anonymous AmiTCP:db/path.msg ^[-A-Za-z0-9._]*$ ^. ^-

      specifies that all upload filenames for anonymous users must be
      made of only the characters A-Z, a-z, 0-9, and "._-" and may not
      begin with a "." or a "-". If the filename is invalid,
      AmiTCP:db/path.msg will be displayed to the user.

upload <root-dir> <dirglob> <yes|no> <owner> <group>
      Define a directory with <dirglob> that permits or denies uploads.

      If it does permit uploads, all files will be owned by <owner> and
      <group> and will have the permissions set according to <mode>.

      Directories are matched on a best-match basis.

```
        For example:
          upload  Work:ftp  *                        no
          upload  Work:ftp  Work:ftp/incoming       yes ftp daemon 0666
          upload  Work:ftp  Work:ftp/incoming/gifs yes jlc guest  0600 nodirs
```

        This would only allow uploads into /incoming and /incoming/gifs.
        Files that were uploaded to /incoming would be owned by ftp/daemon
        and would have permissions of 0666. File uploaded to
        /incoming/gifs would be owned by jlc/guest and have permissions of
        0600. The setting of file owners is only done if MultiUser is
        installed on your system or if you have version 39 of Amiga OS
        or higher (Kickstart 3.x).

        The optional "dirs" and "nodirs" keywords can be specified to allow
        or disallow the creation of new subdirectories using the mkdir com-
        mand.

        The upload keyword only applies to users who have a home directory
        (the argument to the chroot()) of <root-dir>.

        WARNING: The second parameter to "upload" is a pattern. That's why
        you can't use assigns or symbolic links in this path, you must pass
        the full pathname (with the volume name and not the device name, thus
        use "Work:" instead of "DH1:"). F.e. "dh1:archive/incoming" is not
        a valid upload path, but "Work:Archive/incoming" is. The comparison
        is also case sensitive!!!

    Files
       AmiTCP:db/ftpaccess

    See Also
       ftpd , ftpconversions , ftpshut

1.12 Configuration Files: ftpconversions

Name
  ftpconversions – ftpd conversions database

Description
  The conversions known by ftpd and their attributes are stored in an
  ASCII file that is structured as below. Each line in the file provides
  a description for a single conversion. Fields are separated by semicolons
  (;) and not by colons (:) as in the Unix version. This was done
  so that you can use the full Amiga path for the commands.

```
        %s;%s;%s;%s;%s;%s;%s;%s
         1  2  3  4  5  6  7  8

        Field           Description
         1                strip prefix
         2                strip postfix
         3                addon prefix
         4                addon postfix
         5                external command
         6                types
```

```
        7                options
        8                description
```

Known Problems
  The conversions mechanism does not currently support the strip prefix
  and addon prefix fields.

Files
  AmiTCP:db/ftpconversions

See Also
   ftpd , ftpaccess

1.13 MultiUser Support

From version 37.12 on wu-ftpd supports MultiUser. wu-ftpd will
transparently notice if MultiUser is installed on your system and use it.
There are some special things to be considered if MultiUser is installed on
your system (especially as the MultiUser support is not well tested as I
don't have MultiUser installed on my system).

1. You have to login as root before starting AmiTCP. This is needed so
that the ftp daemon process is owned by root. Setting of the SetUID bit on
ftpd does not work because ftpd is started by inetd in a way that is
incompatible with MultiUser. If you don't login as root before starting
AmiTCP then the ftp server will fail (and will output a message to your
logfile). After the user is logged in wu-ftpd will login as that user,
too. It will only change its user ID back to root if it is making things
that need root access (like changing the owner of a file with the "upload"
keyword).

2. The file AmiTCP:db/passwd is no longer used by ftpd if MultiUser is
installed. You have to add all the users to your MultiUser password file
(and also the group file used is the one from MultiUser). This will
probably change in the future when AmiTCP's usergroup.library supports
MultiUser. The main problem is that AmiTCP's and MultiUser's password
files are not compatible (actually only the password encryption is
different).

3. MultiUser does not allow invalid passwords in the password file. Thus
if you have the following entry in your password file MultiUser will
complain:

  test|*|1|10|Test User|Work:|shell

as '*' is not a valid crypted password. You either have to leave the field
empty or put a valid crypted password there.

4. Make all AmiTCP files owned by root and make them only readable for
others. If you don't make them readable for others then ftpd will not be
able to read the configuration files after it is logged in as a normal user
and will fail.

1.14 Important Notes

1. I have created some routines that partly simulate the behaviour of the
Unix chroot() function call. Let's say that a user's home directory is
"Work:Archive". He can now access f.e. "Work:Archive/test.txt" or
"Work:Archive/somedir/test2.txt", but not "Work:test3.txt" or
"S:Startup-Sequence".

This means that the user can only access the directory structure under his
own home directory and nothing else. This could sometimes lead to
problems, f.e. if you wanted to mount CD-ROMs under the anonymous ftp
directory structure. That's why I have added two new keywords, namely
"allowdir" and "denydir". Users are explicitely allowed access to
directories specified with "allowdir" even if the directories are not under
the users home directory. If a directory is specified as "denydir" then no
access to it will be allowed (even if it is under the home directory). For
more information on "allowdir" and "denydir" click here .

Note that it is still possible to access some files even if "denydir" is
active. This can be achieved with the automatic archive feature, where the
server automatically TAR's a directory if you try to get the file
"<dirname>.tar". Even if a directory under this directory is not allowed
access it can be archived if you archive its parent directory. So be
careful! There are only two solutions. Either disable the automatic
archiving (create a file '.notar' in the offending directory) or use
MultiUser and set the protection flags accordingly.

Please be aware that chroot() will only be done for anonymous and guest
users, not for real users. Real users will still be able to access every
directory and file on your Amiga (except for those you deny with
"denydir"). If you want a user to only be able to access the directory
structure under his home directory then you should put him in the guest
group.

WARNING: Other than the above mentioned things there are no other
protection measures, if you did not remove the upload/delete/rename
permission for users everyone can do everything in the directories he can
get into.

Additional security can be achieved if MultiUser is installed on
your system (but only if there are no bugs in my MultiUser support :-).

2. I have added Unix to Amiga path translation. Now you can do "CD ..",
"CD ./incoming" or "GET ../file.txt myfile.txt" like on Unix boxes. Another
feature is that all paths will be exactly Unix-like (e.g. "/pub/incoming"
instead of "Work:Archive/pub/incoming") which makes it possible to use httpd
to access the ftp server. The second feature only works for anonymous and
guest users.

NOTE: If the user accesses a directory that you have explicitely allowed
with the "allowdir" keyword and this directory is outside the users home
directory then he will see the full path! Example: The users home directory
is "Work:Archive" and you have allowed him to also access "Work:". If his
current directory is "Work:Archive" he will see it as "/". If his current
directory is "Work:" he will also see it as "Work:".

3. The server was compiled with SAS's STACKEXTEND option because it needs
a lot of stack (certainly more than the stack that is supplied by inetd).
So watch out, your memory can run out very soon if you have many instances
of ftpd running.

NOTE: It is recommend you that you also read the Common Problems section.

1.15 Common Problems

Q: When I try to login under a certain username with the correct password
   I get the message "User <someone> access denied".

A: Make sure that the password entry for that user has a correct shell entry
   (see Configuration Files). Also make sure that you did not lockout
   yourself in the AmiTCP:db/ftphosts file.

Q: When I try to FTP to my amiga I only get

     Connected to my.machine.si.
     421 Service not available, remote server has closed the connection

   or I get the message "Connection refused".

A: First of all make sure that you are using AmiTCP 3.0 beta 2 or higher.
   Also make sure that the file AmiTCP:db/inetd.conf contains this line:

     ftp   stream   tcp nowait root amitcp:serv/ftpd in.ftpd

   If that does not help look at the syslog (wherever you have it redirected).
   If it says "not running as root" then you are using MultiUser and you were
   not logged in as root when you were starting AmiTCP. Carefully read the
    MultiUser section. If you don't have MultiUser installed then make sure
   that you don't have any library named multiuser.library in your LIBS:
   directory (just rename it to something else).

Q: The server identifies my machine as "localhost" instead of my
   fully qualified domain name.

A: There seems to be a bug in AmiTCP's gethostname() function. Add the line
   "hostname=your.domain.name" (without the quotes) to AmiTCP:db/AmiTCP.config
   (your.domain.name is the fully qualified domain name of your Amiga).

Q: How can I redirect the server log messages to a file?

A: Add the following lines to AmiTCP:db/AmiTCP.config:

     consolename=NIL:
     logfilename=<pathname of logfile>

Q: I have a valid shell entry for my username in the password file, but I still
   can't login and get the message "bad shell" in the logfile.

A: You have probably put your username in the ftpusers file. The ftpusers file
   specifies the users that are not able to login and not those who are able
   to login.

Q: The "upload" keyword doesn't seem to work as I want it too (either I can
   upload everywhere or nowhere, but I don't seem to be able to specify
   the allowable upload paths).

A: Reread the documentation on the "upload" keyword. The second parameter
   to "upload" is a case sensitive pathname with patterns. Device names
   (f.e. DH0:), assigns and hard/soft links won't work! Also watch that if you
   specify the pattern as "Work:Archive/incoming/*" uploads to incoming will not
   be possible, because when your current directory is "incoming" the path is
   "Work:Archive/incoming" so the above pattern does not match (because of the
   superfluous last '/'). You would have to specify "Work:Archive/incoming*" (or
   better yet "Work:Archive/incoming").

Q: I get a Mungwall hit the first time a user accesses the FTP server. I'm
   not using MultiUser.

A: You are probably using AmiTCP 3.0. Its netinfo.device has a bug which causes
   Mungwall hits when it gets opened the first time. You should upgrade to either
   4.0 demo or 4.1.

1.16 Author

This server is a port of the original Unix sources. For copyright
information click here . Author of Amiga port:

  Blaz Zupan
  Ljubljanska 19/b
  62000 Maribor
  Slovenia
  E-Mail: blaz.zupan@uni-mb.si
  IRC: Herbie

Bug reports and suggestions are very much welcomed. A list of beta testers
can be found here .

1.17 Reporting Bugs

It often happens, that users send me mail like "ftpd crashes right after
startup". This doesn't help much, because I can't know what is going on
with this little information. That's why I'm presenting here a list of
things you should send me when reportings bugs.

- wu-ftpd version (you can find out which version you have by typing
"version amitcp:serv/ftpd full" in a shell)

- AmiTCP version

- OS version (Kickstart and Workbench, type "version full" in a shell)

- Do you have MultiUser installed (yes/no)? If yes, which version?

- Hardware setup (Amiga model, CPU type, memory, disks, type of network
connection (Ethernet, SLIP, PPP, ...), any additional expansion boards)

- If there is a problem with configuring wu-ftpd please send me your
AmiTCP:db/passwd and AmiTCP:db/ftpaccess file

- If ftpd crashes, then if possible please run Enforcer, Mungwall and
Segtracker. IMPORTANT: Only run Enforcer together with Segtracker,
because the Enforcer output is almost useless without Segtracker. You can
find Enforcer (Segtracker is included), Mungwall and Sushi (for catching
the debugging output) on Aminet. The best idea is to put Enforcer,
Mungwall, Segtracker and Sushi into your Startup-Sequence (not
User-Startup!) right after the SetPatch command. Add the following lines
after SetPatch:

```
  SegTracker >NIL:
  RUN >NIL: Sushi <>"CON:0/20/640/100/Sushi  CTRL-E=Empty  CTRL-F=File/AUTO/CLOSE/ ←
      WAIT/INACTIVE" ON BUFK=64 NOPROMPT ASKEXIT ASKSAVE
  RUN >NIL: Enforcer RAWIO
  RUN >NIL: MungWall NAMETAG SHOWHUNK
```

If you do all of this you can be sure that I will be able to resolve your
problem much faster. Send a complete bug report to me (see address).

1.18 Copyrights and Credits

First of all I would like to thank my beta testers:

hamster@biolink.com (Steve Fraden)
nverenin@nverenin.extern.ucsd.edu (N. J. Verenini)
jbray@airley.apana.org.au (James Ray)
dseaman@kuhub.cc.ukans.edu (Derek Seaman)
xs@nil.student.uni-tuebingen.de (Andreas Otte)
root@wizard.hip.cam.org (Greg Patterson)

If you were betatesting wu-ftpd and are not included on this list then
please mail me.

This server is a port of the Washington University sources. The following
copyright notice applies:

1.19 Future Plans

- Improve installer script
- Implement a "file filter" so that people are able to lock out certain
  files they don't want to be displayed (f.e. BBS control files if you
  are sharing the FTP server and a BBS up/download area)
- Implement a GUI for controlling the server (you'll be able to see which
  users are currently online, kick users off, shut the server down, all with

    the push of a button.
- Anything you suggest

I do not intend to port wu-ftpd to AS225.

If you have any other suggestions then please contact me (see Author).

1.20   Revision History

Version 37.21

  Fourth released version.

  Fixes yet another (less dangerous) security hole. I really hope this is
  the last one. Please report any security problems immediately!

Version 37.20

  Beta version, not released to the general public.

  Fixed a major security hole. I hope not many people found out
  about it and I hope that everybody upgrades to this new version. And no,
  I won't describe the bug here for the sake of those who still use older
  versions...

  This new version should use much less memory then the previous version.
  I don't know why 37.19 used about 600K for every invocation, a simple
  recompile cured the problem. It seems like I linked the 37.19 executable
  with some old and buggy object file that was lying around somewhere.
  This will be a lesson - next time I will do a rebuild from scratch.

  Fixed problems with MultiUser and file protections. Now uses MuFS
  function for setting file protections if it is installed (thanks to
  Ingolf Koch for the suggestion).

Version 37.19

  Third released version.

Version 37.18

  Beta version, not released to the general public.

  This is really embarasing. I completely screwed up the MultiUser
  support in 37.14 and up. In fact, wu-ftpd immediately crashed when
  MultiUser was installed. I wonder why only one user noticed this.
  Is nobody using wu-ftpd together with MultiUser or what?

Version 37.17

  Beta version, not released to the general public.

  Seems like it was not a good idea to mess with the signal code.
  Sending a Control-C to a ftpd process lead to a stack overflow and
  could also generate Enforcer hits and maybe even a crash. Also

in earlier versions sending a Control-C did not remove the record
from the ftp-pids file so the number of current users was no longer
correct after killing a ftpd process. Now fixed.

Version 37.16

  Beta version, not released to the general public.

  It is now possible to specify where the transfer log will be put.
  See documentation on "logpath".

  Added setproctitle(). Now you can see what your users are doing
  by simply typing "status" in a shell.

  Documented solution to "Upload keyword won't work" problem.

  Completely rewritten binary upload routine. Upload buffer now works
  (previously the buffer was only used on downloads due to a bug).
  Binary uploads should be much faster now.

  If MultiUser is used on your system then usergroup.library won't
  be opened anymore, because it is not needed. This should save some
  memory.

Version 37.15

  Beta version, not released to the general public.

  The Unix root directory emulation is back. This was added because of
  compatibility problems with httpd (httpd expects to be able to do a CWD
  to "/" which means "parent directory" on the Amiga and "root directory"
  under Unix).

  The idle timer finally works. Now if a user is idle more than the number
  of specified seconds (see SITE IDLE command) the ftp server will close the
  connection. No more hanging processes when a network connection breaks...

Version 37.14

  Second released version.

Version 37.13

  Beta version, not released to the general public.

  Improved documentation for "allowdir" and "denydir".

  Fixed a stupid bug in IsParent()/amiga.c. I did not free locks that I got
  with ParentDir(). This meant that every directory you went into was left with
  an open lock (and thus could not be deleted anymore). Stupid, really stupid...

  Added workaround for problems with the AmiCDROM filesystem by Frank Munkert.
  It seems like it does not support the ExamineFH() call so the fstat() function
  of AmiTCP fails (and the user gets the message "not a plain file" when trying to
  get a file from the CD-ROM). wu-ftpd contains a kludge which fixes this (see
  '-k' option for ftpd).

Version 37.12

  Beta version, not released to the general public.

  Added MultiUser support.

  Added Installation script (very simple, but working :-).

  Now correctly reports system type when requested with SYST
  command ('AMIGA Type: L8' instead of 'UNKNOWN Type: L8').

  It was possible to do a CWD to a file. Fixed.

  Sometimes a lock on the last directory you changed into was not
  unlocked. This was a bug in AmiTCP's lstat(). This distribution
  contains a fixed lstat().

  Completely rewritten chroot() emulation because the routine in 37.11
  was the cause of strange behaviour of the server in many cases. Should
  work OK now. Please note that I have removed the "Unix root dir"
  simulation because there were some problems with it. Now anonymous FTP
  users see the full Amiga pathnames!

  The anonymous FTP user count was sometimes incorrectly reported
  as 0 (although there were users online). This was because the original
  Unix code kept the process files open all the time but under Amiga DOS
  other processes can not read from files locked with exclusive locks.
  Because ftpd and ftpcount were unable to open the process file for
  reading they could not count the users and thus reported 0 users. Now
  fixed.

  Added -b option to server. With this option you can specify the
  size of the I/O buffer used for sending and receiving files. In 37.11
  the buffer was 512 bytes big which caused slow transfers. Now you can
  specify your own buffer size. The default buffer size is 100K.

  Fixed NLST. Previously it responded with "Bad directory components"
  if you tried a NLST with patterns (f.e. NLST *.lha). This also caused
  the 'mget' command of ncftp to fail.

  Source code reorganized and cleaned up a bit. Removed unused files from
  distribution.

  The file protection changing with the 'upload' keyword now works,
  also changing the owner now works correctly (under V39 even without
  MultiUser).

  Added new option to ftpshut: 'ftpshut off' now reenables the
  server after a shutdown.

  Unix to Amiga path translation is now done for all commands (not only
  for CWD).

  Changed separator in ftpconversions file from colon to semicolon
  so that one can use full Amiga pathnames for the commands.

  Fixed bug in handling of AmiTCP:db/shells file. Only entries that started

with a '/' were recognized (because shell entries under Unix always start
with '/').

Improved documentation. Added some previously undocumented features,
also some reminders to possible problems.

Version 37.11

First released version.

Version 37.0 – 37.10

Beta versions. Only released to few people for testing.
(Thanks to the beta testers.)