BootManage® TFTP Server

User and Reference Manual

Boot Manage®

TFTP Server

BootManage® TFTP Server

User and Reference Manual

Copyright:

Trademarks:

Colophon:

This document was produced using FrameMaker 5.5 on Windows NT workstations. Page proofs were printed on a 1200 dpi PostScript Level II printer, and final output was run directly to film. The PDF file for online reading was created using Adobe Acrobat 4.0.

Revision history:

January 2000, first version

Project team: Ralf Büttner, Dirk Köppen

Printed in Germany.

Table of Contents

Introduction

The TFTP Standard

The Trivial File Transfer Protocol TFTP is a simple protocol to transfer files, designed to to be small and easy to implement. Therefore, it lacks most of the features of a regular file transfer protocol. The only thing it can do is read and write files from/to a remote server, where each packet is acknowledged separately.

The TFTP protocol is especially useful in remote boot / remote installation environments where a client PC needs to download a boot image file over the network from a boot server.

The TFTP protocol is an open and vendor-independent internet standard defined by the Internet Engineering Task Force (IETF) in a number of Request For Comments (RFC) documents*:

Document	Title
RFC 1350	The TFTP Protocol (Revision 2)
RFC 1785	TFTP Option Negotiation Analysis
RFC 2347	TFTP Option Extension
RFC 2348	TFTP Blocksize Option
RFC 2349	TFTP Timeout Interval and Transfer Size Options

Most UNIX operating systems already contain a TFTP server as part of the operating system distribution, however this TFTP server may not always be full-featured. Other server operating systems like Windows NT, NetWare, and OS/2 do not provide a TFTP server.

* There are more RFC documents on TFTP available, but most of them have been obsoleted by the documents shown in our table. For more information about the IETF and RFC documents, point your web browser to http://www.ietf.org.

The BootManage® TFTP Server

The BootManage® TFTP Server is a standards-conforming server implementation of the TFTP protocol as defined in the IETF RFC document 1350. More, the BootManage® TFTP Server provides a number of custom enhancements which provide extended functionality, especially in combination with the BootManage® TCP/IP BOOT-PROM and the BootManage® PXE Toolkit.

Implementations are available for the following operating system platforms*:

- Windows 2000 Professional and Server (in preparation)
- Windows NT 3.x and 4.0 (Workstation and Server)
- Windows 98 and 95
- Windows 3.x, Windows for Workgroups 3.1x
- NetWare 3.x, 4.x, 5.x
- OS/2

A Word About Licensing

The BootManage® TFTP Server is not free software, and you need a license to use it. However, the BootManage® TCP/IP BOOT-PROM, the BootManage® PXE PROM, and the BootManage® PXE Toolkit already contain a license for the BootManage® TFTP Server. If you purchased one of these products, you do not need a separate license for the BootManage® TFTP Server.

* Support for custom enhancements varies for the different implementations.

TFTPD32

TFTPD32 is a 32-Bit TFTP server that runs on Microsoft Windows 95, Windows 98, Windows NT Workstation and Server, and Windows 2000* Professional and Server machines. On Windows NT, TFTPD32 can be run as a Win32 application or installed as a Windows NT service.

TFTPD32 Features

TFTPD32 adds several features which are not available in standard TFTP implementations:

- Allows restricted file access for security.
- Provides several means of controlling server and network load.
- Allows larger TFTP segment sizes (LTFTP) for higher throughput and fewer network transactions.
- Supports multicast file transfer (MTFTP) for faster concurrent booting of multiple clients.
- Supports filename mapping (in combination with the BPBOOT bootstrap loader from the BootManage® TCP/IP BOOT-PROM distribution).

Installing TFTPD32

First, copy the file *tftpd32.exe* to the local hard disk of a Windows 95/98, Windows NT Workstation/Server, or Windows 2000 Professional/Server machine.

```
C:\> copy a:\tftpd32.exe %WINDIR%                    (on Windows 95/98)
C:\> copy a:\tftpd32.exe %SystemRoot%\system32       (on Windows NT)
```

* Support for Windows 2000 is in preparation and not yet available at this time.

TFTPD32 will look up the port number for the tftp port in the file *%SystemRoot%\system32\drivers\etc\services*. If this entry is not present, TFTPD32 will use the following default:

```
tftp    69/udp
```

If you want to start TFTPD32 as a Win32 application, simply invoke it from the commandline using the -cmd option and any other options which suit your needs, e. g.:

```
C:\> tftpd32 -cmd -v 2
```

If you do not specify any options on the commandline, TFTPD32 will read the options from the text file *c:\etc\tftpd.cnf*. For a detailed list of all supported options, please see *"TFTPD32 Commandline Options"* on page 10.

! If TFTPD32 should operate as a standard TFTP Server, it is not necessary to specify any options on the commandline beyond -cmd. However, to activate one or more of the advanced features TFTPD32 provides, you must specify the corresponding option(s).

Installation As a Service

If you want to install TFTPD32 as a Windows NT service, use the -install option:

```
C:\> tftpd32 -install
```

Please note that when using -install, you cannot specify any other options on the commandline. Therefore, be sure to place any desired options in the text file, *c:\etc\tftpd.cnf,* where TFTPD32 will read them.

When installed as a service, TFTPD32 will automatically start at every Windows NT system startup. Like any other service, you can start and stop TFTPD32 using the Windows NT Service Control Manager. In addtion, you can control TFTPD32 from the commandline:

```
C:\> net start tftpd          (start TFTPD32 service)
C:\> net stop tftpd           (stop TFTPD32 service)
```

TFTPD32 Commandline Options

The following arguments can be given to the TFTPD32 server:

```
tftpd32 -cmd options              (run as Win32 application)
tftpd32 -install                  (install as Windows NT service)
tftpd32 -remove                   (remove service)
```

```
    tftpd32 -?                        (display command line syntax)
```

With the `-cmd` argument, the following options are provided:

Option	Description
`-a [`*`file`*`]`	Read table of mapped filenames from *file*
`-b` *`num`*	Limit TFTP bandwidth to approximately *num* KBytes/s
`-c` *`num`*	Number of concurrent TFTP transfers (max. 64)
`-d [`*`path`*`]`	Restrict TFTP access to directory *path* and its subdirectories.
`-h`	Enable read-ahead buffer.
`-i` *`num`*	Exit after *num* seconds of inactivity.
`-k` *`num`*	Set TFTP transmission keep-alive time.
`-l [`*`file`*`]`	Log all messages to a file.
`-m` *`file port`*	MTFTP configuration file and listen port.
`-p` *`num`*	Set TFTP listen port.
`-r`	Restrict file access to current directory.
`-s` *`num port`*	Set TFTP segment size and listen port.
`-t` *`num`*	Set Multicast Time-to-live value to *num* seconds
`-u` *`num`*	Number of MTFTP unicasts to be sent (default: 4).
`-v` *`level`*	Set level of verbosity (0, 1, or 2).
`-w`	Enable writing to existing files on the TFTP server
`-x`	If `-w` is also set, allow creation of files on the TFTP server

Controlling Server and Network Load

The following options allow to control performance related parameters of `TFTPD32`.

Limit Number Of Concurrent TFTP Transfers (-c)

The number of concurrent TFTP transfers can be limited by using the `-c` option. The following example limits the number of concurrent TFTP transfers to 4:

```
    tftpd32 -c 4
```

Enable Read-Ahead Buffers (-h)

This option enables read-ahead buffers for the TFTP file transfer. That is, `TFTPD32` reads more data than is needed and buffers the additional data for later use. In some environments, the transfer speed increases by enabling the `-h` option. By default, `TFTPD32` does not read-ahead.

Terminate TFTP Server After Inactivity Period (-i)

TFTPD32 can be automatically terminated by using the -i option. If there is no TFTP activity for the number of seconds specified, TFTPD32 terminates. The following example aborts TFTPD32 after 45 seconds of inactivity:

```
tftpd32 -i 45
```

Set TFTP Transmission Keepalive Time (-k)

TFTPD32 retains all information from a TFTP transfer for a specified amount of time so that a client can repeat a prior transfer. The -k option allows one to change the default time of 15 seconds.

Security Related Options

Security related options control in what way TFTP clients can access files on the TFTP server.

Set TFTP Server Root Directory (-d)

A default path can be added to each request issued by a TFTP client. This way, TFTPD32 restricts access to files below a certain directory on the server. If the -d option is given without a directory, this defaults to *c:\tftpboot*. The following examples restrict TFTP access to both files and subdirectories below *c:\tftpboot*:

```
tftpd32 -d /tftpboot
tftpd32 -d
```

Restrict TFTP Access To Directory (-r)

The -r option restricts all file access to the current directory and its subdirectories. That is, no files from any other locations may be transmitted. The current directory can be changed using the -d option.

The following example limits all file transmissions to the directory */tftpboot* and its subdirectories:

```
tftpd32 -d c:\tftpboot -r
```

Enable TFTP Write Access (-w)

Normally, TFTPD32 only permits read operations, i.e. a TFTP client can only read a file, not write one to a TFTP server. The -w option enables TFTP write access so that a TFTP client can write to existing files on the TFTP server. Unless the -x option is also specified, a TFTP client can only write to existing files and cannot create new files.

Allow Creation of Files (-x)

Together with the `-w` option, the `-x` option allows a TFTP client to create new files on the TFTP server.

Logging

TFTPD32 can log its activity to the application window on the screen, to the Windows NT event log and/or to a log file.

Log TFTP Server Activity (-l)

All output of `TFTPD32` can also be logged to a file. The `-l` option enables logging and allows one to specify a filename. If no filename is specified, then the default is *c:\tftpboot\tftpd.log*. The following example enables full debugging and logs all output to *c:\tmp\tftpd.log*:

```
tftpd32 -v 2 -l c:\tmp\tftpd.log
```

Set Verbosity Level (-v)

The debugging and activity output of `TFTPD32` can be changed by the `-v` option. The default is 1, and increasing the number generates more output. A value of 0 does not create any output, and a value of 4 displays very detailed information.

Multicast TFTP Operation

Enable Multicast TFTP (MTFTP) Operation (-m)

The `-m` option enables the `TFTPD32` multicast file transfer support (MTFTP). Two arguments are passed to the `-m` option: The first argument specifies a configuration file, the second argument specifies the UDP port number where `TFTPD32` listens for MTFTP file transfer requests. The following example uses port number 75 for MTFTP requests:

```
tftpd32 -m c:\etc\mtftptab 75
```

The configuration file has the following format:

```
c:\tftpboot\pcnfs.X      225.0.0.1     76
c:\tftpboot\ramd.X       225.0.0.2     76
```

Since there are no reserved port numbers for multicast TFTP, it cannot be guaranteed that the default ports of 75 and 76 will be available on all systems.

The first column holds the filename to be transmitted. If `TFTPD32` receives an MTFTP request for that filename on the specified MTFTP server port (75 in the above example), then it sends the MTFTP packets to the IP address specified in the second column (225.0.0.1). The third column holds the MTFTP client port number where all data is sent.

Set Number Of Unicast Packets In MTFTP Mode (-u)

The `-u` option is used in conjunction with multicast file transfer (MTFTP). The `-u` option specifies the number of unicast packets that `TFTPD32` sends to the MTFTP client after a transmission has been initiated.

The default is 4 packets and should not be changed.

Large Block TFTP Operation

Enable Large Block TFTP (LTFTP) Operation (-s)

This option allows one to set the TFTP segment size. The standard TFTP protocol specifies a segment size of 512 bytes. By increasing the segment size, fewer network transactions are needed and, therefore, faster file transfer can be achieved. Increasing the TFTP segment size makes the TFTP transmission incompatible with standard TFTP clients and the RFC specification. Therefore, `TFTPD32` only sends larger packets to clients which open the TFTP transmission at the specified UDP port:

```
    tftpd32 -s 1408 59
```

This adds support for larger TFTP packets (here 1408 bytes) if the connection is opened at UDP port 59 instead of the standard TFTP port 69. Opening a TFTP transmission at the standard TFTP port 69 lets `TFTPD32` send 512 byte packets. Opening a TFTP transmission at the non-standard UDP port 59 allows `TFTPD32` to send up to 1408 bytes in TFTP packets.

Custom Settings

Specify TFTP Server Listen Port (-p)

By default, `TFTPD32` uses UDP port number 69 to listen for TFTP file transfer requests. This can be overridden by using the `-p` option. `TFTPD32` can also be used together with an existing TFTP server. In this case, `TFTPD32` is used only for extended features.

To disable the standard TFTP functionality of `TFTPD32`, bind the TFTP UDP port number to an unused UDP port, e.g.:

```
tftpd32 -s 1408 59 -m c:\etc\mtftptab 75 -v 1 -h -p 21435
```

Using Mapped Filenames

Enable Mapped Filenames (-a)

The `-a` option enables the `TFTPD32` filename mapping feature. As an argument to the `-a` option, the filename of a map configuration file must be passed. The purpose of the map configuration file is to substitute a requested filename with a mapped filename using the following format:

```
<requested-filename-1>  <mapped-filename-1>
<requested-filename-2>  <mapped-filename-2>
<requested-filename-3>  <mapped-filename-3>
...
$DEFAULT$               <default-filename>
```

Whenever `TFTPD32` receives a TFTP request for a file/pathname which starts with the character string *$MAP$*, it strips the *$MAP$* prefix and looks up the remaining file/pathname in the left-hand side of the map configuration file.

If a matching file/pathname is found, `TFTPD32` replaces the requested file/pathname with the corresponding (mapped) file/pathname listed in the right-hand side of the map configuration file. Then, `TFTPD32` sends the mapped file (instread of the requested file) to the TFTP client.

If a matching file/pathname is not found, `TFTPD32` checks if a `$DEFAULT$` entry is present at the end of the map configuration file. If it finds one, `TFTPD32` uses the mapped file/pathname associated with the `$DEFAULT$` entry.

If a matching file/pathname is not found, and `TFTPD32` also does not find a `$DEFAULT$` entry at the end of the map configuration file, it simply uses the (original) requested file/pathname, but without the *$MAP$* prefix.

`TFTPD32` will only try to perform filename mapping if the requested filename starts with the string *$MAP$*. `TFTPD32` will not perform any filename mapping for files that do not start with the string *$MAP$*. Files in the left-hand side of the map file must be specified without the leading *$MAP$* string.

The filename mapping feature is especially useful in combination with the BPBOOT bootstrap loader which is part of the BootManage® TCP/IP BOOT-PROM distribution. BPBOOT can generate individual filenames using the client's MAC address. This allows to easily define "boot image groups" for PC clients in a DHCP environment without the need of specifying individual boot images in the DHCP server's configuration.

Example Using Mapped Filenames

The following example illustrates the filename mapping feature. Let us assume that the map file *c:\etc\tftpd.map* contains the following information:

```
c:\tftpboot\112233445566.X   c:\tftpboot\winnt40.X
c:\tftpboot\445566778899.X   c:\tftpboot\winnt40.X
c:\tftpboot\778899aabbcc.X   c:\tftpboot\win95.X
c:\tftpboot\aabbccddeeff.X   c:\tftpboot\win98.X
$DEFAULT$                    c:\tftpboot\bootfile.X
```

To enable filename mapping, TFTPD32 must be invoked as follows:

```
C:\> tftpd32 -cmd -a c:\etc\tftpd.map -v 2
```

The commandline option -v is not needed for mapped filename operation, but it is very useful for locating problems.

When receiving a TFTP read request for file *MAPc:\tftpboot\778899aabbcc.X*, TFTPD32 strips the leading *MAP* and looks up the remaining filename *c:\tftpboot\778899aabbcc.X* in the left-hand side of the map file *c:\etc\tftpd.map*. The filename is found in the third row, so TFTPD32 substitutes it with the corresponding entry *c:\tftpboot\win95.X* on the right-hand side and sends this file back to the client. Likewise, when a TFTP client requests *MAPc:\tftpboot\aabbccddeeff.X*, TFTPD32 sends *c:\tftpboot\win98.X* instead.

The first two rows of the map file *c:\etc\tftpd.map* demonstrate how to group boot image files. Clients request different files (*MAPc:\tftpboot\112233445566.X* and *MAPc:\tftpboot\445566778899.X*), but both are substituted by the same file (*c:\tftpboot\winnt40.X*).

To demonstrate the $DEFAULT$ entry, let us assume that TFTPD32 receives a read request for a file named *MAPc:\tftpboot\123456789abc.X*. This file is lot listed in our sample map file, but it starts with the string *MAP*. In this case, TFTPD32 looks for a $DEFAULT$ entry on the left-hand side at the end of the map file. In our map file, a $DEFAULT$ entry is present, so TFTPD32 uses this for substitution and sends the file *c:\tftpboot\bootfile.X* back to the client. If there were no $DEFAULT$ entry, TFTPD32 simply strips the *MAP* prefix and sends the file *c:\tftpboot\123456789abc.X* back to the client instead.