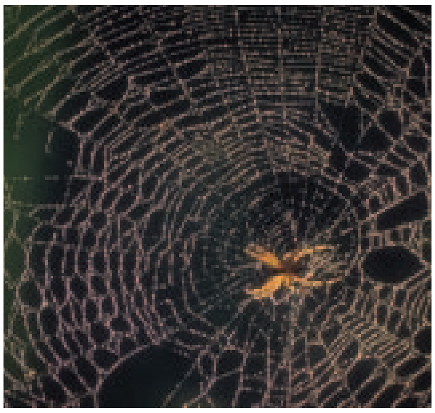


tHTTPd

tiny Web

Need to share a directory temporarily and allow other users browser based access? Setting up an Apache server to do this would be slightly over the top. So why not go for the easy approach with tHTTPd? BY CHRISTIAN PERLE

This issue's "out of the box" is all about a tool with an absolutely unpronouncable name, the "tiny *HTTP* daemon" – that is a miniature **web server**. The server program itself weighs in at a mere 60 to 70 kilobytes. There are several installation methods.

Pre-cooked or do-it-yourself?

Pre-configured binaries are available for the Debian, SuSE, Red Hat, and Mandrake distributions. However, the tHTTPd package is not pre-installed on Red Hat and its offspring – you will need to use **rpmfind** to locate it. (*ftp://speakeasy.rpmfind.net/linux/rhcontrib/7.1/i386/thttpd-2.21b-fr0.4.i386.rpm*).

No matter what distribution you use, you can always revert to compiling the sources available at *http://www.acme.com/software/thttpd/thttpd-2.23beta1.tar.gz*, and this adds the advantage that you will automatically have the latest released version.

If you opt for installing the binaries, you will need to use the package manager supplied with your distribution – *rpm* for SuSE, Red Hat and Mandrake, *dpkg* or *apt-get* for Debian. Installing from the sources requires just a few more steps:

```
tar xzf thttpd-2.23beta1.tar.gz
cd thttpd-2.23beta1
./configure
make
```

```
su  <I>(enter root password)<I>
make WEBGROUP=www install
exit
```

The *thttpd* executable is placed in */usr/local/bin*, or if you used the RPM package in */usr/sbin*. Additional files are placed in */usr/local/www* (source archive), */usr/local/httpd/htdocs* (SuSE) or */var/www* (Red Hat).

User and root flavored Instant Soup

The easiest way to go is to launch the web server in the directory that you want to share – this is often referred to as instant webserving. You do not need superuser privileges to do so, but you will need to tell the program to listen on a different port than the standard HTTP **port** 80. In our example we tell tHTTPd to listen on the **unprivileged port** 4242:

```
thttpd -p 4242
```

If you do not want to launch the web server in the working directory, but prefer to share a different directory instead, you can stipulate a directory by specifying the the *-d directoryname* option. Users accessing the server will not be able to see parent directories or the files they contain.

If you launch the server with *root* privileges, it will be allowed to listen on port 80. After you launch the daemon, tHTTPd will drop its *root* privileges and run as the unprivileged user, *nobody*. You can apply an additional security measure prior to this phase. If you stipulate the *-r* flag, the server uses **chroot** to change to a new root directory that it will not be able to break out of.

Figure 1 shows an example of directories shared for browser based access by tHTTPd. The *Konqueror* web browser was used.

File access is provided to */home/chris/public*, the directory where the

OUT OF THE BOX

There are thousands of tools and utilities for Linux. "Out of the box" takes a pick of the bunch and each month suggests a little program, which we feel is either absolutely indispensable or unduly ignored.

GLOSSARY

Web server: *A service program (server) that allows web browsers to access files via "HyperText Transfer Protocol" (HTTP). The most popular web server is Apache.*

rpmfind: *A search agent for packages in rpm format. http://rpmfind.net/ contains packages mostly for Red Hat and related distributions.*

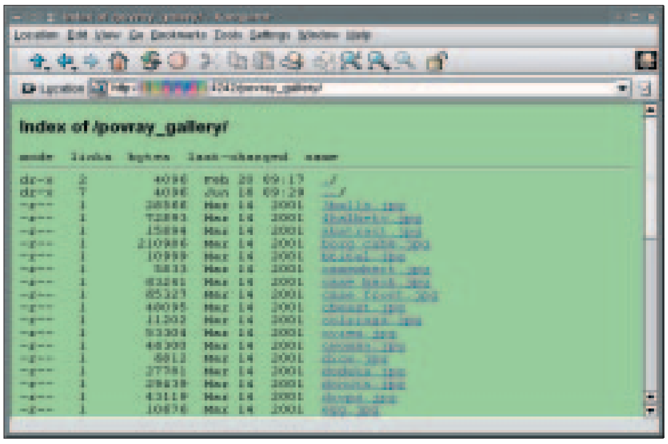
Port: *A docking position for network connections. Ports are designated by number and many of them are assigned to services by this number. Programs that bind to ports can offer their services (such as file transfer or remote login) on these ports.*

Unprivileged port: *Port up to and including 1023 require root privileges to bind to a service. In contrast, ports 1024 through 65535 (so-called unprivileged ports) can also be used by user processes.*

chroot: *The ("change root") system call defines a new root directory for a process. The process cannot break out of this directory and is thus isolated from the rest of the file system.*

Symlink: *Abbreviation for "symbolic link". A symlink is a special file that contains a path (the target referred to). If you attempt to read or write to a symlink, the system will in fact access the target. Symbolic links are created by the ln -s syntax.*

CGI: *Short for "Common Gateway Interface". This mechanism is used by the web server to create dynamic web page content. You could use a CGI shell script to display details of the users currently logged on to the server, for example.*

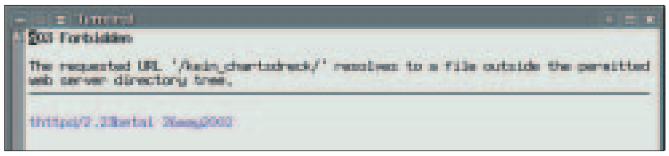


```
Index of /povray_gallery/

mode  links  bytes  last-changed  name

dr-x    2     4096  Feb 20 09:17  ./
dr-x    7     4096  Jun 18 09:29  ../
-r--    1    28568  Mar 14  2001  [illegible]
-r--    1    72893  Mar 14  2001  [illegible]
-r--    1    15894  Mar 14  2001  [illegible]
-r--    1   210986  Mar 14  2001  [illegible]
-r--    1    10959  Mar 14  2001  [illegible]
-r--    1     5833  Mar 14  2001  [illegible]
-r--    1    83281  Mar 14  2001  [illegible]
-r--    1    85327  Mar 14  2001  [illegible]
-r--    1    40095  Mar 14  2001  [illegible]
-r--    1    11202  Mar 14  2001  [illegible]
-r--    1    53308  Mar 14  2001  [illegible]
-r--    1    48309  Mar 14  2001  [illegible]
-r--    1     8812  Mar 14  2001  [illegible]
-r--    1    27781  Mar 14  2001  [illegible]
-r--    1    28438  Mar 14  2001  [illegible]
-r--    1    43118  Mar 14  2001  [illegible]
-r--    1    10876  Mar 14  2001  [illegible]
```

Figure 1: Konqueror Access



```
403 Forbidden

The requested URL '/kein_chartdreck/' resolves to a file outside the permitted
web server directory tree.

thttpd/2.23beta1 26may2002
```

Figure 2: No access is allowed at this point

server was launched. As the server is not listening on the standard port (port 80), the users will need to supply the port in the URL, such as in *http://localhost:4242* if they want to access the local web server.

Forbidden Fruit

Even if tHTTPd is not running in a **chroot** jail, it will still restrict accesss to files and directories below the directory it was launched from.

Figure 2 shows a user attempting to access the *no_charttoppers* directory, which represents a **symlink** to a target outside of the daemon's home directory. This makes tHTTPd issue an HTTP 403 ("forbidden") error – in other words access to this directory is not permitted. Double dot .. (parent directory) attacks designed to access the server's home directory are also refused by the server.

When you launch the web server, you may notice that immediately the program disappears from the terminal where it was launched and retires into the background, just like a well-behaved daemon should. But how do you stop the daemon? You can always use the *ps* command (which outputs a list of the active processes) to obtain the neccessary details, i.e. the process ID. We additionally use *grep* to filter the output, leaving only entries that contain the *thttpd* string, and send the *kill* to the process ID that we find, in order to terminate the process.

```
chris@camera:~ $ ps axc | grep ↵
thttpd
 2752 ?        S      0:00 ↵
thttpd
chris@camera:~ $ kill 2752
```

If messing around with *ps* and *grep* is not your idea of fun, you can use the *-i pid-file* syntax to have the server write the process ID to a file when you launch *thttpd*. In this case, you can use the following command to kill the process: *kill $(cat pid-file)*.

Logbook

As you will normally want to know what is happening on your server, you might like to try the *-l logfile* option to protocol access to the server. In the following example tHTTPd was launched on port 4242 and in the shared directory */home/chris/public*; the logfile is called */home/chris/thttpd.log*. Now let's take a look at the entries the server writes to the logfile – see listing 2.

A host with the IP address *10.0.0.200* has accessed our server twice, using the *Konqueror* browser in both cases. The first request was to read the start directory. The browser has no way of knowing that / really is */home/chris/↵ public*.

The second request attempts to read a file called */favicon.ico*, but as this does not exist, the server issues the HTTP Error 404 ("not found").

Option Sets

If you often launch tHTTPd with the same set of options, you can save yourself some typing by placing them in a configuration file. In this case, the only option you will need is *-C configfile*, in order to assign a configuration file to the server. The entries in the file are slightly different than the command line options. Listing 1 contains an overview of some commands we have already used, and a new option.

To launch the server, you simply type *thttpd -C myweb.conf*. The line with *cgipat=/cgi-bin/*.cgi* allows tHTTPd to run **CGI** scripts, provided they are stored in */cgi-bin* below the server root directory and assuming the *.cgi* suffix.

Another program in the package, *htpasswd*, allows you to password protect specific directories: If you use this tool to create a *.htpasswd* file and then place the file in a shared directory, any users accessing the directory via their web browsers will need to know the stored access credentials.

If you intend to use tHTTPd as a permanent web server, local users can add their own homepages by typing *makeweb*. The best way to find out more about options like this is to read the man pages (*man htpasswd, man makeweb*) and the online documentation on the tHTTPd website. ■

Listing 1: Configuration file myweb.conf

```
# Configuration file for thttpd
port=4242
dir=/home/chris/public
logfile=/home/chris/thttpd.log
cgipat=/cgi-bin/*.cgi
```

Listing 2: logfile thttpd.log

```
chris@camera:~ $ thttpd -p 4242 -d ~/public -l ~/thttpd.log
chris@camera:~ $ tail -f ~/thttpd.log
10.0.0.200 - - [18/Jun/2002:15:03:22 +0200] "GET / HTTP/1.1" 200 50000 ""
"Mozilla/5.0 (compatible; Konqueror/2.2.1; Linux)"
10.0.0.200 - - [18/Jun/2002:15:03:27 +0200] "GET /favicon.ico HTTP/1.1"
404 0 "" "Mozilla/5.0 (compatible; Konqueror/2.2.1; Linux)"
```