Configuring ProFTPD
(via `proftpd.conf`)

The Configuration File
The first step in configuring a `proftpd` daemon is knowing where the configuration file, usually named `proftpd.conf`, is located. The default location for this file is `/etc/proftpd.conf` or `/usr/local/etc/proftpd.conf`, depending on your installation. When starting the daemon, the exact path to the configuration file to be used can always be specified using the `-c` command-line option.

The idea behind `proftpd`'s handling of the configuration file is that a blank file can be used, and the daemon will still operate. This means that, unlike Apache, there is a "default" server configuration in every `proftpd.conf`; ProFTPD does not require that all server configurations be explicitly written in the `proftpd.conf` file. This default server attempts to bind to the IP address of the hostname indicated by the `hostname(1)` command.

Configuration Format
The format of the `proftpd.conf` file is deliberately designed to resemble the format used by Apache: lines of configuration directives contained with different contexts. A list of the configuration directives for ProFTPD is available here:

http://www.proftpd.org/docs/

When reading the description for the configuration directives, this key might be useful:

http://www.castaglia.org/proftpd/doc/contrib/configuration-directive-key.html

It describes the description format, and lists the different contexts in the configuration file. The "server config" context is the one in which most of your configuration directives will most likely be placed.

Two new configuration directives were introduced in `1.2.6rc1`: `<IfModule>` and `<IfDefine>`. These work exactly like Apache's directives of the same names, providing the ability to have conditional sections in the configuration file.

Starting the Daemon
One of the first decisions you will need to make is whether you will be running your ProFTPD server as an `inetd` service, or as a `standalone` server. This is controlled in the `proftpd.conf` using the `ServerType` configuration directive (see the ServerType page).

Server Identity
The daemon must be started with root privileges in order to do things like binding to port 21 and chrooting FTP sessions. However, it is not a good idea to leave a long-lived process running as root. The `User` and `Group` configuration directives are thus recommended. These directives configure the identity to which the daemon will switch, after accomplishing its startup tasks. The daemon will switch to the configured `User` and `Group` in the "server config" context. (Note that this switch uses the *effective* UID/GID, not the *real* UID/GID. Some programs, such as `top`, will continue to report `proftpd` as running as `root`; this is because the program displays the real UID/GID of processes. The `proftpd` daemon retains root privileges for operations such as chroots and binding to port 20 for active data transfers. If you wish `proftpd` to drop all root privileges, use the `RootRevoke` configuration directive.)

For this reason, it is recommended that a non-privileged identity be used. Many sites choose to use user `nobody`. Historically, this role account was used by NFS-related processes; over time, many other applications default to using user `nobody`. This has the side-effect of adding to the "privileges" held by user `nobody`, in terms of files owned and/or accessible by that user. Instead, I personally recommend that a new role account be created for use specifically by the daemon, a user `ftpd`, and perhaps even a group `ftpd`. Many systems that run Apache have a user `www` or user `apache` for use by the `httpd` daemon; similarly, a separate user should be created for the `proftpd` daemon.

In the default configuration file that accompanies the proftpd source code, there appears:

```
User nobody
Group nogroup
```

When trying to start the daemon, many users encounter the "no such group 'nogroup'" error message. There are really no reasonable defaults for those directives. The error message is a way of telling you to create the role accounts mentioned above.

For every connection, `proftpd` creates a new process to handle that client/connection. Once that client has successfully authenticated, then that process switches to the identity/privileges (*e.g.* UID, primary and supplemental GIDs, *etc*) of the authenticated user. Thus all browsing, uploads, and downloads that clients do happen as the user as which they are logged in.

Logging in
By default, the `proftpd` daemon reads the host's `/etc/passwd` file for logging in users. This means that to add FTP users, you simply need to create new system accounts for those users in your `/etc/passwd` file.

Sometimes, though, sites want "virtual", FTP-only users. In order to support such configurations, the `AuthUserFile` configuration directive can be used (see here for details).

For the purpose of authenticating users using other means, there are various authentication modules: `mod_sql`, `mod_ldap`, `mod_radius`, *etc*. Authentication and the login process is discussed here in more detail.

For setting up anonymous logins, there is the `<Anonymous>` configuration context. If there are no `<Anonymous>` sections in your `proftpd.conf`, then no anonymous logins will be allowed - simple. As mentioned in the description, the `User` directive in an `<Anonymous>` context determines what username is treated as an anonymous login. The main other thing to know about anonymous logins is that ProFTPD automatically chroots anonymous logins.

For normal, non-anonymous logins, jails/chroots are configured using the `DefaultRoot` directive. This is the configuration directive used to restrict users to their home directories, to keep them from browsing around the site. There is a page covering chrooting here.

If you use `<VirtualHost>` sections, and it seems that your server configuration is not being seen by connecting clients, you might need to check that, if using a DNS name instead of an IP address in your `<VirtualHost>` line, that name resolves to an IP address different from that of the "default" server. Many people new to ProFTPD get the impression that since the configuration syntax looks similar to Apache's, things like name-based virtual hosting will work as well. Unfortunately, this is not possible. It is not a limitation in ProFTPD, but rather in the RFCs that define FTP. See the virtual server page for more information.

As a workaround, some sites configure virtual servers to run on non-standard ports, using the `Port` configuration directive. As long as the clients are aware of the non-standard port, this scheme works well. One minor little caveat to keep in mind, when using this approach, is

the numbers used: the RFCs mandate that the daemon, for the purposes of active data transfers (as opposed to passive) use port `L-1` as the source port for the data connection, where `L` is the port number at which the client contacted the server. This means that servers that use the standard port 21 for FTP will use port 20 as the source port for their active data transfers. Passive data transfers do not have this restriction. The restriction comes into play when choosing non-standard port numbers for virtual hosts. For example, this configuration would cause problems for clients of the second virtual server that wanted to use active data transfers:

```
<VirtualHost a.b.c.d>
  Port 2121
  ...
</VirtualHost>

<VirtualHost a.b.c.d>
  Port 2122
  ...
</VirtualHost>
```

The second virtual would attempt to use port 2121 as the source port for an active data transfer, but would be blocked, as the first virtual server is already using that port for listening.

Access Restrictions
Many sites like to have specific directories for uploads, and other directories only for downloads; some sites like to allow downloads, but no browsing of directories or their contents. For configurations to achieve this, use combinations of the `<Directory>` and `<Limit>` configuration directives. There are separate pages that cover these configuration sections:

- `<Directory>`
- `<Limit>`

Further Questions
Hopefully this document answers some of your questions, or at least enough to get you started. In addition, you should take a look at some of the example configuration files. Once you are comfortable with the configuration file format, a reading of all the configuration directives' descriptions is recommended, especially if you plan on having more complex configurations. When trying to figure out why something is not working, make use of server debugging output.

If you still have questions, the users mailing list is the best place to post them.

Last Updated: *$Date: 2009/09/04 22:26:49 $*
