LinuxFocus article number 341
http://linuxfocus.org

by Mario M. Knopf
<netzmeister/at/neo5k/dot/org>

About the author:

Mario enjoys to keep busy with Linux, networks and other topics concerning security related issues. In his spare time he takes care of two webpages: neo5k.org and linuxwallpapers.de.

Translated to English by:
Jürgen Pohl
<sept.sapins/at/verizon.net>

vsftpd - An Introduction to the Very Secure FTP Daemon

Abstract:

This article gives a basic introduction to the "Very Secure FTP Daemon". I am beginning with a general description of FTP and vsftpd. After that we will have a look at the installation, configuration and start options of the vsftp daemon. We are closing with a short functions test.

Introduction

The File Transfer Protocol’s purpose is the platform independent data transfer of the internet, it is based on a server/client architecture. RFC 959[1] determins FTP to be split in two different channels, one serves for the data (TCP-port 20) and the other for the control (TCP-port 21). Over the control channel the two sides (server and client) exchange commands for the initiation of the data transfer
A FTP connection involves four steps:

- User authentication
- Establishing the control channel
- Establishing the data channel
- Discontinuing the connection

The FTP is using is the connection controlling TCP (Transmission Control Protocol) as transmission protocol which assures the arrival of the data at the recipient. Therefore there is no need for FTP to be concerned about paket loss or error checking during the data transfer. Simply expressed TCP makes sure that each data paket is arriving only once - without errors and in the correct sequence.

Data transmission differentiates between three different types of transfer whereas the completion of the stream mode is marked by an end-of-file (EOF) and in the two other transfer modii with an end-of-record (EOR) marker.

- Stream
- Block
- Compressed

In addition there are two different transfer modes:

- ASCII
- Binary

The ASCII-mode is being used for the transfer of text files, the binary mode is being used to transfer programs and similar data. The user does not need to select the transfer mode specifically since by now all FTP clients switch to the recognized type of file to be transfered.

Since the user recognicion and the password of the authentification are not encrypted it is very important to point out this potential security risk. This is the reason for some thoughts about the security of the FTP. In October 1997 RFC 2228[2] was finally published, which defined security specific addendums to the File Transfer Protocol.

vsftpd

vsftpd represents a server for unix like operating systems, it runs on platforms like Linux, ?BSD, Solaris, HP-UX and IRIX. It supports many features which are very much missed on other FTP-servers. Some of them are:

- very high security requirements
- band width limits
- good scalability
- the possibililty to create virtual users
- IPnG support
- better than average performance
- the possibility to assign virtual IPs
- high speed

The name *vsftpd* stands for "very secure FTP daemon", which is one of the primary concerns of developer Chris Evans. From the very beginning of the development and the design of the FTP server high security was one of the concerns.

One example is the fact that *vsftpd* is operated in *chroot* mode, which means a program (in this case *vsftpd*) is assigned a new root directory (l), it can no longer access programs or files outside of that directory - it is so to speak ’locked up’. Should a FTP-server be compromised the potential attacker will be isolated from the rest of the system and extensive damage will be prevented. More information on *chroot* can be found in the the artikel under [3]. Article[4] is recommended for those interested in the specific security mechanism of *vsftpd*

With the many features - of which the requirements for the security of the FTP-service should be the highest priority - *vsftpd* elevates itself significantly above the other FTP-servers. WU-FTPD[5] may be mentioned here as a negative example due to a variety of numerous security gaps in the past couple of years.

Installation

The installation of the *vsftpd* daemon is fairly easy since complete RPM packages of *vsftpd* can be found in each main distribution, in many cases it is already installed. Otherwise the source can be found at [6] and manually installed.

After getting the source unpack the tarball, go to the directory created and execute *make*. Here is a demonstration of the necessary commands:

```
neo5k@phobos> tar xzvf vsftpd-x.x.x.tar.gz
neo5k@phobos> cd vsftpd-x.x.x
neo5k@phobos> make
```

Prior to this we should check if the user *"nobody"* and the directory *"/usr/share/empty"* exists and if necessary we need to create them. If we are planning access for anonymous users a user *"ftp"* with the home directory *"/var/ftp"* needs to be generated. This will be accomplished by entering the following two commands:

```
neo5k@phobos> mkdir /var/ftp
neo5k@phobos> useradd -d /var/ftp ftp
```

Because of security reasons the directory *"/var/ftp"* should not belong to the user *"ftp"*, nor should he have writing privileges in it. With the next two commands we can change the owner and take writing privileges from other users if the user is already existing:

```
neo5k@phobos> chown root.root /var/ftp
neo5k@phobos> chmod og-w /var/ftp
```

After all pre-conditions are met we can now install the *vsftp*-daemon:

```
neo5k@phobos> make install
```

The manpages and the program should now be copied to the correct location in the data system. In case of unexpected complications manual copying of the files should do the job.

```
neo5k@phobos> cp vsftpd /usr/sbin/vsftpd
neo5k@phobos> cp vsftpd.conf.5 /usr/share/man/man5
```

```
neo5k@phobos> cp vsftpd.8 /usr/share/man/man8
```

Since the sample of our configuration file at this point has not been copied - it would make the introduction easier - we need another manual entry:

```
neo5k@phobos> cp vsftpd.conf /etc
```

Configuration

The configuration file may be found under *"/etc/vsftpd.conf"*. Like with most configuration files comments are being marked with an initial hash key.

```
# Comment line
```

An examplary configuration could look like this:

```
# Anonymus FTP-access permitted? YES/NO
anonymous_enable=NO

# Permit anonymus upload? YES/NO
anon_upload_enable=NO

# Permission for anonymus users to make new directories? YES/NO
anon_mkdir_write_enable=NO

# Permission for anonymus users to do other write operations - like renaming or deleting? YES/NO
anon_other_write_enable=NO

# Log on by local users permitted? YES/NO
local_enable=YES

# Shall local users be locked into their home directory? YES/NO
chroot_local_user=YES

# Highest permitted data transfer rate in bytes per second for local logged on users. Default = 0
(unlimited)
local_max_rate=7200

# General write permission? YES/NO
write_enable=YES

# Enable messages when changing directories? YES/NO
dirmessage_enable=YES

# Welcome banner at users logon.
ftpd_banner="Welcome to neo5k’s FTP service."
```

```
# Activate logging? YES/NO
xferlog_enable=YES

# Logging of all FTP activities? YES/NO
# Careful! This can generate large quantities of data.
log_ftp_protocol=NO

# Confirm connections are established on port 20 (ftp data) only. YES/NO
connect_from_port_20=YES

# Timeout during idle sessions
idle_session_timeout=600

# Data connection timeout
data_connection_timeout=120

# Access through Pluggable Authentication Modules (PAM)
pam_service_name=vsftpd

# Standalone operation? YES/NO - depending on operation mode (inetd, xinetd, Standalone)
# The author's FTP service is being startet with xinetd, therefore the value here is NO.
listen=NO
```

Starting the FTP-Service

vsftpd may operate in three different ways. One is through *inetd* or *xinetd*, the third is standalone operation.

inetd

If the FTP service shall be operated with *inetd* we open the configuration file *"/etc/inetd.conf"* with an editor:

```
neo5k@phobos> vi /etc/inetd.conf
```

We search for the lines pertaining to the FTP services and remove the comment mark in front of the *vsftpd* entry. If there is no such entry we may enter it. After that we have to restart *inetd*. The entry should look like this:

```
# ftp   stream   tcp   nowait   root   /usr/sbin/tcpd   in.ftpd
ftp   stream   tcp   nowait   root   /usr/sbin/tcpd   vsftpd
```

xinetd

It is recommended to start the *vsftp* daemon with *xinetd* which is more up to date than *inetd*. Some updates are e.g. logging of requests, access control, binding of the service to a specific network interface and so on. A very good introduction to *xinetd* can be found under [7]. After the modification restart of *xinetd* is necessary. The configuration of *xinetd* could look like this:

```
# vsftp daemon.
service ftp
{
     disable = no
     socket_type = stream
     wait = no
     user = root
     server = /usr/sbin/vsftpd
     per_source = 5
     instances = 200
     no_access = 192.168.1.3
     banner_fail = /etc/vsftpd.busy_banner
     log_on_success += PID HOST DURATION
     log_on_failure += HOST
     nice = 10
}
```

Standalone Operation

There is also the possibility to operate the *vsftp* daemon in standalone mode. For this we open again the file *"/etc/vsftpd.conf"* and make the following changes:

```
# Shall the vsftp daemon run in standalone operation? YES/NO
listen=YES
```

After that entry the daemon can be startet with following entry

```
neo5k@phobos> /usr/sbin/vsftpd &
```

If the search path has been entered correctly this entry will do the start

```
neo5k@phobos> vsftpd &
```

With the next entry we can check if the search path was entered correctly:

```
neo5k@phobos> echo $PATH
/usr/sbin:/bin:/usr/bin:/sbin:/usr/X11R6/bin
```

In standalone mode we have, of course, to watch that the *vsftp* daemon is not startet with *inetd* or *xinetd*.

Operation Test

After successful installation and configuration we are able to access our FTP server for the first time.

```
neo5k@phobos> ftp phobos
Connected to phobos
220 "Welcome to neo5k’s FTP service."
Name (phobos:neo5k): testuser
331 Please specify the password.
Password:
230 Login successful
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -l
229 Entering Extended Passive Mode
150 Here comes the directory listing
drwxr-xr-x      11  500      100        400  May 07 16:22  docs
drwxr-xr-x       9  500      100        464  Feb 01 23:05  hlds
drwxr-xr-x      39  500      100       4168  May 10 09:15  projects
226 Directory send OK.
ftp>
```

Conclusion

As we noticed the *vsftp* daemon is not hard to install nor to configure. It offers many features and a high degree of security.

Of course, this introduction offers only a small glimpse into the environment offered by *vsftpd*, since the FTP server is offering numerous possibilities of configuration. Those of you who would like to research *vsftpd* more in depth should visit the project page[6] and review the extensive documentation.

Links

[1] ftp://ftp.rfc-editor.org/in-notes/rfc959.txt [RFC 959 - File Transfer Protocol]
[2] ftp://ftp.rfc-editor.org/in-notes/rfc2228.txt [RFC 2228 - FTP Security Extensions]
[3] linuxfocus.org: article225, January2002 [chroot]
[4] http://vsftpd.beasts.org/DESIGN [Security vsftpd]
[5] http://www.wu-ftpd.org/ [WU-FTPD]
[6] http://www.vsftpd.beasts.org/ [Home of vsftpd]
[7] linuxfocus.org: article 175, November2000 [xinetd]
