

Name: glftpd ftp server
Vendor URL: <http://www.glftpd.com/>
Author: Karol Więsek <appelast@drumbass.art.pl>

Issue:

Logged in users to glftpd server could gain root privileges in the system, destroy any file appending to it or unzip any zip archive.

Description:

Glftpd is a ftp server. It is a kind of warez ftp (like serv-u, war-ftp). It has its own virtual users, groups etc. (it doesn't use system files). It has built in request and message system, which allow to communicate between users. After connecting it chroots, and moreover users are chrooted second time. The second chroot is not typical chroot (for example commands could modify files outside of a chroot), but users could not get out of it.

Details:

1) Writing to any file with effective uid equal 0

Messaging system, which allows users to communicate each other is vulnerable to simple attack, which allows any logged user to append his own line (unfortunately formatted - which disallows uid escalation) to any file.

Sending messages uses following algorithm :

- Check if user exists by stat'ing /ftp-data/users/username, where username is not checked at all.
- Open file /ftp-data/users/username in append mode and write formatted line.

Attacker by setting destination username to "../..site/incoming/file" could destroy target file (f.ex. zipfile).

2) Unpacking any file

Version 1.25 allows users to check, list and verify uploaded zip files using unzip. This id done using following commands.

system command	glftpd command
unzip -tqq file	site zipchk file
unzip -l -v file	site ziplist file
unzip -p -C file *.nfo *.NFO	site nfo file

There is possibility to pass filename with additional parameters to unzip.

After checking argument parsing function in unzip, there is possibility to trick unzip, using the following command unzip is obligated extract target file.

unzip -l -v --l --v file
(for more details see unzip.c)

So, command `site ziplist --l --v file`, will extract target file.

3) Bad `eid` restoring

All config, messages, help files are owned by root, so any password, tagline changing, message sending are connected with temporary changing of `eid`. There is one situation where restoring old `eid` is broken.

```
1301 read(0, "SITE onel zupa\r\n", 1024) = 16
1301 alarm(900) = 888
1301 getuid() = 0
1301 getpid() = 1301
1301 rt_sigprocmask(SIG_BLOCK, ~[STKFLT CONT PWR RT_0 RT_1 RT_2 RT_3
RT_4 RT_5 RT_6 RT_7 RT_8 RT_9 RT_10 RT_11 RT_12 RT_13 RT_14 RT_15 RT_16
RT_17 RT_18 RT_19 RT_20 RT_21 RT_22 RT_23 RT_24 RT_25 RT_26 RT_27 RT_28
RT_29 RT_30], [], 8) = 0
1301 setresuid(ruid 4294967295, eid 0, suid 4294967295) = 0
1301 open("/ftp-data/misc/oneliners.1301.temp", O_WRONLY|O_CREAT|O_TRUNC,
0666) = 6
1301 open("/ftp-data/misc/oneliners", O_RDONLY) = 7
1301 fstat(7, {st_mode=S_IFREG|0666, st_size=428, ...}) = 0
1301 old_mmap(NULL, 4096, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x4015e000
1301 read(7, "appelast 10409213"..., 4096) = 428
1301 read(7, "", 4096) = 0
1301 fstat(6, {st_mode=S_IFREG|0666, st_size=0, ...}) = 0
1301 old_mmap(NULL, 4096, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x4015f000
1301 time(NULL) = 1040922613
1301 close(7) = 0
1301 munmap(0x4015e000, 4096) = 0
1301 write(6, "appelast 10409213"..., 535) = 535
1301 close(6) = 0
1301 munmap(0x4015f000, 4096) = 0
1301 unlink("/ftp-data/misc/oneliners") = 0
1301 rename("/ftp-data/misc/oneliners.1301.temp", "/ftp-data/misc/oneliners") = 0
1301 setresuid(ruid 4294967295, eid 0, suid 4294967295) = 0
1301 rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
1301 write(1, "200 Your oneliner has been added"..., 35) = 35
1301 rt_sigaction(SIGALRM, {0x8058f60, [ALRM], SA_RESTART|0x4000000},
{0x8058f60, [ALRM], SA_RESTART|0x4000000}, 8) = 0
1301 gettimeofday({1040922613, 745799}, NULL) = 0
1301 time([1040922613]) = 1040922613
1301 read(0, <unfinished ...>
```

glftpd 1.25 is confirmed vulnerable to all 3 attacks, since glftpd 1.26
commands `site [info|ziplist|zipchk]` was disabled and changed, but rest 2 bugs

are present in all version including newest 1.28.