Reverse engineering 2 Write up

Download the crackme from github. You can check the permissions with the command **ls –la**.

Run **chmod u+x crackme** to get execute permission

```
[osboxes@osboxes Downloads]$ ls -la
total 24
drwxr-xr-x.  2 osboxes osboxes  4096 Feb 19 22:26 .
drwx------. 16 osboxes osboxes  4096 Feb 21 12:32 ..
-rwxrw-r--.  1 osboxes osboxes 13000 Feb 18 22:01 crackme
```

We can run the program to see what it does.

```
Success![osboxes@osboxes Downloads]$ ./crackme
Enter the password
password
Try harder.
[osboxes@osboxes Downloads]$
```

Run the executable with **./crackme** in a different terminal to get the process in a sleep state

Run the **ps -elf | grep crackme** and **pstree -p <pid>** to see the process and the child process

- The -e writes info out to standard output about all processes
- The l generates a long listing
- The f generates a full listing

A process with a T means that the process has terminated so we are looking for an S for a process that is sleeping.

```
[osboxes@osboxes Downloads]$ ps -elf | grep ./crackme
0 T osboxes   2223  2032  0  80   0 -  1120 signal 12:34 pts/0    00:00:00 ./crackme
1 T osboxes   2227  2223  0  80   0 -  1087 signal 12:34 pts/0    00:00:00 ./crackme
0 S osboxes   2239  2032  0  80   0 -  1120 wait_w 12:34 pts/0    00:00:00 ./crackme
1 t osboxes   2240  2239  0  80   0 -  1087 ptrace 12:34 pts/0    00:00:00 ./crackme
0 S osboxes   2242  2190  0  80   0 - 29932 pipe_w 12:34 pts/1    00:00:00 grep --color=auto ./crackme
```

```
[osboxes@osboxes Downloads]$ pstree -p 2239
crackme(2239)───crackme(2240)
```

Run **gdb –pid= <pid>** to attach a debugger to the parent process. In this case it is the process with the id 2239.

Run the **quit** command in gdb to see if the right process is attached

```
[osboxes@osboxes Downloads]$ gdb --pid=2239
GNU gdb (GDB) Fedora 8.0.1-36.fc27
Copyright (C) 2017 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word".
Attaching to process 2239
Reading symbols from /home/osboxes/Downloads/crackme...(no debugging symbols found)...done.
Reading symbols from /lib64/libc.so.6...(no debugging symbols found)...done.
Reading symbols from /lib64/ld-linux-x86-64.so.2...(no debugging symbols found)...done.
0x00007f0dc1b18e01 in read () from /lib64/libc.so.6
Missing separate debuginfos, use: dnf debuginfo-install glibc-2.26-15.fc27.x86_64
(gdb) quit
A debugging session is active.

        Inferior 1 [process 2239] will be detached.

Quit anyway? (y or n) n
Not confirmed.
```

Run **gcore** to generate a core dump for the parent process

```
Not confirmed.
(gdb) gcore
Saved corefile core.2239
```

Run **detach** to detach the core dump

```
(gdb) detach
Detaching from program: /home/osboxes/Downloads/crackme, process 2239
```

Type **quit** to exit the debugger

Use **ls** to see the core.* file generated.

```
[osboxes@osboxes Downloads]$ ls
core.2239  crackme
```

Use **strings gcore.<pid>** to get all the strings from the core dump

Use **strings gcore.<pid> > FILENAME.txt** to dump the strings into a text file if it is easier to view the text that way.

```
[osboxes@osboxes Downloads]$ strings core.2239
CORE
crackme
./crackme
CORE
CORE
LINUX
IGISCORE
CORE
ELIFCORE
/home/osboxes/Downloads/crackme
/home/osboxes/Downloads/crackme
/home/osboxes/Downloads/crackme
/usr/lib64/libc-2.26.so
/usr/lib64/libc-2.26.so
/usr/lib64/ld-2.26.so
/usr/lib64/ld-2.26.so
D$      a
D$ H
D$ H
Enter the password
Success!
Try harder.
;*3$"
sssss
Enter the password
Success!
Try harder.
;*3$"
Enter the password
aliases
```

Looking through the output from the strings the success and try harder are visible and a couple of lines under that the string th15_wAs_ahard_0ne!!** is suspicious looking, so that might be the password.

```
linux-vdso.so.1
tls/
\C&S
T<,
&4XM2$
th15_wAs_a_hard_0ne!!**
////////////////
siggetmask
```

Run ./crackme with **th15_wAs_a_hard_0ne!!****.

The output is success so that would be the password.

```
[osboxes@osboxes Downloads]$ ./crackme
Enter the password
th15_wAs_a_hard_0ne!!**
Success![osboxes@osboxes Downloads]$
```