22SP CS 232: Operating System

Mystery

Author1: David Yoo (gy24)
Author2: Seong Chan Cho (sc77)
Date: Feb 22, 2022

What does this program do?

- The program runs by given command lines. It may randomize data by seeding, it may output data of the desired amount. It may sort the data. But most importantly it allows you to send data out to the network through the port. It is sent to TCP, and by connecting to that port we can see what is being sent (data). We can use netcat to connect to a certain address. The program itself just has functionalities to organize and manage items of data and send out data through different networks or servers.

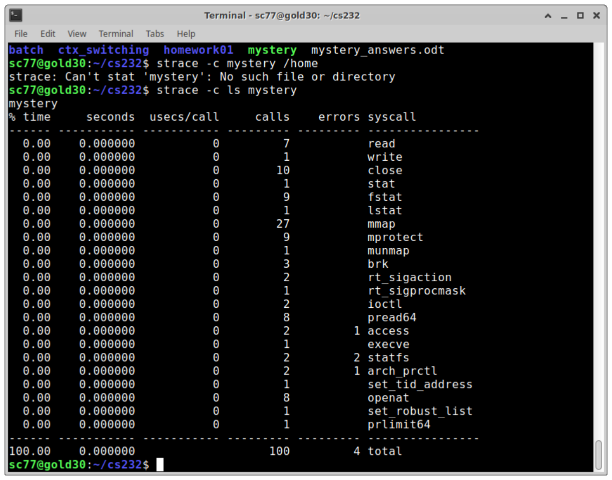
Tools used (11 tools), and 12 significant findings

Tool Used	What we learned (significant findings)	What the tool does
1. Bash[./filename] (execution)	1. It outputs a message saying that it deleted all my files but after a few seconds it follows by not, so we are guessing there is some kind of wait method in here.	It runs the main program.
	2. It deletes itself from the current directory.	
2. Command lines	3. Command lines: -h, -n, -s, -e, -p	It lets the user type in the command to execute results.
	-h: It displays the help message of what commands exist in the program.	
	-n <i>: print out <I> amount of items from the data.	
	-s: sorts the datas in ascending order.	
	-e <seed>: It randomly generates numbers (numbers in the data) in a specific seed. It will have the same sets of items if the same seed is input in the command. It starts from the previous value generated by the generator.	
	-p <port>: It sends out the data to TCP through a given port. We had to figure out how to access that port to see the values that were being sent out.	

3. strings	4. It prints on the terminal relatively and is more readable than just trying to open the program on the editor.	It extracts all the strings inside the file and displays on terminal.
4. file	5. This file is a 64 bit LSB executable, which is dynamically linked and includes so.2 library.	It displays the file type.
5. ldd	6. It contains so.1, so.6 and so.2. These have address values inside which has the location of the matching object and address where it is loaded.	Print this shared objects (shared libararies)
6. readelf	7. The specification of this executable file.	It prints out the file format and information about this file.
7. ls-lh	8. The size of this program is 14K, and the file was last modified on February 16th, at 08:44.	Gives out the details of the file information.
8. Objdump -d	9. It displays the assembly language of this executable file. We were able to see such functions like srandom(), sleep(), unlink(), puts(), qsort(), write(),close(),…etc functions implemented in the program. Then we were also able to see which memory spaces it used through the main().	It shows the disassembly of the executable code.
9. Netcat (nc local host [port])	10. We learned that in this program, we can use nectar to receive data that was sent from -p (port). For example, we used ./mystery -p 19494 to send the data from mystery and checked the sent data with netcat localhost 19494. We had to open two terminals to send and listen from two different windows.	Netcat allows to read the data from a specific port.
10. Strace (Strace -c ls mystery)	11. All the system calls are used in the program. It uses calls like close(), fstat(), mmap(), write(), read(), set_tid_address(),…etc. These system calls are called throughout the program whenever needed to access the OS.	Strace -c ls mystery: It displays a table of how many of all the system calls were called throughout the system. For example, read was called 7 times, close was called 10 times… (see picture attached below)
11. Mstrace	12. We have checked there is no memory leak in this program.	This command allows user to check if there is any memory leaks in the program.

Bug

- When ./mystery -p is inputted in the terminal, it returns segmentation fault, kills the program.



```
batch  ctx_switching  homework01  mystery  mystery_answers.odt
sc77@gold30:~/cs232$ strace -c mystery /home
strace: Can't stat 'mystery': No such file or directory
sc77@gold30:~/cs232$ strace -c ls mystery
mystery
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
  0.00    0.000000           0         7           read
  0.00    0.000000           0         1           write
  0.00    0.000000           0        10           close
  0.00    0.000000           0         1           stat
  0.00    0.000000           0         9           fstat
  0.00    0.000000           0         1           lstat
  0.00    0.000000           0        27           mmap
  0.00    0.000000           0         9           mprotect
  0.00    0.000000           0         1           munmap
  0.00    0.000000           0         3           brk
  0.00    0.000000           0         2           rt_sigaction
  0.00    0.000000           0         1           rt_sigprocmask
  0.00    0.000000           0         2           ioctl
  0.00    0.000000           0         8           pread64
  0.00    0.000000           0         2         1 access
  0.00    0.000000           0         1           execve
  0.00    0.000000           0         2         2 statfs
  0.00    0.000000           0         2         1 arch_prctl
  0.00    0.000000           0         1           set_tid_address
  0.00    0.000000           0         8           openat
  0.00    0.000000           0         1           set_robust_list
  0.00    0.000000           0         1           prlimit64
------ ----------- ----------- --------- --------- ----------------
100.00    0.000000                   100         4 total
sc77@gold30:~/cs232$
```

(Image of the output of strace -c ls mystery)

Sources

ls -lh (https://unix.stackexchange.com/questions/336845/how-to-view-the-size-of-the-binary-files-using-linux-command)
Objdump -d (https://www.codementor.io/@packt/reverse-engineering-a-linux-executable-hello-world-rjceryk5d)
netcat (https://www.varonis.com/blog/netcat-commands) (https://www.digitalocean.com/community/tutorials/how-to-use-netcat-to-establish-and-test-tcp-and-udp-connections)