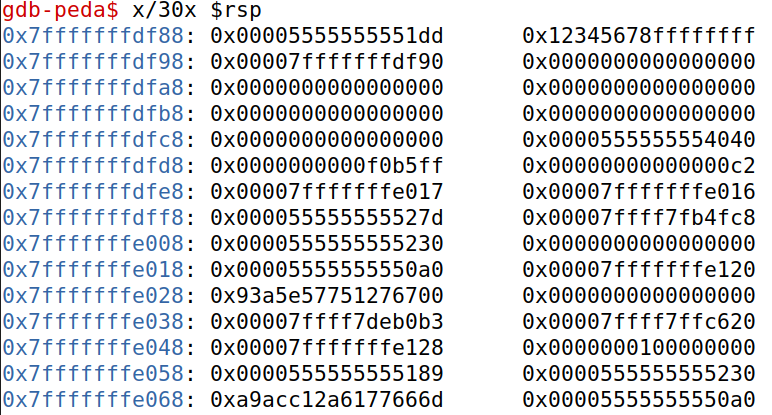
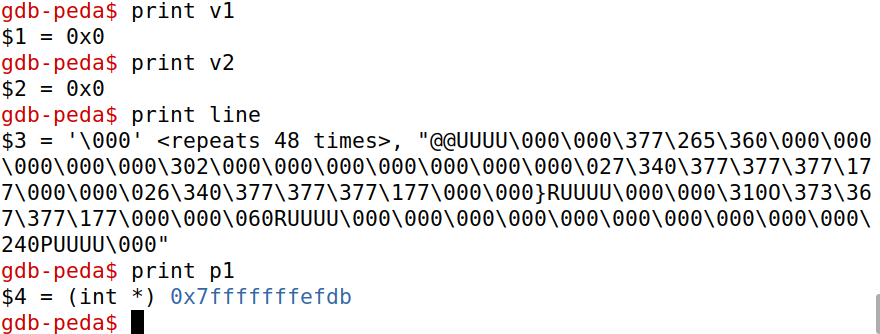
|  |  |  |
| --- | --- | --- |
| Luke Pepin | CSE 4400 - HW 4 | 4/20/2025 |

1. q1.c:

a.

|  |  |
| --- | --- |
| **Task** | **Commands** |
| Compile q1 | gcc –g q1.c -o q1 |
| Run program in GDB | gdb q1 |
| Set breakpoint | break gets |
| Find Variable Addresses | break main |
| Run Program | run |
| Inspect Stack | x/30x $rsp |



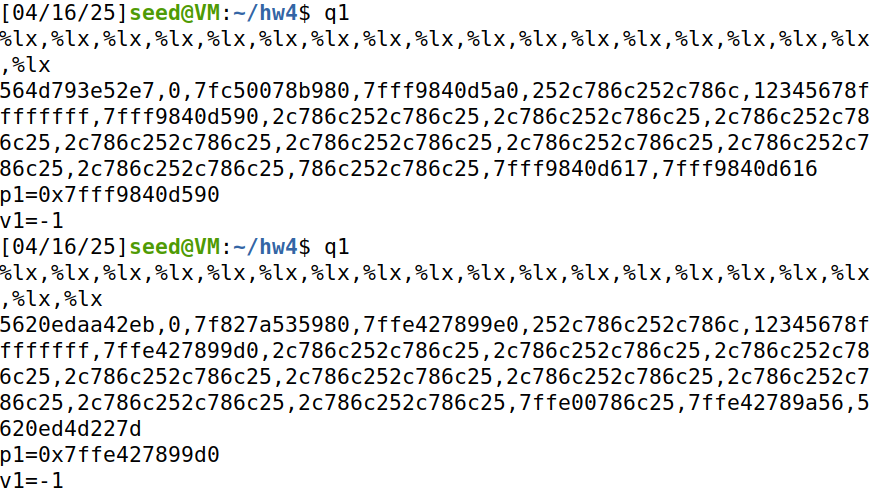


|  |  |  |
| --- | --- | --- |
| Local Variable | Address | Value |
| $rsp | 0x7fffffffdf88 | 0x00005555555551dd |
| v1 | 0x7fffffffdf98 | 0x0 |
| v2 | 0x7fffffffdfa8 | 0x0 |
| p1 | 0x7fffffffefdb | 0x7fffffffefdb |
| lineq1 | 0x7fffffffdfa0 | '\000' <repeats 48 times>, "@@UUUU\000\000\377\265\360\000\000\000\000\000\302\000\000\000\000\000\000\000\027\340\377\377\377\177\000\000\026\340\377\377\377\177\000\000}RUUUU\000\000\310O\373\367\377\177\000\000\060RUUUU\000\000\000\000\000\000\000\000\000\000\240PUUUU\000" |

b.

|  |  |
| --- | --- |
| **Task** | **Commands** |
| Run q1 in bash | q1 |
| Enter inputs | “Luke Pepin”, “%lx”, “%lx,%lx”,”%lx,%lx,%lx”, etc. |





Gave up before byte array line could be found.

c.

|  |  |
| --- | --- |
| **Task** | **Commands** |
| Run q1 in bash | q1 |
| Construct a input to make v1: 100 | “%100x%5$n” |

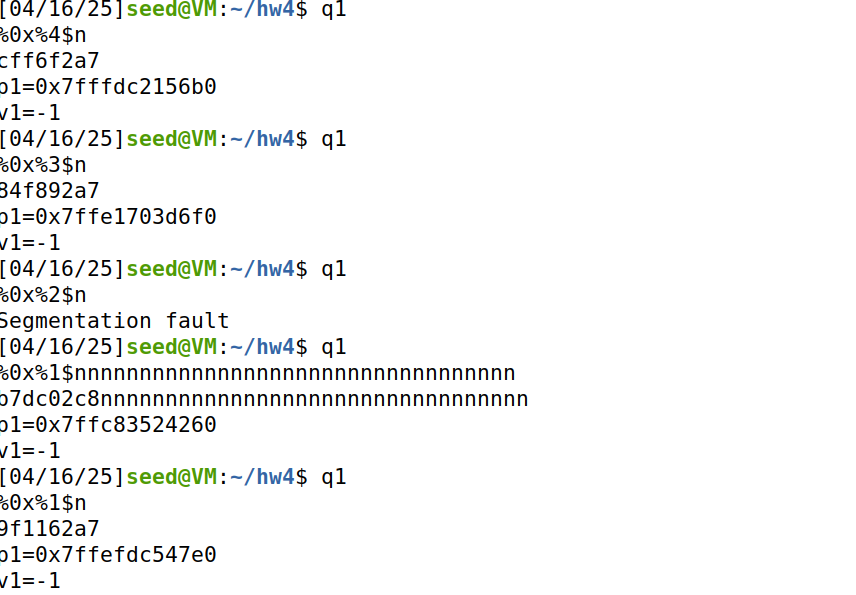
The input consists of 2 values the first prints that many characters (100) the second then tries to write the value to the stack location it points to (5). My results only vielded segmentation faults on stack locations over 4 or incorrect results.



d.

|  |  |
| --- | --- |
| **Task** | **Commands** |
| Run q1 in bash | q1 |
| Construct a input to make v1: 0 | “%0x%5$n” |

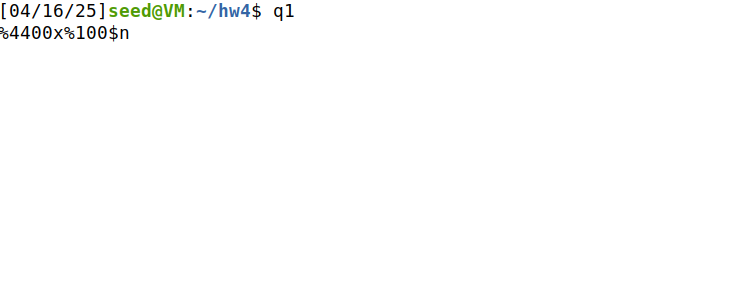
Similar to before the payload is the same however the first character is changed to 0. Results do not differ.q

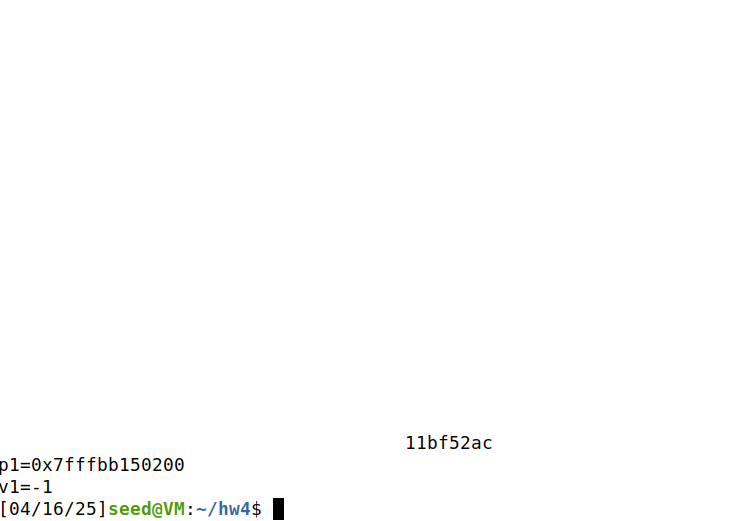


e.

|  |  |
| --- | --- |
| **Task** | **Commands** |
| Run q1 in bash | q1 |
| Construct a input to make v1: 4400 | “%4400x%100$n” |

No changes in result.

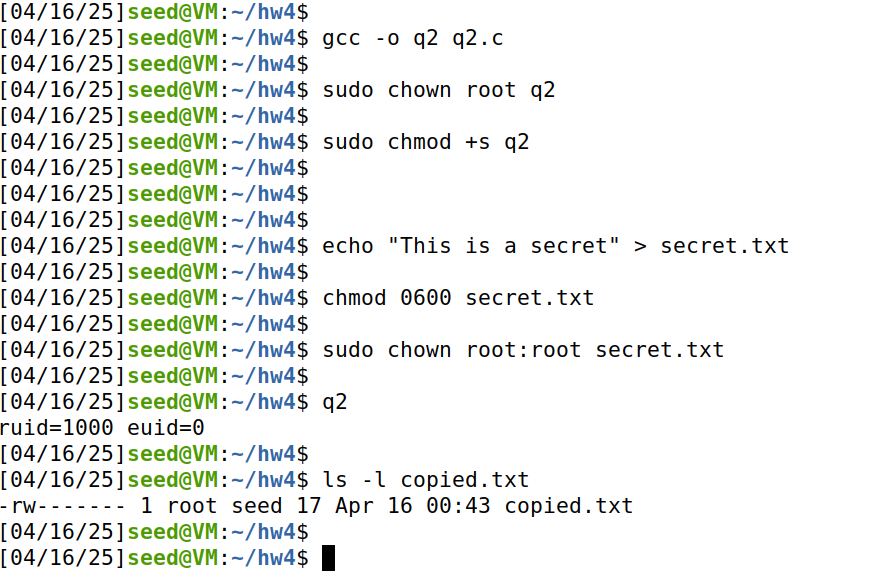




1. setuid() and seteuid():

a.

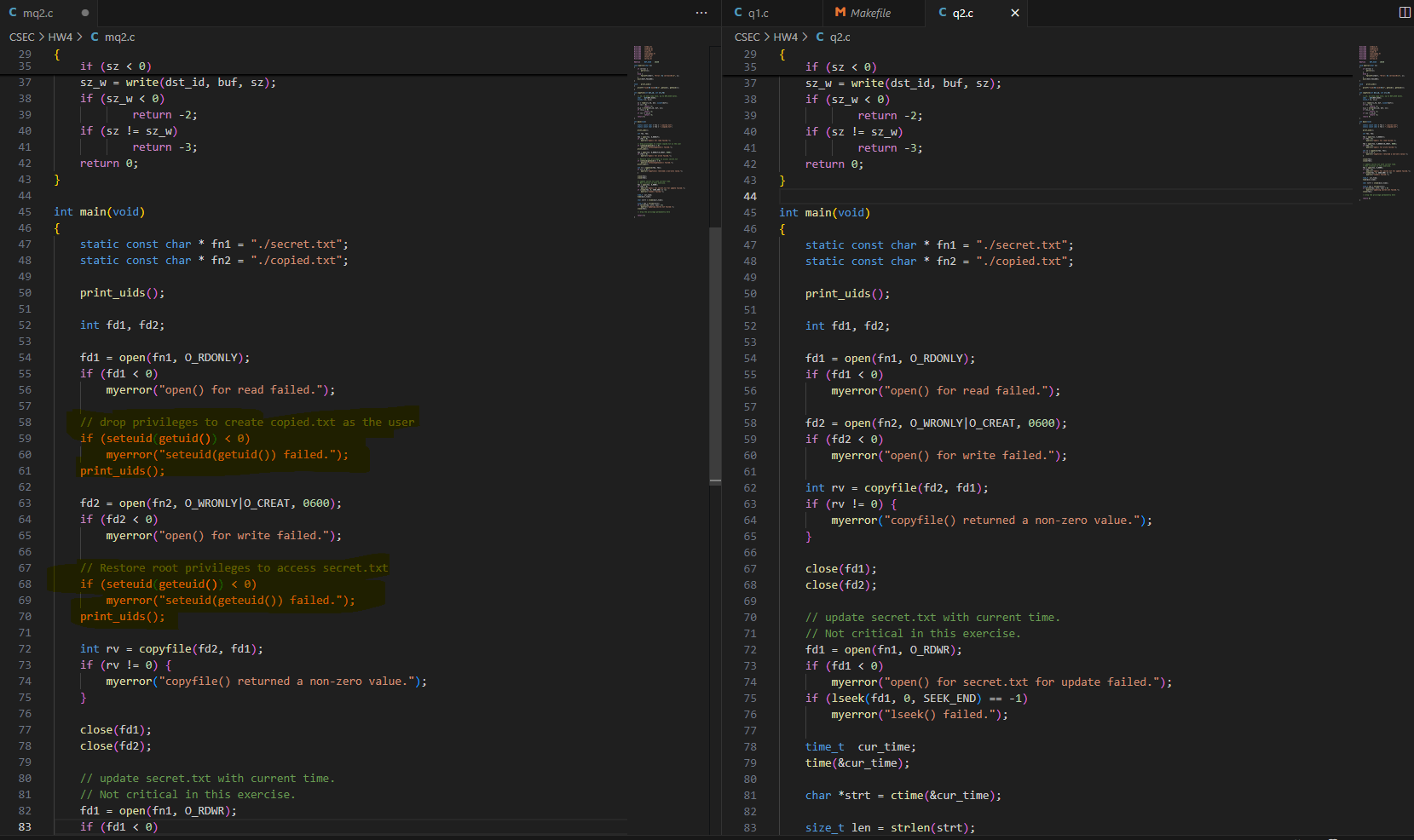
|  |  |
| --- | --- |
| **Task** | **Commands** |
| Compile q2.c, change the ownership to root, set the SUID | gcc -o q2 q2.c  sudo chown root q2  sudo chmod +s q2 |
| Make secret.txt | echo "This is a secret" > secret.txt  chmod 0600 secret.txt  sudo chown root:root secret.txt |
| Run q2 | q2 |
| Check ownership of q2 | ls -l copied.txt |

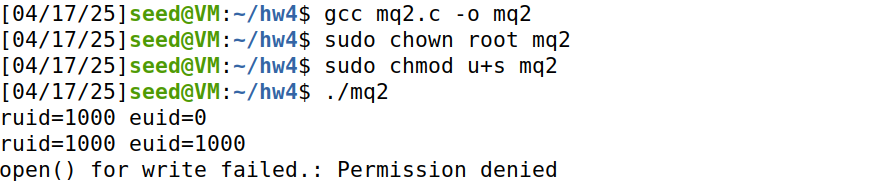
If q2 successfully runs the owner of copied.txt is root. This is because the process that creates the file runs with an effective user ID of 0 (root). 

|  |  |
| --- | --- |
| **Task** | **Commands** |
| Modify q2.c | nano mq2.c |
| Complie and run mq2.c | gcc mq2.c -o mq2  sudo chown root mq2  sudo chmod u+s mq2  ls -l mq2  mq2 |

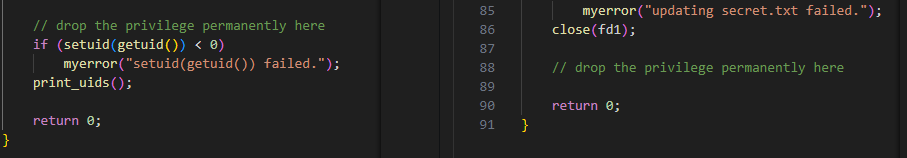
Changes to code:

Drop the root privileges with seteuid(getuid()) before creating copied.txt and restore root privileges soon after. Call to print\_uids() were also added to verify changes.



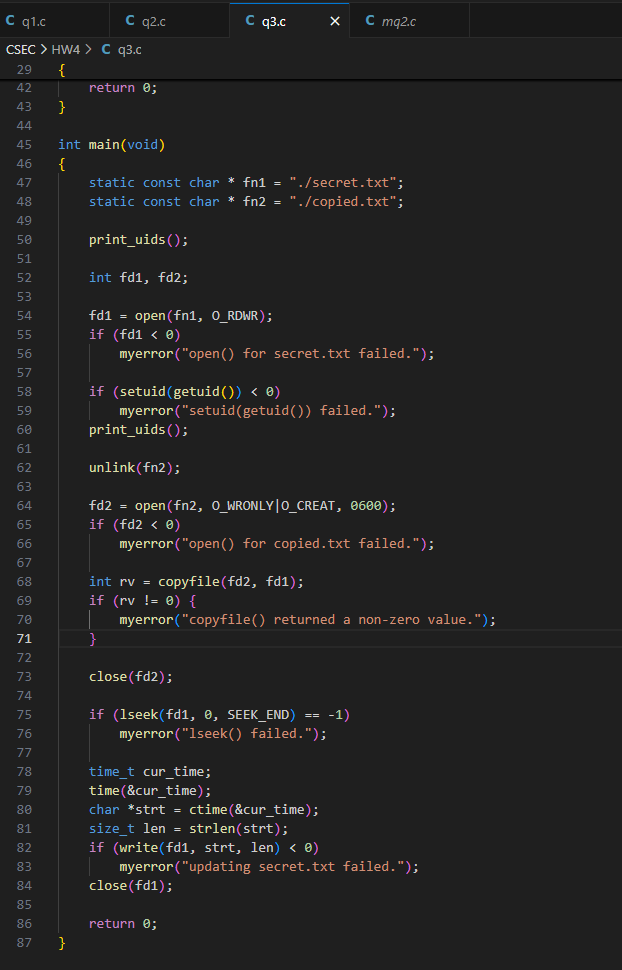
Task Failed

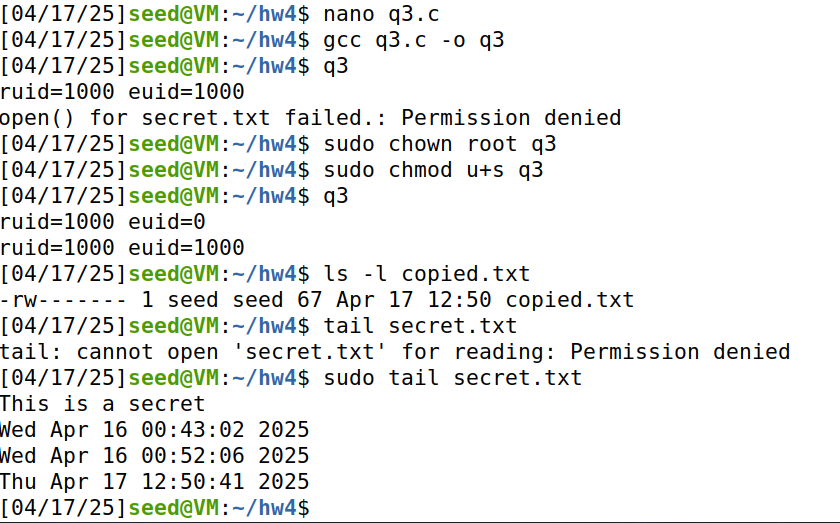
c.



1. q3.c:

Q3.c is created from q2.c, it opens secret.txt once with read/write access before dropping privileges and them uses the same open file descriptor for both copying and updating the file.





1. ELF-Virus:

a. If an executable is infected, what is the layout of the infected file?

The first part of infected file is the virus code. The second part is the original code and its contents. Third a signature is appended. This ensures the infected file performs the virus actions first.

b. Does the virus infect the file that is already infected? If not, how does the virus know if a file has been infected?

No, due to the signature appended to the back the infected code (the signature is if the last few bytes of the file equal a constant “4033”), the virus reads it as already infected and does not reinfect it.

c. While the host program is running, is the virus process running? Why?

No, while the original host program is running the virus process has already ran, due to the structure of the infected file the virus is first.

d. If a user runs an infected file in a directory that they do not have write access, what would happen? Why?

Any virus code that involves modifying files will fail, the host part of the program might still run but the features of creating, writing, renaming will result in permission denied.