COMP 3000 A3
Prepared By: Imran Gabrani-Juma
Prepared For: Professor Anil Somayaji
Course: COMP 3000
SN: 101036672

1. **Are man pages a good documentation source when developing kernel modules? Why or why not?**

   When looking at the different online resources that that we have available I would suggest that man pages are not ideal and not the correct way when looking for a good documentation course for this kind of development. During lecture and tutorial, tutorial 5 to be more specific we where instructed to use two different courses. These two sources where

   1.1 https://elixir.bootlin.com/linux/latest/source/kernel/sys.c

   1.2 https://elixir.bootlin.com/linux/latest/source

   These two sources would be a better documentation as they provide more information as well as give a better understanding for the task at hand. Given that these links where provided in class and tutorial, it shows that these are a better source than using man pages as per the professors instructions.

2. **When you load a kernel module, what code is guaranteed to run? (Which functions will definitely be called?) With what privileges will this code run?**

   The code that is “Guaranteed” to run would be the code that runs under the process of the init function, in regards to looking at our kernel module. This is done because when we go to execute this process we will be running it with the supervisor mode rather than the normal execution. As a result this process will now join the kernel, this is especially important because the kernel requires special privileged access to do this process actually and properly, now that it has this, the portion of code will be “guaranteed to run”.

3. **You attach a new USB webcam to your computer that is running Linux. The webcam is recognized and works properly.**

    1. **How could you find out what kernel module(s) were loaded, if any, when you attached the webcam?**

        once the device is plugged in one of the most efficient ways to see what is working and what kernel module(s) are running would be to run lsmode, this would give us all the information that we need.

    2. **What is a simple way you could prevent those modules from being loaded in the future?**

        In the future If we wanted to prevent this, the easiest way to complete this task would to blacklist the module. This should prevent the module from loading and protect the system. This process can be executed by running the command modprobe.blacklist= and then the name of the module in questions.

    3. **What sort of problems could the webcam drivers (modules) cause if they sometimes use dynamically allocated memory improperly (say, by using a buffer after it had been freed)?**

        Normally when we try to diagnose these kinds of problems it is often best to run a segmentation fault this is a great first step at diagnosing the problem first hand, this would be the ideal way to see this problem because we are trying to write in a created memory space that has not been properly allocated yet.

4. **What would changing CLASS_NAME in newgetpid.c to "carleton" change in the observable behavior of newgetpid?**

The clear error that we have here is that once we change the particular name of the class, it also works in hand with changing the config files that we have unevenly saved in the file system and this change has caused an issue.

5. **How do you safely access userspace memory from a kernel module? Give an example of where we do this in a kernel module covered in class.**

As explained in lecture by professor, the correct method to solving is problem is a multi-step process, in order to access userspace memory in the system we must do the following first we have to make calls to the user, from here we must

-> Copy_to_user
-> copy_from_user
-> get_user
-> put_user

from here, we must make another call call to

-> put_user in newgetpid.c

This call MUST be done in the newgetpid_read() function.

From here we can make another function call of copy copy_to_user in remember.c in function remember_read and a call to copy_to_user in the function remember_write function.

6. **What changes need to be made to newgetpid.c to allow it to respond to write requests? Hint: How does newgetpid.c respond to read requests?**

When working with this problem the ideal method would be to allow the command newgetpid.c to allow the write request. We must used add.write = newgetpid_write to the file_opperation structure. Once this process is completed we can then create the function newgetpid_write(struct file *f, char *buff, size_t len, loff_t *offset). By using this command in the process we can write a retest directly from the user. This will effectively write the process we need to the working device file.

7. **If you destroy all of the superblocks in a filesystem, will fsck be able to recover the filesystem? Explain.**

When understanding this problem we have to first understand that the file will not be able to be recovered. In conjunction fsck.ext4 cannot re[air the system as a result of us destroying all of the super blocks in the filesystem. This is a result of the superblocks that contain key information about where the file blocks are stored and saved within memory. An important concept that we must understand is then because the file is corrupted it can no longer store the map of where and data is held and allocated. Thus coming to the conclusion that fsck.ext4 cannot be fixed as the system doesn't not know where to look for the process.

8. **When would you expect to find files in the lost+found directory?**

When encountering a problem just like this I would highly say that its defiantly possible to find these files within the lost+found directory. We can come to this conclusion based on many factors but one important being that because the file has no link to it within the system as well as the fact that it cannot be found in any directory because of these links it is almost guaranteed that we will find it in the lost+found directory for this reason. Because of these idefnties we would find the files in the lost+found directory, this can be accessed when we run the process in the command line to run fsck

9. **When first connecting to a remote host via ssh, you will normally get a message saying something similar to this: "The authenticity of host 'access.scs.carleton.ca (134.117.29.72)' can't be established. RSA key fingerprint is SHA256:MexEKZF0Os0Vl6VTObN70lRf2DFsGfD8DTQ7FKKqVJ4. Are you sure you want to continue connecting (yes/no)?"**

1. Why is this question important?

When looking at this problem, is it important that we understand what we are trying to do, here we can see that the system is informing us that the user at the computer cannot successfully verify and validate if the host is who they claim to be. Thus as a result instead of the connection letting us through successfully we are asked to escape in the event that we do not trust the legitimacy of the host

2. **What is the "fingerprint" for?**

this problem the term fingerprint is a unique system token that is given to your machine, this unique token is used to allow the host to determine who can connect to the system by the fingerprint that has been assigned to the system.

10. **As user student (uid=1000), you run "sshfs <scs-username>@access.scs.carleton.ca:. scs-files", where <scs-username> is your username.**

1. **Will the files in scs-files have a uid=1000? Why or why not?**

In this problem we can identify that the system will not have the same UID, this is a result of the different mapping methods that files have in the local system. As a result both systems will have different UID

2. **Assume a file in scs-files is marked as being readable only by the owner. Will you be able to read the contents of this file? Why or why not?**

This will now allow, only the owner of the system file will be able to read it due to the current permissions, likewise anyone else who tires will be dined access. Furthermore, when the system uses SSHSF to connect to the host, we can see that it provides different types of access that are allowed permissions. Thus we cannot access the files because the SSHSF does not give us the correct permissions needed.

11. **How can you make a file that is 20 GB in size (you can read 20 GB of information from it) but takes up essentially no space on disk?**

In this case, in order for the file to take up this amount of memory we need to make a current filesystem out of the file, this it will allow for great amounts of megabytes in the hundreds and turn them into smaller kilobytes of data, thus this data will be stored on the disk and the remaining can be stored virtually.