**My Implementation**

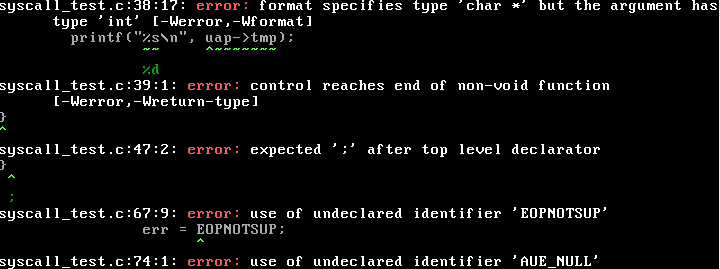
We’re going to do two things here: first, we’re going to make a rootkit – this is just because the book I’m using has it as an example, so if I run into issues it’ll serve as a good practice run. Once that’s all working, I’ll then move onto what I planned – hiding a specific folder from the system.

**Syscall Module**

So, I started by copying most of the stuff from the example given in FreeBSD book (I cut out some of the comments I had so it would fit:



When I tried to make the file I got the following errors:



Each one:

1. Easy – I changed the argument to int (from the example) but forgot to change the print.
2. Easy – I forgot to return 0 at the end of the sys\_call\_example function.
3. No ; after struct.
4. EOPNOTSUP should be EOPNOTSUPP, as supplied in sys/module.h
5. This error made me want to shoot myself: here’s why.

Error 5:

* So it turns out since the Rootkit book was written, there was an extra addition to the includes you need to run SYSCALL\_MODULE. This is #include <sys/sysproto.h>. This took me like an hour of searching, and then looking through the example code the OS provides in /sys.
* Easy right? No, and if this wasn’t going on open learning I would have some much stronger things to say about it. Anyway, I chucked this into my includes and it did fix the AUE\_NULL error. However, it then gave me like a billion “uint32\_t” has not been defined. After another half hour, I found placing this #include after #include <sys/param.h> fixed it.
* This was too hacky for me, so I found that just making sure to #include <sys/types.h> before fixed it.
* You know what’s very dumb? This:
* 
* This is in the header file for sysproto.h. So it’s obviously a dependency. But guess what? For god only knows what reason, IT DOESN’T RUN. I do not have nearly enough energy to trace this problem to its core, so I’m just going to assume something changed somewhere else in the OS at some point, which cause this #include to not run.

Sick, so this finally makes. I ran it and it didn’t print! Yay! This was a quick fix though. I was doing printf in my load function. However, printf will just print to the kernel’s buffer, only viewable with dmesg. Changing this to uprintf made it output the the userspace, and so I finally had this System call module working.