RECITATION 11

INFO

- Jon Rutkauskas
- Recitation: Tue 12-12:50
- Office Hours: Tue 11-11:50
  Thur 11-12:50
  SENSQ 5806
  (additional hours by appointment if needed)

- On discord: @jrutkauskas
- By email: jsr68@pitt.edu
- Website:
  https://github.com/jrutkauskas/spring2019-449-rec
- Ask me any questions you have!!!

WARMUP POLLS

"STICK" AROUND…

- "Stick" around until after this presentation is over for a fun surprise I brought for all of you…
- I think you'll find it very a*PEEL*ing
- …
- …
- …
- The surprise is stickers. I bought you all some fun stickers and you each can have one every time you come to recitation.

WHICH OF THE FOLLOWING IS –NOT- TRUE REGARDING THE DIFFERENT TYPES OF DEVICES

Character devices send and receive data 1 byte at a time, and only work in one direction. Ex: a mouse.

Block devices let you access data in chunks, and have random access. Ex: your ram.

There are many devices that are virtual, representing not a physical thing but an interface with the OS or other software in some way. Ex. /dev/urandom

Kernel modules are a type of device the OS/kernel uses to load plugins.

WHICH OF THE FOLLOWING IS –NOT- TRUE REGARDING THE DIFFERENT TYPES OF DEVICES

- Kernel Modules are *like* plugins for the OS.
- We write device drivers as kernel modules so the OS can load them dynamically
- All the other statements are true definitions.

Character devices send and receive data 1 byte at a time, and only work in one direction. Ex: a mouse.

Block devices let you access data in chunks, and have random access. Ex: your ram.

There are many devices that are virtual, representing not a physical thing but an interface with the OS or other software in some way. Ex. /dev/urandom

Kernel modules are a type of device the OS/kernel uses to load plugins.

WHICH OF THE FOLLOWING IS –NOT- A LIMITATION OF WRITING CODE THAT RUNS IN THE KERNEL

Very restricted access to memory for security reasons

Must use special kernel functions like kmalloc, kfree, kprintf

No access to the C std library

No floating point access

WHICH OF THE FOLLOWING IS –NOT- A LIMITATION OF WRITING CODE THAT RUNS IN THE KERNEL

1. The kernel actually has basically UNRESTRICTED access to memory. You can access kernel memory and user memory, though you do need the copy_to_user(…) function.
2. The kernel has no C std library, so malloc, free, printf are out of the question, but we have kernel ~equivalents. They differ in some ways, though, so it's a good idea to read the man pages (as always).
3. ^^
4. To save time on context switches, the kernel doesn't allow access to the floating point registers, so it doesn't have to save them… meaning it can't use them, so no floating point access!

Very restricted access to memory for security reasons

Must use special kernel functions like kmalloc, kfree, kprintf

No access to the C std library

No floating point access

WHICH OF THESE IS AN ADVANTAGE A USER-MODE DRIVER HAS OVER A KERNEL-MODE ONE?

Fewer context switches

Ability to read and write to files in /dev and /sys

Memory leaks and crashes only stop the driver, not the whole system.

Ability to respond to hardware interrupts

WHICH OF THESE IS AN ADVANTAGE A USER-MODE DRIVER HAS OVER A KERNEL-MODE ONE?

1. User-mode drivers actually have MORE context switches, since all operations need to go through the kernel. Program → kernel → driver → kernel → Program.
   - Vs. Program → kernel (and driver) → Program
2. Daemons, how we run user-mode drivers, have higher privileges and can use /dev and /sys… kernel mode drivers, of course, can do this too, so this is not an *advantage*.
3. When any user-mode program leaks memory, all you have to do to free the memory is kill the program, in kernel, you'd have to restart the system. When a user-mode program crashes… you've seen it. It just stops working and maybe prints a message to the screen (Segmentation Fault. Core dumped)
4. Only *kernel-mode* drivers can respond to hardware interrupts.

Fewer context switches

Ability to read and write to files in /dev and /sys

Memory leaks and crashes only stop the driver, not the whole system.

Ability to respond to hardware interrupts

OTHER QUESTIONS ON THIS MATERIAL

- This is all getting a lot more conceptual and a lot harder to make examples of
- Are there questions about the concepts you've learned in the last few lectures?

NO LAB THIS WEEK!

- Project 4 is due this weekend, though.
- Remember not to forkbomb thoth. See the instructions on the project description, or from your class slides, or from my Recitation 10 slides… or just `ulimit -u 30` every time you login
- I want you to have some time to work on that and be able ask me questions.
- Is anyone having trouble with strtok?
- Anything else, I am here!

- Also you can get a sticker now. :-)