Quiz: Limited Direct Execution

Total points 50/55

Take the quiz solo, but feel free to consult a partner student, the book, the videos or other resources if needed. Re-take quiz if your score is less than 80% or if you just want some more practice.

The respondent's email (**parth2@pdx.edu**) was recorded on submission of this form.

✓ What's the primary benefit of "Direct Execution"? * 5/5

○ Ease of Use

○ Security

○ Portability

● Speed/Performance ✓

○ Honesty

✓ Why do we need limits on direct execution? * 5/5

○ prevent programmer errors (bugs)

○ honesty

● fairness of resource use ✓

○ prevent OS crashes and process crashes caused by other processes' behavior

○ performance

○ prevent all uses of I/O devices

✓ Is a segmentation fault an example of limited direct execution? * 5/5

○ Yes, concurrent access to I/O devices is not allowed

○ No, seg faults are caused by programmer error

● Yes, the MMU limits which memory addresses can be accessed by a program ✓

○ No, it is a fault not an interrupt

✗ How do system calls help to implement limits on direct execution? * 0/5

○ because there are a limited number of system calls

○ system calls allow the OS to expose a limited set of functionality to application programs

● system calls are invoked mainly via timer interrupts which may not by altered by the application programmer ✗

○ by allowing the OS to run before and during the application's request. the OS can then enforce whatever limits it needs to.

○ by handling faults for such problems as divide by zero

✓ A typical processor chip has multiple cores. The OS runs a process on one core and a different process on a different core. This is an example of what kind of sharing? * 5/5

○ time sharing

● space sharing ✓

✓ An OS allocates some memory for a process. Then, when that process is not running, the OS reclaims the physical memory and allows another process to use the same frames of memory. This is an example of what kind of sharing? * 5/5

● time sharing ✓

○ space sharing

How frequently is each type of limit used, checked or invoked? *

	Multiple times per instruction	Approximately once per instruction	Once per many (variable number of) instructions	Approximately once per 10ms	Score	
timer interrupt	○	○	○	●	1/1	✓
system call interface	○	○	●	○	1/1	✓
MMU address translation	●	○	○	○	1/1	✓
status register mode bit	○	●	○	○	1/1	✓

Which limit prevents which bad thing from occurring? *

	MMU	Timer Interrupt	System Call Interface	OS does not limit this	Mode bit in Status Register	Score	
program attempts to run forever	○	●	○	○	○	1/1	✓
program has a logical bug	○	○	○	●	○	1/1	✓
changing the mode bit	○	○	○	○	●	1/1	✓
rewriting an interrupt handler	●	○	○	○	○	1/1	✓
program tries to read memory of another program	●	○	○	○	○	1/1	✓
changing the interrupt vector	●	○	○	○	○	1/1	✓
program tries to write file for which it does not have privileges	○	○	●	○	○	1/1	✓
modify the time period of	○	○	●	○	○	1/1	✓

the timer interrupt

✓ Can the trap instruction (called "int" on Intel x86 chips) be called from user application code? * 5/5

● Yes ✓

○ No

When does the OS run? *

	yes, OS runs at this time	no, OS does not run at this time	Score	
within a program's while loop	○	●	1/1	✓
during a system call	●	○	1/1	✓
on every I/O device interrupt	●	○	1/1	✓
when a fault occurs	●	○	1/1	✓
at boot time	●	○	1/1	✓
when a timer interrupt occurs	●	○	1/1	✓
during a function call	○	●	1/1	✓
during each CPU instruction	○	●	1/1	✓