Raymond Ng: JQG999
IS3033-Summer 2022
June 18, 2022

Exercise 03

Part 1: Memory Management and Process Scheduling Algorithms

Linux:

Linux uses *Demand Paging*, *Segmentation* and *Swapping* memory management algorithms.
References:
https://tldp.org/LDP/tlk/mm/memory.html
https://certsimple.com/does-linux-use-segmentation/
https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/9_VirtualMemory.html

Linux uses *Completely Fair Scheduling* (*CFS*) as its process scheduling algorithm.
References:
https://blog.acolyer.org/2016/04/26/the-linux-scheduler-a-decade-of-wasted cores/#:~:text=Linux%20uses%20a%20Completely%20Fair,must%20run%20at%20least%20once.

CFS specifically was not mentioned in class; however, it's based on process scheduling, which was mentioned in class. Process scheduling can be thought as a program that is loaded as process in RAM and then CPU executes the process according to the priority of the process. In *CFS*, it fairly or equally divvies the CPU time among all the processes.

Mac:

Mac uses *Demand Paging* and *Swapping*.
References:
https://www.cs.csustan.edu/~john/classes/previous_semesters/cs3750_operatingsys_i/2003_04_Fall/Notes/ch10_Virtual_Memory.html
https://lemp.io/does-mac-os-x-use-paging/

Mac uses the *Mach scheduler* as its process scheduling algorithm.
References:
https://developer.apple.com/library/archive/documentation/Darwin/Conceptual/KernelProgramming/scheduler/scheduler.html
https://docs.huihoo.com/darwin/kernel-programming-guide/scheduler/chapter_8_section_1.html

Mach scheduling was not specifically mentioned in class; however, I know *Scheduling* was identified to be one of the processes for moving memory. According to *developer.apple.com*, *Mach Scheduling* is based on a system of run queues at various priorities and handled in different ways. The priority levels are divvied into four bands by characteristics: normal, system high priority, kernel mode only, and real-time threads.

Windows:

Windows uses *Demand Paging*, *Segmentation*, and *Paging*
References:
https://answers.microsoft.com/en-us/windows/forum/all/physical-and-virtual-memory-in-windows-10/e36fb5bc-9ac8-49af-951c-e7d39b979938

Windows uses *Multilevel Feedback Queue* as its process scheduling algorithm.
References:
https://practice.geeksforgeeks.org/problems/which-type-of-scheduling-is-done-in-windows-10-os

This particular scheduling algorithm was not mentioned in class. According to *geeksforgeeks.org*, in Multilevel Feedback Queue Scheduling processes are permanently assigned to a queue on entry to the system and processes are not allowed to move between queues, allowing low scheduling overhead. Reportedly, it's the most complex algorithm.

Raymond Ng: JQG999
IS3033-Summer 2022
June 18, 2022

References:
https://www.geeksforgeeks.org/multilevel-feedback-queue-scheduling-mlfq-cpu-scheduling/#:~:text=Advantages%20of%20Multilevel%20Feedback%20Queue%20Scheduling%3A&text=It%20allows%20different%20processes%20to,to%20the%20higher%20priority%20queue.

Part 2: Medium regarding Meltdown and Spectre

i. I'm a bit of novice when interpreting these problem sets, but I was not surprised that the user-space and kernel-space tables were not previously separated. My understanding is that user-space table and kernel space are two different data structures, and because they both require access to the same information they are not segregated. The kernel-space table and the user-space table need to be able to retrieve information about kernel- and user-space processes. The reason the user-space and kernel-space tables were not separated previously is because it would have been very difficult to do so. separating the two tables would have required a lot of extra work, and it would have added a lot of complexity to the system.

ii. Yes, it makes sense that the two tables should be separated. The kernel-space table contains sensitive information that should not be accessible to user-space programs. By separating the two tables, challenges the ability of nefarious/bad actors (or attackers) to gain access to the kernel-space table.

iii. I think this type of attack underscores the importance of implementing/practicing appropriate security measures to mitigate threats. From my understanding, sometimes users may become complacent over time and skip critical steps in the interest of performance, or it's easier/simple, and neglecting the protection and/or defense of a system. Example would be not keeping your system up to date with all the latest security updates. I'm curious to know how common these Meltdown attacks are in the industry and what best practices cybersecurity companies are instituting to mitigate these threats.