Labs

Neo Pi

Neo.Pi1@Marist.edu

October 1, 2024

1 Lab One

1.1 What are the advantages and disadvantages of using the same system call interface for manipulating both files and devices?

The advantage of using the same system call interface for both files and devices is so that each device can be accessed as thought it was a file within the system. Since kernel already deals with devices through a file interface, adding code to support a different kind of file interface is possible. The disadvantage of using the same interface is that the functionality of some devices might not be able to be accessed. Loss of functionality or performance could be a result.

1.2 Would it be possible for the user to develop a new command interpreter using the system call interface provide by the operating system? How?

Yes, a user can develop a new command interpreter by using the system call interface provided by the operating system. You can develop it by specifically using system calls to manage processes and file-related system calls for i/o operations.

2 Lab Two

2.1 How is your console like the ancient TTY subsystem in Unix as described in https://www.linusaskesson/programming/tty/?

Originally, the TTY system was designed for text-based input and output. Similarly, the current console we are working on operates the same way. It takes character-based commands, and handles commands by processing input as text characters. The input is buffered line-by-line, just as it is in our own OS. Anohter important factor, is that the TTV subsystem was the primary interface for the system, and similarly our OS provides a similar interactive interface. Lastly, the interaction is all in real time, when issuing commands and receiving feedback.

3 Lab Three

3.1 Explain the difference between internal and external fragmentation.

Internal fragmentation happens when memory is allocated in block larger than what is required. The memory that is not used inside the allocated block becomes useless if the process does not require the whole block. External fragmentation happens when there is enough free memory, but the free memory is scattered in different blocks. Even though there is enough free memory, there are no single blocks large enough for the request. Overall, internal fragmentation contains wasted space within memory blocks, while external fragmentation has wasted space between memory blocks.

3.2 Given five memory partitions of 100KB, 500KB, 200KB, 300KB, and 600LB (in that order), how would optimal, first-fit best-fit, and worst-fit algorithms place processes of 212KB, 417KB, 112KB, and 426KB (in that order)?

For optimal placement, it would work like this: 212KB : 300KB 417KB : 500KB 112KB : 200KB 426KB : 600KB

First-Fit: 212KB : 500KB 417KB : 600KB 112KB : 200KB 426KB :

Best-Fit: 212KB : 300KB, 417KB : 500KB, 112KB : 200KB, 426KB : 600KB

Worst-Fit: 212KB : 600KB, 417KB : 500KB, 112KB : 300KB, 426KB :

4 Lab 4

4.1 What is the relationship between a guest operating system and a host operating system like VMware? What factors need to be considered in choosing the host operating system?

The relationship involves how the host manages the hardware resources and virtualizes them for the guest OS. The guest OS runs as if it were operating directly on a physical machine, but it instead operates on virtual hardware. The guest OS relies on the host OS for certain things such as CPU and RAM. Some factors to be considered are: The compatibility with the virtualization software, Performance, Resource Allocation, Security.

5 Lab 5

5.1 The processes are assumed to have arrived in the order P1, P2, P3, P4, P5, all at time 0.

Process	Burst Time	Priority
P1	10	3
P2	1	1
P3	2	3
P4	1	4
P5	5	2

Table 5.1: Consider the set of processes, with the length of the CPU burst given in milliseconds.

a. Draw four Gantt charts that illustrate the execution of these processes using the following scheduling algorithms. FCFS, SJF, non preemptive priority (a smaller priority number implies a higher priority), and RR (quantum = 1).

P1	P2	P3	P4	P5	
0	10	11	13	14	19

Table 5.2: FCFS

P2	P4	P3	P5	P1	
0	1	2	4	9	19

Table 5.3: SJF

P2	P5	P1	P3	P4	
0	1	6	16	18	19

Table 5.4: Non-preemptive Priority

P1	P2	P3	P4	P5	P1	P3	P5	P1
0	1	2	3	4	5	6	7	8

Table 5.5: RR Part1

P5	P1	P5	P1	P5	P1	
9	10	11	12	13	14	19

Table 5.6: RR Part2

b. What is the turnaround time of each process for each of the scheduling algorithms in part a?

Process	FCFS	SJF	Priority	RR
P1	10	19	16	19
P2	11	1	1	1
P3	13	4	18	12
P4	14	2	19	3
P5	19	9	6	13

Table 5.7: Turnaround Time

c. What is the waiting time of each process for each of these scheduling algorithms?

Process	FCFS	SJF	Priority	RR
P1	0	9	6	9
P2	10	0	0	0
P3	11	2	16	10
P4	13	1	18	2
P5	14	4	1	8

Table 5.8: Waiting Time

d. Which of the algorithms results in the minimum average waiting time (over all processes)?

FCFS Average Waiting Time: (0 + 10 + 11 + 13 + 14) / 5 = 9.6 ms.
SJF Average Waiting Time: (9 + 0 + 2 + 1 + 4) / 5 = 3.2 ms.
Priority Average Waiting Time: (6 + 0 + 16 + 18 + 1) / 5 = 8.2 ms.
RR Average Waiting Time: (9 + 0 + 10 + 2 + 8) / 5 = 5.8 ms.

Shortest Job First has the minimum average waiting time.

6 Appendix