REPORT

Ayush Agrawal
Roll No: 2020101025

Implementation of Specification 1: Syscall Tracing

The trace system call has the following prototype:

int trace(uint64 mask)

struct proc has a new variable called mask. The trace system call stores this number in this variable and the syscall function in syscall.c checks if the binary representation of mask contains the ith bit set where i is the system call index that the process calls. If it is set, then it prints the pid of process, name of syscall, its arguments and return value. The user program strace is used to trace the syscalls called by the process. For eg, strace 32 grep hello README, traces all read calls(index=5) called by the process grep.

Implementation of Specification 2:

- FCFS:

According to FCFS scheduling algorithm, the process which was created first should be given the CPU first. Therefore, a new variable ctime was added in struct proc to store the creation time of the process. The scheduler selects the process with the minimum creation time and gives it the CPU. Timer interrupts were disabled in trap.c as FCFS is non-preemptive in nature. In allocproc, where processes are created, the ctime variable is set to ticks to store the creation time.

-PBS:

According to PBS scheduling algorithm, the process which has the highest priority is given to CPU first. Three variables staticpriority, niceness and dynamicpriority were added to struct proc in order to store priority. Some other variables like lastruntime, sleepstarttime, sleeptime were also added in order to calculate these priorities. Static Priority is set to 60 in allocproc. Niceness and dynamic priority is calculate in the scheduler function by calling calculatedp function. Each time the scheduler selects the process with lowest dynamic priority and gives it the CPU. If dynamic priorities are same, it schedules the process which has run lesser number of times, if that is also same then it schedules the process which was created first. A new system call set_priority was also added to change the static priority of the process and reschedule it if its dynamic priority decreased as a result of change. The user program setpriority.c is used to change the static priority of a process by user.

Implementation of Specification 3: Procdump

For PBS, the procdump shows process's pid, its dynamic priority, it's state, it's total running time (stored in variable runtime in struct proc), it's total waiting time(stored in variable total sleep time in struct proc), and the number of times it was scheduled.

Q: Explain in the README how could this be exploited by a process.

In MLFQ scheduling algorithm, a process is run till it uses it’s time slice and is then put into the next lower priority queue. However, if the process relinquishes the CPU before it’s time slice is over, it is put at the back of the same queue. This fact can be exploited by a process to get maximum CPU time. The process can go to sleep before it’s time slice gets over and then placed at the same queue. In this manner, it will never be transferred to a lower priority queue and hence can be completed first among all the processes.

Performace of ROUND_ROBIN, FCFS, PBS using schedulertest.c:

Scheduling Algorithm	Total Run Time(Avg)	Total Wait Time(Avg)
ROUND_ROBIN	18	115
FCFS	31	55
PBS	10	118

We observe that FCFS has the lowest average waiting time among all the algorithms while PBS has the highest waiting time among all the algorithms.
FCFS is favourable where processes have low CPU bursts as all the processes will then get CPU time and it is straightforward to implement.
ROUND_ROBIN is favourable in time-sharing systems where each process gets an equal share of CPU time.
PBS is preffered in situtations where priority of process matters(for eg. Money paid for using the system) and higher priority processes have to be scheduled first.