

# Operating Systems Laboratory

## Assignment-3

### REPORT

#### **Part-1:**

By modifying the `schedule_process()` function in `minix/servers/sched/schedule.c` file, we get the user-level process brought in by scheduler. This can be achieved by printing endpoint of the process being handled with highest priority.

```
/*=====*
 *          schedule_process          *
 *=====*/
static int schedule_process(struct schedproc * rmp, unsigned flags)
{
    int err;
    int new_prio, new_quantum, new_cpu;

    pick_cpu(rmp);

    if(rmp->priority >= 7){
        printf("\nMinix 3: PID %d swapped in\n", _ENDPOINT_P(rmp->endpoint));
    }
}
```

## Part-2: UnixBench benchmark analysis

### Observations:

The following tasks are carried in the Unixbench benchmark suite:

### 1. Arithoh-CPU bound intensive task

To see how scheduler handles two CPU intensive tasks run simultaneously.

Run two instances of arithoh.sh which replicates CPU intensive task and doesn't wait for any I/O operation.

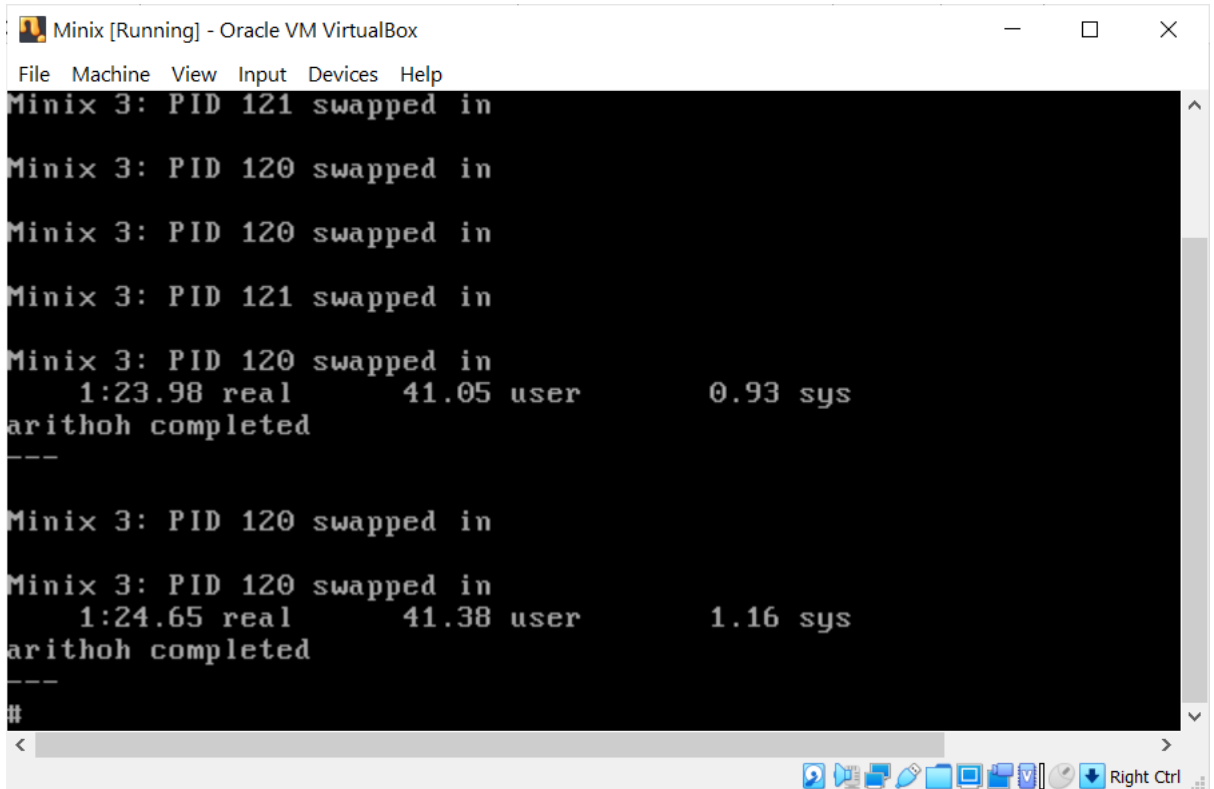
A screenshot of a Windows desktop with a virtual machine window titled "Minix [Running] - Oracle VM VirtualBox". The window has a menu bar with "File", "Machine", "View", "Input", "Devices", and "Help". The main area shows a black terminal window with white text displaying a repeating pattern: "Minix 3: PID 206 swapped in" followed by "Minix 3: PID 205 swapped in". This sequence repeats eight times. At the bottom of the screen, the Windows taskbar is visible, showing icons for various applications and a system tray with a clock and a "Right Ctrl" button.

Minix [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Minix 3: PID 206 swapped in  
Minix 3: PID 205 swapped in  
Minix 3: PID 206 swapped in  
Minix 3: PID 206 swapped in  
Minix 3: PID 205 swapped in  
Minix 3: PID 206 swapped in  
Minix 3: PID 205 swapped in  
Minix 3: PID 206 swapped in  
Minix 3: PID 205 swapped in  
Minix 3: PID 206 swapped in  
Minix 3: PID 205 swapped in  
Minix 3: PID 206 swapped in

Windows taskbar: Right Ctrl



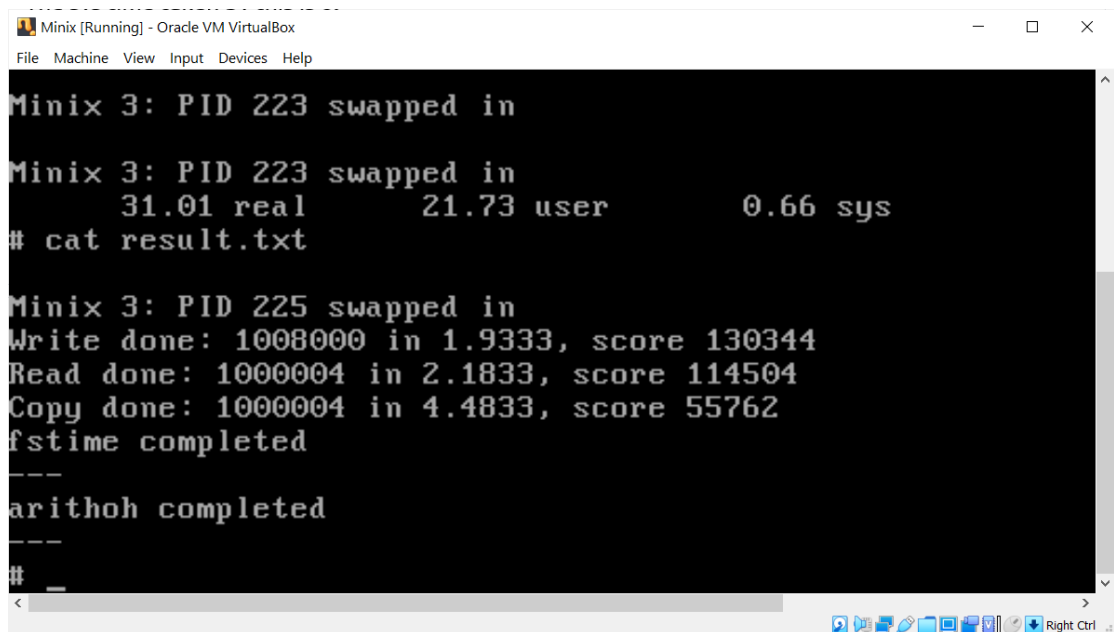
```
Minix [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Minix 3: PID 121 swapped in
Minix 3: PID 120 swapped in
Minix 3: PID 120 swapped in
Minix 3: PID 121 swapped in
Minix 3: PID 120 swapped in
1:23.98 real    41.05 user    0.93 sys
arithoh completed
---
Minix 3: PID 120 swapped in
Minix 3: PID 120 swapped in
1:24.65 real    41.38 user    1.16 sys
arithoh completed
---
#
```

Above two images show the PID swapping between two numbers indicating PM switching between two processes as both have equal priorities. Finally, first process ends quite earlier than the second one this is because parent process is given higher priority than child process while scheduling. Later, to be allotted same priority. That explains the time difference between the two.

## 2. Fstime - I/O bound task

### Fstime vs arithoh

Run arithoh and fstimecall processes simultaneously. As I/O processes are given higher priority, fstime seems to complete first followed by arithoh



```
Minix 3: PID 223 swapped in
Minix 3: PID 223 swapped in
      31.01 real      21.73 user      0.66 sys
# cat result.txt

Minix 3: PID 225 swapped in
Write done: 1008000 in 1.9333, score 130344
Read done: 1000004 in 2.1833, score 114504
Copy done: 1000004 in 4.4833, score 55762
fstime completed
---
arithoh completed
---
#
```

### 3. Pipe- CPU bound task

System time is very high compared to user time. This is because of the Inter Process Communication protocols which are system based protocols.

```

Minix [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
.project          Run          pgms          src          workload_mix
Makefile          USAGE        res.txt       testdir
# cd workload_mix/
# ls

Minix 3: PID 128 swapped in
arithoh.sh        pipe.sh        run.sh        syscall.sh
fstime.sh         result.txt    spawn.sh     workload_mix.sh
# ./pipe.sh

Minix 3: PID 129 swapped in
Minix 3: PID 130 swapped in
Minix 3: PID 131 swapped in
Minix 3: PID 131 swapped in
Minix 3: PID 131 swapped in
Minix 3: PID 131 swapped in
Minix 3: PID 131 swapped in
9.81 real      0.75 user      9.05 sys
pipe completed
---
#
  
```

#### Pipe vs arithoh:

Pipe's system time is more than user time and completes ahead of arithoh when run simultaneously.

```

Minix 3: PID 245 swapped in
Minix 3: PID 245 swapped in
Minix 3: PID 244 swapped in
Minix 3: PID 244 swapped in
Minix 3: PID 245 swapped in
Minix 3: PID 245 swapped in
Minix 3: PID 244 swapped in
Minix 3: PID 244 swapped in
Minix 3: PID 245 swapped in
21.58 real      1.80 user      18.30 sys
pipe completed
---
Minix 3: PID 244 swapped in
Minix 3: PID 244 swapped in
  
```

Spawn does the work of creating n number of process iteratively. The system time of the task is more than that of the user time. From this it can be inferred that initiated process work is less than that of the time it takes to initiate the process.

Here, it is because of that, the user time of the arithoh is more than that of spawn because arithoh is CPU intensive task unlike processes created by spawn.

```
Minix 3: PID 43 swapped in
Minix 3: PID 44 swapped in
Minix 3: PID 45 swapped in
Minix 3: PID 46 swapped in
Minix 3: PID 47 swapped in
Minix 3: PID 48 swapped in
28.86 real      0.71 user      23.33 sys
spawn completed
---
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
```

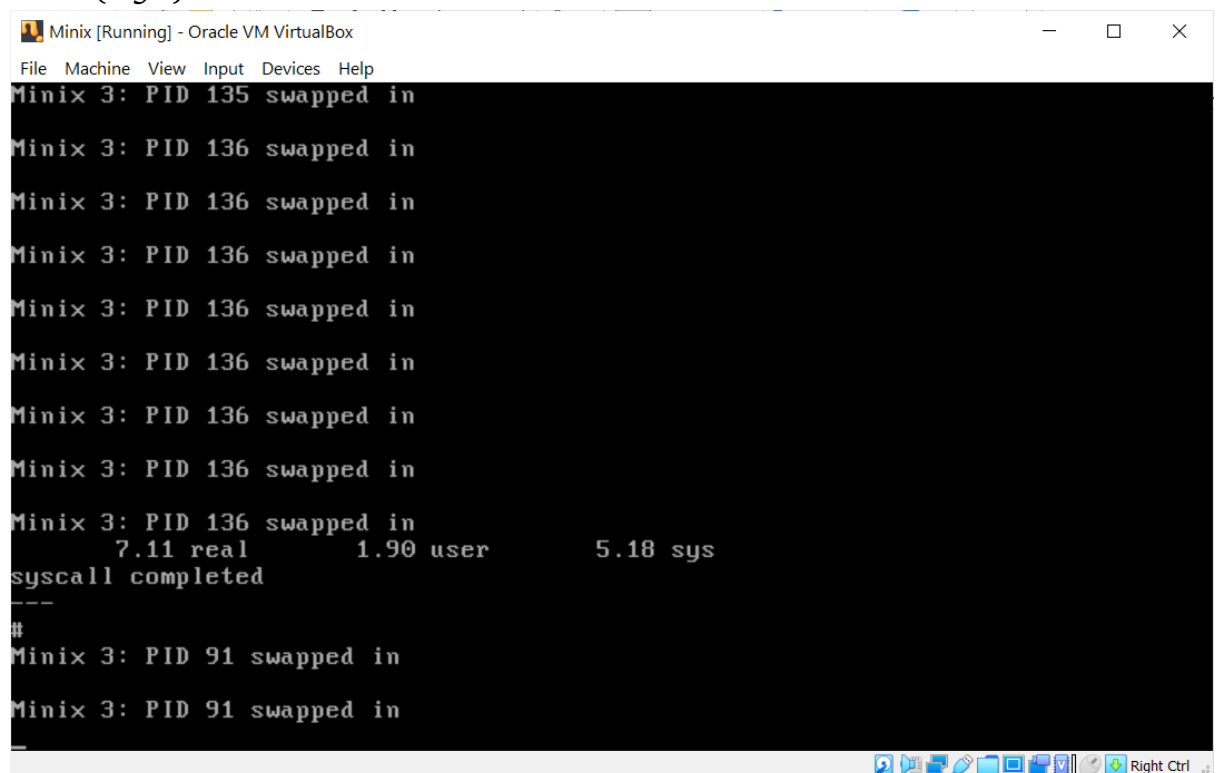
```

Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
Minix 3: PID 7 swapped in
50.26 real    22.13 user    0.86 sys
arithoh completed
---
#

```

## 5. Syscall- CPU bound benchmark

Most of the time spent is in system mode(5.18) and user time is very small(1.90)



The screenshot shows a terminal window titled "Minix [Running] - Oracle VM VirtualBox". The terminal output displays the results of a syscall benchmark. It shows multiple instances of "Minix 3: PID 135 swapped in" and "Minix 3: PID 136 swapped in". The benchmark results are: 7.11 real, 1.90 user, and 5.18 sys. The text "syscall completed" is followed by a separator line "----" and a prompt "#". Below the prompt, there are two more lines of "Minix 3: PID 91 swapped in". The window has a standard menu bar (File, Machine, View, Input, Devices, Help) and a taskbar at the bottom with various icons and a "Right Ctrl" key indicator.

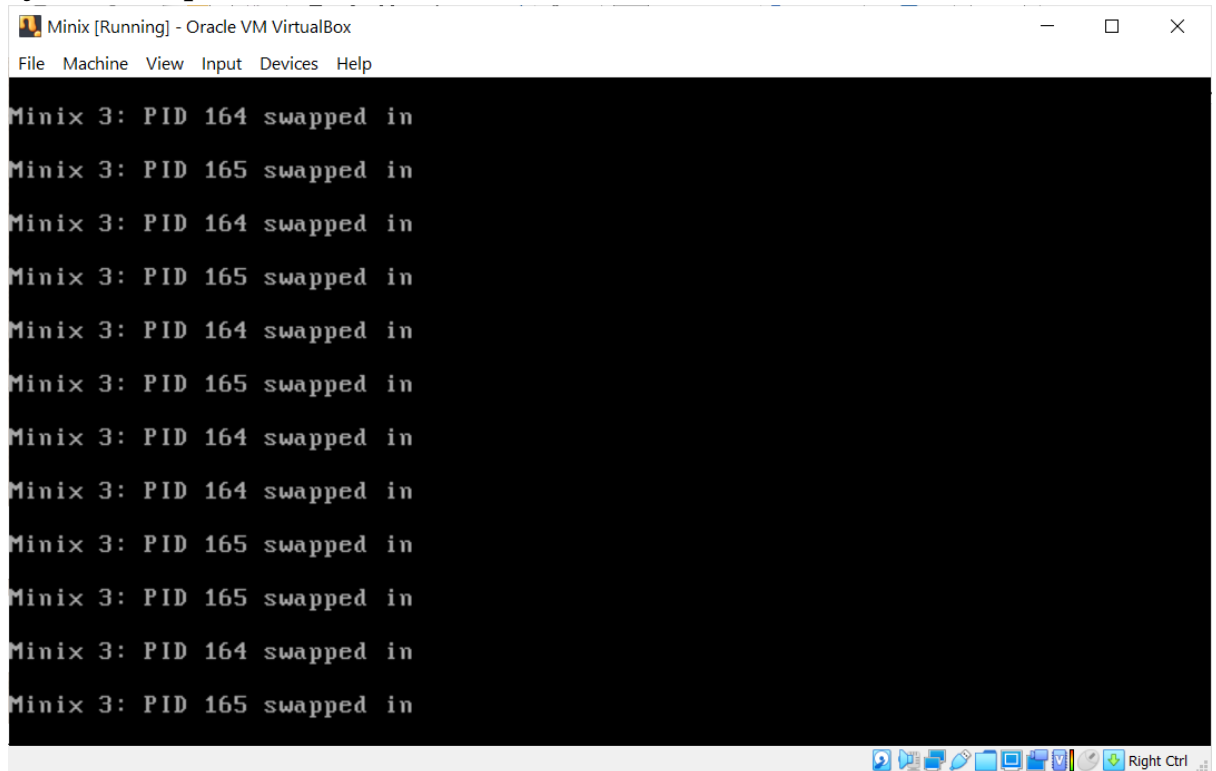
```

Minix 3: PID 135 swapped in
Minix 3: PID 136 swapped in
Minix 3: PID 136 swapped in
Minix 3: PID 136 swapped in
Minix 3: PID 136 swapped in
Minix 3: PID 136 swapped in
Minix 3: PID 136 swapped in
Minix 3: PID 136 swapped in
Minix 3: PID 136 swapped in
Minix 3: PID 136 swapped in
Minix 3: PID 136 swapped in
Minix 3: PID 136 swapped in
7.11 real    1.90 user    5.18 sys
syscall completed
----
#
Minix 3: PID 91 swapped in
Minix 3: PID 91 swapped in

```

## Syscall vs arithoh

When syscall and arithoh are run simultaneously, the scheduler handles it the same as fstime vs arithoh and uses round robin protocol. syscall completes task earlier than arithoh.



The screenshot shows a terminal window titled "Minix [Running] - Oracle VM VirtualBox". The terminal output displays a sequence of messages indicating that two processes, PID 164 and PID 165, are being swapped in and out of memory. The messages alternate between the two PIDs, demonstrating a round-robin scheduling pattern. The messages are as follows:

```
Minix 3: PID 164 swapped in
Minix 3: PID 165 swapped in
Minix 3: PID 164 swapped in
Minix 3: PID 165 swapped in
Minix 3: PID 164 swapped in
Minix 3: PID 165 swapped in
Minix 3: PID 164 swapped in
Minix 3: PID 164 swapped in
Minix 3: PID 165 swapped in
Minix 3: PID 165 swapped in
Minix 3: PID 164 swapped in
Minix 3: PID 165 swapped in
```

The window includes a menu bar with "File", "Machine", "View", "Input", "Devices", and "Help". The bottom of the window shows a taskbar with various icons and a "Right Ctrl" button.