

---

## CS 314 – Operating Systems Lab

### Lab-3 Report

**Student 1:** Sahaja Nandyala  
**Roll-No:** 200010032

**Student 2:** Kavali Sri Vyshnavi Devi  
**Roll-No:** 200010023

---

## 1 Part I

The schedule.c file in the location `usr/src/minix/servers/sched` has been made following changes for printing "PID pid swapped in", whenever a user-level process is brought in by the scheduler.

The following if statement has been added to "schedule\_process" function in schedule.c file. Here the USER\_Q signifies  $((\text{MIN\_USER\_Q} - \text{MAX\_USER\_Q}) / 2 + \text{MAX\_USER\_Q})$  which is specified in **config.h** file. Datatype of `_ENDPOINT_P` is defined in **proc.c** file.

```
//-----  
if (rmp->priority >= USER_Q){  
    printf("Minix(200010032): PID %d swapped in\n", _ENDPOINT_P(rmp->endpoint));  
}  
//-----
```

Figure 1: Part1 code

To build the changes on MINIX3 OS, a `run.sh` is also made which copies `schedule.c` from the present working directory to `usr/src/minix/servers/sched` and building the OS again.

```
echo "copying files"  
cp schedule.c /usr/src/minix/servers/sched/schedule.c  
echo "going to src directory and building the updated code"  
cd /usr/src  
make build MKUPDATE=yes >log.txt 2>log.txt  
echo "build completed successfully"  
exit 0
```

The screenshot of build success for Part1.

```
install -N /usr/src/etc -c -p -r ../minix/servers/ds/ds /boot/minix/.temp/mod01_
ds
install -N /usr/src/etc -c -p -r ../minix/servers/rs/rs /boot/minix/.temp/mod02_
rs
install -N /usr/src/etc -c -p -r ../minix/servers/pm/pm /boot/minix/.temp/mod03_
pm
install -N /usr/src/etc -c -p -r ../minix/servers/sched/sched /boot/minix/.temp/
mod04_sched
install -N /usr/src/etc -c -p -r ../minix/servers/vfs/vfs /boot/minix/.temp/mod0
5_vfs
install -N /usr/src/etc -c -p -r ../minix/drivers/storage/memory/memory /boot/mi
nix/.temp/mod06_memory
install -N /usr/src/etc -c -p -r ../minix/drivers/tty/tty/tty /boot/minix/.temp/
mod07_tty
install -N /usr/src/etc -c -p -r ../minix/fs/mfs/mfs /boot/minix/.temp/mod08_mfs
install -N /usr/src/etc -c -p -r ../minix/servers/vm/vm /boot/minix/.temp/mod09_
vm
install -N /usr/src/etc -c -p -r ../minix/fs/pfs/pfs /boot/minix/.temp/mod10_pfs
install -N /usr/src/etc -c -p -r ../sbin/init/init /boot/minix/.temp/mod11_init
rm /dev/c0d0p0s0:/boot/minix/3.3.0r2
Done.
Build started at: Mon Jan 16 21:31:04 IST 2023
Build finished at: Mon Jan 16 21:33:34 IST 2023
Minix: PID 293 exited
# _
```

Figure 2: Build Successful

Later reboot the OS on successful build to observe the changes. Executing printenv command after rebuilding the OS.

```
# printenv
Minix: PID 221 created
MINIX(200010023): PID 196 swapped in
PWD=/root
LD_LIBRARY_PATH=/usr/local/lib:/usr/pkg/lib:/usr/lib:/lib
HOME=/root
PATH=/usr/local/sbin:/usr/pkg/sbin:/usr/sbin:/sbin:/usr/local/bin:/usr/pkg/bin:/
usr/bin:/bin:/usr/games
TERM=minix
USER=root
PAGER=less
TZ=Asia/Calcutta
EDITOR=vi
LOGNAME=root
SHELL=/bin/sh
Minix: PID 221 exited
```

Figure 3: Executing printenv command

Executing ls command after rebuilding the OS.

```
# ls
Minix: PID 357 created
MINIX(200010023): PID 107 swapped in
.bash_history .exerc .gitconfig .profile
Minix: PID 357 exited
#
```

Figure 4: Executing ls command

## 2 Part II

The source code for UnixBench benchmark suite was copied into the home folder in the guest Minix3 VM and built using gmake command. Later six work mix files are written which tries various combinations of UnixBench programs to study.

```
Minix: PID 290 exited
# pwd
/home/byte-unixbench-mod/byte-unixbench-mod/UnixBench
# gmake
```

Figure 5: gmake command

```
Minix: PID 320 created
MINIX(200010023): PID 69 swapped in
Minix: PID 320 exited
Minix: PID 321 created
MINIX(200010023): PID 70 swapped in
Minix: PID 321 exited
Minix: PID 319 exited
clang -o pgms/whetstone-double -Wall -pedantic -O0 -ffast-math -I ./src -DTIME -
DDP -DGTODay -DUNIXBENCH src/whets.c -lm
Minix: PID 322 created
MINIX(200010023): PID 71 swapped in
Minix: PID 323 created
MINIX(200010023): PID 72 swapped in
Minix: PID 323 exited
Minix: PID 324 created
MINIX(200010023): PID 73 swapped in
Minix: PID 324 exited
Minix: PID 322 exited
gmake[1]: Leaving directory '/home/byte-unixbench-mod/UnixBench'
Minix: PID 266 exited
Minix: PID 254 exited
```

Figure 6: Executing gmake command

## 2.1 Study On Programs In Src Folder

- **arith.c(aritoh.sh)** long arithmetic operations more CPU bound operations.
- **fstime.c(fstime.sh)** IO performance is measured using this. file copy, 1024 byte buffer size, 500 max blocks.
- **pipe.c(pipe.sh)** The pipe benchmark measures the number of times a process can write 512 bytes to a pipe and read them back per second. The pipe switching benchmark measures the number of times two processes can exchange an increasing integer through a pipe. Here more context switches occur.
- **spawn.c(spawn.sh)** It does fork and write to files both CPU and I/O bound.
- **syscall.c(syscall.sh)** Sit in a loop calling getpid where constant switching between user space and kernel also cause some overhead.

## 2.2 Study of Programs by workload mix

Using the UnixBench Benchmark Suite, some workload mix shell scripts were created in order to study the behavior of the scheduler by observing the sequence in which processes are swapped in.

### 2.2.1 workload\_mix1.sh

In this shell script we have used **arithoh.sh** and **fstime.sh** as following code

```
#!/bin/sh
./arithoh.sh &
./fstime.sh &
wait
```

Figure 7: workload\_mix1.sh code

Here, PID of aritoh is 201 and PID of fstime is 202. We have noticed that the aritoh.sh is doing big arithmetic operations which require CPU time long, where as fstime.sh mostly perform I/O operations for which the requirement of CPU is less but goes to blocked state frequently. So, when fstime.sh is waiting for response from I/O device aritoh.sh is being executed which that is why we can observe that it is getting swapped more frequently.

Also airtoh.sh took more time to execute and so it is exited later as compared to fstime.sh as their dependency on CPU differs.

```

MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 202 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 202 swapped in

```

Figure 8: workmix\_load1: processes getting swapped

```

MINIX(200010023): PID 202 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 202 swapped in
MINIX(200010023): PID 202 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 202 swapped in
Copy done: 1000004 in 14.0833, score 17751
COUNT:17751:0:KBps
TIME:14.1
Minix: PID 442 exited
      39.41 real      3.51 user      24.68 sys
Minix: PID 440 exited
fstime completed
---
Minix: PID 438 exited
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in

```

Figure 9: workmix\_load1: exit of fstime process

```

MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
MINIX(200010023): PID 201 swapped in
Minix: PID 441 exited
      46.91 real      18.70 user      0.00 sys
Minix: PID 439 exited
arithoh completed
---
Minix: PID 437 exited
Minix: PID 436 exited
#

```

Figure 10: workmix\_load1: exit of arithoh process

### 2.2.2 workload\_mix2

In this shell script we have used **arithoh.sh** and **syscall.sh** as following

```
#!/bin/sh
./arithoh.sh &
./syscall.sh &
wait
```

Figure 11: workload\_mix2.sh code

Here, PID of aritoh is 213 and PID of syscall is 214. we can observe that aritoh.sh is meant for making big arithmetic operations where as syscall.sh calls kernel functions that is getpid() which makes more context switches as compared to aritoh.sh. Here aritoh.sh and syscall.sh are being swapped in alternatively depending on their intensity of consuming the CPU. At last syscall.sh was completed as it is much CPU intense work as compared to aritoh.sh which completed execution earlier. So, from this we can interpret that the cost taken to perform context switches in syscall.sh is more than the CPU time taken by aritoh.sh which we can tell by the order of completion of 2 processes.

```
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 214 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 214 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
Minix: PID 453 exited
    59.88 real    18.80 user    0.00 sys
Minix: PID 451 exited
arithoh completed
---
Minix: PID 449 exited
```

Figure 12: workload\_mix2: exit of aritoh process

```

Minix: PID 449 exited
MINIX(200010023): PID 214 swapped in
MINIX(200010023): PID 214 swapped in
MINIX(200010023): PID 214 swapped in
MINIX(200010023): PID 214 swapped in
MINIX(200010023): PID 214 swapped in
MINIX(200010023): PID 214 swapped in
Minix: PID 454 exited
      1:11.75 real      15.71 user      37.23 sys
Minix: PID 452 exited
syscall completed
---
Minix: PID 450 exited
Minix: PID 448 exited
#

```

Figure 13: workload\_mix2: exit of syscall process

```

MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 214 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 214 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 213 swapped in
MINIX(200010023): PID 214 swapped in
MINIX(200010023): PID 213 swapped in

```

Figure 14: work2-swap

### 2.2.3 workload\_mix3

In this shell script we have used **pipe.sh** and **fstime.sh** as following //

```
#!/bin/sh
./pipe.sh &
./fstime.sh &
wait
```

Figure 15: workload\_mix3.sh code

Here, PID of pipe is 55 and PID of fstime is 57 .We can observe that pipe.sh makes more system calls and make more context switches, where as fstime.sh is an I/O bound. So, when fstime.sh is present in the blocked state that is waitting for an interrupt from I/O device pipe.sh can be scheduled that is why pipeline (55) is being swapped in more frequently.

Also as the fstime.sh is I/O bound it is completed first and later pipe.sh completed execution.

```
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 57 swapped in
MINIX(200010023): PID 57 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 57 swapped in
MINIX(200010023): PID 57 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 57 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 57 swapped in
Copy done: 1000004 in 27.8000, score 8992
COUNT:8992:0:KBps
TIME:27.8
Minix: PID 523 exited
1:05.90 real      3.06 user      25.46 sys
Minix: PID 522 exited
fstime completed
---
Minix: PID 518 exited
```

Figure 16: work3 fstime exit



```

Minix: PID 518 exited
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
Minix: PID 521 exited
      1:27.73 real      6.25 user      52.91 sys
Minix: PID 519 exited
pipe completed
---
Minix: PID 516 exited
Minix: PID 515 exited
#

```

Figure 17: work3 pipe exit

```

MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 57 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 57 swapped in
Read done: 1000004 in 14.6667, score 17045
COUNT:17045:0:KBps
TIME:14.7
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 57 swapped in
MINIX(200010023): PID 57 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 57 swapped in
MINIX(200010023): PID 57 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 57 swapped in
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 57 swapped in

```

Figure 18: work3 swap

## 2.2.4 workload\_mix4

In this shell script we have used **spawn.sh** as following

```
#!/bin/sh
./spawn.sh &
wait
```

Figure 19: workload\_mix4.sh code

Here, spawn do many fork system calls due to which many context switches happens between child process and parent process by which many process id's are created and so one can notice that many different pid's are showing while a process is getting swapped in. As many context switches happened in spawning it also require more CPU time in saving and restoring data such as registers, program counters, kernel stacks and virtual memory.

```
Minix: PID 15264 created
MINIX(200010023): PID 176 swapped in
Minix: PID 15264 exited
Minix: PID 15265 created
MINIX(200010023): PID 177 swapped in
Minix: PID 15265 exited
Minix: PID 15266 created
MINIX(200010023): PID 178 swapped in
Minix: PID 15266 exited
Minix: PID 15267 created
MINIX(200010023): PID 179 swapped in
Minix: PID 15267 exited
Minix: PID 15268 created
MINIX(200010023): PID 180 swapped in
Minix: PID 15268 exited
Minix: PID 15269 created
MINIX(200010023): PID 181 swapped in
Minix: PID 15269 exited
Minix: PID 15270 created
MINIX(200010023): PID 182 swapped in
Minix: PID 15270 exited
Minix: PID 15271 created
MINIX(200010023): PID 183 swapped in
```

Figure 20: workload\_mix4 swapping

```
Minix: PID 20549 created
MINIX(200010023): PID 161 swapped in
Minix: PID 20549 exited
Minix: PID 20550 created
MINIX(200010023): PID 162 swapped in
Minix: PID 20550 exited
Minix: PID 10548 exited
25.21 real 0.46 user 15.05 sys
Minix: PID 10546 exited
Minix: PID 10543 exited
ls
Minix: PID 20551 created
MINIX(200010023): PID 163 swapped in
arithoh.sh spawn.sh work1.txt work4.sh
fstime.sh syscall.sh work2.sh work4.txt
pipe.sh work1.sh work3.sh workload_mix.sh
Minix: PID 20551 exited
# cat work4.txt
Minix: PID 20552 created
MINIX(200010023): PID 164 swapped in
spawn completed
---
Minix: PID 20552 exited
#
```

Figure 21: workload\_mix4 spawn exit

### 2.2.5 workload\_mix5

In this shell script we have scheduled two **arithoh.sh** jobs are executed as following

```
#!/bin/sh
./arithoh.sh &
./arithoh.sh &
wait
```

Figure 22: workload\_mix5.sh code

Here, the PID of first aritoh is 35 and the PID of second aritoh is 36. As the jobs that are schedule do the same functionality there shouldn't be change in the intensity on which they use CPU. So, here both the processes are being swapped in alternatively. Also all the processes scheduled here are CPU intense as they perform long long arithmetic operations. All of them are scheduled alternatively till both the processes complete execution.

```
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 35 swapped in
Minix: PID 286 exited
      36.98 real      18.60 user      0.00 sys
Minix: PID 284 exited
arithoh completed
----
Minix: PID 282 exited
Minix: PID 281 exited
#
```

Figure 23: work5-1st-aritoh-exit

```
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 36 swapped in
Minix: PID 287 exited
      32.05 real      18.38 user      0.00 sys
Minix: PID 285 exited
arithoh completed
---
Minix: PID 283 exited
```

Figure 24: work5-2nd-aritoh-exit

```
MINIX(200010023): PID 36 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 36 swapped in
MINIX(200010023): PID 36 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 36 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 36 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 36 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 36 swapped in
MINIX(200010023): PID 35 swapped in
MINIX(200010023): PID 36 swapped in
```

Figure 25: work5-swap

## 2.2.6 workload\_mix6

In this shell script we have schedule three **fstime.sh** jobs are executed as following

```
#!/bin/sh
./fstime.sh &
./fstime.sh &
./fstime.sh &
wait
```

Figure 26: workload\_mix6.sh code

Here the PID of first, second, third fstime are 52, 54, 55 respectively. As the jobs are same there shouldn't be change in the intensity on which they use CPU. So, here all the three processes are being swapped in alternatively. Also all the processes scheduled here are I/O dependent, one can confirm this by observing the execution in which all the processes will be waiting to receive interrupt from I/O device. All of them are scheduled alternatively till all the processes complete execution.

```
MINIX(200010023): PID 24 swapped in
MINIX(200010023): PID 25 swapped in
Copy done: 1000004 in 43.3500, score 5767
COUNT:5767:0:KBps
TIME:43.4
Minix: PID 303 exited
    1:36.36 real      3.28 user      25.91 sys
Minix: PID 301 exited
fstime completed
---
Minix: PID 298 exited
Copy done: 1000004 in 43.6167, score 5731
COUNT:5731:0:KBps
TIME:43.6
Minix: PID 305 exited
    1:36.90 real      3.35 user      23.50 sys
Minix: PID 302 exited
fstime completed
---
Minix: PID 299 exited
```

Figure 27: work6-execution

```

Minix: PID 298 exited
Copy done: 1000004 in 43.6167, score 5731
COUNT:5731:0:KBps
TIME:43.6
Minix: PID 305 exited
1:36.90 real      3.35 user      23.50 sys
Minix: PID 302 exited
fstime completed
---
Minix: PID 299 exited
Copy done: 1000004 in 44.0667, score 5673
COUNT:5673:0:KBps
TIME:44.1
Minix: PID 306 exited
1:37.60 real      3.41 user      28.08 sys
Minix: PID 304 exited
fstime completed
---
Minix: PID 300 exited
Minix: PID 297 exited
#

```

Figure 28: work6-execution-completion

```

Write done: 1008000 in 21.8000, score 11559
COUNT:11559:0:KBps
TIME:21.8
MINIX(200010023): PID 55 swapped in
MINIX(200010023): PID 52 swapped in
MINIX(200010023): PID 54 swapped in
MINIX(200010023): PID 52 swapped in
MINIX(200010023): PID 54 swapped in
MINIX(200010023): PID 55 swapped in
Read done: 1000004 in 20.6333, score 12116
COUNT:12116:0:KBps
TIME:20.6
Read done: 1000004 in 20.6333, score 12116
COUNT:12116:0:KBps
TIME:20.6
Read done: 1000004 in 20.7000, score 12077
COUNT:12077:0:KBps
TIME:20.7

```

Figure 29: work6-swap