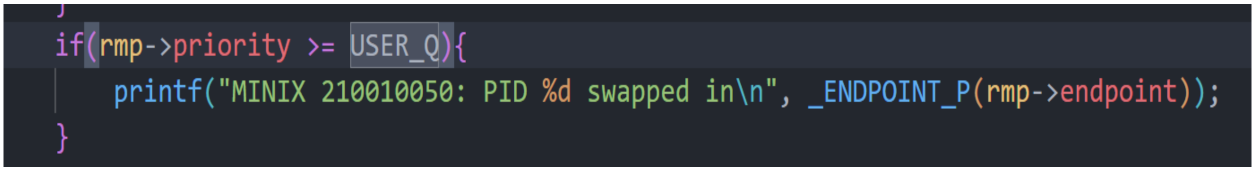
CS-314 OS LAB-3 REPORT

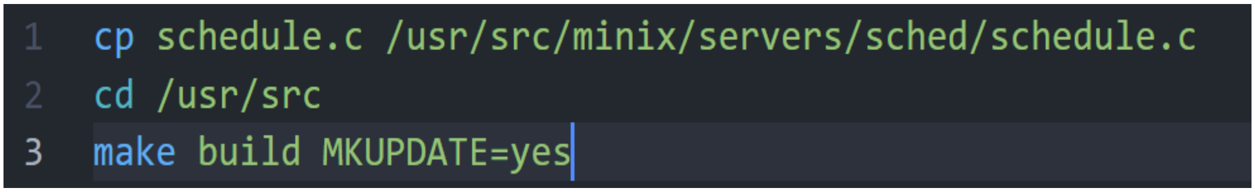
PART-1

- To print the string "MINIX <Rollnumber>: PID <pid> swapped in" whenever a user-level process is brought in by the scheduler, we need to modify the source code in the file 'schedule.c' at the location: /usr/src/minix/servers/sched
We add the following statement at line no. 327 inside the function 'schedule_process()' in a copy of file 'schedule.c'.
⇨ schedule.c:



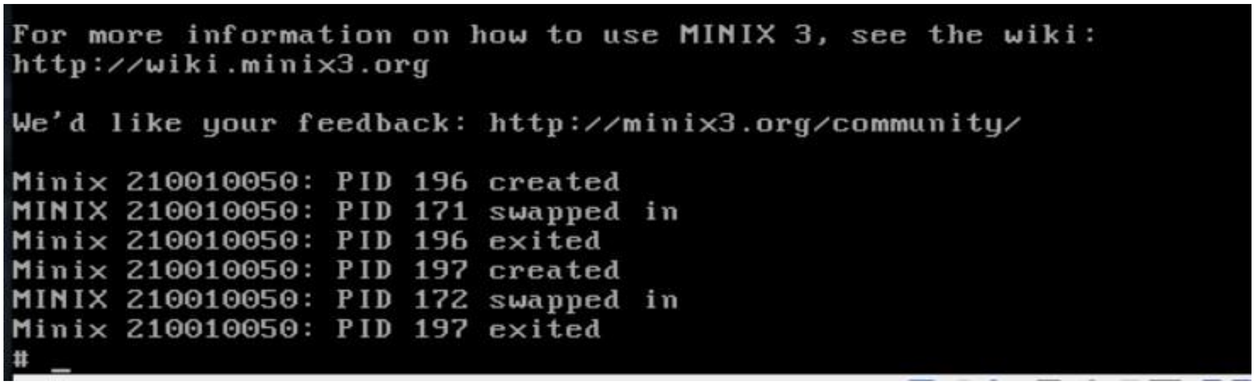
```
if(rmp->priority >= USER_Q){
    printf("MINIX 210010050: PID %d swapped in\n", _ENDPOINT_P(rmp->endpoint));
}
```

- For this to happen, we need to run the bash file shown below, which copies the modified code to the correct location and deploys the changes on our Minix OS.
⇨ run.sh:



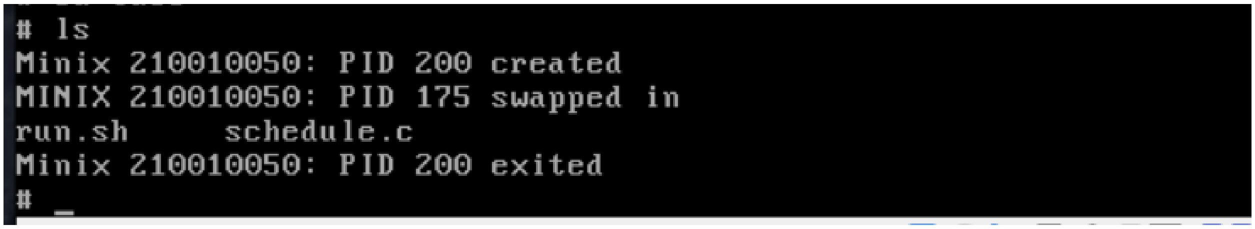
```
cp schedule.c /usr/src/minix/servers/sched/schedule.c
cd /usr/src
make build MKUPDATE=yes
```

- Now, after rebooting the system, we can see the following:

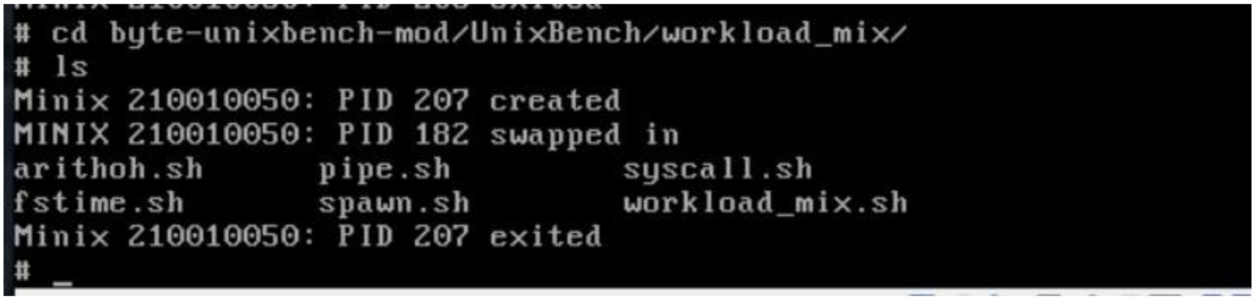


```
For more information on how to use MINIX 3, see the wiki:
http://wiki.minix3.org

We'd like your feedback: http://minix3.org/community/

Minix 210010050: PID 196 created
MINIX 210010050: PID 171 swapped in
Minix 210010050: PID 196 exited
Minix 210010050: PID 197 created
MINIX 210010050: PID 172 swapped in
Minix 210010050: PID 197 exited
# _
```

- On running ls command, you can see the following:



```
# ls
Minix 210010050: PID 200 created
MINIX 210010050: PID 175 swapped in
run.sh     schedule.c
Minix 210010050: PID 200 exited
# _
```

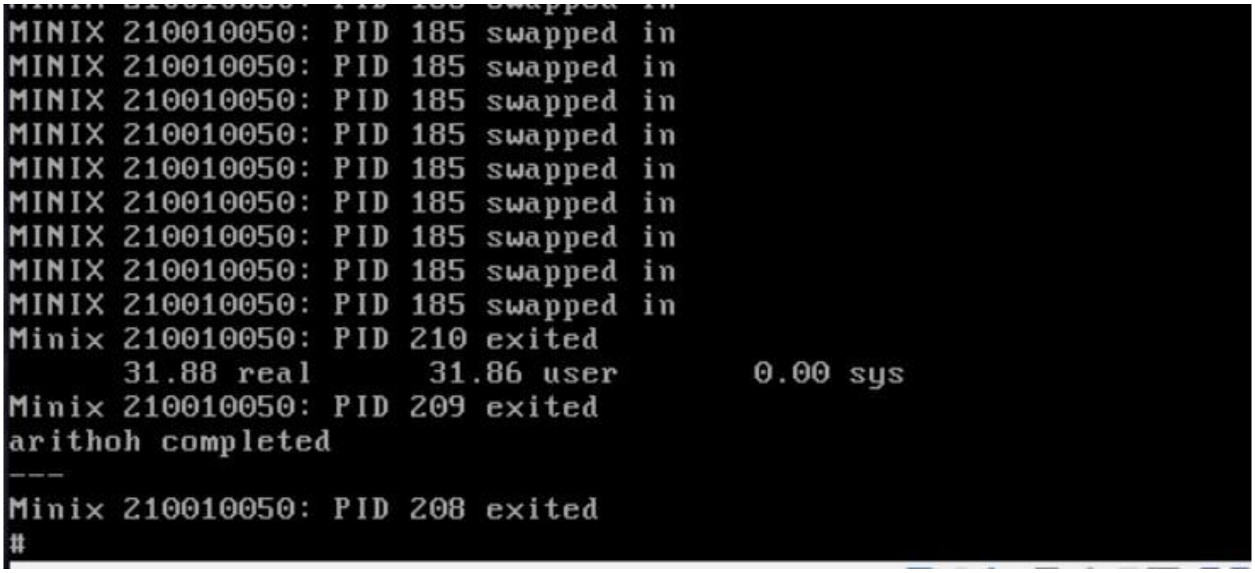


```
# cd byte-unixbench-mod/UnixBench/workload_mix/
# ls
Minix 210010050: PID 207 created
MINIX 210010050: PID 182 swapped in
arithoh.sh      pipe.sh         syscall.sh
fstime.sh       spawn.sh        workload_mix.sh
Minix 210010050: PID 207 exited
# _
```

PART-2

- We move the source code of the UnixBench benchmark to the home folder in our Minix OS. We then run gmake to build the benchmarks. We then go inside the directory ‘byte-unixbench-mod/UnixBench/workload_mix’ and run the following benchmarks:

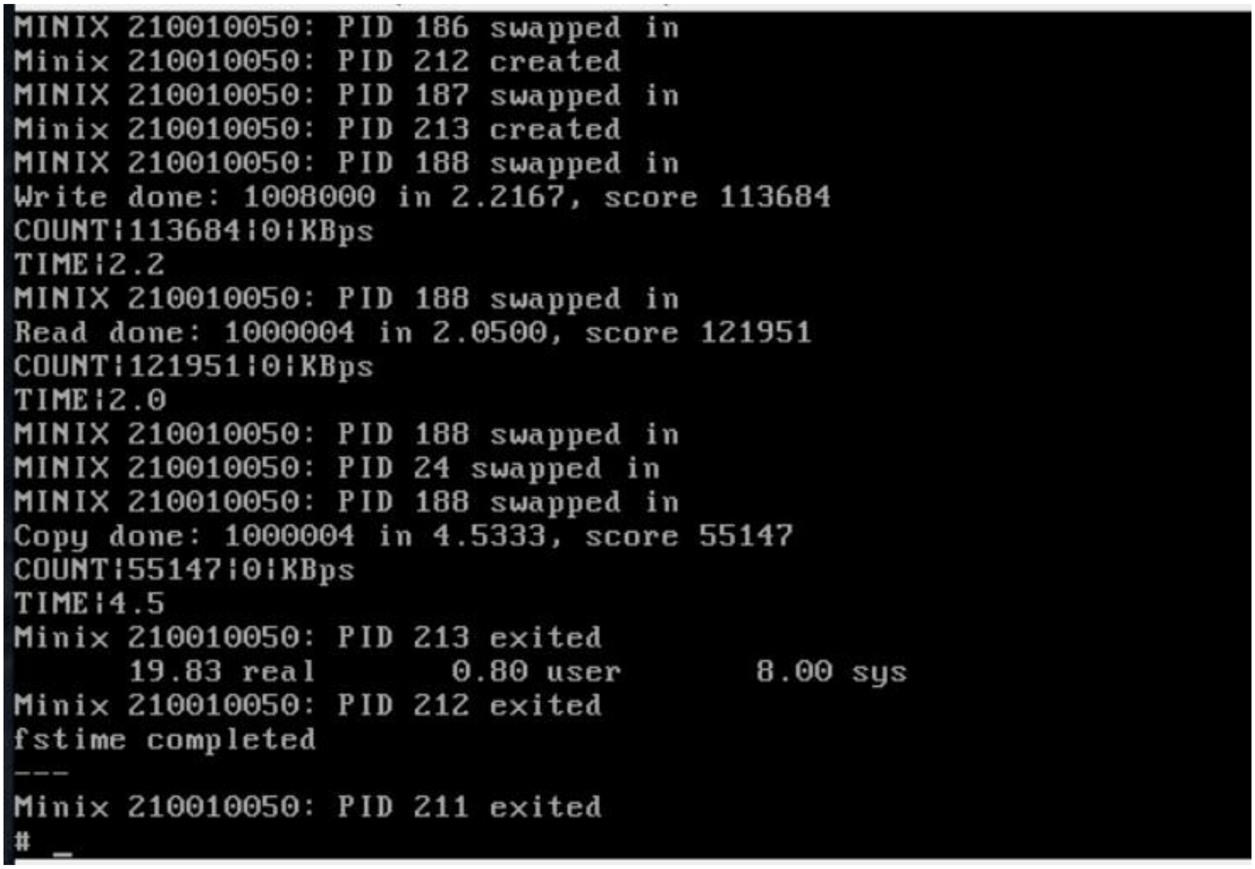
- arithoh.sh:



```
MINIX 210010050: PID 185 swapped in
MINIX 210010050: PID 185 swapped in
MINIX 210010050: PID 185 swapped in
MINIX 210010050: PID 185 swapped in
MINIX 210010050: PID 185 swapped in
MINIX 210010050: PID 185 swapped in
MINIX 210010050: PID 185 swapped in
MINIX 210010050: PID 185 swapped in
MINIX 210010050: PID 185 swapped in
Minix 210010050: PID 210 exited
      31.88 real        31.86 user        0.00 sys
Minix 210010050: PID 209 exited
arithoh completed
---
Minix 210010050: PID 208 exited
#
```

The arithoh.sh contains arithmetic operations so it is a **CPU Bound** Process since repeated arithmetic operations are rather computationally intensive, and don’t have an I/O component. It is observed that there is no system time used. The entire process is run in user mode as they are cpu intensive processes. Since it is a CPU bound process it has a lower priority hence it’s preemption frequency is high.

- fstime.sh:

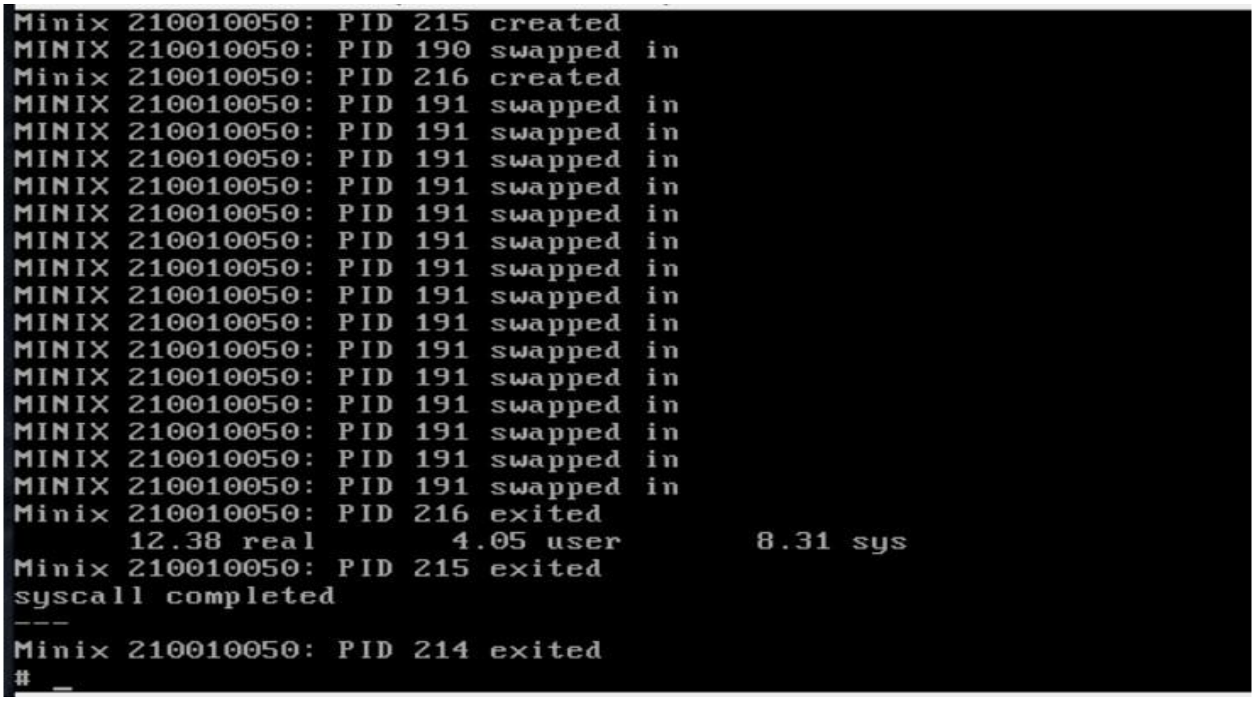


```
MINIX 210010050: PID 186 swapped in
Minix 210010050: PID 212 created
MINIX 210010050: PID 187 swapped in
Minix 210010050: PID 213 created
MINIX 210010050: PID 188 swapped in
Write done: 1008000 in 2.2167, score 113684
COUNT|113684|0|KBps
TIME|2.2
MINIX 210010050: PID 188 swapped in
Read done: 1000004 in 2.0500, score 121951
COUNT|121951|0|KBps
TIME|2.0
MINIX 210010050: PID 188 swapped in
MINIX 210010050: PID 24 swapped in
MINIX 210010050: PID 188 swapped in
Copy done: 1000004 in 4.5333, score 55147
COUNT|55147|0|KBps
TIME|4.5
Minix 210010050: PID 213 exited
      19.83 real       0.80 user       8.00 sys
Minix 210010050: PID 212 exited
fstime completed
---
Minix 210010050: PID 211 exited
# _
```

The fstime contains file operations so it is **I/O Bound** Process. Here, the real time is high as compared to user and sys time because the process consists of several calls to time, date, sleep etc.

- syscall.sh:

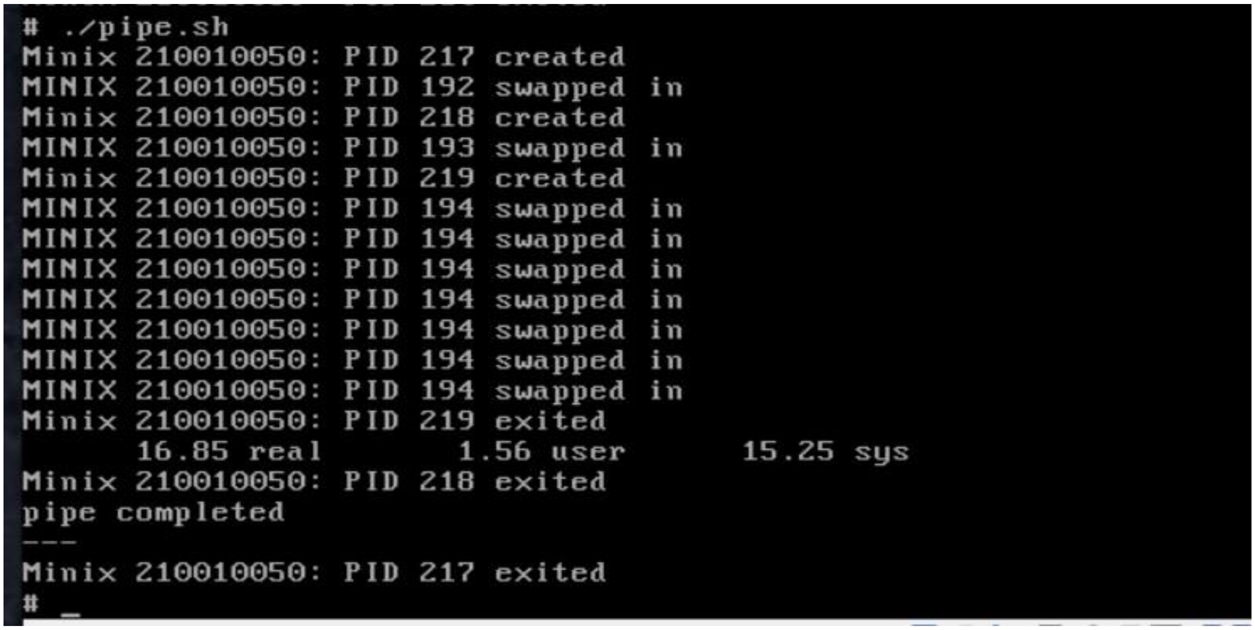
The syscall script seems to mainly consist of **CPU-bound** processes, which does system calls and deals with file descriptors. Hence, we can see that most of the process is run in kernel mode rather than user mode. Their preemption frequency is lower as switching between user and kernel mode has an overhead.



```
Minix 210010050: PID 215 created
MINIX 210010050: PID 190 swapped in
Minix 210010050: PID 216 created
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
MINIX 210010050: PID 191 swapped in
Minix 210010050: PID 216 exited
      12.38 real        4.05 user        8.31 sys
Minix 210010050: PID 215 exited
syscall completed
---
Minix 210010050: PID 214 exited
# _
```

- pipe.sh:

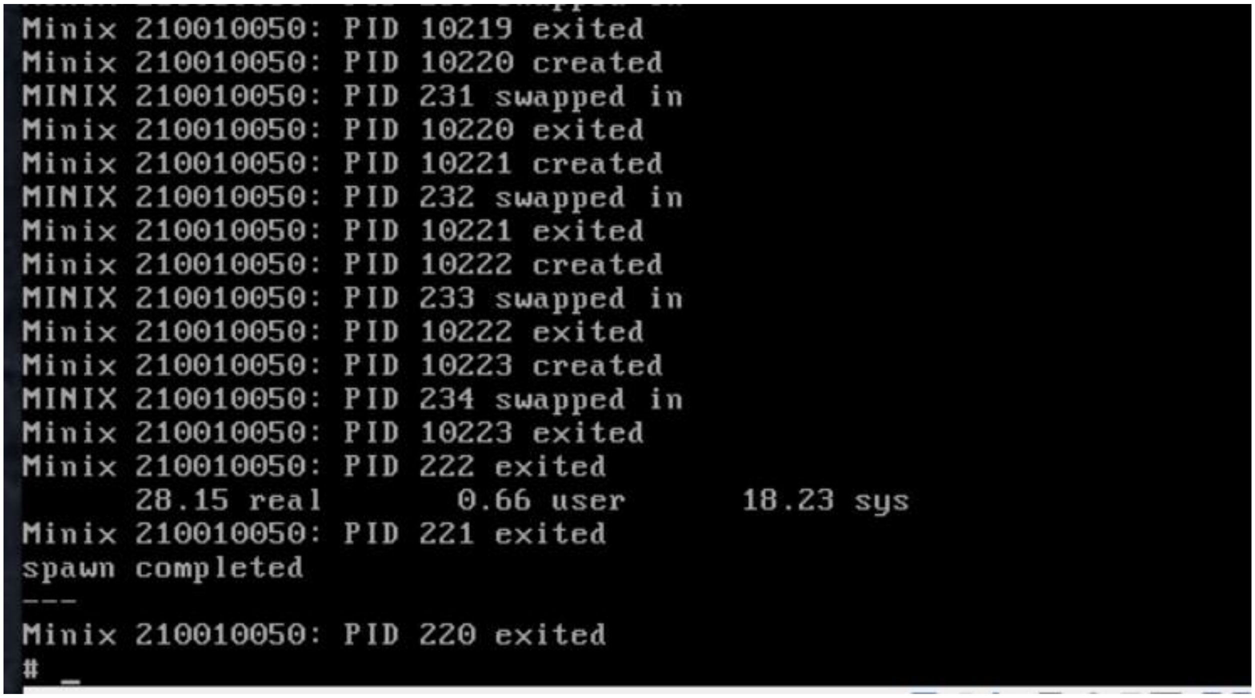
  The pipe script seems to be more **I/O-bound**; it involves repeated reads and writes to a pipe. It also has less CPU usage, due to its smaller bursts.



```
# ./pipe.sh
Minix 210010050: PID 217 created
MINIX 210010050: PID 192 swapped in
Minix 210010050: PID 218 created
MINIX 210010050: PID 193 swapped in
Minix 210010050: PID 219 created
MINIX 210010050: PID 194 swapped in
MINIX 210010050: PID 194 swapped in
MINIX 210010050: PID 194 swapped in
MINIX 210010050: PID 194 swapped in
MINIX 210010050: PID 194 swapped in
MINIX 210010050: PID 194 swapped in
MINIX 210010050: PID 194 swapped in
Minix 210010050: PID 219 exited
      16.85 real        1.56 user       15.25 sys
Minix 210010050: PID 218 exited
pipe completed
---
Minix 210010050: PID 217 exited
# _
```

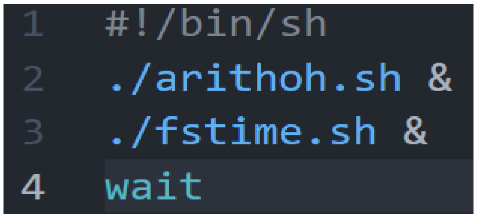
- spawn.sh:

  This benchmark consists of system calls and thus seems to be **CPU-bound**. Hence, we can see that most of the process is run in kernel mode rather than user mode. Their preemption frequency is lower as switching between user and kernel mode has an overhead.

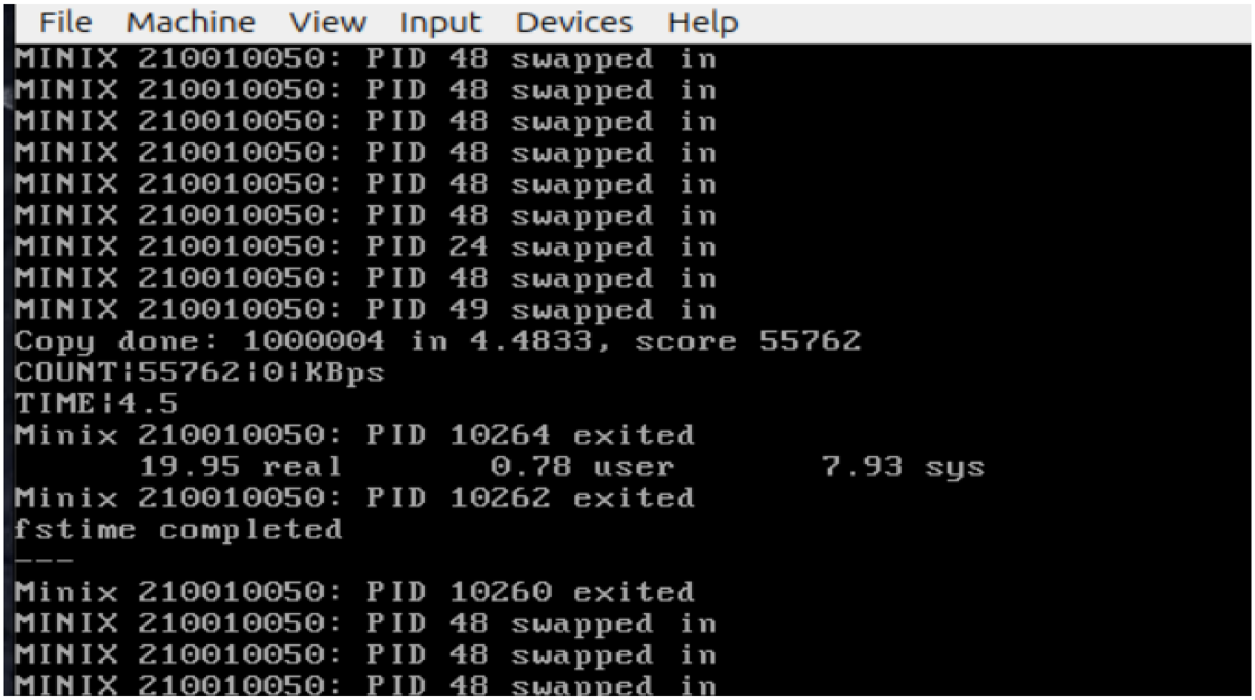


```
Minix 210010050: PID 10219 exited
Minix 210010050: PID 10220 created
MINIX 210010050: PID 231 swapped in
Minix 210010050: PID 10220 exited
Minix 210010050: PID 10221 created
MINIX 210010050: PID 232 swapped in
Minix 210010050: PID 10221 exited
Minix 210010050: PID 10222 created
MINIX 210010050: PID 233 swapped in
Minix 210010050: PID 10222 exited
Minix 210010050: PID 10223 created
MINIX 210010050: PID 234 swapped in
Minix 210010050: PID 10223 exited
Minix 210010050: PID 222 exited
      28.15 real        0.66 user       18.23 sys
Minix 210010050: PID 221 exited
spawn completed
---
Minix 210010050: PID 220 exited
# _
```

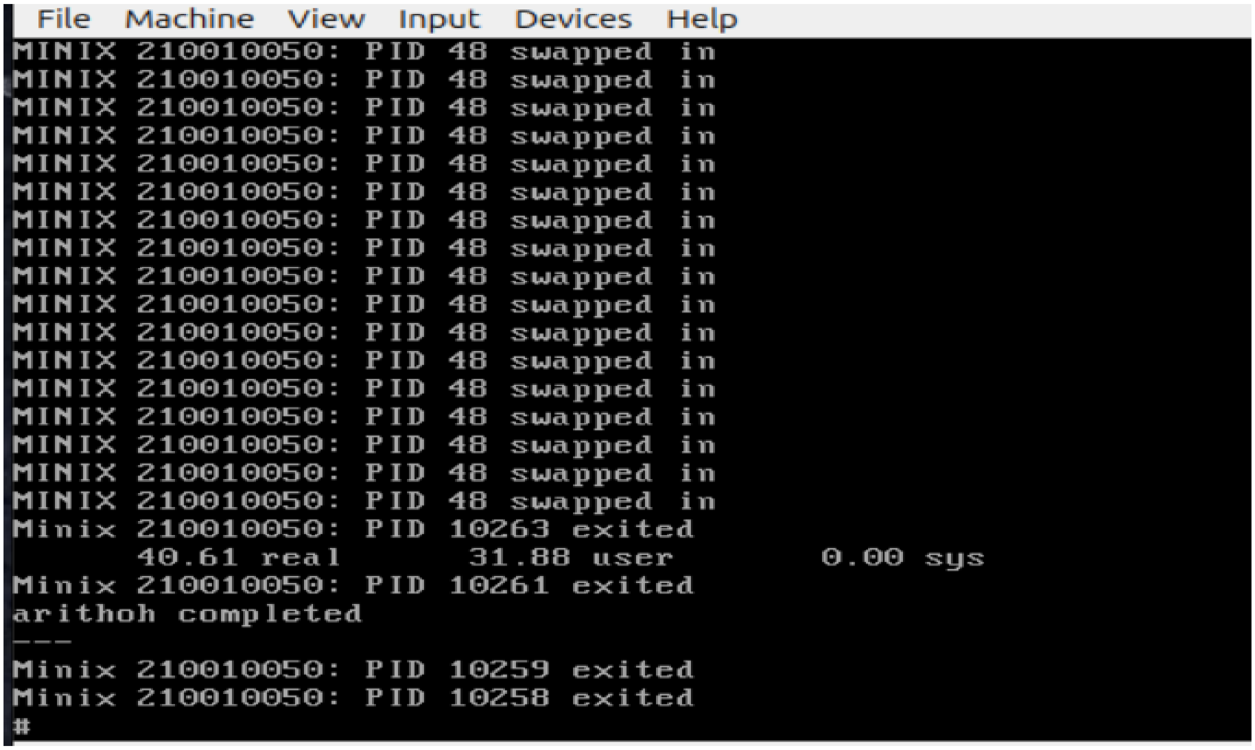
- Running a workload_mix :
  - workload_mix1.sh:



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



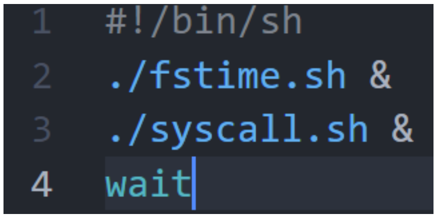
```
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 24 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 49 swapped in
Copy done: 1000004 in 4.4833, score 55762
COUNT|55762|0|KBps
TIME|4.5
Minix 210010050: PID 10264 exited
      19.95 real         0.78 user         7.93 sys
Minix 210010050: PID 10262 exited
fstime completed
---
Minix 210010050: PID 10260 exited
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
```



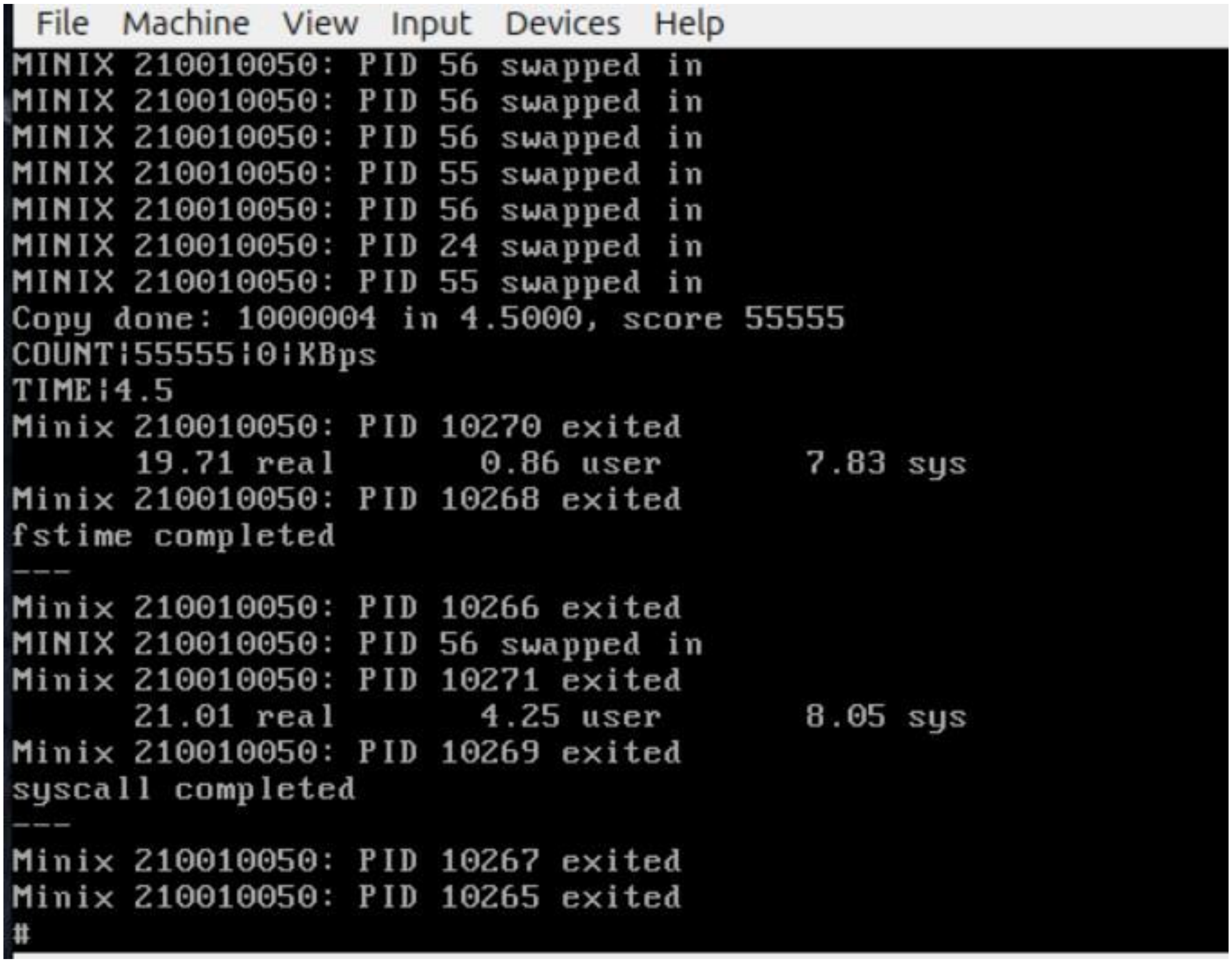
```
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
MINIX 210010050: PID 48 swapped in
Minix 210010050: PID 10263 exited
      40.61 real        31.88 user         0.00 sys
Minix 210010050: PID 10261 exited
arithoh completed
---
Minix 210010050: PID 10259 exited
Minix 210010050: PID 10258 exited
#
```

It is observed that, fstime completes its execution before arithoh as fstime mostly consists of real time processes which have the highest priority when compared to arith operations. Also among the two processes, arithoh.sh has a higher pre-emption frequency as it is a lower priority process. We observe that arithoh swaps occur in the time where fstime waits for I/O. The fstime.sh process is switched back to user mode only when all the I/O routines are completed.

- workload_mix2.sh:



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



```
MINIX 210010050: PID 56 swapped in
MINIX 210010050: PID 56 swapped in
MINIX 210010050: PID 56 swapped in
MINIX 210010050: PID 55 swapped in
MINIX 210010050: PID 56 swapped in
MINIX 210010050: PID 24 swapped in
MINIX 210010050: PID 55 swapped in
Copy done: 1000004 in 4.5000, score 55555
COUNT|55555|0|KBps
TIME|4.5
Minix 210010050: PID 10270 exited
      19.71 real        0.86 user        7.83 sys
Minix 210010050: PID 10268 exited
fstime completed
---
Minix 210010050: PID 10266 exited
MINIX 210010050: PID 56 swapped in
Minix 210010050: PID 10271 exited
      21.01 real        4.25 user        8.05 sys
Minix 210010050: PID 10269 exited
syscall completed
---
Minix 210010050: PID 10267 exited
Minix 210010050: PID 10265 exited
#
```

In this, we run a syscall and an I/O bound process- fstime to observe the process swaps. In this case, the CPU-intensive parts of the syscall code swap in when fstime waits for I/O.