Operating Systems Lab - CS 314

Rishit Saiya - 180010027, Assignment - 3

January 29, 2021

1 Part-1

In Part-1, we had to modify the Minix3 source code such that the string *"PID <pid> swapped in"* is printed, whenever a user-level process is brought in by the scheduler.

Using the help of Source-1 & Source-2, the file at location: `minix/servers/sched` called `schedule.c`. The following piece of code was added in *schedule_process* function:

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

In order to deploy the changes on Minix3 VM, a simple script `run.sh` was written whose main function is to copy `schedule.c` to correct location and build the system to see changes.

```
cp schedule.c /usr/src/minix/servers/sched/schedule.c
cd /usr/src && make build MKUPDATE=yes >log.txt 2>log.txt
```

After a successful build, Figure 1-4 shows the effective changes done.

2 Part-2

Using the UnixBench Benchmark Suite, some modifications in `UnixBench/workload_mix` were done in order to study the behavior of the scheduler by seeing the sequence of PID prints when these workloads are run. In the following subsections, 4 different combinations of `workload_mix` to understand how workloads are run.

2.1 `workload_mix1.sh`

In this shell script, 2 scripts `arithoh.sh` & `fstime.sh` are used as follows:

```
Minix: PID 24793 created
Minix: PID 24793 exited
Minix: PID 24794 created
Minix: PID 24794 exited
Minix: PID 24795 created
Minix: PID 24795 exited
Minix: PID 24796 created
Minix: PID 24796 exited
Minix: PID 24797 created
Minix: PID 24797 exited
Minix: PID 24798 created
Minix: PID 24798 exited
Minix: PID 24743 exited
Minix: PID 24711 exited
Minix: PID 24634 exited
Minix: PID 24633 exited
Minix: PID 24799 created
Minix: PID 24799 exited
Minix: PID 24800 created
Minix: PID 24801 created
Minix: PID 24801 exited
Minix: PID 24800 exited
Minix: PID 259 exited
Minix: PID 257 exited
# _
```

Figure 1: `run.sh` successfully executed

```
 extracting: byte-unixbench-mod/UnixBench/workload_mix/workload_mix.sh
 extracting: byte-unixbench-mod/UnixBench/workload_mix/workload_mix2.sh
 extracting: byte-unixbench-mod/UnixBench/workload_mix/workload_mix3.sh
 extracting: byte-unixbench-mod/UnixBench/workload_mix/workload_mix4.sh
Minix: PID 234 exited
# ls
Minix: PID 235 created
PID 210 swapped in
byte-unixbench-mod        byte-unixbench-mod.zip
Minix: PID 235 exited
# cd byte-unixbench-mod
# ls
Minix: PID 236 created
PID 211 swapped in
LICENSE.txt  README.md    UnixBench
Minix: PID 236 exited
# cd UnixBench/
# ls
Minix: PID 237 created
PID 212 swapped in
.cproject       README          WRITING_TESTS   src             workload_mix
.project        Run             pgms            testdir
Makefile        USAGE           results         tmp
Minix: PID 237 exited
# gmake_
```

Figure 2: Building using `gmake` command

```
do-hdboot ===> releasetools
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
Done.
Build started at:  Tue Jan 26 20:59:55 IST 2021
Build finished at: Tue Jan 26 21:03:58 IST 2021
Minix: PID 224 exited
#
```

Figure 3: Build Successfully Completed

```
Minix: PID 214 exited
# ls
Minix: PID 215 created
PID 190 swapped in
.exrc     .profile
Minix: PID 215 exited
#
```

Figure 4: `PID <PID> swapped in` shown in Terminal

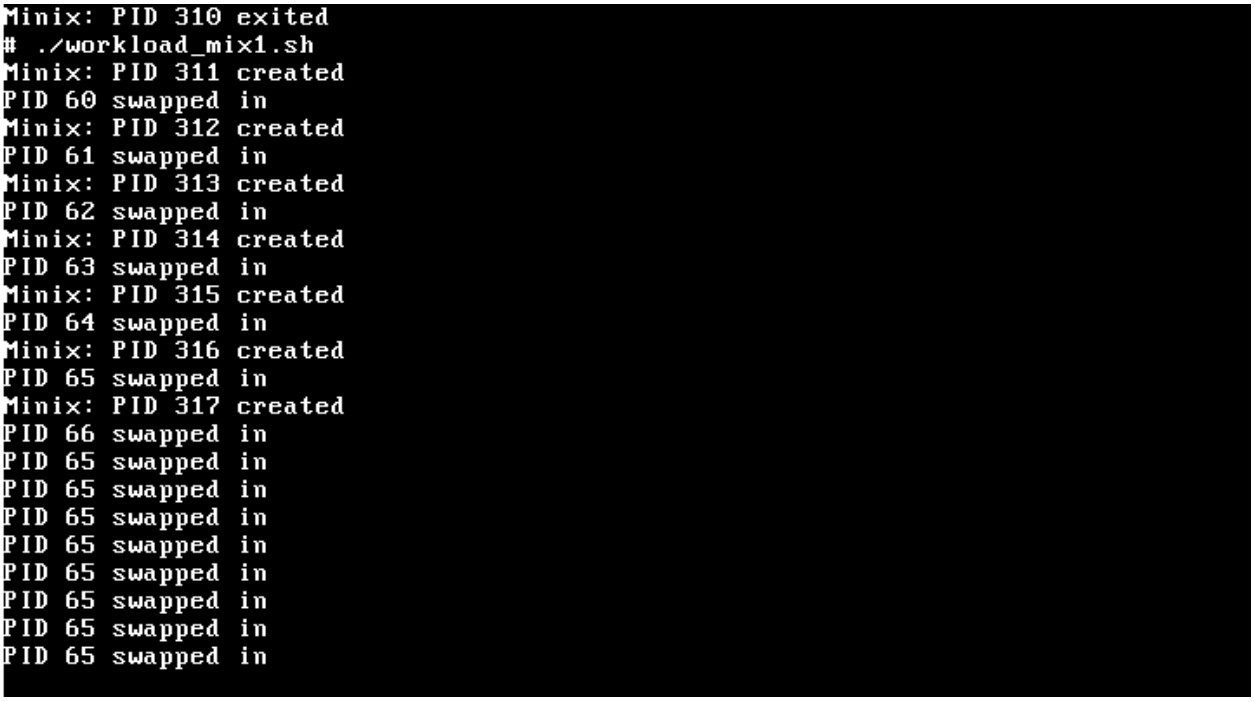

Figure 5: Execution of `workload_mix1.sh`

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

In `workload_mix1.sh`, an instance of file `arithoh.sh` and an instance of file `fstime.sh` are run. It is clear that the instructions in `arithoh.sh` are computationally intensive, whilst those in `fstime.sh` are I/O bound. Clearly in Figure 5-8 it is seen that only the "PID ¡PID¿ swapped in" statements corresponding to `arithoh.sh` get printed as `fstime.sh` waits for its I/O operations to be completed.

In here, `arithoh.sh` has PID 65 and `fstime.sh` has PID 66. So, PID 65 process is scheduled and PID process 66 is waiting for an input till then `arithoh.sh` is scheduled and is using processor for its CPU intensive tasks. When `fstime.sh` with PID 66 receives input, it is scheduled and process is completed. Towards the end, next 65 PID task is scheduled till it completed.

2.2 `workload_mix2.sh`

In this shell script, only script `arithoh.sh` is used as follows:

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

```
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
Read done: 1000004 in 1.5500, score 161290
COUNT¦161290¦0¦KBps
TIME¦1.5
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
```

Figure 6: PID load alternatively till input for `fstime.sh` is received

```
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 65 swapped in
PID 66 swapped in
PID 65 swapped in
PID 66 swapped in
Copy done: 1000004 in 2.7833, score 89820
COUNT¦89820¦0¦KBps
TIME¦2.8
Minix: PID 317 exited
      16.65 real       0.40 user       5.01 sys
Minix: PID 315 exited
fstime completed
---
Minix: PID 313 exited
PID 65 swapped in
```

Figure 7: `fstime.sh` scheduled and completed prior

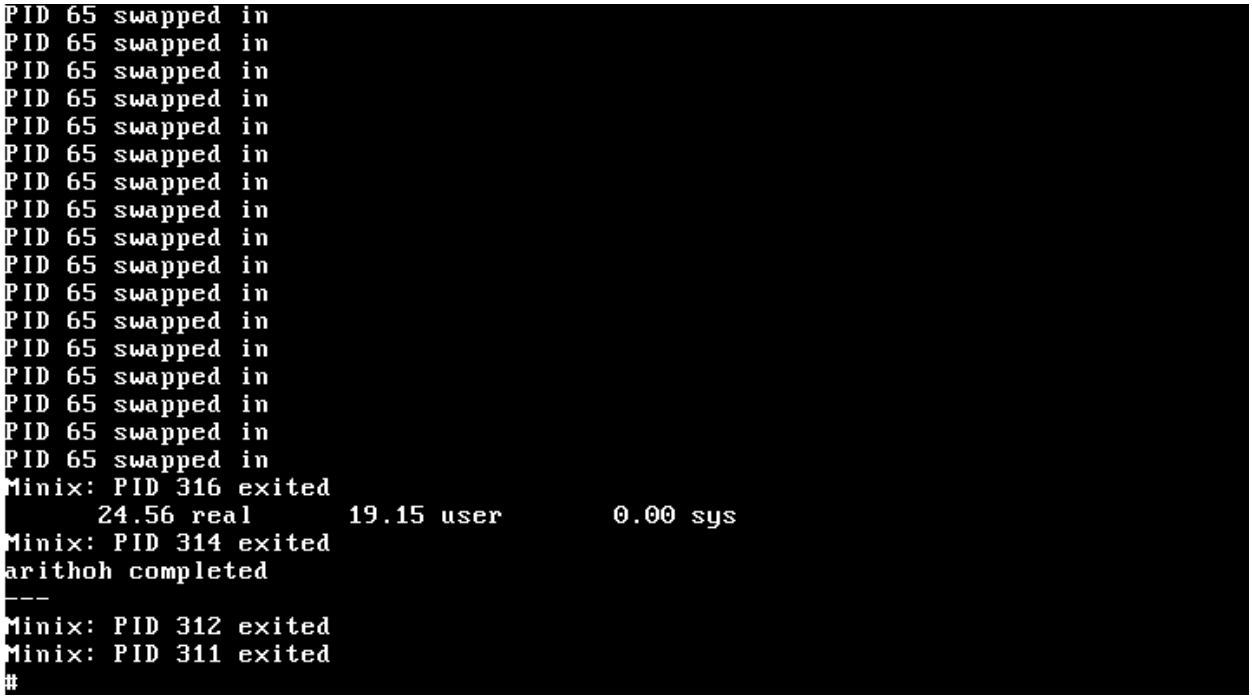

Figure 8: `arithoh.sh` completed towards the end

```
./arithoh.sh &
./arithoh.sh &
wait
```

In here, first `arithoh.sh` has PID of 74, second `arithoh.sh` has PID 75 and the last `arithoh.sh` has PID 76. Clearly, if multiple instances are run of `arithoh.sh` which are CPU Intensive in nature as in this workload, it can be seen that th getting scheduled alternatively. Figure 9-10 show that process with PID 74,75,76 correspond to 3 instances of workload in `arithoh.sh` that are scheduled alternatively until they are completed.

2.3 `workload_mix3.sh`

In this shell script, 2 scripts `arithoh.sh`, `syscall.sh` are used as follows:

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

In here, `arithoh.sh` has PID 82 and `syscall.sh` has PID 83 By literature, it is known that `arithoh.sh` and `syscall.sh` are different kind of CPU Intensive tasks. Figure 11-13, show that during the execution at start, both are alternatively scheduled depending upon

```
PID 74 swapped in
PID 75 swapped in
PID 76 swapped in
PID 74 swapped in
PID 75 swapped in
PID 76 swapped in
PID 75 swapped in
PID 76 swapped in
PID 74 swapped in
PID 76 swapped in
PID 74 swapped in
PID 75 swapped in
PID 74 swapped in
PID 75 swapped in
PID 76 swapped in
PID 75 swapped in
PID 76 swapped in
PID 74 swapped in
PID 76 swapped in
PID 74 swapped in
PID 75 swapped in
PID 74 swapped in
PID 75 swapped in
PID 76 swapped in
```

Figure 9: PID 74,75, 76 of `arithoh.sh` scheduled in order

```
PID 74 swapped in
PID 75 swapped in
PID 74 swapped in
PID 75 swapped in
PID 75 swapped in
PID 74 swapped in
PID 74 swapped in
PID 75 swapped in
PID 75 swapped in
PID 74 swapped in
PID 75 swapped in
Minix: PID 325 exited
      58.40 real      19.43 user       0.00 sys
Minix: PID 322 exited
arithoh completed
---
Minix: PID 319 exited
Minix: PID 326 exited
      58.55 real      19.70 user       0.00 sys
Minix: PID 323 exited
arithoh completed
---
Minix: PID 320 exited
Minix: PID 318 exited
#
```

Figure 10: `arithoh.sh` all process complete

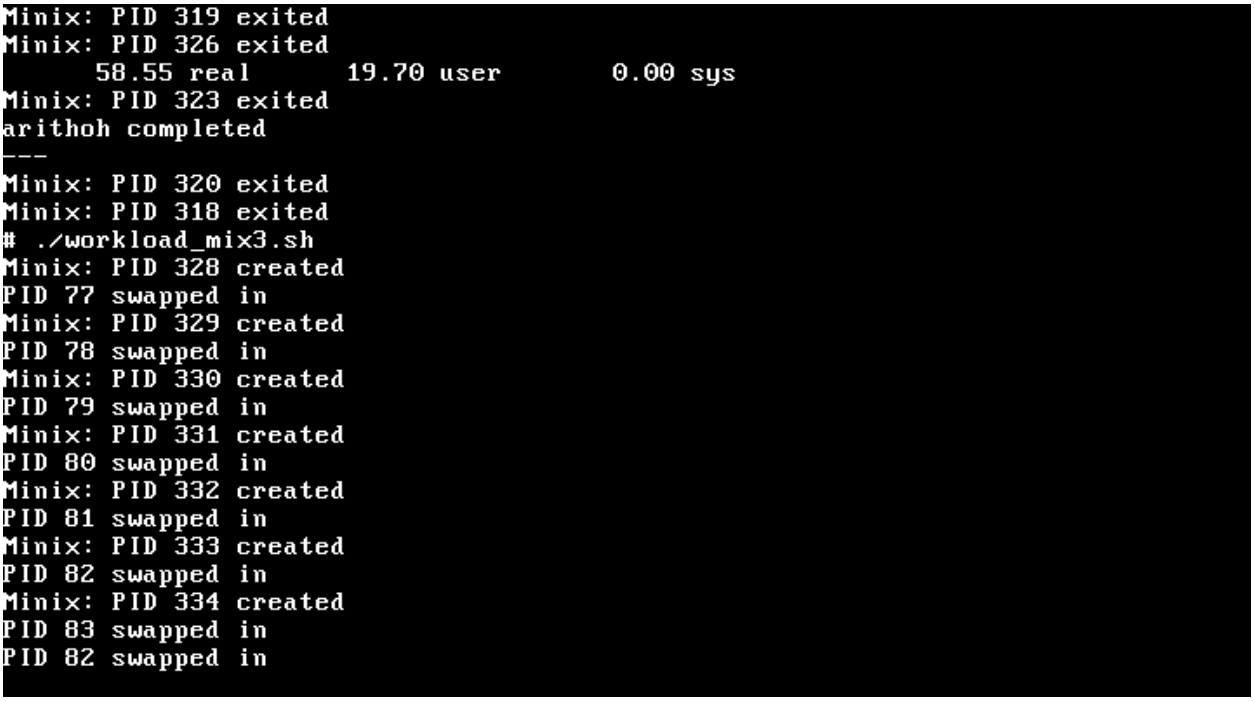

Figure 11: Execution of `workload_mix3.sh`

their intensiveness and then when `syscall.sh` with PID 83, which is relatively less CPU Intensive as compared to `arithoh.sh` is completed prior. Towards the end, `arithoh.sh` is scheduled until finished completely.

2.4 `workload_mix4.sh`

In this shell script, script `fstime.sh` is used as follows:

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

In Figure 14-16, 2 same workloads *fstime.sh* with PID 90, 92, 93. All the 3 are I/O processes that are waiting for input. When a process receives input, then that PID process is scheduled and finished prior. In the similar fashion, in the order of received inputs, processes are scheduled and then finished.

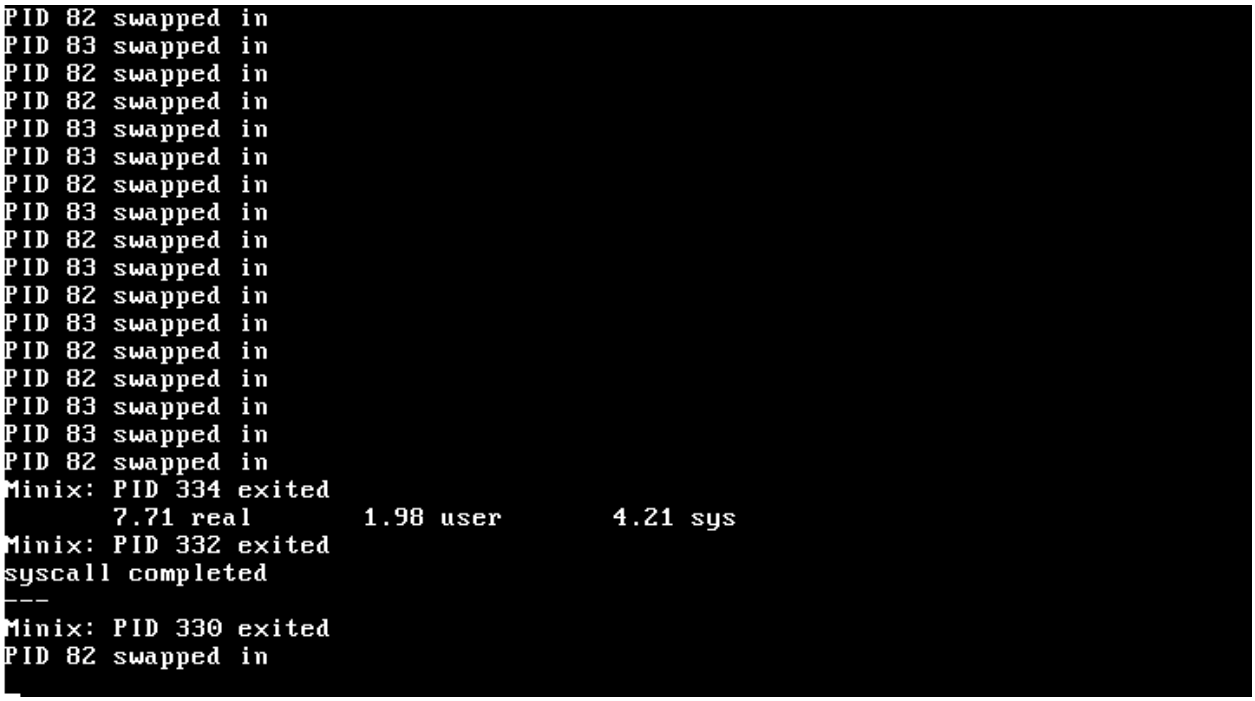

Figure 12: `syscall.sh` scheduled prior and completed

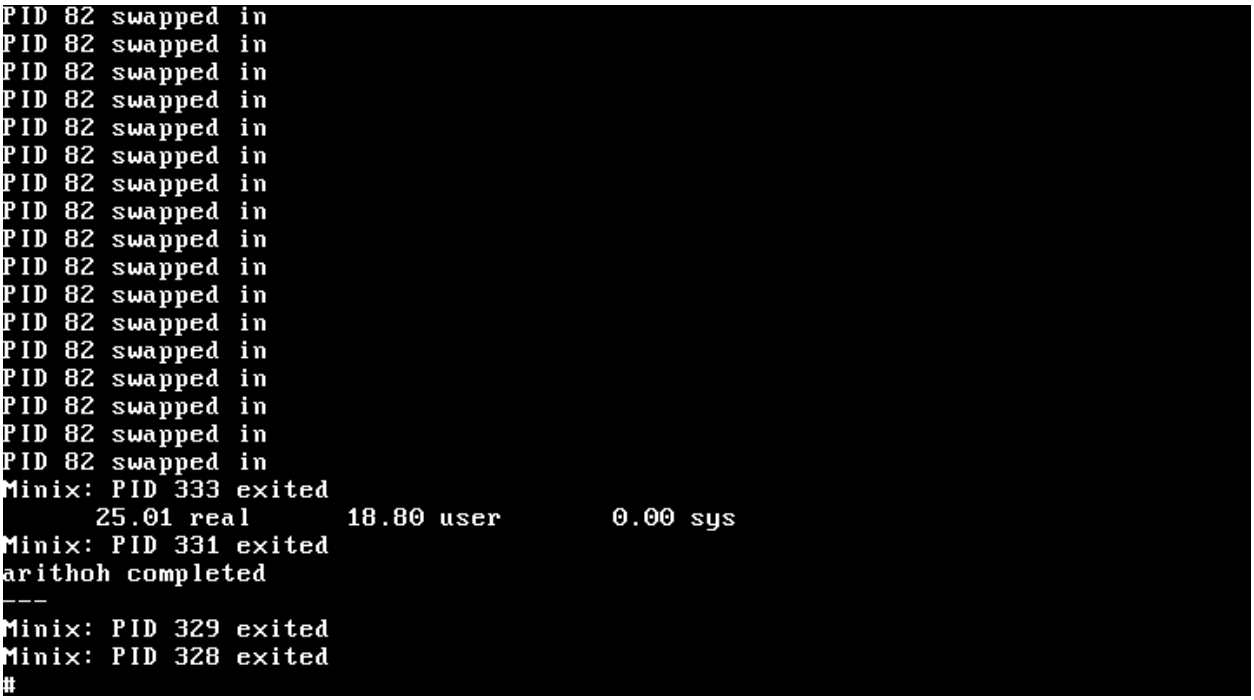

Figure 13: `arithoh.sh` process complete

```
# ./workload_mix4.sh PID 91 swapped in
PID 91 swapped in

# ./workload_mix4.sh
Minix: PID 335 created
PID 84 swapped in
Minix: PID 336 created
PID 85 swapped in
Minix: PID 337 created
PID 86 swapped in
Minix: PID 338 created
PID 87 swapped in
Minix: PID 339 created
PID 88 swapped in
Minix: PID 340 created
PID 89 swapped in
Minix: PID 341 created
PID 90 swapped in
Minix: PID 342 created
PID 92 swapped in
Minix: PID 343 created
PID 93 swapped in
Minix: PID 344 created
PID 94 swapped in
```

Figure 14: Execution of `workload_mix4.sh`

```
Minix: PID 342 created
PID 92 swapped in
Minix: PID 343 created
PID 93 swapped in
Minix: PID 344 created
PID 94 swapped in
Write done: 1008000 in 3.0667, score 82173
Write done: 1008000 in 3.0667, score 82173
Write done: 1008000 in 3.0667, score 82173
COUNT|82173|0|KBps
COUNT|82173|0|KBps
COUNT|82173|0|KBps
TIME|3.1
TIME|3.1
TIME|3.1
Read done: 1000004 in 2.9833, score 83799
Read done: 1000004 in 2.9833, score 83799
Read done: 1000004 in 2.9833, score 83799
COUNT|83799|0|KBps
COUNT|83799|0|KBps
COUNT|83799|0|KBps
TIME|3.0
TIME|3.0
TIME|3.0
```

Figure 15: I/O execution done for all `fstime.sh` processes

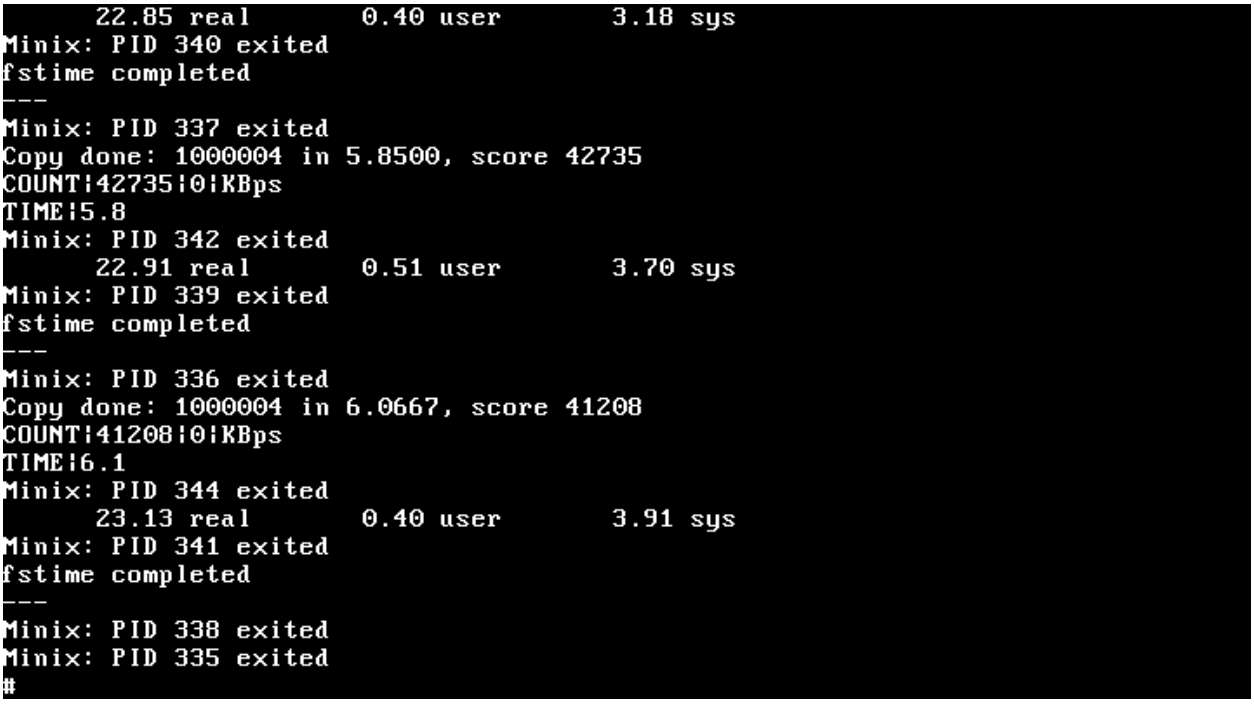

Figure 16: All `fstime.sh` process complete