Project Two Report
Introduction to Operating Systems
New Beginnings Spring 2018

Gavin Megson

22 April 2018

Description

For this assignment, I learned about kernel level data structures and concurrency; implementing system calls regarding process ownership and other information; tracking the amount of time the CPU spends on processes; and implementing user-level commands to display process state.

Deliverables

The following features were added to xv6:

- New fields in the process structure to track the User ID, Group ID, and Parent ID of a process, as well as time spent in CPU. System calls for getting UID, GID, and PPID and setting UID and GID were added.
- The Ctrl-P command was updated to display active process information, including the above mentioned fields.
- A new user command to time the execution of other processes was added, which simply allows another process to execute and times it. This required no new system calls.

Implementation

Updates to Process Structure

The proc struct in `proc.h` had the following fields added (lines 75 – 79):

- Fields UID and GID, unsigned integers representing the User and Group IDs. The range of possible values are from 0 to 32767, and the default value is defined in `param.h` as 0.
- Fields cpu ticks total and cpu ticks in, unsigned integers. Field cpu ticks in captures the CPU's ticks at the time the process enters or re-enters its timeslice, while total uses that field to accumulate the total number of ticks the process has spent running.

Associated System Calls

The following functions related to the above proc updates were added or changed:

- System calls for setting the UID and GID of a process were added (`sysproc.c` lines 128 – 148). These return an error if the ID is outside the prescribed range (0 to 32767). Associated function testuidgid in new file idtests.c confirms these functions are in working order.
- System call `fork`, as a function that creates new processes, was edited to ensure proper inheritance of UID and GID (lines 192 – 194).
- In the special case of `init`, the first process, the function `userinit` was modified (lines 122 – 125) to set UID and GID to 0. Field `parent` also was changed to point to its own process; this helps ensure methods returning a Parent PID will show `init` as its own parent.
- System calls for returning the UID, GID, and Parent PID of a process were added (lines 111 – 126).
- Edits to functions `scheduler` (line 60) and `sched` (line 385) respectively in `proc.c` update the `cpu ticks` fields when the process begins and ends running, respectively.

Ctrl-P

- The Ctrl-P display was updated with the code emailed out to students (`proc.c` lines 540 – 577, new file `procdump.c`).

Time Function

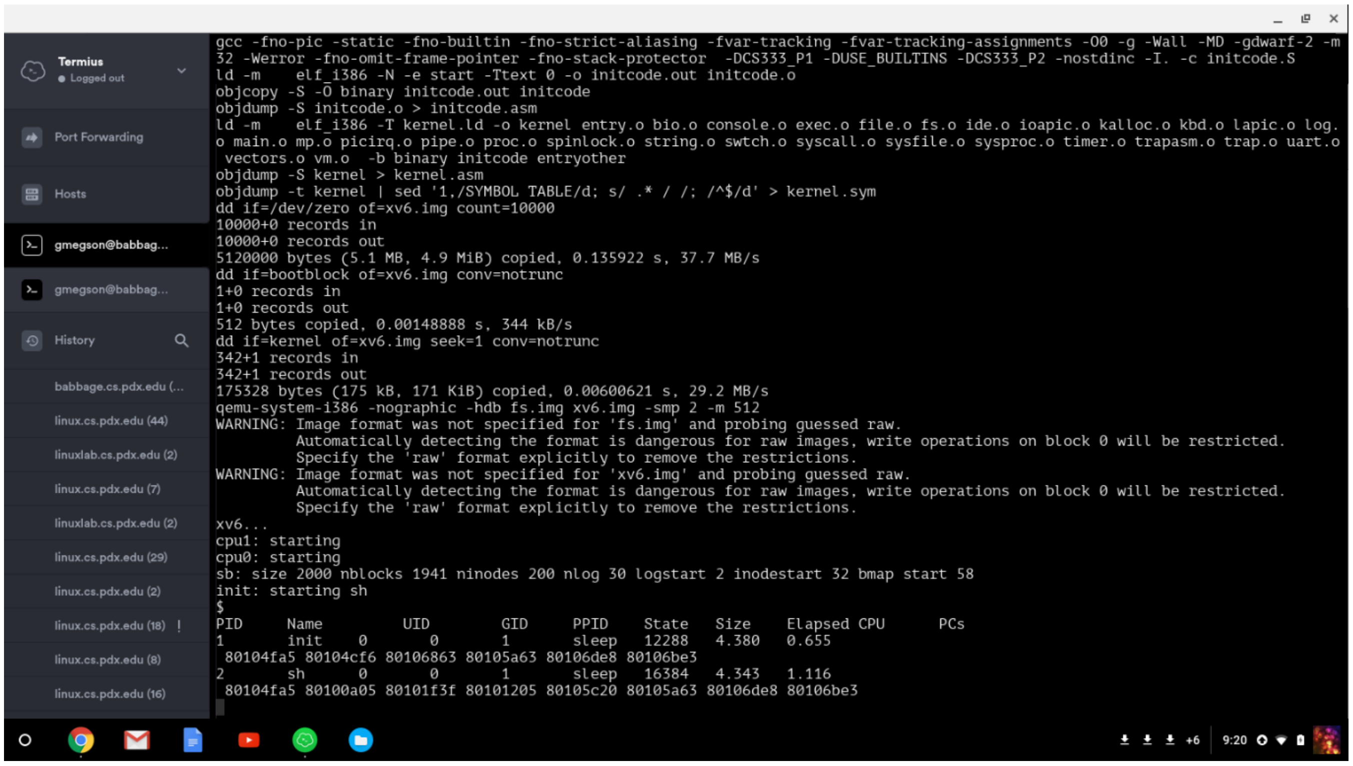
User command `time` (`time.c`) takes another user command and its own associated arguments as arguments and times how long the process takes to elapse, printing to the console the total time it took.

- The function works by forking a child process to exec the rest of the arguments, then having the parent wait on its execution and calculating the difference in ticks.
- No new system calls were needed, as existing call `uptime` was sufficient to get the ticks necessary to calculate time.

Testing

Ctrl-P Updates

Ctrl-P has been updated to display as follows:



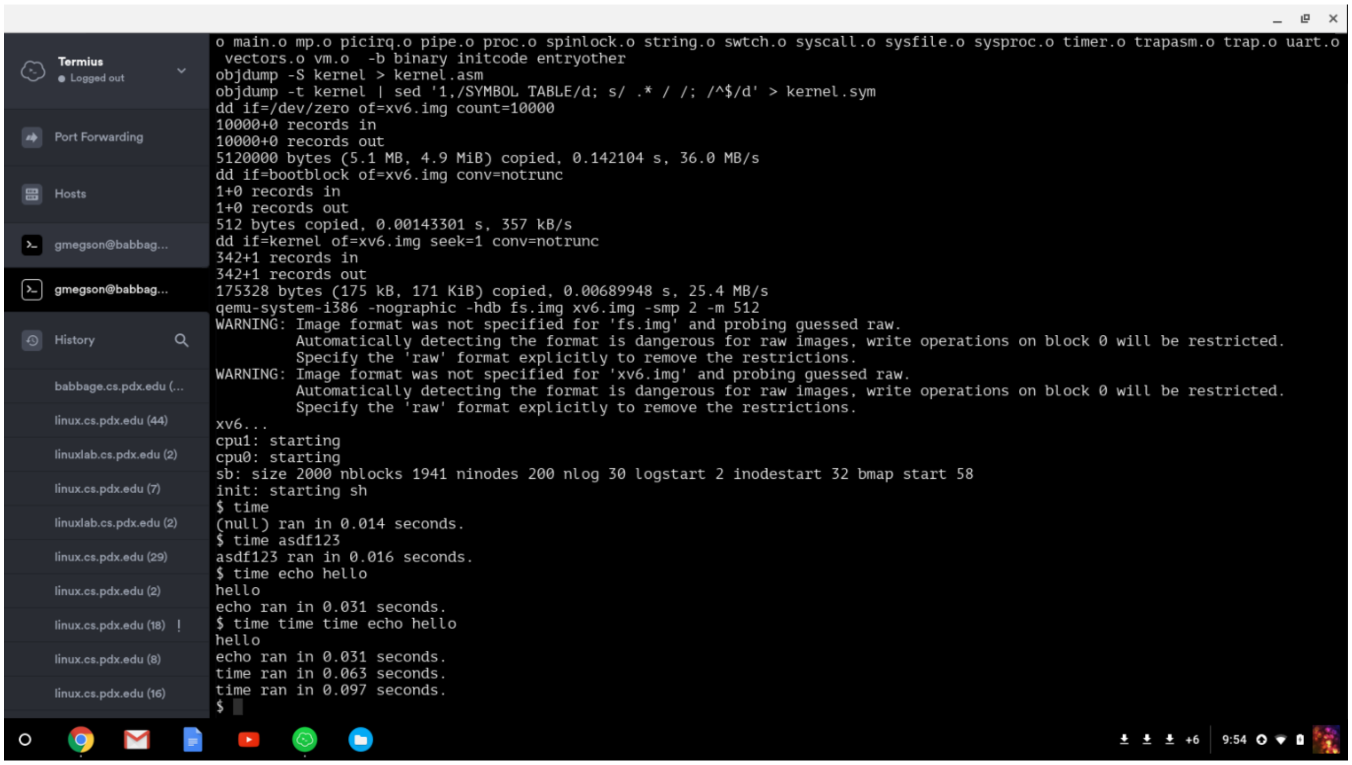
```
gcc -fno-pic -static -fno-builtin -fno-strict-aliasing -fvar-tracking -fvar-tracking-assignments -O0 -g -Wall -MD -gdwarf-2 -m
32 -Werror -fno-omit-frame-pointer -fno-stack-protector  -DCS333_P1 -DUSE_BUILTINS -DCS333_P2 -nostdinc -I. -c initcode.S
ld -m    elf_i386 -N -e start -Ttext 0 -o initcode.out initcode.o
objcopy -S -O binary initcode.out initcode
objdump -S initcode.o > initcode.asm
ld -m    elf_i386 -T kernel.ld -o kernel entry.o bio.o console.o exec.o file.o fs.o ide.o ioapic.o kalloc.o kbd.o lapic.o log.
o main.o mp.o picirq.o pipe.o proc.o spinlock.o string.o swtch.o syscall.o sysfile.o sysproc.o timer.o trapasm.o trap.o uart.o
 vectors.o vm.o  -b binary initcode entryother
objdump -S kernel > kernel.asm
objdump -t kernel | sed '1,/SYMBOL TABLE/d; s/ .* / /; /^$/d' > kernel.sym
dd if=/dev/zero of=xv6.img count=10000
10000+0 records in
10000+0 records out
5120000 bytes (5.1 MB, 4.9 MiB) copied, 0.135922 s, 37.7 MB/s
dd if=bootblock of=xv6.img conv=notrunc
1+0 records in
1+0 records out
512 bytes copied, 0.00148888 s, 344 kB/s
dd if=kernel of=xv6.img seek=1 conv=notrunc
342+1 records in
342+1 records out
175328 bytes (175 kB, 171 KiB) copied, 0.00600621 s, 29.2 MB/s
qemu-system-i386 -nographic -hdb fs.img xv6.img -smp 2 -m 512
WARNING: Image format was not specified for 'fs.img' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
WARNING: Image format was not specified for 'xv6.img' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
xv6...
cpu1: starting
cpu0: starting
sb: size 2000 nblocks 1941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
$
PID     Name        UID        GID     PPID    State   Size    Elapsed CPU     PCs
1       init    0       0       1       sleep   12288   4.380   0.655
 80104fa5 80104cf6 80106863 80105a63 80106de8 80106be3
2       sh      0       0       1       sleep   16384   4.343   1.116
 80104fa5 80100a05 80101f3f 80101205 80105c20 80105a63 80106de8 80106be3
```

Figure 1: Ctrl-P New Output

The headers and information associated with each process has been updated to specifications.
This test `PASSES`.

Test `time` Command

The `time` command was invoked with null and invalid arguments, and with a function that takes its own argument.



```
o main.o mp.o picirq.o pipe.o proc.o spinlock.o string.o swtch.o syscall.o sysfile.o sysproc.o timer.o trapasm.o trap.o uart.o
 vectors.o vm.o  -b binary initcode entryother
objdump -S kernel > kernel.asm
objdump -t kernel | sed '1,/SYMBOL TABLE/d; s/ .* / /; /^$/d' > kernel.sym
dd if=/dev/zero of=xv6.img count=10000
10000+0 records in
10000+0 records out
5120000 bytes (5.1 MB, 4.9 MiB) copied, 0.142104 s, 36.0 MB/s
dd if=bootblock of=xv6.img conv=notrunc
1+0 records in
1+0 records out
512 bytes copied, 0.00143301 s, 357 kB/s
dd if=kernel of=xv6.img seek=1 conv=notrunc
342+1 records in
342+1 records out
175328 bytes (175 kB, 171 KiB) copied, 0.00689948 s, 25.4 MB/s
qemu-system-i386 -nographic -hdb fs.img xv6.img -smp 2 -m 512
WARNING: Image format was not specified for 'fs.img' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
WARNING: Image format was not specified for 'xv6.img' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
xv6...
cpu1: starting
cpu0: starting
sb: size 2000 nblocks 1941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
$ time
(null) ran in 0.014 seconds.
$ time asdf123
asdf123 ran in 0.016 seconds.
$ time echo hello
hello
echo ran in 0.031 seconds.
$ time time time echo hello
hello
echo ran in 0.031 seconds.
time ran in 0.063 seconds.
time ran in 0.097 seconds.
$
```

Figure 2: Time Test

The program handles null arguments properly, but displays misleading output on incorrect arguments. Calling a function which takes its own argument works correctly.

Because one of the subtests failed, his test `FAILS`.