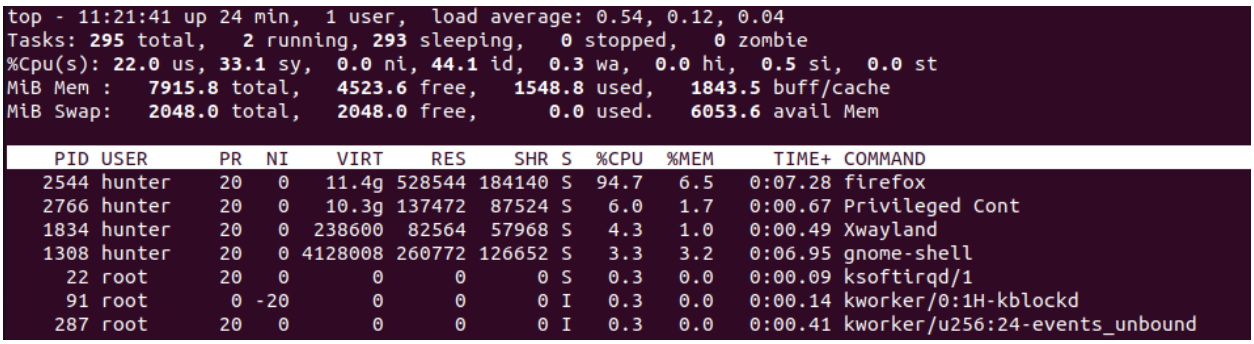
Hunter Pohl lab3

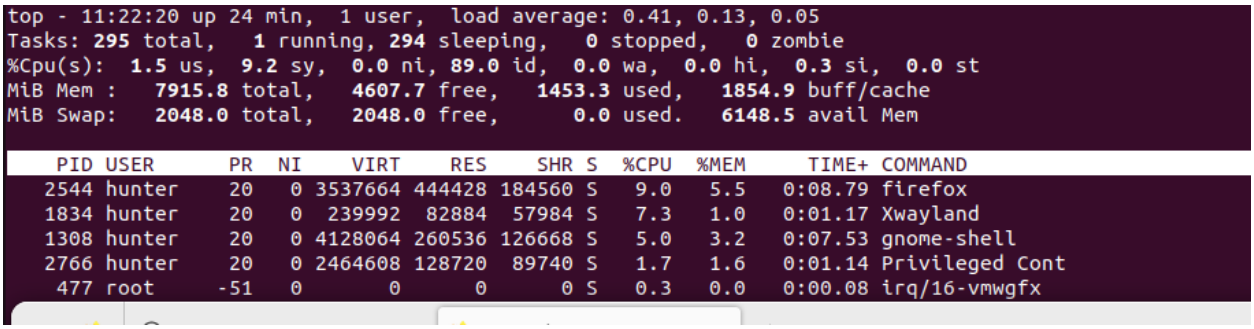
1. There are 4 child processes created. 2^n and there are two n, so =4.

2. starting a browser from Firefox

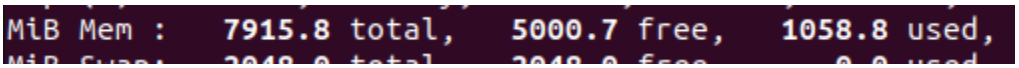
It consumed a lot of CPU at the start



Then the CPU % goes down and memory stays around the same



3. Amount of memory available



In total there is 7915.8 MiB

4. Process consuming CPU most

Is the ‘gnome-shell’ at 4.7%

```
top - 11:20:08 up 22 min,  1 user,  load average: 0.00, 0.00, 0.00
Tasks: 287 total,   1 running, 286 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.2 us,  1.0 sy,  0.0 ni, 97.8 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   7915.8 total,   5363.6 free,   1032.3 used,   1519.9 buff/cache
MiB Swap:   2048.0 total,   2048.0 free,      0.0 used.   6589.7 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1308 hunter    20   0 4130216 260256 126640 S   4.7   3.2   0:06.27 gnome-shell
    665 systemd+  20   0   14824   6068   5268 S   0.3   0.1   0:00.80 systemd-oomd
   1683 hunter    20   0  355912  28212  17496 S   0.3   0.3   0:00.62 ibus-extension-
   2430 hunter    20   0  577868  58948  46420 S   0.3   0.7   0:00.35 gnome-terminal-
   2527 hunter    20   0   22004   4084   3220 R   0.3   0.1   0:00.04 top
      1 root      20   0  167856  13112   8160 S   0.0   0.2   0:01.51 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.02 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
      8 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0H-events_highpri
     10 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
     11 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_rude_
     12 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_trace
     13 root      20   0       0      0      0 S   0.0   0.0   0:00.04 ksoftirqd/0
     14 root      20   0       0      0      0 I   0.0   0.0   0:00.22 rcu_sched
     15 root      rt   0       0      0      0 S   0.0   0.0   0:00.00 migration/0
     16 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/0
     18 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/0
     19 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/1
     20 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/1
     21 root      rt   0       0      0      0 S   0.0   0.0   0:00.70 migration/1
     22 root      20   0       0      0      0 S   0.0   0.0   0:00.08 ksoftirqd/1
     24 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/1:0H-events_highpri
     25 root      20   0       0      0      0 S   0.0   0.0   0:00.00 kdevtmpfs
     26 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 inet_frag_wq
     27 root      20   0       0      0      0 S   0.0   0.0   0:00.00 kauditd
     29 root      20   0       0      0      0 S   0.0   0.0   0:00.00 khungtaskd
     30 root      20   0       0      0      0 S   0.0   0.0   0:00.00 oom_reaper
     31 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 writeback
     32 root      20   0       0      0      0 S   0.0   0.0   0:00.02 kcompactd0
     33 root      25   5       0      0      0 S   0.0   0.0   0:00.00 ksmd
     34 root      39  19       0      0      0 S   0.0   0.0   0:00.00 khugepaged
     39 root      20   0       0      0      0 I   0.0   0.0   0:00.02 kworker/1:1-rcu_par_gp
     81 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kintegrityd
     82 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kblockd
     83 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 blkcg_punt_bio
     84 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 tpm_dev_wq
     85 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 ata_sff
     86 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 md
     87 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 edac-poller
     88 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 devfreq_wq
     89 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 watchdogd
     91 root       0 -20       0      0      0 I   0.0   0.0   0:00.12 kworker/0:1H-kblockd
     93 root      20   0       0      0      0 S   0.0   0.0   0:00.00 kswapd0
     94 root      20   0       0      0      0 S   0.0   0.0   0:00.00 ecryptfs-kthrea
     96 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kthrotld
     97 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 irq/24-pciehp
     98 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 irq/25-pciehp
     99 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 irq/26-pciehp
    100 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 irq/27-pciehp
    101 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 irq/28-pciehp
    102 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 irq/29-pciehp
    103 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 irq/30-pciehp
```

5. the process that has the most memory when the browser is running is Firefox but then when I close the browser it is: ‘gnome-shell’

```
    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1308 hunter    20   0 4128036 260452 126648 S   1.0   3.2   0:07.93 gnome-shell
```

Which is the terminal so it makes sense, since that is the only thing running

6. apt-get: can update, remove, and install packages using it.

yum: install, remove, and update packages and functions similarly to apt-get.

wget: to download files from the internet. The HTTP, HTTPS, and FTP protocols are supported.

gzip: This program uses the Gzip algorithm to compress files. Before files are transferred over a network or to conserve disk space, compression is frequently used.

tar: Using this archive tool, you can put several files into one archive file. Since tape backups were the original purpose of "tar," its name is derived from "tape archive".

rar: file compression and archiving

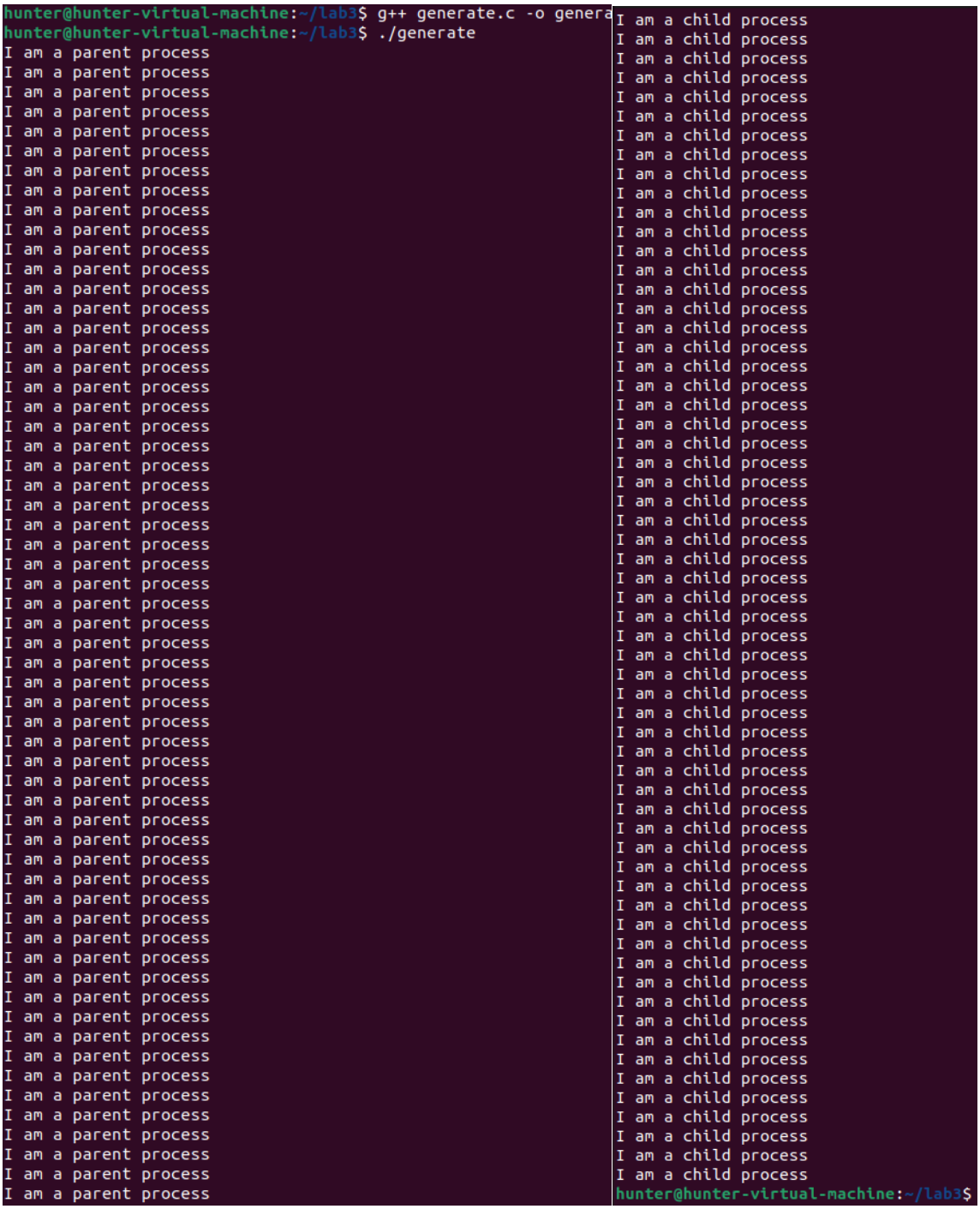
7. Write a program that will generate a child process. In a loop, the child process writes "I am a child process" 200 times and the parent process repeatedly prints "I am a parent process" in a loop.

```
#include <unistd.h>
#include <iostream>

int main() {
  pid_t pid = fork();

  if (pid == 0) {
    for (int i = 0; i < 200; ++i) {
      std::cout << "I am a child process" << std::endl;
    }
  } else {
    for (int i = 0; i < 200; ++i) {
      std::cout << "I am a parent process" << std::endl;
    }
  }

  return 0;
}
```



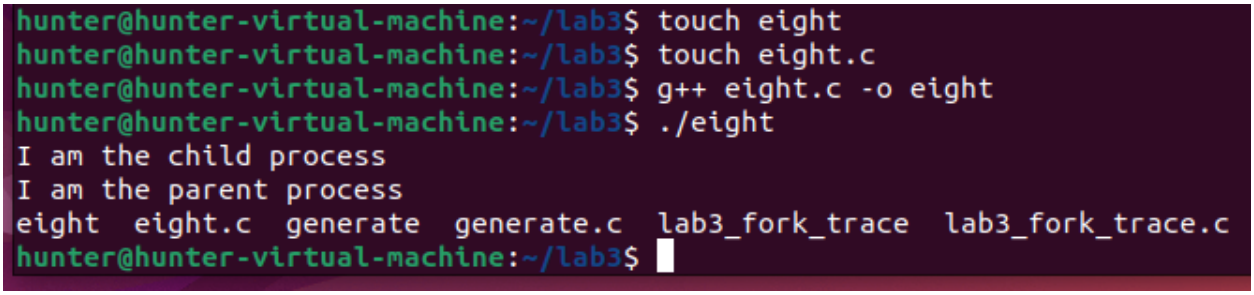
8. Write a program that create a child process with the fork () system call. The parent process waits for the child process to finish before printing the contents of the current directory.

```
#include <unistd.h>
#include <iostream>
#include <sys/wait.h>
#include <cstdlib>

int main() {
  pid_t pid = fork();

  if (pid == 0) {
    std::cout << "I am the child process" << std::endl;
    exit(0);
  } else {
    int status;
    waitpid(pid, &status, 0);
    std::cout << "I am the parent process" << std::endl;
    system("ls");
  }

  return 0;
}
```



9. Write a program that create a child process with the fork () system call and print its PID. Following a fork () system call, both parent and child processes print their process type and PID. Additionally, the parent process prints the PID of its child, and the child process prints the PID of its parent.

```
#include <unistd.h>
#include <iostream>
#include <sys/types.h>

int main() {
  pid_t pid = fork();

  if (pid == 0) {
```

```
    std::cout << "I am a child process" << std::endl;
    std::cout << "My PID: " << getpid() << std::endl;
    std::cout << "My parent's PID: " << getppid() << std::endl;
  } else {
    std::cout << "I am a parent process" << std::endl;
    std::cout << "My PID: " << getpid() << std::endl;
    std::cout << "My child's PID: " << pid << std::endl;
  }

  return 0;
}
```