PRACTICAL-1

Aim:

Collect the following basic information about your machine using proc.

a. How many CPU cores does the machine have?
b. How much memory, and what fraction of it is free?
c. How many context switches has the system performed since bootup?
d. How many processes has it forked since bootup?
e. How many processors does your machine have?
f. What is the frequency of each processor?
g. Find out various states of process at time of observation.

Program Code:

a. How many CPU cores does the machine have?

Command : cat cpuinfo

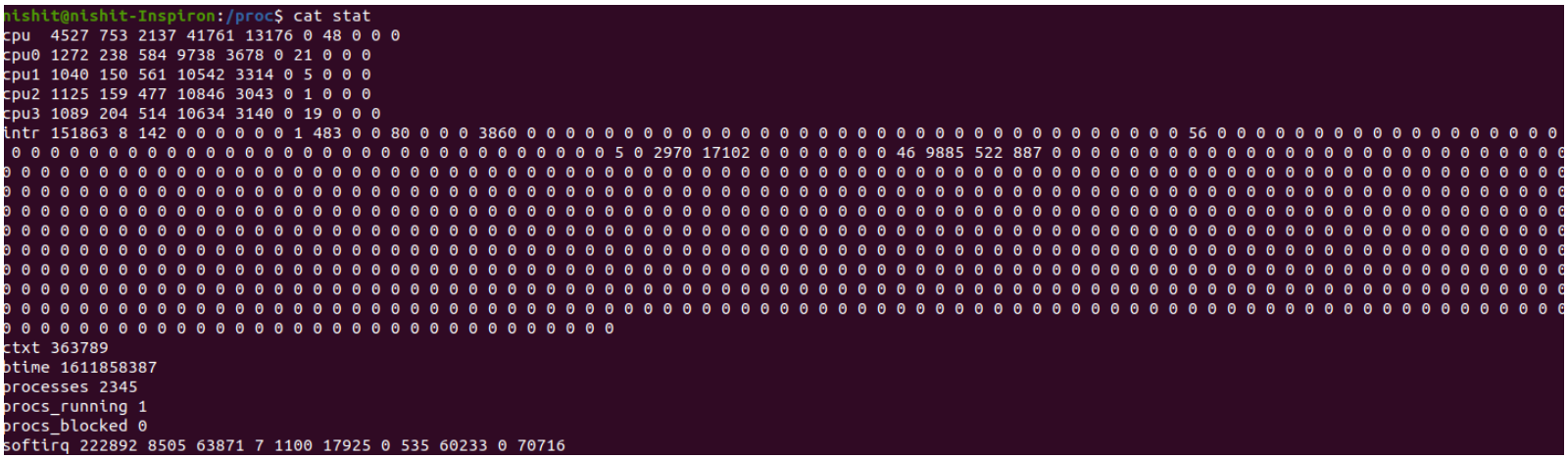
```
nishit@nishit-Inspiron:/proc$ cat cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 142
model name      : Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz
stepping        : 9
microcode       : 0xd6
cpu MHz         : 600.017
cache size      : 3072 KB
physical id     : 0
siblings        : 4
core id         : 0
cpu cores       : 2
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 22
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep
```

b. How much memory, and what fraction of it is free?
Command : cat meminfo

```
nishit@nishit-Inspiron:/proc$ cat meminfo
MemTotal:        8014980 kB
MemFree:         3057436 kB
MemAvailable:    4856968 kB
Buffers:           72764 kB
Cached:          2374112 kB
SwapCached:            0 kB
Active:          3199492 kB
Inactive:        1154860 kB
Active(anon):    2168640 kB
Inactive(anon):   213136 kB
Active(file):    1030852 kB
Inactive(file):   941724 kB
Unevictable:      259688 kB
Mlocked:               0 kB
SwapTotal:       3999740 kB
SwapFree:        3999740 kB
Dirty:               136 kB
Writeback:             0 kB
AnonPages:       2167216 kB
Mapped:           620308 kB
Shmem:            474308 kB
KReclaimable:      98484 kB
Slab:             225204 kB
SReclaimable:      98484 kB
SUnreclaim:       126720 kB
KernelStack:       16032 kB
PageTables:        29160 kB
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:     8007228 kB
Committed_AS:    8514740 kB
VmallocTotal:   34359738367 kB
VmallocUsed:       33588 kB
VmallocChunk:          0 kB
Percpu:             3824 kB
HardwareCorrupted:     0 kB
AnonHugePages:         0 kB
ShmemHugePages:        0 kB
ShmemPmdMapped:        0 kB
FileHugePages:         0 kB
FilePmdMapped:         0 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
Hugetlb:               0 kB
DirectMap4k:      272752 kB
DirectMap2M:     5881856 kB
DirectMap1G:     3145728 kB
```

Here total Memory : 8014980 kb
Free Memory : 3057436 kb
Fraction of memory that is free : 38.14 %

c. How many context switches has the system performed since bootup?
Command: cat stat
Ctxt line gives the total number of context switches across all cpu.



d. How many processes has it forked since bootup?
Command: vmstat -f or /proc/stat

e. How many processors does your machine have?

```
nishit@nishit-Inspiron:/proc$ cat cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 142
model name      : Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz
stepping        : 9
microcode       : 0xd6
cpu MHz         : 528.034
cache size      : 3072 KB
physical id     : 0
siblings        : 4
core id         : 0
cpu cores       : 2
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 22
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs b
ts rep_good nopl xtopology nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave a
vx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid mpx rdse
ed adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear flush_l1d
vmx flags       : vnmi preemption_timer invvpid ept_x_only ept_ad ept_1gb flexpriority tsc_offset vtpr mtf vapic ept vpid unrestricted_guest ple pml ept_mode_based_exec
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs itlb_multihit srbds
bogomips        : 5399.81
clflush size    : 64
cache_alignment : 64
address sizes   : 39 bits physical, 48 bits virtual
power management:

processor       : 1
vendor_id       : GenuineIntel
cpu family      : 6
model           : 142
model name      : Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz
stepping        : 9
microcode       : 0xd6
cpu MHz         : 568.967
cache size      : 3072 KB
physical id     : 0
siblings        : 4
core id         : 1
cpu cores       : 2
apicid          : 2
initial apicid  : 2
```

f. What is the frequency of each processor?
Command: cat cpuinfo | grep “MHz”

```
nishit@nishit-Inspiron:/proc$ cat cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 142
model name      : Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz
stepping        : 9
microcode       : 0xd6
cpu MHz         : 600.017
cache size      : 3072 KB
physical id     : 0
siblings        : 4
core id         : 0
cpu cores       : 2
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 22
wp              : yes
```

```
nishit@nishit-Inspiron:/proc$ cat cpuinfo | grep "MHz"
cpu MHz         : 600.007
cpu MHz         : 600.015
cpu MHz         : 600.015
cpu MHz         : 600.016
```

g. Find out various states of process at time of observation.

Running: In this state file either running or ready to run state.
Waiting: In this state, a process is waiting for an event to occur or for a system resource.
Stopped: in this state, a process has been stopped, usually by receiving a signal.
Zombie: In this state process is dead, it has been halted but it’s still has an entry in the process table.

Learning from practical: In this practical I have learnt about different files and commands of proc as well as different information about processes.

Assignment-1

Q-1 What is the time spent by a process in user mode and Kernel mode?

Ans: The difference is whether the time is spent in user space or kernel space.
User CPU time is time spent on the processor running your program's code (or code in libraries).
System CPU time is the time spent running code in the operating system kernel on behalf of your program.
Whenever we want to determine the time spent by a process in user mode and Kernel mode so at that time run “thread-times.stp” file and get the output like below:

```
   tid   %user %kernel (of 20002 ticks)
     0   0.00%  87.88%
 32169   5.24%   0.03%
  9815   3.33%   0.36%
  9859   0.95%   0.00%
  3611   0.56%   0.12%
  9861   0.62%   0.01%
 11106   0.37%   0.02%
 32167   0.08%   0.08%
  3897   0.01%   0.08%
  3800   0.03%   0.00%
  2886   0.02%   0.00%
  3243   0.00%   0.01%
  3862   0.01%   0.00%
  3782   0.00%   0.00%
 21767   0.00%   0.00%
  2522   0.00%   0.00%
  3883   0.00%   0.00%
  3775   0.00%   0.00%
  3943   0.00%   0.00%
  3873   0.00%   0.00%
```

Q-2 Find the version of linux kernel installed in your system.
Command: uname -r

```
nishit@nishit-Inspiron:/proc$ uname -r
5.8.0-40-generic
```

Q-3 Mention the difference between MemFree and MemAvailable fields of the file /proc/meminfo.

/proc/meminfo: provide estimated available memory

Many load balancing and workload placing programs check /proc/meminfo to estimate how much free memory is available. They generally do this by adding up "free" and "cached", which was fine ten years ago, but is pretty much guaranteed to be wrong today.

It is wrong because Cached includes memory that is not freeable as page cache, for example shared memory segments, tmpfs, and ramfs, and it does not include reclaimable slab memory, which can take up a large fraction of system memory on mostly idle systems with lots of files.

Currently, the amount of memory that is available for a new workload, without pushing the system into swap, can be estimated from MemFree, Active(file), Inactive(file), and SReclaimable, as well as the "low" watermarks from /proc/zoneinfo.

However, this may change in the future, and user space really should not be expected to know kernel internals to come up with an estimate for the amount of free memory.

It is more convenient to provide such an estimate in /proc/meminfo. If things change in the future, we only have to change it in one place.

```
nishit@nishit-Inspiron:/proc$ cat meminfo| grep "MemTotal"
MemTotal:        8014980 kB
nishit@nishit-Inspiron:/proc$ cat meminfo| grep "MemFree"
MemFree:         6406132 kB
nishit@nishit-Inspiron:/proc$ cat meminfo| grep "MemAvailable"
MemAvailable:    6837532 kB
```

Q-4 Explain the content of /proc/key-users

At the core, keys are stored in the aptly named struct key which has the following kinds of fields:
- a unique serial number
- a key type that can identify the filesystem that the key belongs to
- a description string that is used for searching for the key
- a payload that contains the actual key data
- user and group information including permissions
- an expiration time
- a key state that tracks instantiation, revocation, deletion, etc.

The key types provide a way for a filesystem to configure its own set of key operations. The operations that a key type can specify are:

- instantiate - create a key of that type • update - modify a key
- match - match a key to a description, which is used in the key search
- revoke - clear some key data and change the state to KEY_FLAG_REVOKED
- destroy - clear all key data
- describe - summarize the key's description and payload as text
- read - read the key data
- request_key - called when the key is not available in order to retrieve the key from elsewhere

The /proc/keys and /proc/key-users entries in proc enable a user to view the keys and key users currently managed by the kernel.

```
nishit@nishit-Inspiron:/proc$ cat key-users
    0:   188 187/187 86/1000000 1775/25000000
  101:     1 1/1 1/200 9/20000
  102:     1 1/1 1/200 9/20000
  116:     2 2/2 2/200 18/20000
  121:     1 1/1 1/200 9/20000
  125:     2 2/2 2/200 26/20000
 1000:     3 3/3 3/200 37/20000
```

Q-5 How many processes are running & how many are blocked?

- We can use GREP command to find out total number of running and total number of blocked processes that are in PROC are checked by below process:

```
nishit@nishit-Inspiron:/proc$ cat stat | grep "procs_blocked"
procs_blocked 0
nishit@nishit-Inspiron:/proc$ cat stat | grep "proc_running"
nishit@nishit-Inspiron:/proc$ cat stat | grep "procs_running"
procs_running 2
```