

## TASK5.3

### Part1

1. How many states could has a process in Linux?

A process can have the following states: created, ready, waiting, running, completed

2. Examine the pstree command. Make output (highlight) the chain (ancestors) of the current process.

```
student@CsnKhai:~$ pstree -s
init--cron
    |--dbus-daemon
    |--dhclient
    |--5*[getty]
    |--login--bash--pstree
    |--rsyslogd--3*[{rsyslogd}]
    |--sshd
    |--systemd-logind
    |--systemd-udev
    |--upstart-file-br
    |--upstart-socket-
    |--upstart-udev-br
student@CsnKhai:~$
```

3. What is a proc file system?

**The proc file system** - is as an interface to internal data structures in the kernel. It can be used to obtain information about the system and to change certain kernel parameters at runtime.

4. Print information about the processor (its type, supported technologies, etc.).

```
student@CsnKhai:~$ lscpu
Architecture:          i686
CPU op-mode(s):        32-bit
Byte Order:             Little Endian
CPU(s):                 1
On-line CPU(s) list:   0
Thread(s) per core:    1
Core(s) per socket:    1
Socket(s):              1
Vendor ID:              GenuineIntel
CPU family:             6
Model:                  141
Stepping:               1
CPU MHz:                2688.027
BogoMIPS:               5376.05
L1d cache:              48K
L1i cache:              32K
L2 cache:               1280K
L3 cache:               12288K
student@CsnKhai:~$
```

5. Use the ps command to get information about the process. The information should be as follows: the owner of the process, the arguments with which the process was launched for execution, the group owner of this process, etc.

|          |     |     |     |       |      |      |     |       |      |                 |
|----------|-----|-----|-----|-------|------|------|-----|-------|------|-----------------|
| root     | 46  | 0.0 | 0.0 | 0     | 0    | ?    | S   | 13:27 | 0:00 | [kworker/u2:2]  |
| root     | 67  | 0.0 | 0.0 | 0     | 0    | ?    | S<  | 13:27 | 0:00 | [deferwq]       |
| root     | 68  | 0.0 | 0.0 | 0     | 0    | ?    | S<  | 13:27 | 0:00 | [charger_manage |
| root     | 114 | 0.0 | 0.0 | 0     | 0    | ?    | S<  | 13:27 | 0:00 | [kpsmouse]      |
| root     | 115 | 0.0 | 0.0 | 0     | 0    | ?    | S   | 13:27 | 0:00 | [kworker/0:2]   |
| root     | 116 | 0.0 | 0.0 | 0     | 0    | ?    | S<  | 13:27 | 0:00 | [kworker/u3:1]  |
| root     | 117 | 0.0 | 0.0 | 0     | 0    | ?    | S   | 13:27 | 0:00 | [scsi_eh_2]     |
| root     | 126 | 0.0 | 0.0 | 0     | 0    | ?    | S   | 13:27 | 0:00 | [jbd2/sda1-8]   |
| root     | 127 | 0.0 | 0.0 | 0     | 0    | ?    | S<  | 13:27 | 0:00 | [ext4-rsv-conve |
| root     | 275 | 0.0 | 0.2 | 3008  | 616  | ?    | S   | 13:27 | 0:00 | upstart-udev-br |
| root     | 280 | 0.0 | 0.6 | 12096 | 1544 | ?    | Ss  | 13:27 | 0:00 | /lib/systemd/sy |
| message+ | 328 | 0.0 | 0.3 | 4236  | 984  | ?    | Ss  | 13:27 | 0:00 | dbus-daemon --s |
| root     | 359 | 0.0 | 0.6 | 4212  | 1724 | ?    | Ss  | 13:27 | 0:00 | /lib/systemd/sy |
| syslog   | 361 | 0.0 | 0.4 | 30476 | 1112 | ?    | Ssl | 13:27 | 0:00 | rsyslogd        |
| root     | 368 | 0.0 | 0.2 | 2880  | 596  | ?    | S   | 13:27 | 0:00 | upstart-file-br |
| root     | 462 | 0.0 | 0.2 | 2868  | 592  | ?    | S   | 13:27 | 0:00 | upstart-socket- |
| root     | 589 | 0.0 | 0.7 | 5512  | 1852 | ?    | Ss  | 13:27 | 0:00 | dhclient -1 -v  |
| root     | 709 | 0.0 | 0.3 | 4644  | 824  | tty4 | Ss+ | 13:27 | 0:00 | /sbin/getty -8  |
| root     | 711 | 0.0 | 0.3 | 4644  | 824  | tty5 | Ss+ | 13:27 | 0:00 | /sbin/getty -8  |
| root     | 714 | 0.0 | 0.3 | 4644  | 836  | tty2 | Ss+ | 13:27 | 0:00 | /sbin/getty -8  |
| root     | 715 | 0.0 | 0.3 | 4644  | 832  | tty3 | Ss+ | 13:27 | 0:00 | /sbin/getty -8  |
| root     | 717 | 0.0 | 0.3 | 4644  | 824  | tty6 | Ss+ | 13:27 | 0:00 | /sbin/getty -8  |
| root     | 747 | 0.0 | 0.9 | 7796  | 2476 | ?    | Ss  | 13:27 | 0:00 | /usr/sbin/sshd  |
| root     | 748 | 0.0 | 0.3 | 3052  | 792  | ?    | Ss  | 13:27 | 0:00 | cron            |
| root     | 810 | 0.0 | 0.8 | 4400  | 2016 | tty1 | Ss  | 13:27 | 0:00 | /bin/login --   |
| root     | 823 | 0.0 | 0.0 | 0     | 0    | ?    | S   | 13:28 | 0:00 | [kauditd]       |
| student  | 840 | 0.0 | 1.2 | 6668  | 3008 | tty1 | S   | 13:28 | 0:00 | -bash           |
| root     | 855 | 0.0 | 0.0 | 0     | 0    | ?    | S   | 13:33 | 0:00 | [kworker/u2:1]  |
| student  | 865 | 0.0 | 0.4 | 5216  | 1164 | tty1 | R+  | 13:44 | 0:00 | ps aux          |

ps aux

6. How to define kernel processes and user processes?

If the process name in [], it means that it is a kernel process, otherwise it is a user process

7. Print the list of processes to the terminal. Briefly describe the statuses of the processes. What condition are they in, or can they be arriving in?

R – runnin, s – sleeping, z – zombie, s – stopped, d – uninterruptable sleep

8. Display only the processes of a specific user.

```
student@CsnKhali:~$ ps -u student
  PID TTY          TIME CMD
   840 tty1        00:00:00 bash
   867 tty1        00:00:00 ps
```

9. What utilities can be used to analyze existing running tasks (by analyzing the help for the ps command)?

## Pc, top, proc

10. What information does top command display?

```
top - 10:12:21 up 4 min, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 66 total, 1 running, 61 sleeping, 2 stopped, 2 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 247792 total, 99692 used, 148100 free, 11704 buffers
KiB Swap: 0 total, 0 used, 0 free. 64724 cached Mem
```

| PID | USER | PR | NI  | VIRT | RES  | SHR  | S | %CPU | %MEM | TIME+   | COMMAND      |
|-----|------|----|-----|------|------|------|---|------|------|---------|--------------|
| 1   | root | 20 | 0   | 4196 | 2176 | 1392 | S | 0.0  | 0.9  | 0:00.56 | init         |
| 2   | root | 20 | 0   | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kthreadd     |
| 3   | root | 20 | 0   | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | ksoftirqd/0  |
| 4   | root | 20 | 0   | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kworker/0:0  |
| 5   | root | 0  | -20 | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kworker/0:0H |
| 6   | root | 20 | 0   | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.10 | kworker/u2:0 |
| 7   | root | 20 | 0   | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.36 | rcu_sched    |
| 8   | root | 20 | 0   | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | rcu_bh       |
| 9   | root | rt | 0   | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | migration/0  |
| 10  | root | rt | 0   | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | watchdog/0   |
| 11  | root | 0  | -20 | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | khelper      |
| 12  | root | 20 | 0   | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kdevtmpfs    |
| 13  | root | 0  | -20 | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | netns        |
| 14  | root | 0  | -20 | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | writeback    |
| 15  | root | 0  | -20 | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kintegrityd  |
| 16  | root | 0  | -20 | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | bioset       |
| 17  | root | 0  | -20 | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kworker/u3:0 |
| 18  | root | 0  | -20 | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kblockd      |
| 19  | root | 0  | -20 | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | ata_sff      |
| 20  | root | 20 | 0   | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | khubd        |
| 21  | root | 0  | -20 | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | md           |
| 22  | root | 0  | -20 | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | devfreq_wq   |
| 23  | root | 20 | 0   | 0    | 0    | 0    | S | 0.0  | 0.0  | 0:00.11 | kworker/0:1  |

It displays PID, User, Load Averages, CPU and memory usage, commands...

11. Display the processes of the specific user using the top command.

```
top - 10:18:01 up 9 min, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 65 total, 1 running, 59 sleeping, 3 stopped, 2 zombie
%Cpu(s): 0.1 us, 0.3 sy, 0.0 ni, 99.5 id, 0.1 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 247792 total, 100076 used, 147716 free, 11728 buffers
KiB Swap: 0 total, 0 used, 0 free. 64732 cached Mem
```

| PID | USER    | PR | NI | VIRT | RES  | SHR  | S | %CPU | %MEM | TIME+   | COMMAND |
|-----|---------|----|----|------|------|------|---|------|------|---------|---------|
| 848 | student | 20 | 0  | 6668 | 3016 | 1652 | S | 0.0  | 1.2  | 0:00.03 | bash    |
| 862 | student | 20 | 0  | 4560 | 960  | 832  | T | 0.0  | 0.4  | 0:00.14 | info    |
| 864 | student | 20 | 0  | 0    | 0    | 0    | Z | 0.0  | 0.0  | 0:00.00 | man     |
| 866 | student | 20 | 0  | 4560 | 960  | 832  | T | 0.0  | 0.4  | 0:00.01 | info    |
| 868 | student | 20 | 0  | 0    | 0    | 0    | Z | 0.0  | 0.0  | 0:00.00 | man     |
| 872 | student | 20 | 0  | 5420 | 1308 | 988  | T | 0.0  | 0.5  | 0:00.12 | top     |
| 877 | student | 20 | 0  | 5416 | 1280 | 968  | R | 0.0  | 0.5  | 0:00.00 | top     |

12. What interactive commands can be used to control the top command? Give a couple of examples.

**q** – quit

**k** – kill process

**r** – renice

**o** – add filter

13. Sort the contents of the processes window using various parameters (for example, the amount of processor time taken up, etc.)

**Shift + m** – CPU

**Shift + t** – time

**Shift + n** – PID

14. Concept of priority, what commands are used to set priority?

**nice -n value command**

15. Can I change the priority of a process using the top command? If so, how?

```
top - 10:37:08 up 28 min, 1 user, load average: 0.00, 0.00, 0.00
Threads: 68 total, 1 running, 61 sleeping, 4 stopped, 2 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 247792 total, 100576 used, 147216 free, 11728 buffers
KiB Swap: 0 total, 0 used, 0 free. 64732 cached Mem
```

| PID | USER    | PR | NI | VIRT | RES  | SHR  | S | %CPU | %MEM | TIME+   | COMMAND |
|-----|---------|----|----|------|------|------|---|------|------|---------|---------|
| 879 | student | 32 | 12 | 5420 | 1316 | 992  | R | 0.0  | 0.5  | 0:00.28 | top     |
| 877 | student | 20 | 0  | 5420 | 1316 | 992  | T | 0.0  | 0.5  | 0:00.28 | top     |
| 872 | student | 20 | 0  | 5420 | 1308 | 988  | T | 0.0  | 0.5  | 0:00.11 | top     |
| 868 | student | 20 | 0  | 0    | 0    | 0    | Z | 0.0  | 0.0  | 0:00.00 | man     |
| 866 | student | 20 | 0  | 4560 | 960  | 832  | T | 0.0  | 0.4  | 0:00.01 | info    |
| 864 | student | 20 | 0  | 0    | 0    | 0    | Z | 0.0  | 0.0  | 0:00.00 | man     |
| 862 | student | 20 | 0  | 4560 | 960  | 832  | T | 0.0  | 0.4  | 0:00.14 | info    |
| 848 | student | 20 | 0  | 6668 | 3016 | 1652 | S | 0.0  | 1.2  | 0:00.03 | bash    |

By using the key **r**

16. Examine the kill command. How to send with the kill command process control signal? Give an example of commonly used signals.

**kill [options] <pid>**

**SIGTERM (15)** – is used to ask a process to stop

**SIGKILL (9)** - is used to force a process to stop

**SIGHUP (1)** - is used to hang up a process

17. Commands jobs, fg, bg, nohup. What are they for? Use the sleep, yes command to demonstrate the process control mechanism with fg, bg.

**jobs** — display status of jobs in the current session

**fg** — run jobs in the foreground

**bg** — run jobs in the background

**nohup** - run a command immune to hangups, with output to a non-tty

## Part2

1. Check the implementability of the most frequently used OPENSSH commands in the MS Windows operating system. (Description of the expected result of the commands + screenshots: command – result should be presented)

2. Implement basic SSH settings to increase the security of the client-server connection (at least

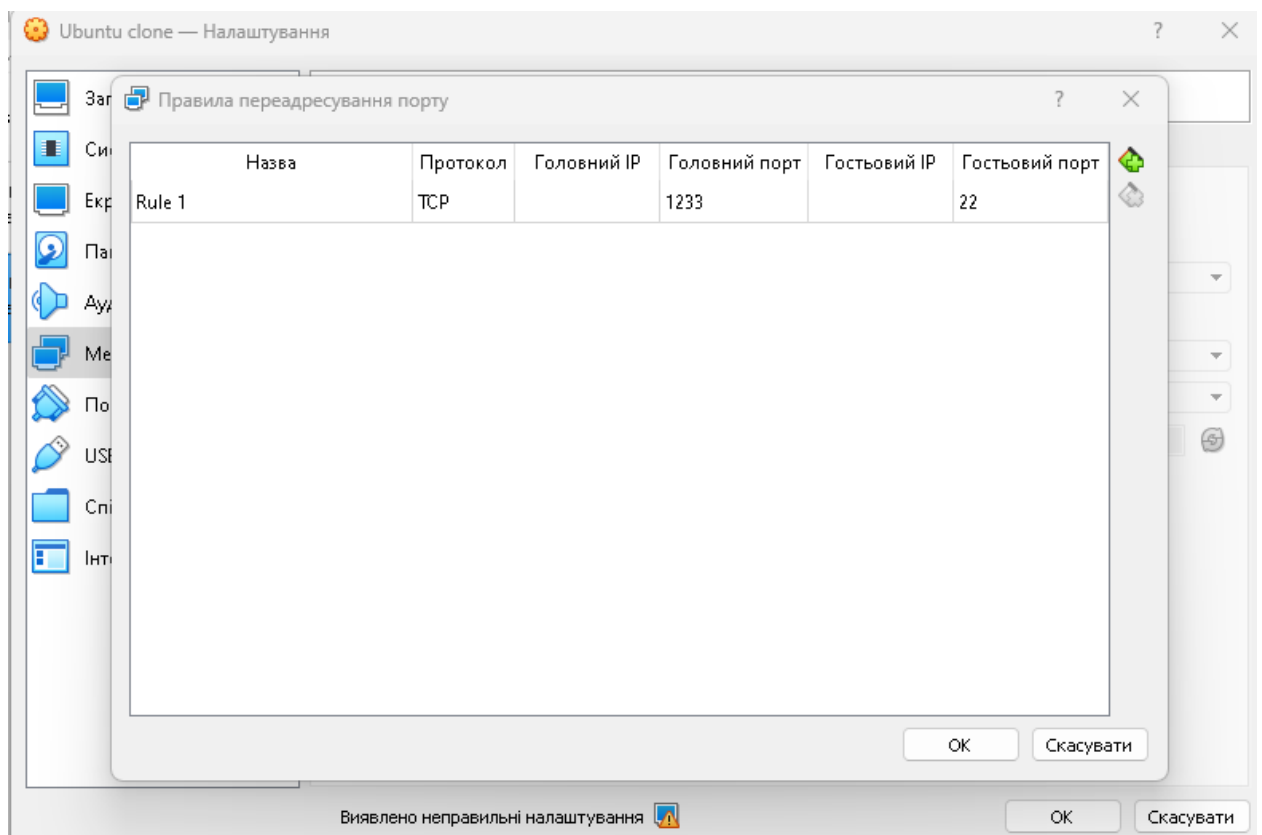
```
PermitRootLogin no
```

```
PasswordAuthentication no
```

3. List the options for choosing keys for encryption in SSH. Implement 3 of them.

```
# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
```

4. Implement port forwarding for the SSH client from the host machine to the guest Linux virtual machine behind NAT.



5\*. Intercept (capture) traffic (tcpdump, wireshark) while authorizing the remote client on the server using ssh, telnet, rlogin. Analyze the result.