

Task.Linux.3

Pashynskyi Maxim

Task3.Part1

1) How many states could have a process in Linux?

Running: The process is currently using the CPU and executing its instructions.

Waiting: The process is waiting for an event to occur before it can proceed.

Ready: The process is waiting to be assigned to a CPU. It's ready to execute, but the scheduler has not yet selected it to run.

Terminated: The process has finished executing or has been explicitly terminated by the user or another process.

Stopped: The process has been stopped, usually as a result of receiving a signal.

Zombie: A terminated process that is still listed in the process table. It's waiting for its parent process to collect its exit status.

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

```
student@CsnKhai:~$ pstree -s -p
init(1)---cron(758)
        |---dbus-daemon(361)
        |---dhclient(599)
        |---getty(719)
        |---getty(721)
        |---getty(724)
        |---getty(725)
        |---getty(727)
        |---irqbalance(760)
        |---login(821)---bash(847)
        |---rsyslogd(393)---{rsyslogd}(400)
        |                  |---{rsyslogd}(401)
        |                  |---{rsyslogd}(402)
        |---sshd(747)---sshd(867)---sshd(887)---bash(888)---pstree(945)
        |              |---sshd(869)---sshd(920)---sftp-server(921)
        |---systemd-logind(381)
        |---systemd-udevd(276)
        |---upstart-file-br(463)
        |---upstart-socket-(499)
        |---upstart-udev-br(272)
student@CsnKhai:~$
```

Here we can see that parent process **sshd(747)** created 2 separate subprocesses, one for establishing secure shell connection with bash

command interpreter and *ps*tree executed inside, and another to make *SFTP* connection.

3) What is a proc file system?

The */proc* file system in Linux is a virtual filesystem that provides an interface to kernel data structures and information about running processes. It's not a regular file system that stores files on disk, it's a way to expose system and process-related information as files and directories in a hierarchical structure.

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

```
student@CsnKhai:~$ inxi -C
CPU:      Dual core Intel Core i7-7700HQ CPU (-MCP-) cache: 6144 KB flags: (nx pae sse sse2 sse3 sse4_1 sse4_2 ssse3)
Clock Speeds: 1: 0.00 MHz 2: 0.00 MHz
student@CsnKhai:~$ lscpu
Architecture:        i686
CPU op-mode(s):      32-bit
Byte Order:          Little Endian
CPU(s):              2
On-line CPU(s) list: 0,1
Thread(s) per core:  1
Core(s) per socket:  2
Socket(s):           1
Vendor ID:           GenuineIntel
CPU family:          6
Model:               158
Stepping:            9
CPU MHz:             0.000
BogoMIPS:            6082.56
L1d cache:           32K
L1i cache:           32K
L2 cache:            256K
L3 cache:            6144K
student@CsnKhai:~$
```

```
student@CsnKhai:~$ cat /proc/cpuinfo | head -n 20
processor           : 0
vendor_id          : GenuineIntel
cpu family         : 6
model              : 158
model name         : Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz
stepping           : 9
microcode          : 0xffffffff
cpu MHz            : 0.000
cache size         : 6144 KB
physical id        : 0
siblings           : 2
core id            : 0
cpu cores          : 2
apicid             : 0
initial apicid     : 0
fdiv_bug           : no
f00f_bug           : no
coma_bug           : no
fpu                : yes
fpu_exception      : yes
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.

```
student@CsnKhai:~$ ps -efo user,pid,args,group
USER      PID COMMAND                                GROUP
student    888 -bash USER=student LOGNAME= student
student    2774 \_ ps -efo user,pid,args,g student
student    847 -bash TERM=linux HOME=/home student
student@CsnKhai:~$
```

- 6) How to define kernel processes and user processes?

Kernel processes highlighted with `[]`.

- 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?

```
student@CsnKhai:~$ ps ax
  PID TTY          STAT       TIME COMMAND
    1 ?           Ss          0:01 /sbin/init
    2 ?           S            0:00 [kthreadd]
    3 ?           S            0:00 [ksoftirqd/0]
    4 ?           S            0:00 [kworker/0:0]
    5 ?           S<          0:00 [kworker/0:0H]
    7 ?           S            0:00 [rcu_sched]
    8 ?           S            0:00 [rcu_bh]
    9 ?           S            0:00 [migration/0]
   10 ?           S            0:00 [watchdog/0]
   11 ?           S            0:00 [watchdog/1]
   12 ?           S            0:00 [migration/1]
   13 ?           R            0:01 [ksoftirqd/1]
   14 ?           S            0:00 [kworker/1:0]
   15 ?           S<          0:00 [kworker/1:0H]
   16 ?           S<          0:00 [khelper]
   17 ?           S            0:00 [kdevtmpfs]
   18 ?           S<          0:00 [netns]
```

The first letter of the value under the **STAT** column indicates the state that the process is in.

R: Running or Runnable;
S: Interruptible Sleep;
D: Uninterruptible Sleep;
T: Stopped;
Z: Zombie.

8) Display only the processes of a specific user.

```
student@CsnKhai:~$ ps -u student
  PID TTY          TIME CMD
  847 tty1        00:00:00 bash
  887 ?            00:00:00 sshd
  888 pts/0        00:00:00 bash
  920 ?            00:00:00 sshd
  921 ?            00:00:00 sftp-server
 2759 pts/0        00:00:00 ps
student@CsnKhai:~$
```

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

ps aux or *-ef*: displays a detailed list of all running processes for all users.

--sort: sorts processes by some column (*%cpu*, *%mem*, etc).

ps -eo: allows you to specify the columns you want to display.

--forest: displays a hierarchical tree view of processes, showing parent-child relationships between processes.

ps -C: lists all instances of the specified process.

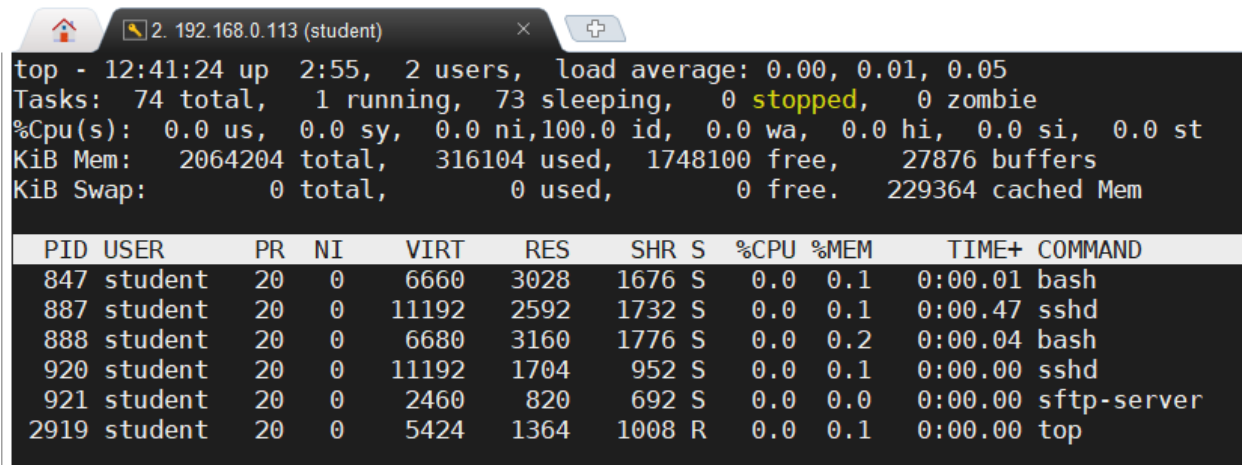
etc...

10) What information does the top command display?

Detailed process information and usage of system resources (CPU, Memory, etc.).

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

```
student@CsnKhai:~$ top -u student
```



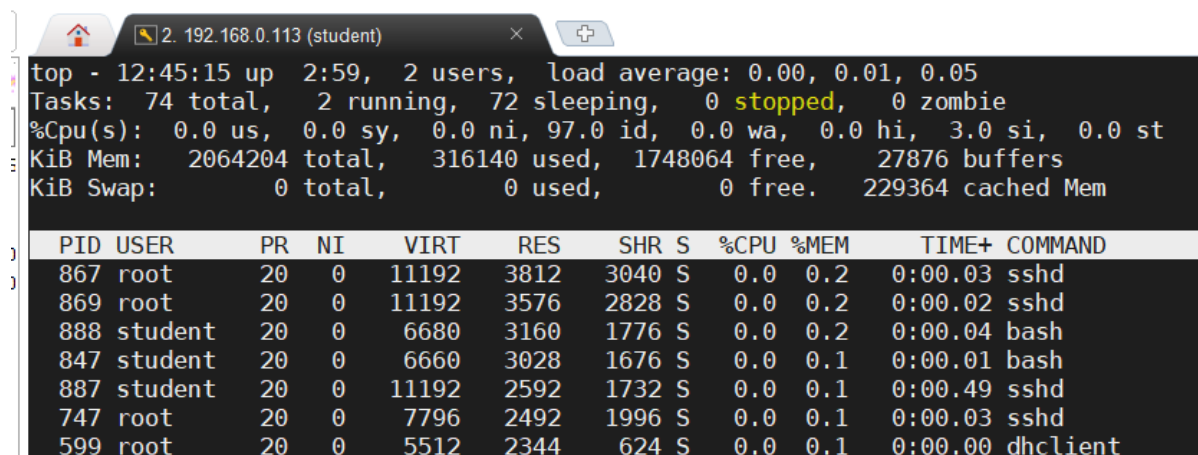
top - 12:41:24 up 2:55, 2 users, load average: 0.00, 0.01, 0.05
Tasks: 74 total, 1 running, 73 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 2064204 total, 316104 used, 1748100 free, 27876 buffers
KiB Swap: 0 total, 0 used, 0 free. 229364 cached Mem

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
847	student	20	0	6660	3028	1676	S	0.0	0.1	0:00.01	bash
887	student	20	0	11192	2592	1732	S	0.0	0.1	0:00.47	sshd
888	student	20	0	6680	3160	1776	S	0.0	0.2	0:00.04	bash
920	student	20	0	11192	1704	952	S	0.0	0.1	0:00.00	sshd
921	student	20	0	2460	820	692	S	0.0	0.0	0:00.00	sftp-server
2919	student	20	0	5424	1364	1008	R	0.0	0.1	0:00.00	top

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

Sorting by columns:

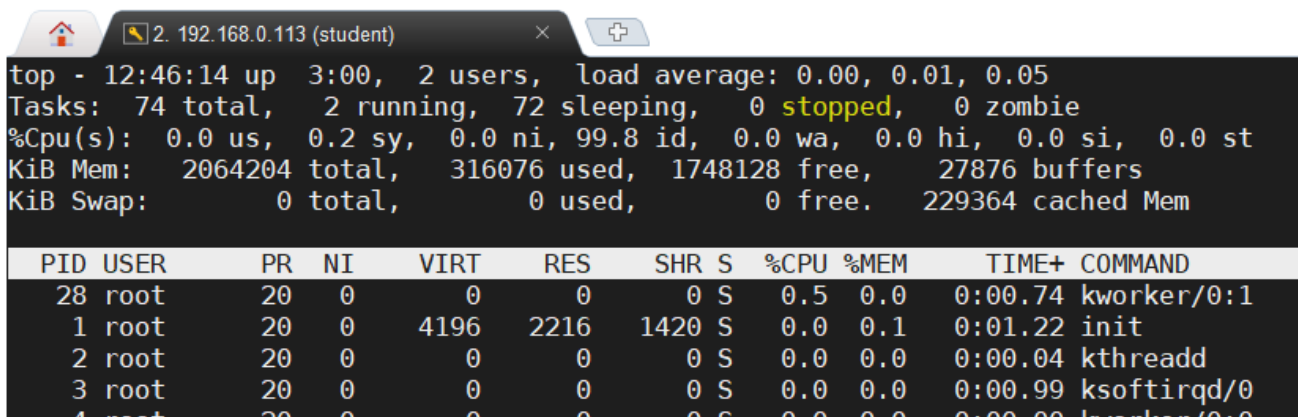
M (sort by mem usage)



top - 12:45:15 up 2:59, 2 users, load average: 0.00, 0.01, 0.05
Tasks: 74 total, 2 running, 72 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 97.0 id, 0.0 wa, 0.0 hi, 3.0 si, 0.0 st
KiB Mem: 2064204 total, 316140 used, 1748064 free, 27876 buffers
KiB Swap: 0 total, 0 used, 0 free. 229364 cached Mem

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
867	root	20	0	11192	3812	3040	S	0.0	0.2	0:00.03	sshd
869	root	20	0	11192	3576	2828	S	0.0	0.2	0:00.02	sshd
888	student	20	0	6680	3160	1776	S	0.0	0.2	0:00.04	bash
847	student	20	0	6660	3028	1676	S	0.0	0.1	0:00.01	bash
887	student	20	0	11192	2592	1732	S	0.0	0.1	0:00.49	sshd
747	root	20	0	7796	2492	1996	S	0.0	0.1	0:00.03	sshd
599	root	20	0	5512	2344	624	S	0.0	0.1	0:00.00	dhclient

P (sort by CPU usage)



top - 12:46:14 up 3:00, 2 users, load average: 0.00, 0.01, 0.05
Tasks: 74 total, 2 running, 72 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.2 sy, 0.0 ni, 99.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 2064204 total, 316076 used, 1748128 free, 27876 buffers
KiB Swap: 0 total, 0 used, 0 free. 229364 cached Mem

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
28	root	20	0	0	0	0	S	0.5	0.0	0:00.74	kworker/0:1
1	root	20	0	4196	2216	1420	S	0.0	0.1	0:01.22	init
2	root	20	0	0	0	0	S	0.0	0.0	0:00.04	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.99	ksoftirqd/0
4	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kworker/0:0

Filtering by user:

```
top - 12:47:19 up 3:01, 2 users, load average: 0.00, 0.01, 0.05
Tasks: 75 total, 2 running, 73 sleeping, 0 stopped, 0 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: 2064204 total, 316052 used, 1748152 free, 27876 buffers
KiB Swap: 0 total, 0 used, 0 free, 229364 cached Mem

Which user (blank for all) student
  PID USER      PR  NI   VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    847 student    20   0   6660   3028   1676 S   0.0   0.1   0:00.01 bash
    887 student    20   0  11192   2592   1732 S   0.0   0.1   0:00.52 sshd
    888 student    20   0   6680   3160   1776 S   0.0   0.2   0:00.04 bash
    920 student    20   0  11192   1704    952 S   0.0   0.1   0:00.00 sshd
    921 student    20   0   2460    820    692 S   0.0   0.0   0:00.00 sftp-server
   2921 student    20   0   5420   1452   1084 R   0.0   0.1   0:00.00 top
```

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

Sorting by process execution time (T):

```
top - 12:49:44 up 3:03, 2 users, load average: 0.00, 0.01, 0.05
Tasks: 75 total, 1 running, 74 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.2 sy, 0.0 ni, 99.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 2064204 total, 316116 used, 1748088 free, 27876 buffers
KiB Swap: 0 total, 0 used, 0 free, 229364 cached Mem

  PID USER      PR  NI   VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    29 root        20   0     0      0      0 S   0.0   0.0   0:06.19 kworker/1:1
    13 root        20   0     0      0      0 S   0.0   0.0   0:02.91 ksoftirqd/1
    53 root        20   0     0      0      0 S   0.0   0.0   0:02.10 kworker/u4:2
     1 root        20   0   4196   2216   1420 S   0.0   0.1   0:01.22 init
     3 root        20   0     0      0      0 S   0.0   0.0   0:01.00 ksoftirqd/0
   134 root        20   0     0      0      0 S   0.0   0.0   0:00.88 jbd2/sda1-8
    28 root        20   0     0      0      0 S   0.0   0.0   0:00.75 kworker/0:1
    887 student    20   0  11192   2592   1732 S   0.0   0.1   0:00.53 sshd
```

Sorting by virtual memory usage (V):

```
top - 12:52:50 up 3:06, 2 users, load average: 0.00, 0.01, 0.05
Tasks: 74 total, 1 running, 73 sleeping, 0 stopped, 0 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: 2064204 total, 316204 used, 1748000 free, 27876 buffers
KiB Swap: 0 total, 0 used, 0 free, 229364 cached Mem

  PID USER      PR  NI   VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
     1 root        20   0   4196   2216   1420 S   0.0   0.1   0:01.22 init
   272 root        20   0   3008    884    664 S   0.0   0.0   0:00.19 - upstart-udev-br
   276 root        20   0  12024   1420    980 S   0.0   0.1   0:00.09 - systemd-udev
   361 message+    20   0   4236    988    708 S   0.0   0.0   0:00.10 - dbus-daemon
   381 root        20   0   4212   1736   1428 S   0.0   0.1   0:00.00 - systemd-logind
   393 syslog     20   0  31368   1192    856 S   0.0   0.1   0:00.22 - rsyslogd
   463 root        20   0   3012    684    348 S   0.0   0.0   0:00.04 - upstart-file-br
   499 root        20   0   2868    592    436 S   0.0   0.0   0:00.06 - upstart-socket-
   599 root        20   0   5512   2344    624 S   0.0   0.1   0:00.00 - dhclient
   719 root        20   0   4644    832    720 S   0.0   0.0   0:00.00 - getty
   721 root        20   0   4644    836    720 S   0.0   0.0   0:00.00 - getty
   724 root        20   0   4644    832    720 S   0.0   0.0   0:00.00 - getty
   725 root        20   0   4644    832    720 S   0.0   0.0   0:00.00 - getty
   727 root        20   0   4644    840    720 S   0.0   0.0   0:00.00 - getty
   747 root        20   0   7796   2492   1996 S   0.0   0.1   0:00.03 - sshd
```

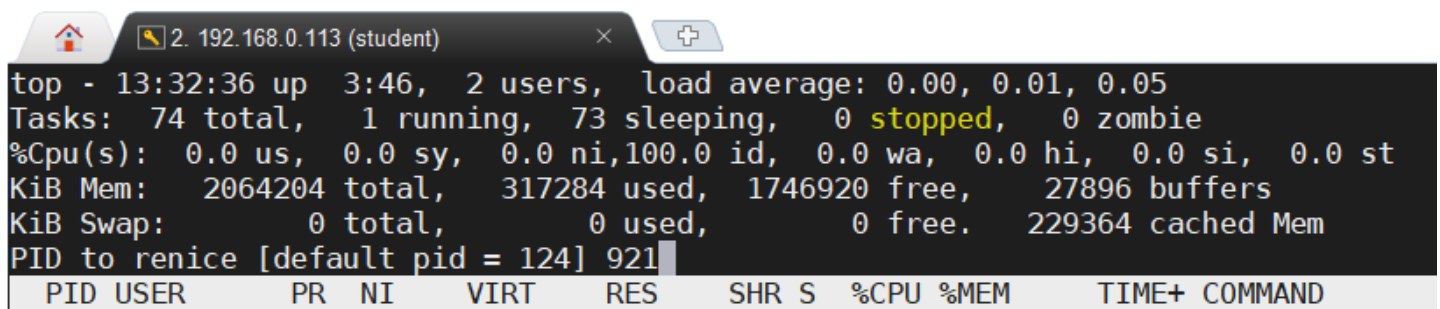

14) Concept of priority, what commands are used to set priority?

The concept of priority determines the order in which the kernel schedules processes to run on the CPU. A higher priority value typically results in a process being scheduled more frequently and receiving *more CPU time*, while a lower priority value means the process is scheduled less frequently and may have *less CPU time* allocated.

To set the priority of the process commands *nice*, *renice* and *top/htop* can be used.

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

Yes, by pressing *r* key, entering *PID* and *nice*ness:

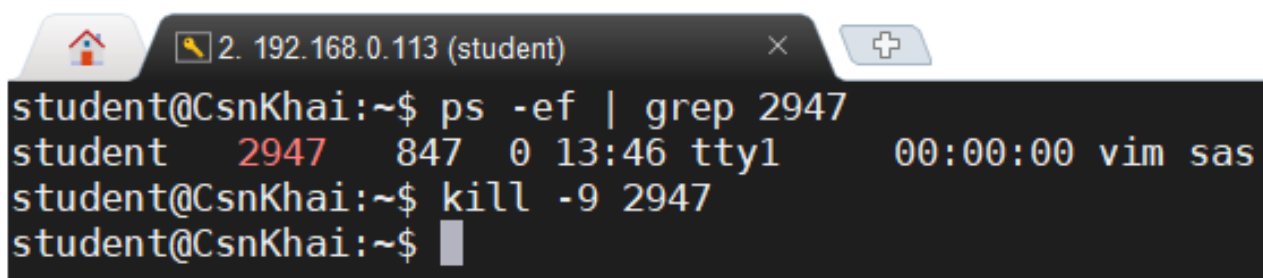


```
top - 13:32:36 up 3:46, 2 users, load average: 0.00, 0.01, 0.05
Tasks: 74 total, 1 running, 73 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 2064204 total, 317284 used, 1746920 free, 27896 buffers
KiB Swap: 0 total, 0 used, 0 free. 229364 cached Mem
PID to renice [default pid = 124] 921
PID USER      PR  NI  VIRT  RES  SHR  S   %CPU  %MEM   TIME+  COMMAND
124 root        20   0    0     0   0     0  S    0.0   0.0   0:00  sleep
```

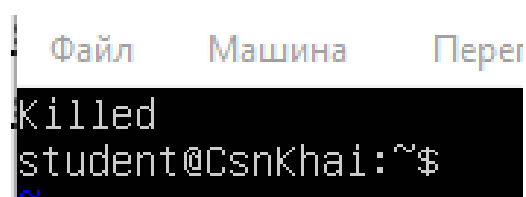
16) Examine the kill command. How to send with the kill command process control signal? Give an example of commonly used signals.

Some commonly used signals are:

SIGKILL (9) to immediately terminate a process:



```
student@CsnKhai:~$ ps -ef | grep 2947
student  2947  847  0 13:46 tty1      00:00:00 vim sas
student@CsnKhai:~$ kill -9 2947
student@CsnKhai:~$
```



```
Файл  Машина  Перег
Killed
student@CsnKhai:~$
```

SIGTERM (15) to ask a process to exit:

```
student@CsnKhai:~$ ps -ef | grep sigkill
student 2956 847 0 13:49 tty1 00:00:00 vim sigkill
student 2958 888 0 13:49 pts/0 00:00:00 grep --color=auto sigkill
student@CsnKhai:~$ kill -15 2956
student@CsnKhai:~$
```

```
Vim: Caught deadly signal TERM
Vim: Finished.
Terminated
student@CsnKhai:~$
```

Also, there are **SIGSTOP (19)**, **SIGINT (2)**, **SIGHUP (1)**, etc.

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: displays a list of currently running jobs (background processes) associated with the current shell session.

fg: is used to bring a background job into the foreground. It resumes a suspended or backgrounded job and makes it the active process in the terminal.

bg: is used to move a suspended job to the background, allowing it to continue running while freeing up the terminal for other commands.

nohup: is used to run a command in the background and ignore the SIGHUP signal. This allows a process to continue running even after you close the terminal session.

Let's run a **sleep** command in the background:

```
student@CsnKhai:~$ nohup sleep 300 &
[1] 2968
student@CsnKhai:~$ nohup: ignoring input and appending output to 'nohup.out'

student@CsnKhai:~$ ps
  PID TTY          TIME CMD
   888 pts/0        00:00:00 bash
  2968 pts/0        00:00:00 sleep
  2969 pts/0        00:00:00 ps
student@CsnKhai:~$
```


See the *jobs*:

```
student@CsnKhai:~$ jobs
[1]+  Running                  nohup sleep 300 &
student@CsnKhai:~$
```

Move it to the foreground with *fg* and suspend it with *ctrl+Z*:

```
student@CsnKhai:~$ fg %1
nohup sleep 300
^Z
[1]+  Stopped                  nohup sleep 300
student@CsnKhai:~$
```

And moving it back to the background with *bg*:

```
student@CsnKhai:~$ fg %1
nohup sleep 300
^Z
[1]+  Stopped                  nohup sleep 300
student@CsnKhai:~$ bg
[1]+ nohup sleep 300 &
student@CsnKhai:~$
```

Task3.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).

Standard *ssh* connection to VM using CMD:

```
C:\Users\Win10>ssh student@192.168.0.113
The authenticity of host '192.168.0.113 (192.168.0.113)' can't be established.
ECDSA key fingerprint is SHA256:yp8INOs6pk/gVv7G84N/cRT3KsgxLPiH81jZ/cRpz0o.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.113' (ECDSA) to the list of known hosts.
student@192.168.0.113's password:
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-63-generic i686)

* Documentation:  https://help.ubuntu.com/
Last login: Fri Aug 18 09:46:31 2023 from 192.168.0.108
student@CsnKhai:~$
```

Generating *key pair*:

```
C:\Windows\system32>ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\Win10\.ssh/id_rsa): C:\Users\Win10\Desktop\id_rsa
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\Win10\Desktop\id_rsa.
Your public key has been saved in C:\Users\Win10\Desktop\id_rsa.pub.
The key fingerprint is:
SHA256:M5m1eNM3Mc9Cx/lr4Uo9jP2huvv7xS8jA7PMdyxKOTY win10@DESKTOP-JT48RV1
The key's randomart image is:
+----[RSA 3072]-----+
|
| . .
| . + +
| * . * .
| . S . + o..
| . +oo o*.o|
| oE+ + Xo|
| o+o* 0 *|
| .*=Xo+o|
+-----[SHA256]-----+
```

Trying to transfer *key pair* to VM (only manual transfer is possible):

```
C:\Windows\system32>ssh-copy-id student@192.168.0.113
'ssh-copy-id' is not recognized as an internal or external command,
operable program or batch file.

C:\Windows\system32>
```

2) Implement basic SSH settings to increase the security of the client-server connection.

Establishing authentication without password (*from host WSL*):

```
max@DESKTOP-JT48RV1:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/max/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/max/.ssh/id_rsa
Your public key has been saved in /home/max/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:94Xy3xLf22beo/1CPjMoLacSlJeF4utb35751T25+qI max@DESKTOP-JT48RV1
The key's randomart image is:
+----[RSA 3072]-----+
|
| . . .
| . o o
| + o .
| .Soo . .
| o. + .o +
| . . .o+ ==
| o.o.++o=@
| .o.Eo+@/@@
+-----[SHA256]-----+
max@DESKTOP-JT48RV1:~$
```

```

max@DESKTOP-JT48RV1: ~
max@DESKTOP-JT48RV1:~$ ssh-copy-id student@192.168.0.113
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/max/.ssh/id_rsa.pub"
The authenticity of host '192.168.0.113 (192.168.0.113)' can't be established.
ECDSA key fingerprint is SHA256:yp8IN0s6pk/gVv7G84N/cRT3KsgxLPiH81jZ/cRpz0o.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
student@192.168.0.113's password:

Number of key(s) added: 1

Now try logging into the machine, with:  "ssh 'student@192.168.0.113'"
and check to make sure that only the key(s) you wanted were added.

max@DESKTOP-JT48RV1:~$

```

```

max@DESKTOP-JT48RV1:~$ ssh student@192.168.0.113
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-63-generic i686)

 * Documentation:  https://help.ubuntu.com/
Last login: Fri Aug 18 17:35:49 2023
student@CsnKhai:~$ w
 17:47:21 up 12 min,  2 users,  load average: 0.00, 0.01, 0.03
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT
student   tty1                    17:35    11:33   0.02s   0.00s  -bash
student   pts/0    192.168.0.108    17:47    1.00s   0.01s   0.00s  w
student@CsnKhai:~$

```

Disabling root authentication:

```

# Authentication:
LoginGraceTime 120
PermitRootLogin no
StrictModes yes

```

```

# Authentication:
LoginGraceTime 120
PermitRootLogin no
StrictModes yes
AllowUsers student

```

```

student@CsnKhai:~$ sudo service ssh restart
ssh stop/waiting
ssh start/running, process 990
student@CsnKhai:~$ sudo service ssh status
ssh start/running, process 990
student@CsnKhai:~$

```

```

student@CsnKhai: ~
max@DESKTOP-JT48RV1:~$ ssh root@192.168.0.113
root@192.168.0.113's password:
Permission denied, please try again.
root@192.168.0.113's password:
Permission denied, please try again.
root@192.168.0.113's password:
root@192.168.0.113: Permission denied (publickey,password).

```

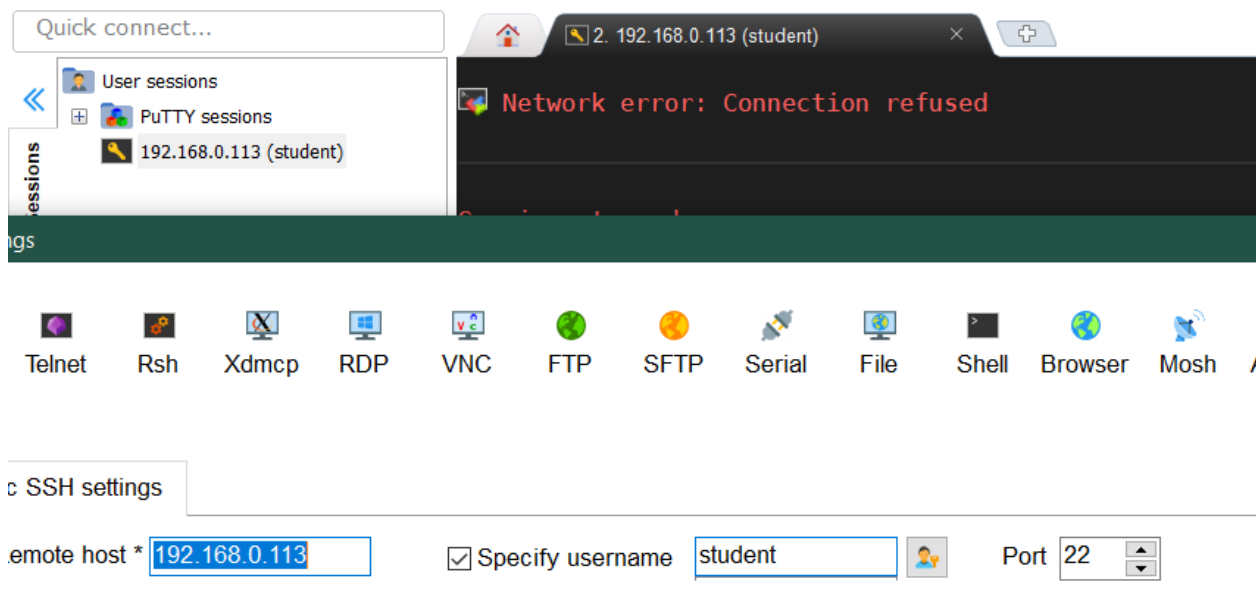
Disabling password based auth:

```
PasswordAuthentication no  
RSAAuthentication yes
```

```
max@DESKTOP-JT48RV1: ~  
max@DESKTOP-JT48RV1:~$ ssh root@192.168.0.113  
root@192.168.0.113: Permission denied (publickey).  
max@DESKTOP-JT48RV1:~$
```

Changing *ssh port* from 22 to 6669:

```
# What ports, IPs and protocols we listen for  
Port 6669  
# Use these options to restrict which interface
```

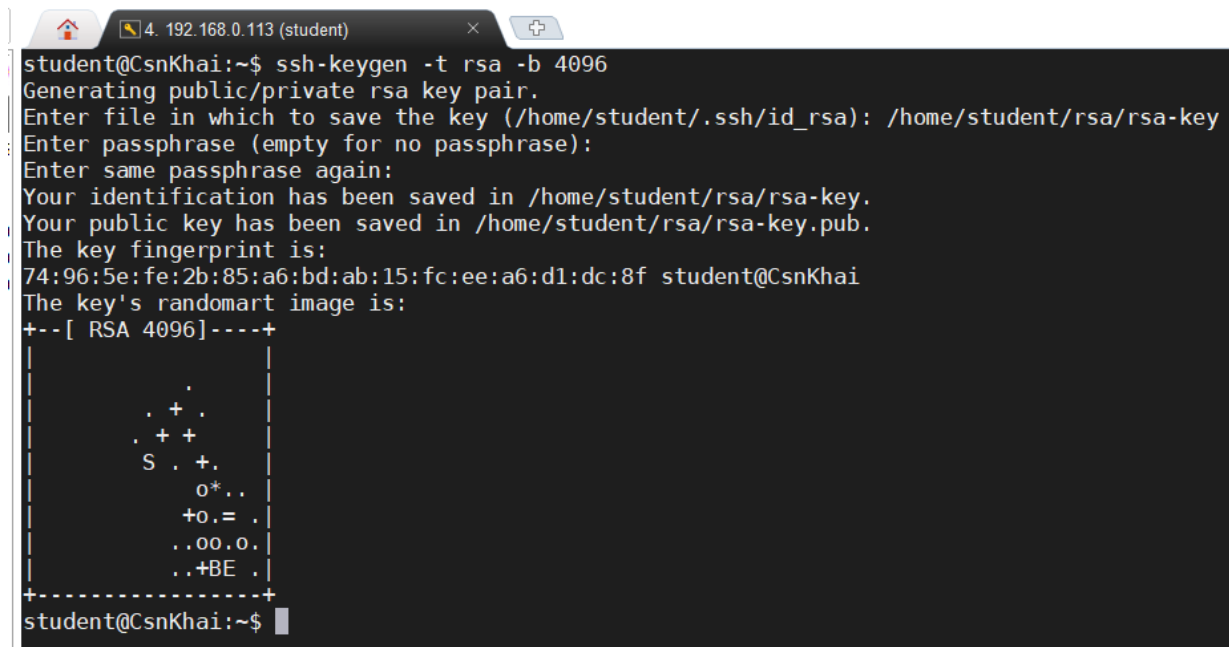


```
max@DESKTOP-JT48RV1:~/.ssh$ ssh student@192.168.0.113  
ssh: connect to host 192.168.0.113 port 22: Connection refused  
max@DESKTOP-JT48RV1:~/.ssh$ ssh student@192.168.0.113 -p 6669  
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-63-generic i686)  
  
* Documentation:  https://help.ubuntu.com/  
Last login: Fri Aug 18 18:01:44 2023 from 192.168.0.108  
student@CsnKhai:~$
```


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

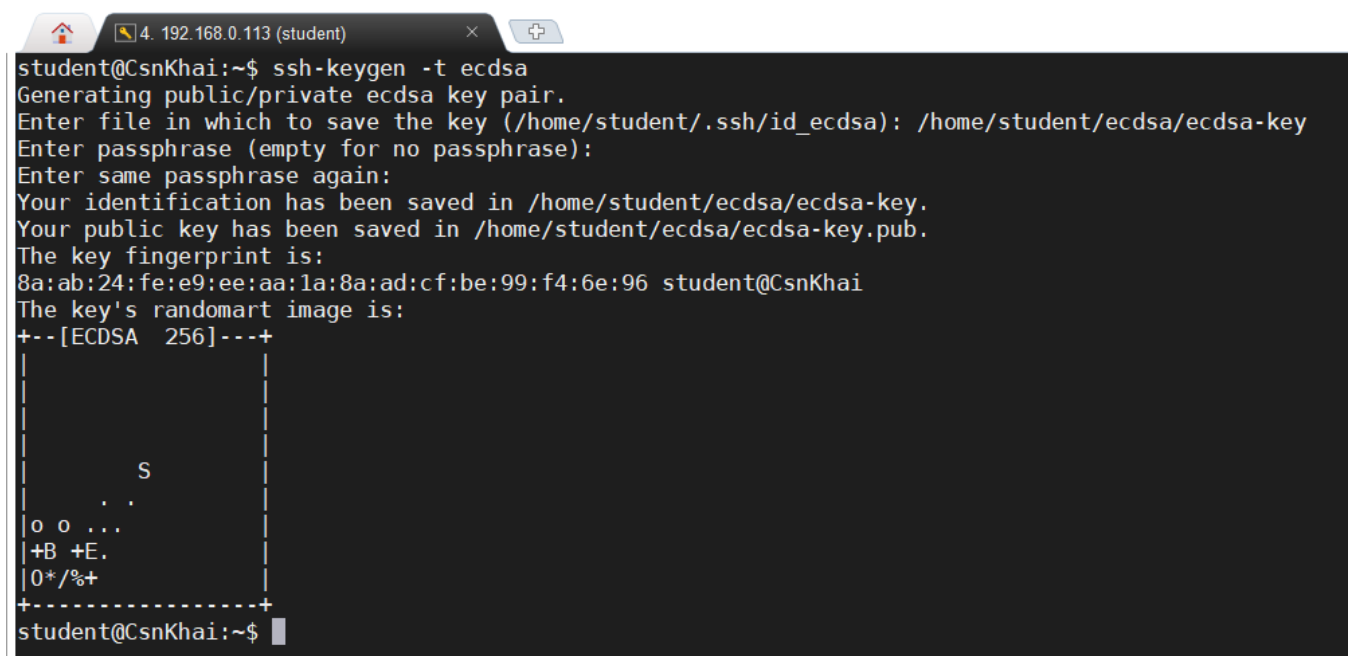
There are many cryptographic algorithms used for encryption of ssh connections. Here are three of them:

- **RSA:** one of the most widely used encryption algorithms for SSH key pairs. It's known for its security and versatility:

A terminal window titled '4. 192.168.0.113 (student)' showing the execution of 'ssh-keygen -t rsa -b 4096'. The output includes prompts for file location, passphrase, and confirmation, followed by the key fingerprint and a randomart image for the RSA 4096 key pair.

```
student@CsnKhai:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/student/.ssh/id_rsa): /home/student/rsa/rsa-key
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/student/rsa/rsa-key.
Your public key has been saved in /home/student/rsa/rsa-key.pub.
The key fingerprint is:
74:96:5e:fe:2b:85:a6:bd:ab:15:fc:ee:a6:d1:dc:8f student@CsnKhai
The key's randomart image is:
+---[ RSA 4096]-----+
|          .          |
|       . + .         |
|      .+ +          |
|     S . +.         |
|          o*..       |
|         +o.= .      |
|        ..oo.o.     |
|       ..+BE .      |
+-----+-----+
student@CsnKhai:~$
```

- **ECDSA:** Elliptic Curve Digital Signature Algorithm, another option for generating SSH key pairs. It uses elliptic curve cryptography and provides a good balance between security and key size:

A terminal window titled '4. 192.168.0.113 (student)' showing the execution of 'ssh-keygen -t ecdsa'. The output includes prompts for file location, passphrase, and confirmation, followed by the key fingerprint and a randomart image for the ECDSA 256 key pair.

```
student@CsnKhai:~$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/student/.ssh/id_ecdsa): /home/student/ecdsa/ecdsa-key
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/student/ecdsa/ecdsa-key.
Your public key has been saved in /home/student/ecdsa/ecdsa-key.pub.
The key fingerprint is:
8a:ab:24:fe:e9:ee:aa:1a:8a:ad:cf:be:99:f4:6e:96 student@CsnKhai
The key's randomart image is:
+---[ECDSA 256]---+
|          S          |
|       . . .         |
|    o o . . .        |
| +B +E.             |
| 0*/%+              |
+-----+-----+
student@CsnKhai:~$
```

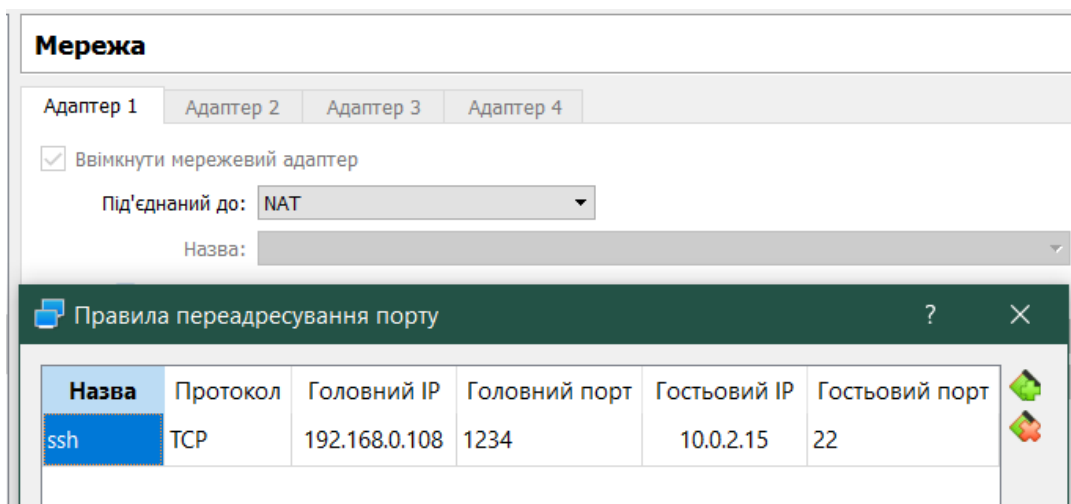
- **ED25519:**

```

student@CsnKhai:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/student/.ssh/id_ed25519): /home/student/ed/ed-key
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/student/ed/ed-key.
Your public key has been saved in /home/student/ed/ed-key.pub.
The key fingerprint is:
a8:e6:41:53:da:a4:2f:79:db:be:95:97:85:71:03:06 student@CsnKhai
The key's randomart image is:
+--[ED25519 256]--+
|      E.o         |
|      . .         |
|      . o         |
|    * . . + .     |
|   = o S . .      |
|  . = . o         |
| * o   o o        |
| o + o . .        |
|  . .+.          |
+-----+
student@CsnKhai:~$

```

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



```

4. 192.168.0.108 (student)
• MobaXterm 12.2 •
(SSH client, X-server and networking tools)

> SSH session to student@192.168.0.108
• SSH compression : ✓
• SSH-browser      : ✓
• X11-forwarding   : ✓ (remote display is forwarded through SSH)
• DISPLAY          : ✓ (automatically set on remote server)

> For more info, ctrl+click on help or visit our website

Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-63-generic i686)

* Documentation: https://help.ubuntu.com/
Last login: Fri Aug 18 18:37:19 2023
student@CsnKhai:~$

```