TASK5.3

Part1

1. How many states could has a process in Linux?
   Generally, a process can have the following states:
   - Running: The process is actively executing and using the CPU.
   - Sleeping (Interruptible): The process is waiting for an event to occur. It can be woken up by receiving a signal or an interrupt. For example, a process waiting for input from a user falls into this state.
   - Sleeping (Uninterruptible): Similar to the sleeping state, but the process cannot be woken up by signals. It's often waiting for some I/O operation to complete, and it's considered to be in a more critical state than regular sleeping.
   - Stopped: The process has been stopped, often by a user or a debugger. It can be restarted using the bg (background) or fg (foreground) command.
   - Zombie: The process has completed its execution, but its exit status hasn't been collected by its parent process yet. It remains in this state until its parent acknowledges its termination, at which point it is removed from the process table.
2. Examine the pstree command. Make output (highlight) the chain (ancestors) of the current process.

```
student@CsnKhai:~$ echo $$
885
student@CsnKhai:~$ pstree -s -p | grep --color=auto -E "885|$"
init(1)-+-cron(715)
        |-dbus-daemon(340)
        |-dhclient(485)
        |-getty(653)
        |-getty(655)
        |-getty(658)
        |-getty(659)
        |-getty(661)
        |-login(796)---bash(837)
        |-rsyslogd(349)-+-{rsyslogd}(350)
        |               |-{rsyslogd}(351)
        |               `-{rsyslogd}(352)
        |-sshd(699)-+-sshd(850)---sshd(883)---bash(885)-+-grep(4666)
        |           |                                   `-pstree(4665)
        |           `-sshd(852)---sshd(892)---sftp-server(893)
        |-systemd-logind(360)
        |-systemd-udevd(255)
        |-upstart-file-br(377)
        |-upstart-socket-(478)
        `-upstart-udev-br(249)
```

3. What is a proc file system?
   Proc file system (procfs) is a virtual file system created on the fly when the system boots and is dissolved at the time of system shutdown. It contains useful information about the processes that are currently running, it is regarded as a control and information center for the kernel.
4. Print information about the processor (its type, supported technologies, etc.).

```
student@CsnKhai:~$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 158
model name      : Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz
stepping        : 10
microcode       : 0xffffffff
cpu MHz         : 0.000
cache size      : 9216 KB
physical id     : 0
siblings        : 1
core id         : 0
cpu cores       : 1
apicid          : 0
initial apicid  : 0
fdiv_bug        : no
f00f_bug        : no
coma_bug        : no
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 mmx fxsr sse sse2 ht nx rdtscp constant_tsc xtopolog
y nonstop_tsc pni pclmulqdq ssse3 cx16 pcid sse4_1 sse4_2 movbe popcnt aes rdrand lahf_lm abm 3dnowprefetch fsgsbase bmi1 bmi2 invpcid rdseed
bogomips        : 9187.32
clflush size    : 64
cache_alignment : 64
address sizes   : 39 bits physical, 48 bits virtual
power management:
```

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 -o  user,args,group
USER     COMMAND                     GROUP
student  -bash                       student
```

6. How to define kernel processes and user processes?

Kernel processes are processes that are managed by the kernel itself. They are responsible for performing critical system-level tasks and managing the operation of the operating system. Kernel processes operate in a privileged mode with access to kernel memory and resources.

User processes are processes that are created and managed by users or user-level applications. These processes perform tasks and run applications that interact with users and provide various services. User processes run in user mode, which is a restricted mode that doesn't have direct access to kernel memory or resources.

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 -eo  user,args,stat,grou
USER     COMMAND                     STAT GROU
root     /sbin/init                  Ss   root
root     [kthreadd]                  S    root
root     [ksoftirqd/0]               S    root
root     [kworker/0:0H]              S<   root
root     [rcu_sched]                 S    root
root     [rcu_bh]                    S    root
root     [migration/0]               S    root
root     [watchdog/0]                S    root
root     [khelper]                   S<   root
root     [kdevtmpfs]                 S    root
root     [netns]                     S<   root
root     [writeback]                 S<   root
root     [kintegrityd]               S<   root
root     [bioset]                    S<   root
root     [kworker/u3:0]              S<   root
root     [kblockd]                   S<   root
root     [ata_sff]                   S<   root
root     [khubd]                     S    root
root     [md]                        S<   root
root     [devfreq_wq]                S<   root
root     [kworker/0:1]               S    root
root     [khungtaskd]                S    root
root     [kswapd0]                   S    root
root     [ksmd]                      SN   root
root     [fsnotify_mark]             S    root
root     [ecryptfs-kthrea]           S    root
root     [crypto]                    S<   root
root     [kthrotld]                  S<   root
root     [scsi_eh_0]                 S    root
root     [scsi_eh_1]                 S    root
root     [deferwq]                   S<   root
root     [charger_manager]           S<   root
```

These statuses provide information about the current condition or state of each process. Here's a summary of common process statuses and what they indicate:

- R (Running): The process is actively executing and using the CPU. It's in a running state and is ready for execution.
- S (Sleeping): The process is in a sleeping state, waiting for an event or condition to occur. It's temporarily inactive but can be woken up by signals or events.
- D (Uninterruptible Sleep): Similar to the sleeping state (S), but the process cannot be interrupted by signals. It's often waiting for I/O operations to complete or other kernel-related tasks.
- T (Stopped): The process has been stopped, often by a user or a debugger. It's suspended and can be resumed using the bg or fg commands.
- Z (Zombie): The process has completed its execution, but its exit status hasn't been collected by its parent process. It remains in this state until its parent acknowledges its termination.
- I (Idle): The process is idle, meaning it's not performing any active tasks. This status is often seen with kernel processes.

- < (High-Priority): The process has a higher priority compared to other processes. This status is not present on all systems.
- N (Low-Priority): The process has a lower priority compared to other processes. This status is not present on all systems.

Processes can transition between these states based on their execution, I/O operations, and interactions with the kernel and other processes.

8. Display only the processes of a specific user.

```
student@CsnKhai:~$ ps -u student
  PID TTY          TIME CMD
  883 ?        00:00:07 sshd
  885 pts/0    00:00:00 bash
  892 ?        00:00:00 sshd
  893 ?        00:00:00 sftp-server
 4702 pts/0    00:00:00 ps
```

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

The “ps” command has several options that allow you to customize the output and focus on specific information about running tasks. Some useful options include:

“-aux”: Displays all processes with detailed information.

“-ef”: Displays a full listing of processes.

“-l”: Long format, providing more detailed information.

10. What information does top command display?

The “top” command is a powerful and interactive utility that displays real-time information about system processes and resource usage. When you run the “top” command in the terminal, it provides a dynamic view of various system statistics and process details.

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

```
student@CsnKhai:~$ top -u student
top - 20:45:16 up  9:47,  2 users,  load average: 0.00, 0.01, 0.05
Tasks:  65 total,   1 running,  64 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.7 sy,  0.0 ni, 99.3 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:    247792 total,   237356 used,    10436 free,    50816 buffers
KiB Swap:        0 total,        0 used,        0 free.   103132 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
  883 student   20   0   11192   2332   1540 S  0.3  0.9   0:07.67 sshd
 4703 student   20   0    5428   1368   1008 R  0.3  0.6   0:00.03 top
  885 student   20   0    6668   3052   1692 S  0.0  1.2   0:00.13 bash
  892 student   20   0   11192   1708    960 S  0.0  0.7   0:00.00 sshd
  893 student   20   0    2460    624    528 S  0.0  0.3   0:00.00 sftp-server
```

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

“h”,”?”. Typing “h” or “?” on that help screen takes you to help for those interactive commands applicable to alternate-display mode.

“=”. This command reverses any 'i' (idle tasks) and 'n' (max tasks) commands that might be active. It also provides for an 'exit' from pid monitoring and user filtering.

“A”. This command switches between full-screen mode and alternate-display mode.

“B”. This command influences use of the 'bold' terminfo capability and alters both the summary area and task area for the current window.

“q”. Quit.

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

Sorting by VIRT:

```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
  883 student   20   0   11192   2332   1540 R   0.1   0.9   0:07.88 sshd
  892 student   20   0   11192   1708    960 S   0.0   0.7   0:00.00 sshd
  885 student   20   0    6668   3052   1692 S   0.0   1.2   0:00.13 bash
 4706 student   20   0    5428   1364   1008 R   0.0   0.6   0:00.06 top
  893 student   20   0    2460    624    528 S   0.0   0.3   0:00.00 sftp-server
```

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

The priority of processes can be adjusted to make more system resources available for other processes.

The “nice” command is used to launch a command with a specified niceness value, which influences the priority of the command.

The “renice” command is used to change the niceness value of an already running process.

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

To renice a running process from top, type “r” . You are first prompted for the PID of the process you want to renice. After entering the PID, you are prompted for the nice value you want to use. Enter a positive value to increase process priority or a negative value to decrease process priority.

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

man kill:

```
KILL(1)                                    User Commands                                    KILL(1)

NAME
       kill - send a signal to a process

SYNOPSIS
       kill [options] <pid> [...]

DESCRIPTION
       The  default  signal  for kill is TERM.  Use -l or -L to list available signals.  Particularly useful signals include HUP, INT, KILL, STOP,
       CONT, and 0.  Alternate signals may be specified in three ways: -9, -SIGKILL or -KILL.  Negative PID values may be  used  to  choose  whole
       process  groups;  see  the  PGID  column  in ps command output.  A PID of -1 is special; it indicates all processes except the kill process
       itself and init.

OPTIONS
       <pid> [...]
              Send signal to every <pid> listed.

       -<signal>
       -s <signal>
       --signal <signal>
              Specify the signal to be sent.  The signal can be specified by using name or number.  The behavior of signals is explained  in  sig-
              nal(7) manual page.

       -l, --list [signal]
              List signal names.  This option has optional argument, which will convert signal number to signal name, or other way round.

       -L, --table
              List signal names in a nice table.

       NOTES  Your shell (command line interpreter) may have a built-in kill command.  You may need to run the command described here as /bin/kill
              to solve the conflict.

EXAMPLES
       kill -9 -1
              Kill all processes you can kill.

       kill -l 11
              Translate number 11 into a signal name.

       kill -L
              List the available signal choices in a nice table.

       kill 123 543 2341 3453
 Manual page kill(1) line 1 (press h for help or q to quit)
```

- The signal SIGTERM (15) is used to ask a process to stop.
- The signal SIGKILL (9) is used to force a process to stop.
- The SIGHUP (1) signal is used to hang up a process. The effect is that the process will reread its configuration files, which makes this a useful signal to use after making modifications to a process configuration file.

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

These commands allow you to control the execution and behavior of processes, especially in the context of terminal sessions.

The “jobs” command lists the background jobs associated with the current terminal session. It displays the job numbers and their statuses, indicating whether they are running or stopped.

The “fg” command brings a background job to the foreground, making it the active process in the terminal. You can specify the job number or use % followed by the job number.

The “bg” command resumes a suspended background job, allowing it to continue running in the background. It is often used after suspending a job using Ctrl+Z.

The “nohup” command is used to run a command or process in the background, even if the terminal session is closed. It's often used to prevent processes from being terminated when you log out.

```
student@CsnKhai:~$ man kill
student@CsnKhai:~$ sleep 300 &
[1] 4730
student@CsnKhai:~$ yes
y
y

^Z[2]+  Stopped                 yes
student@CsnKhai:~$ jobs
[1]-  Running                 sleep 300 &
[2]+  Stopped                 yes
student@CsnKhai:~$ bg %1
-bash: bg: job 1 already in background
student@CsnKhai:~$ fg %1
sleep 300
student@CsnKhai:~$ jobs
[1]+  Stopped                 sleep 300
[2]-  Stopped                 yes
```

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)

For example, ssh-keygen which must generate SSH key pairs for authentication purposes:

```
C:\Windows\System32>ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\User/.ssh/id_rsa): new_ssh
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in new_ssh.
Your public key has been saved in new_ssh.pub.
The key fingerprint is:
SHA256:6LBgAdvXKyRozxEWtrdmkqn6w+gTwYO+5wYXjdkbXXE user@RomanK
The key's randomart image is:
+---[RSA 2048]----+
|.  =.    ..E     |
| =o o.   ..      |
|+.*oB.o .        |
|.+oX=+.+         |
|. +*++= S        |
| *.++*           |
| +B . .          |
|o.oo             |
|oo+.             |
+----[SHA256]-----+
```

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

```
C:\Windows\System32>ssh student@192.168.31.28
The authenticity of host '192.168.31.28 (192.168.31.28)' can't be established.
ECDSA key fingerprint is SHA256:yp8INOs6pk/gVv7G84N/cRT3KsgxLPiH81jZ/cRpzOo.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.31.28' (ECDSA) to the list of known hosts.
student@192.168.31.28's password:
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-63-generic i686)

 * Documentation:  https://help.ubuntu.com/
New release '16.04.7 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Wed Aug 16 11:00:04 2023 from 192.168.31.225
student@CsnKhai:~$
```

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

Ciphers:

```
student@CsnKhai:~$ ssh -Q cipher
3des-cbc
blowfish-cbc
cast128-cbc
arcfour
arcfour128
arcfour256
aes128-cbc
aes192-cbc
aes256-cbc
rijndael-cbc@lysator.liu.se
aes128-ctr
aes192-ctr
aes256-ctr
aes128-gcm@openssh.com
aes256-gcm@openssh.com
chacha20-poly1305@openssh.com
```

Key Exchange Algorithms:

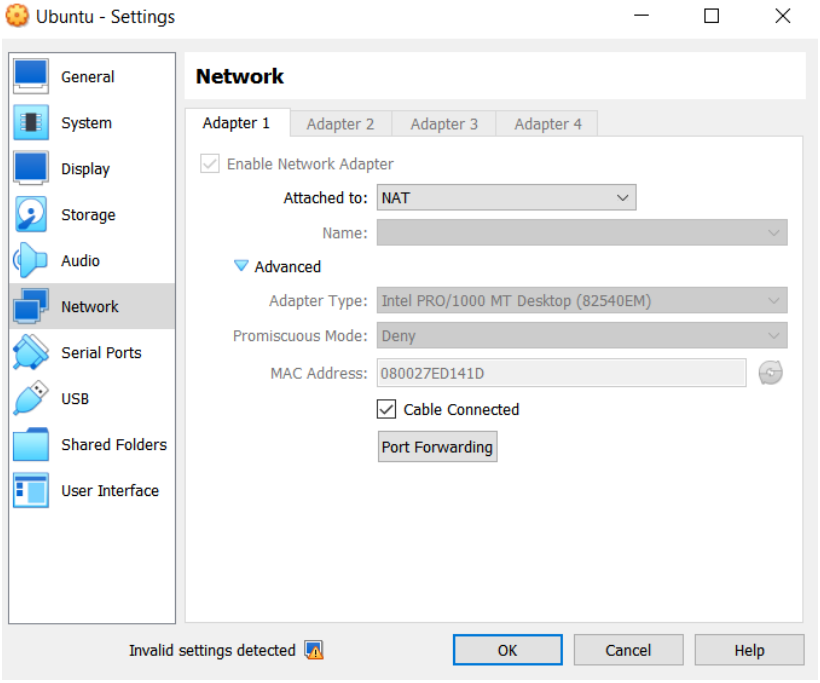
```
student@CsnKhai:~$ ssh -Q kex
diffie-hellman-group1-sha1
diffie-hellman-group14-sha1
diffie-hellman-group-exchange-sha1
diffie-hellman-group-exchange-sha256
ecdh-sha2-nistp256
ecdh-sha2-nistp384
ecdh-sha2-nistp521
diffie-hellman-group1-sha1
curve25519-sha256@libssh.org
```

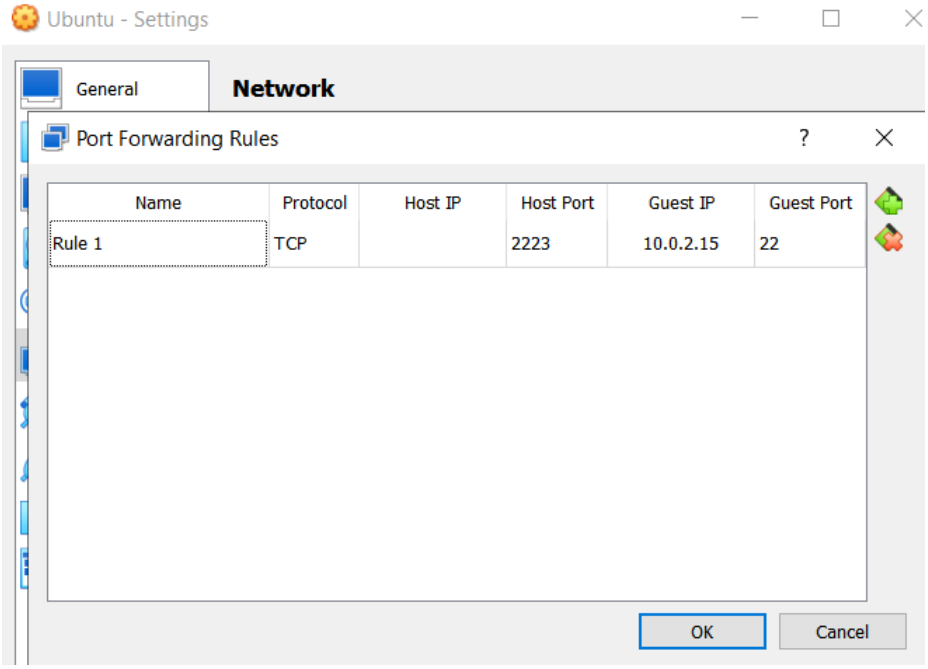
MAC (Message Authentication Code) Algorithms:

```
student@CsnKhai:~$ ssh -Q mac
hmac-sha1
hmac-sha1-96
hmac-sha2-256
hmac-sha2-512
hmac-md5
hmac-md5-96
hmac-ripemd160
hmac-ripemd160@openssh.com
umac-64@openssh.com
umac-128@openssh.com
hmac-sha1-etm@openssh.com
hmac-sha1-96-etm@openssh.com
hmac-sha2-256-etm@openssh.com
hmac-sha2-512-etm@openssh.com
hmac-md5-etm@openssh.com
hmac-md5-96-etm@openssh.com
hmac-ripemd160-etm@openssh.com
umac-64-etm@openssh.com
umac-128-etm@openssh.com
```

```
 GNU nano 2.2.6                                File: /etc/ssh/sshd_config

X11Forwarding yes
X11DisplayOffset 10
PrintMotd no
PrintLastLog yes
TCPKeepAlive yes
#UseLogin no

#MaxStartups 10:30:60
#Banner /etc/issue.net

# Allow client to pass locale environment variables
AcceptEnv LANG LC_*

Subsystem sftp /usr/lib/openssh/sftp-server

# Set this to 'yes' to enable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be allowed through the ChallengeResponseAuthentication and
# PasswordAuthentication.  Depending on your PAM configuration,
# PAM authentication via ChallengeResponseAuthentication may bypass
# the setting of "PermitRootLogin without-password".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and ChallengeResponseAuthentication to 'no'.
UsePAM yes

Ciphers aes256-ctr,aes192-ctr,aes128-ctr
KexAlgorithms curve25519-sha256,diffie-hellman-group-exchange-sha256
MACs hmac-sha2-256,hmac-sha2-512
```

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





```
C:\Windows\System32>ssh -p 2223 student@localhost
The authenticity of host '[localhost]:2223 ([127.0.0.1]:2223)' can't be established.
ECDSA key fingerprint is SHA256:yp8INOs6pk/gVv7G84N/cRT3KsgxLPiH81jZ/cRpzOo.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[localhost]:2223' (ECDSA) to the list of known hosts.
student@localhost's password:
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-63-generic i686)

 * Documentation:  https://help.ubuntu.com/
New release '16.04.7 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Thu Aug 17 06:23:36 2023 from 10.0.2.2
student@CsnKhai:~$
```

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

Intercept traffic using tcpdump:

```
student@CsnKhai:~$ sudo tcpdump -s 0 -i eth0 -w tcpdump.pcap
[sudo] password for student:
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
^C6 packets captured
7 packets received by filter
0 packets dropped by kernel
student@CsnKhai:~$ sudo chmod 644 tcpdump.pcap
student@CsnKhai:~$
```

Downloading pscp on windows:

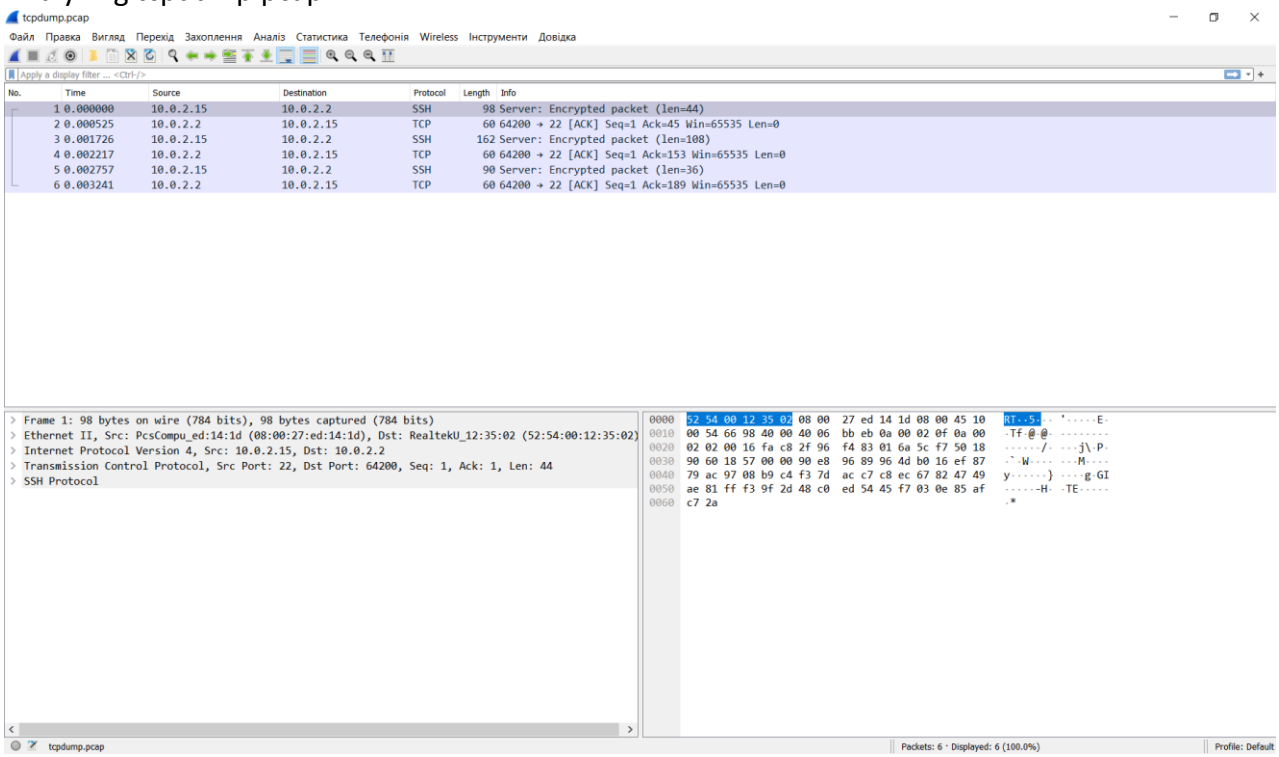
pscp.exe (an SCP client, i.e. command-line secure file copy)

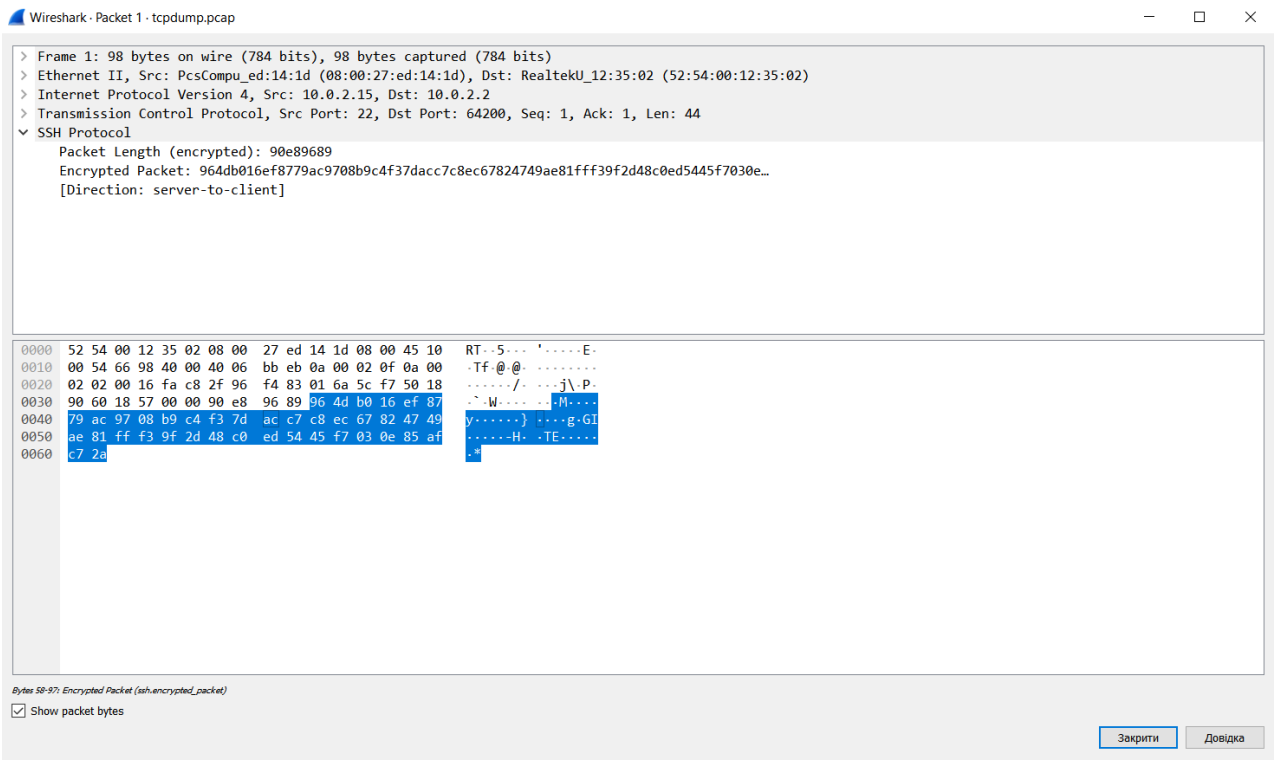
64-bit x86:	pscp.exe	(signature)
64-bit Arm:	pscp.exe	(signature)
32-bit x86:	pscp.exe	(signature)

Transfer tcpdump.pcap:

```
C:\Windows\System32>pscp.exe -P 2223 student@localhost:/home/student/tcpdump.pcap "C:\GitHub"
student@localhost's password:
tcpdump.pcap                 | 0 kB |   0.6 kB/s | ETA: 00:00:00 | 100%
```

Analyzing tcpdump.pcap:





As we can see, result is encrypted.