Learning diary and answers

Name:

Selina Kallio

Email:

selina.kallio@gmail.com

Phone:

040 840 0274

This file shows a bit of my Linux knowledge and skills. It was done for a Linux course in AMK. Word definitions and short tasks are mostly included in it, feel free to test out my code or ask me for more information on this project. Overall, the completion of this file took hundreds of hours and lots of hard work.

Week 1

- Describe following commands and concepts:
    - man

      = display the user manual of any command that we can run on the terminal
        - apropros

      = searches the Linux man page with the help of the keyword
    - man date

      = manual for date usage/writing
    - ls

      = list files or directories
    - ls –help

      = help page of ls command
    - date

      = show date
    - date –help

      help page of date command
    - cd

      = change directory
    - cd –

      = previously used directory and changes to it
    - cd ..

      = move one level up from the current directory
    - ls -lat

      = list of all files sorted by date
    - ls -s aaaa*

      = determines a files size
    - pwd

      = print working directory, shows current dir
    - chown

      = change ownership

- chmod

  = modifies File Permissions

- chgrp

  = change the group ownership

- chmod 644 file

  = give owner r+w access, others r

- chmod g+x myfile

  = give group access to execute

- which

  = locate the executable file

- rm

  = remove objects

- rm -r mydoc

  = remove a directory and all its contents

- cp primary secondary

  cp = copy

  primary = Specifies a group that the operating system assigns to files that are created by the user. Each user must belong to a primary group.

  secondary = Specifies one or more groups to which a user also belongs. Users can belong to up to 15 secondary groups.

- mv file2 file1

  = if both filenames are on the same filesystem, this results in a simple file rename; otherwise the file content is copied to the new location and the old file is removed

- wc -l myfile

  = prints the number of lines present in a file

- mkdir mydata

  = make new directory

- rmdir mydata

  =removes directory

- more, less

  = move up/down

- file

= determine the type of a file

- stat

  = returns file attributes about an inode displays detailed information about given files or file systems

- df

  = available and used disk space usage of the file system

- ln

  = create a hard link or a symbolic link to an existing file or directory

- which, whereis

  = used to locate the binary, source, and manual page files for a command

- find

  = locates files based on some user-specified criteria

- touch

  = create, change and modify timestamps of a file

- touch mynewfile

  = touch command is primarily used to change file timestamps, but if the file (whose name is passed as an argument) doesn't exist, then the tool creates it.

- cp /tmp/test.txt ~/temp/

  = moves test.txt to temp

- Answer shortly:
  - What is the difference between Linux kernel and GNU/Linux distribution?

    GNU is an operating system designed as a replacement for UNIX with many software programs while Linux is an operating system with a combination of GNU software and Linux kernel.

  - Name some very common Linux distributions

    Debian, Ubuntu, Mint, Fedora

  - What is GPLv2/v3 license? And BSD style license?

    GPL= copyleft license, BSP= permissive free software licenses

  - What is (operating system) shell?

    computer program which exposes an operating system's services to a human user or other program

  - What are case sensitive file names?

    In linux, everything

Learning diary and answers

- Describe common purpose of files and directories in “/etc”, “/usr/bin” and “/var”

  Etc = gonfig files

  Var = files important for running services

  Usr = user-accessible applications

- What is shell PATH? What is the difference between absolute and relative path?

  colon-delimited list of directories that your shell searches through when you enter a command

  absolute path specifies the location from the root directory whereas relative path is related to the current directory.

- What is the purpose of tilde character (~) for most Linux shells. For example ls ~/

  omitted folder layers

- How do you recognise a hidden file in any common Unix/Linux file systems?

  Starts with “.”

- What is the meaning of “piping data between commands”?

  make one command's output the standard input of another command

- What are seti-uid (suid) and set-gid (sgid) bits for file permissions?

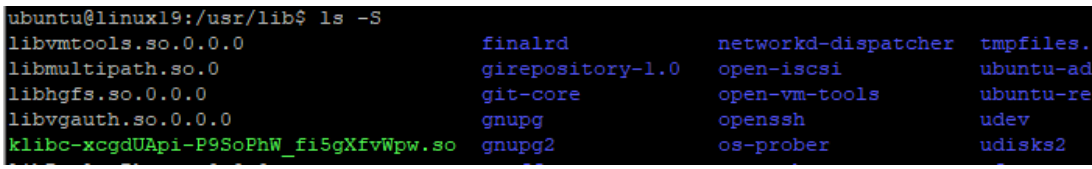
  SUID: a special file permission for executable files

  SGID: special file permission that also applies to executable files and enables other users to inherit the effective GID of file group owner

- What is “sticky-bit”?

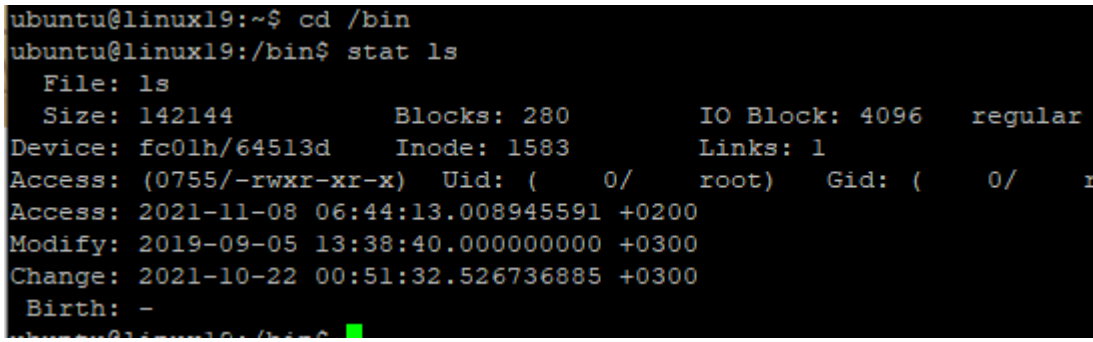
  user ownership access right flag that can be assigned to files and directories

- List five largest files in /usr/lib -directory



- With your personal Linux host or with students.oamk.fi server:

  - Find out what is the group for /bin/ls file?



- How do you change file or directory owner and group?

  chown

- How do you change file permissions so that file user has all rights (read, write and execute), group and others have none?

  chmod 700

- How do you change file permissions so that file user has read and write access (no execute), group and others have read access?

  chmod 644

- How do you change file permissions so that file user, group and others have only read and execute (no write) access?

  chmod 555

- Describe following file permissions and ownership:
  drwxr-x--- 2 teemu root 4096 Jul 2 2002 webalizer

  user: read, write, exe

  group: exe

  others: nothing

- Use manual pages and look what will command “uname -a” do?

  provides users with important system information

- Use manual pages and look what will command “wc -l” do?

  counts lines

- Create directory “exercise1” under you home directory

  ubuntu@linux19:/home$ sudo mkdir exercise1

- Create empty file (length 0 bytes) “qwerty.txt” to that directory

  ubuntu@linux19:/home/exercise1$ sudo touch qwerty.txt

- Change directory name “exercise1” to “exer2”?

  ubuntu@linux19:/home$ sudo mv exercise1 exer2

- Change file qwerty.txt file permissions so that only you have just a read access to it and nothing else

  ubuntu@linux19:/home/exer2$ sudo chmod 400 qwerty.txt

- Create symbolic link to you home directory “this_is_my_link” and make it point to the exer2-directory

  ubuntu@linux19:/home$ sudo ln -s exer2 this_is_my_link

- Remove files and directories which you created on this exercise.

  ubuntu@linux19:/home$ sudo rm this_is_my_link

  ubuntu@linux19:/home$ sudo rm -r exer2

  - How can you find out your current directory location and path? How far (in directories) are you from file system root?

    ubuntu@linux19:/home$ pwd

    /home

    1 directory away from root

Week 2

- Watch some live coding Twitch streams, select one and answer:
  - What is the programming language/languages used?

    C#

  - Which programming libraries and frameworks are being used?

    .NET framework

  - What is the code editor / IDE being used?

    visual studio code

- If you have a Windows host: Install the Cmder and try common file commands there (ls, cd, mkdir etc) and bash shell
- Study and explain shortly following commands and concepts:
  - zip, unzip

    zip = condensed files

    unzip = extracting files condensed as zip

  - tar

    = extract a file compressed with tar

  - gzip

    = file format used for file compression and decompression

  - xz

    = command line data compression utility

  - zcat, zgrep

    zcat= display the contents of a gz file

zgrep= search through compressed files without having to unzip them first

- compress

  making files smaller by using variables for repetitive units

- bzip2

  single file compression (free, open-source)

- 7z

  high compression rate file group archiving

- ldd

  prints the shared libraries required by each program or shared library specified on the command line

- gnu gcc / gcc / g++

  optimizing compiler (gcc for C, g++ for C++)

- Install build-essential meta package (containing development tools) to your server with: sudo apt install build-essential
- Get the source code for curses-based ("text-graphics") worm game nibbles-1.2.tar.gz
  - Unpack the source package to a some temporary directory under your home directory
  - Compile the game and try playing it. Note: Ubuntu does not have ncursed development libraries installed by default. Use apt install to install the missing library dependencies: sudo apt install libncurses-dev
- Get the source code for another curses-based ("text-graphics") Tetris game nct-1.4.tar.gz
  - Unpack source package to temporary directory in your home directory.
  - Use source package's configure script to generate Makefile with installation prefix pointing to your home directory
  - Compile source code and install compiled files

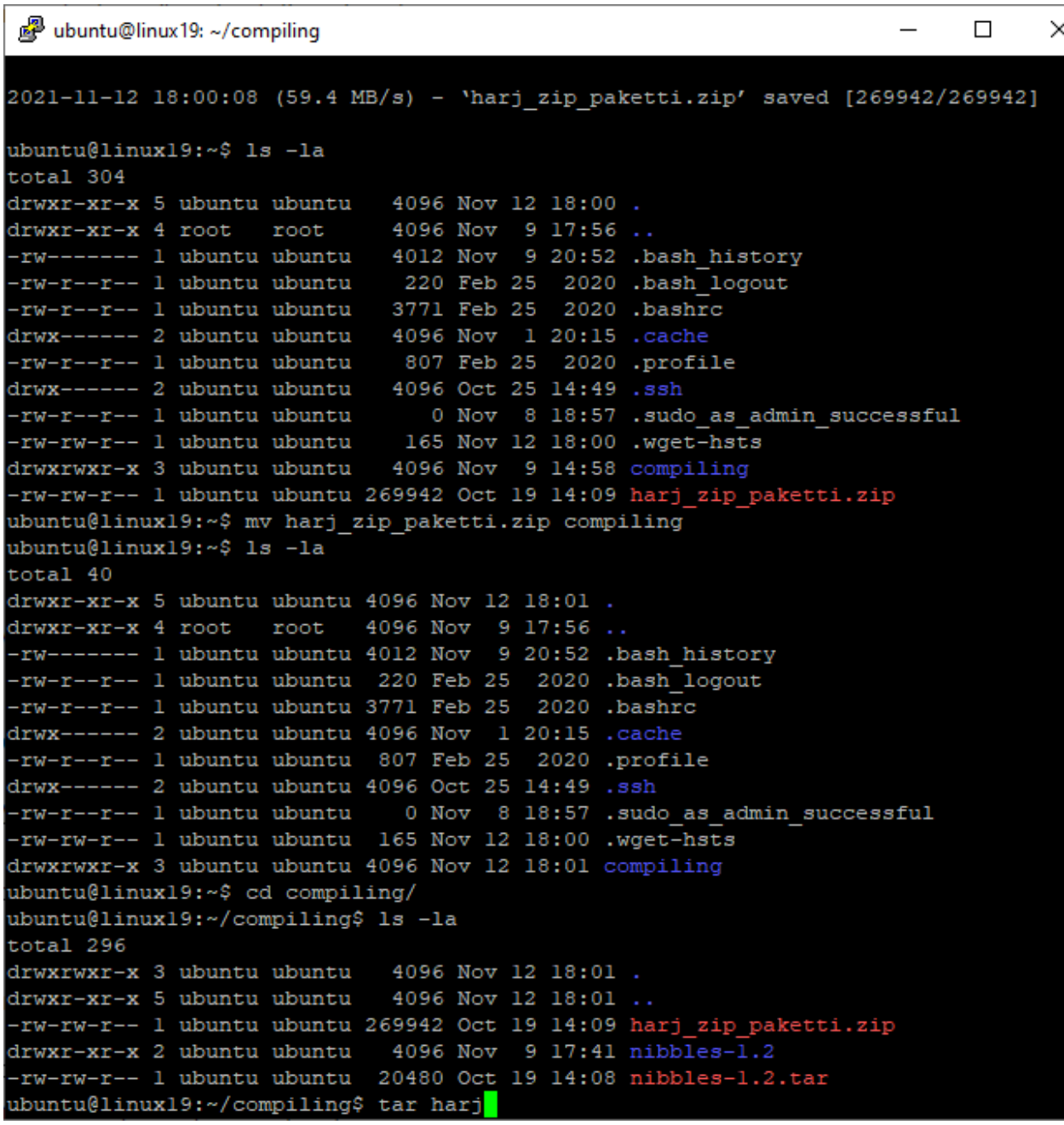
    - Test if game works
    - Remove temporary game directory
- Download the file harj_zip_paketti.zip. Zip-package has following hierarchy:

```
paahakemisto
        hakemisto_a
                karate_kat.jpg
                lazy.jpg

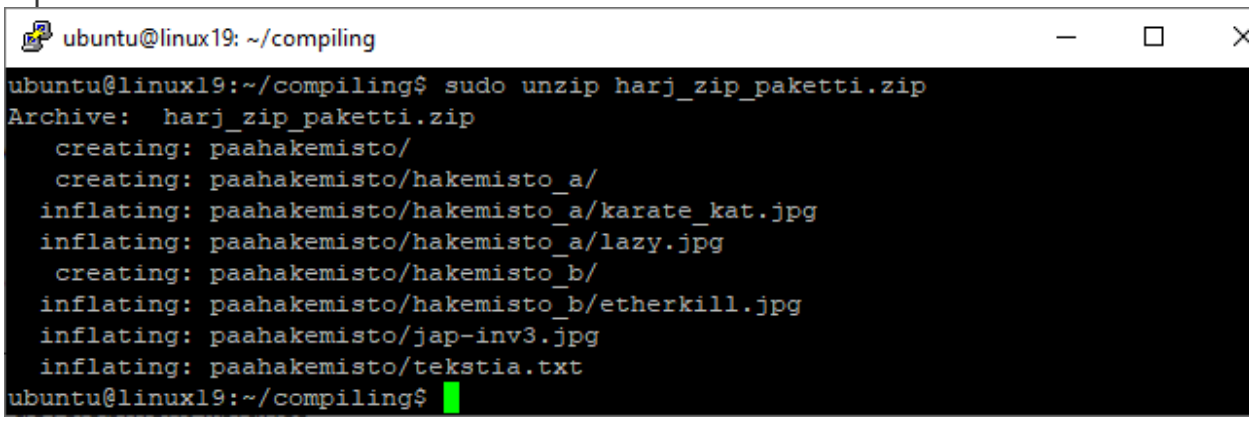
        hakemisto_b
                etherkill.jpg

        jap-inv3.jpg
        tekstia.txt
```

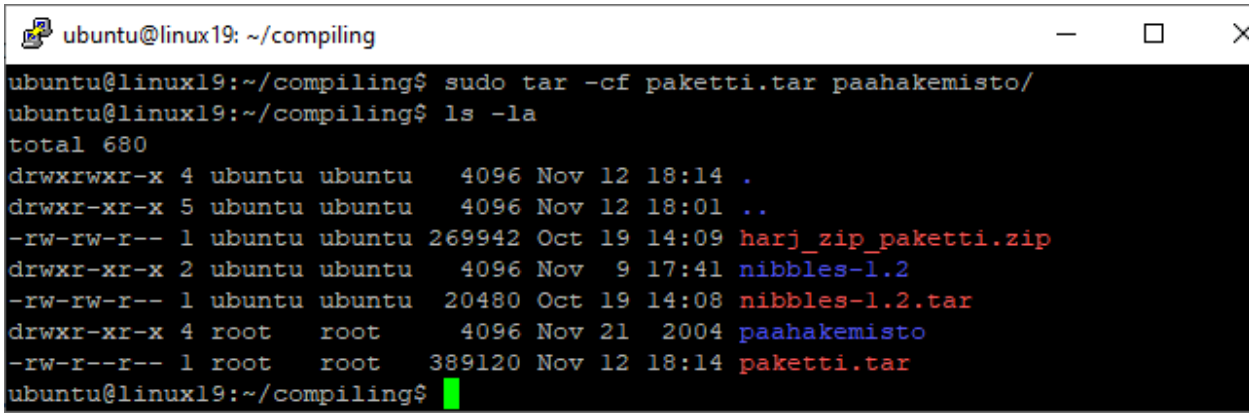
- With the ZIP file:
    - Unpack package and all subdirectories to a temporary directory in your home directory

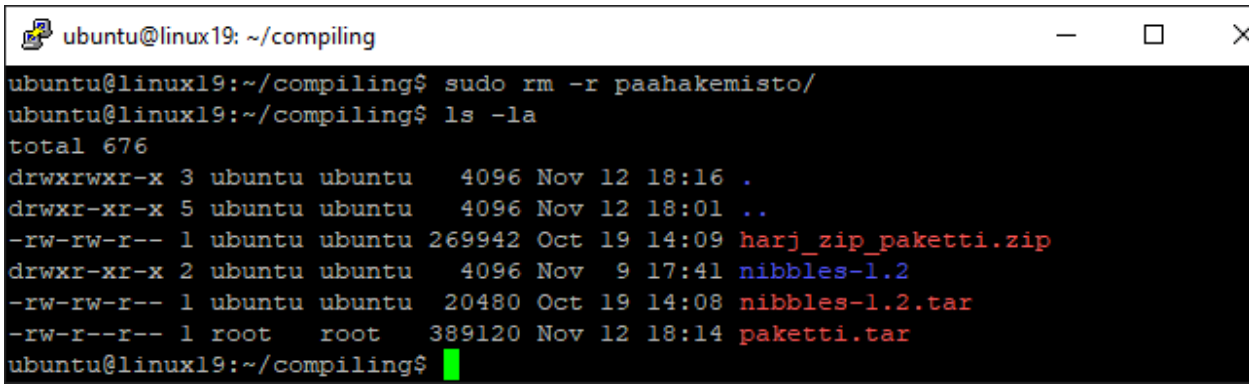


- Create tar archive from unpacked files and directories and name it to a paketti.tar

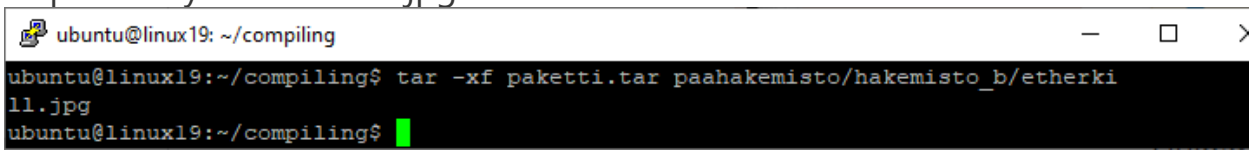


- List contents of the paketti.tar. If everything is correct, delete paahakemisto directory and all subdirectories under it. Don't delete the paketti.tar -file.

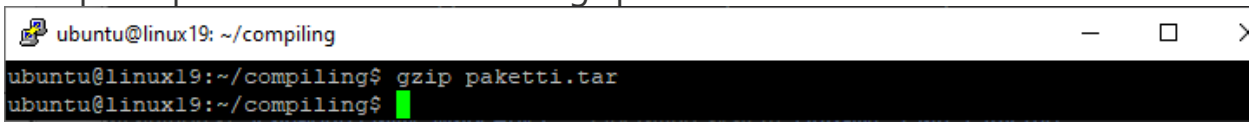




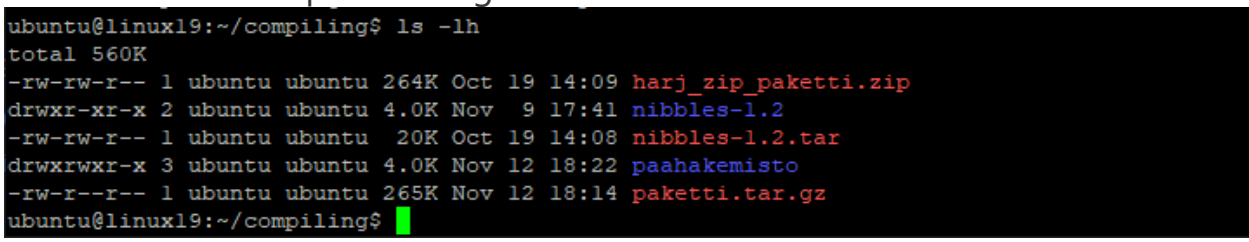
- Unpack only the etherkill.jpg file from tar archive.



- Compress paketti.tar archive with a gzip command.



- What is the size of paketti.tar.gz now?



- Uncompress paketti.tar.gz and compress it again, but now with bzip2. Check the size again. Any difference?

ubuntu@linux19:~/compiling$ gzip -d paketti.tar.gz

ubuntu@linux19:~/compiling$ tar -xvf paketti.tar

ubuntu@linux19:~/compiling$ ls -lh

total 1.4M

-rw-rw-r-- 1 ubuntu ubuntu **675K** Nov 12 18:44 paketti.bz2

-rw-r--r-- 1 ubuntu ubuntu **265K** Nov 12 18:14 paketti.tar.gz

- Delete temporary files and directories created on this practice

- Compile this C source code with gcc and check if it works. helloworld.c source code:

```
#include <stdio.h>

int main(void) {
    printf("Hello, world!\n");
    return 0;
}
```

ubuntu@linux19:~$ nano helloworld.c

ubuntu@linux19:~/hwoof$ gcc -o resulting_binary hello.c

ubuntu@linux19:~/hwoof$ ./resulting_binary

Hello, world!

- Compile this C++ source code with g++ and test it. helloworld.cpp sourse code:

```
#include <iostream>

using namespace std;
int main()
{
     cout << "Hello World!\n";
}

ubuntu@linux19:~/hwoof$ nano plushello.cpp
ubuntu@linux19:~/hwoof$ g++ -o resultingcpp plushello.cpp
```

- With previously compiled helloworld C++ binary:
  - What are statically linked libraries? Why would you use them?

contents of that file are included at link time, physically in the file. usable eg if offline or without contact to outer world

- Inspect the size of ready binary file (that compiled helloworld binary). Compile it again and use some different output filename. Withc gcc, use now statically linked libraries (with compiler's -static parameter). Compare the file sizes of statically and dynamically linked binaries

20 -rwxrwxr-x 1 ubuntu ubuntu 16696 Nov 17 21:11 resulting_binary

ubuntu@linux19:~/hwoof$ gcc -o resulting_binary2 hello.c

ubuntu@linux19:~/hwoof$ gcc -static -o resulting_binary2 hello.c

852 -rwxrwxr-x 1 ubuntu ubuntu 871760 Nov 17 21:30 resulting_binary2

The statically linked file must include the necessary library parts, while the dynamic does not, saving it a lot of data space. (871kB vs 17kB)

- Use strace to inspect interiors (system calls) of ls command: "strace ls" and compare the output to a "strace chmod". Check _exit -values. Why chmod returns 1 and ls returns 0?

Its just a random given value to show correctly executed program.

- Why and when Unix administrators and programmers use system call tracing programs and debuggers such as gdb and strace?

easier to use strace but gdb detects more varied problems; usually debugging purposes but also diagnostics.

- Create some gzipped tar archive and use SSH (scp) to copy it to students.oamk.fi

```
ubuntu@linux19:~/targz$ tar -czvf targz.tar.gz file1 file2

file1
```

```
file2

ubuntu@linux19:~/targz$ ls -la

total 16

drwxrwxr-x 2 ubuntu ubuntu 4096 Nov 22 19:18 .

drwxr-xr-x 8 ubuntu ubuntu 4096 Nov 22 19:13 ..

-rw-rw-r-- 1 ubuntu ubuntu    0 Nov 22 19:13 file1

-rw-rw-r-- 1 ubuntu ubuntu    0 Nov 22 19:14 file2

-rw-rw-r-- 1 ubuntu ubuntu  131 Nov 22 19:18 targz.tar.gz

ubuntu@linux19:~/targz$ scp targz.tar.gz students.oamk.fi
```

- Solve these service management tasks (Note: most tasks will require root access):
    - Check what network adapters your Linux host/server has with command: ip addr or ifconfig (ifconfig is not necessary installed by default)

```
ubuntu@linux19:~$ ip addr

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000

    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

    inet 127.0.0.1/8 scope host lo

       valid_lft forever preferred_lft forever

    inet6 ::1/128 scope host

       valid_lft forever preferred_lft forever

2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000

    link/ether 52:54:00:6a:1c:33 brd ff:ff:ff:ff:ff:ff

    inet 172.20.241.19/23 brd 172.20.241.255 scope global enp1s0

       valid_lft forever preferred_lft forever

    inet6 2001:708:510:665:5054:ff:fe6a:1c33/64 scope global dynamic mngtmpaddr
noprefixroute

       valid_lft 2591996sec preferred_lft 604796sec
```

```
    inet6 2001:708:510:665::19/64 scope global
       valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:fe6a:1c33/64 scope link
       valid_lft forever preferred_lft forever
```

```
ubuntu@linux19:~$ ifconfig
enp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.20.241.19  netmask 255.255.254.0  broadcast 172.20.241.255
        inet6 2001:708:510:665:5054:ff:fe6a:1c33  prefixlen 64  scopeid
0x0<global>
        inet6 fe80::5054:ff:fe6a:1c33  prefixlen 64  scopeid 0x20<link>
        inet6 2001:708:510:665::19  prefixlen 64  scopeid 0x0<global>
        ether 52:54:00:6a:1c:33  txqueuelen 1000  (Ethernet)
        RX packets 1312665  bytes 176417817 (176.4 MB)
        RX errors 0  dropped 321940  overruns 0  frame 0
        TX packets 22324  bytes 3321568 (3.3 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 560  bytes 53908 (53.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 560  bytes 53908 (53.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
ubuntu@linux19:~$ lshw -class network

WARNING: you should run this program as super-user.

  *-network

       description: Ethernet controller

       product: Virtio network device

       vendor: Red Hat, Inc.

       physical id: 0

       bus info: pci@0000:01:00.0

       version: 01

       width: 64 bits

       clock: 33MHz

       capabilities: bus_master cap_list rom

       configuration: driver=virtio-pci latency=0

       resources: irq:22 memory:fe880000-fe880fff memory:fca00000-fca03fff
memory:fe800000-fe87ffff

     *-virtio0

          description: Ethernet interface

          physical id: 0

          bus info: virtio@0

          logical name: enp1s0

          serial: 52:54:00:6a:1c:33

          capabilities: ethernet physical

          configuration: autonegotiation=off broadcast=yes driver=virtio_net
driverversion=1.0.0 ip=172.20.241.19 link=yes multicast=yes
```

- Listen inbound ICMP traffic in your server with tcpdump command line protocol analyzer and test if you can see the traffic when you

ping your server: tcpdump -n -i YOUR_NETWORK_ADAPTER_NAME_HERE icmp

```
ubuntu@linux19:~$ sudo tcpdump -n -i enp1s0 icmp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on enp1s0, link-type EN10MB (Ethernet), capture size 262144
bytes
```

```
ubuntu@linux19:~$ ping 172.20.241.19

PING 172.20.241.19 (172.20.241.19) 56(84) bytes of data.

64 bytes from 172.20.241.19: icmp_seq=1 ttl=64 time=0.137 ms

64 bytes from 172.20.241.19: icmp_seq=2 ttl=64 time=0.033 ms

64 bytes from 172.20.241.19: icmp_seq=3 ttl=64 time=0.028 ms

64 bytes from 172.20.241.19: icmp_seq=4 ttl=64 time=0.030 ms

64 bytes from 172.20.241.19: icmp_seq=17 ttl=64 time=0.030 ms

64 bytes from 172.20.241.19: icmp_seq=18 ttl=64 time=0.028 ms

64 bytes from 172.20.241.19: icmp_seq=19 ttl=64 time=0.030 ms

64 bytes from 172.20.241.19: icmp_seq=20 ttl=64 time=0.029 ms

64 bytes from 172.20.241.19: icmp_seq=21 ttl=64 time=0.032 ms

64 bytes from 172.20.241.19: icmp_seq=22 ttl=64 time=0.029 ms

64 bytes from 172.20.241.19: icmp_seq=23 ttl=64 time=0.029 ms

64 bytes from 172.20.241.19: icmp_seq=24 ttl=64 time=0.028 ms

64 bytes from 172.20.241.19: icmp_seq=25 ttl=64 time=0.028 ms

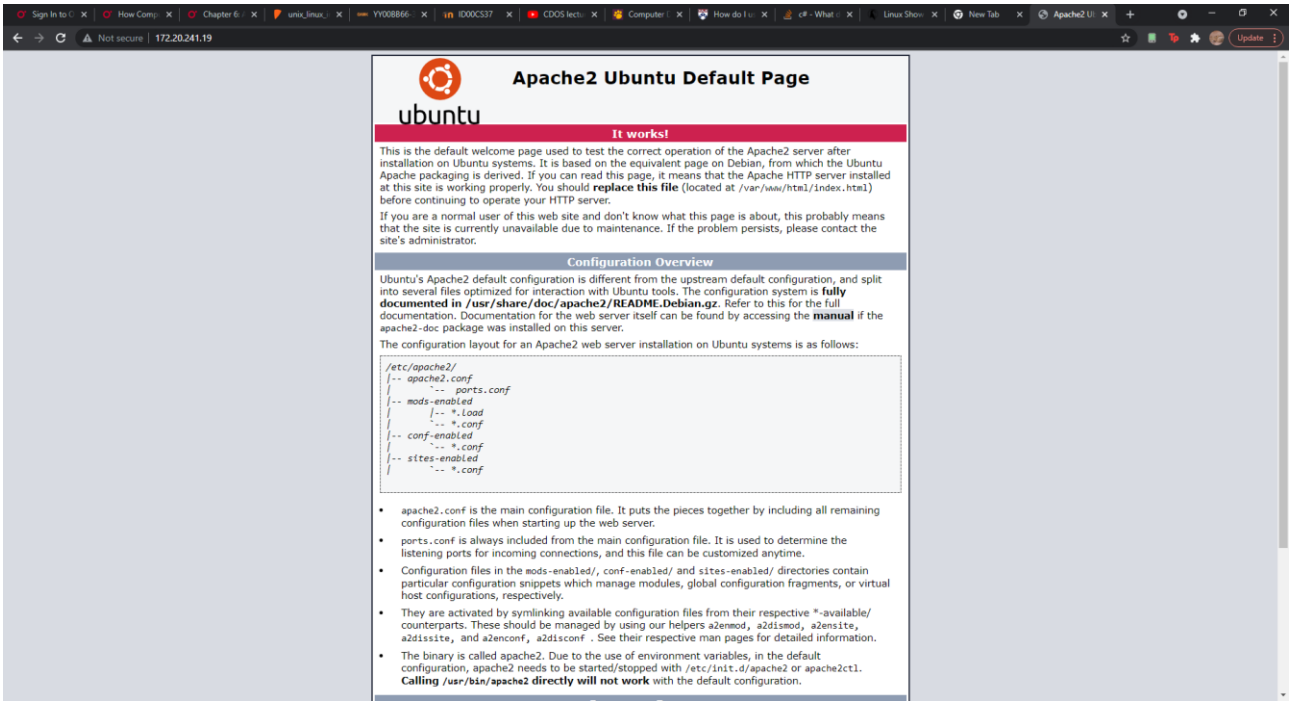
^C

--- 172.20.241.19 ping statistics ---

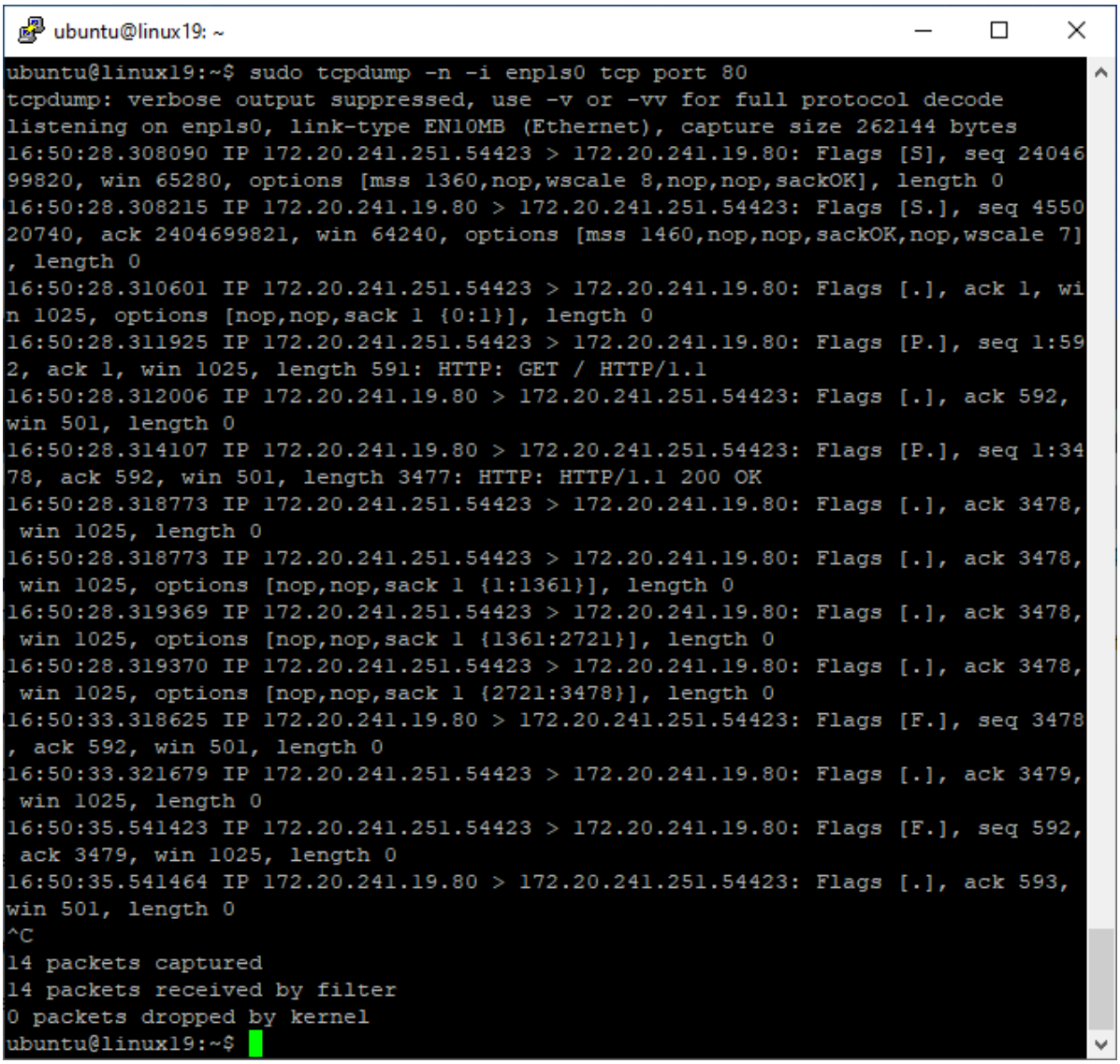
25 packets transmitted, 25 received, 0% packet loss, time 24561ms
```

```
rtt min/avg/max/mdev = 0.027/0.034/0.137/0.021 ms
```

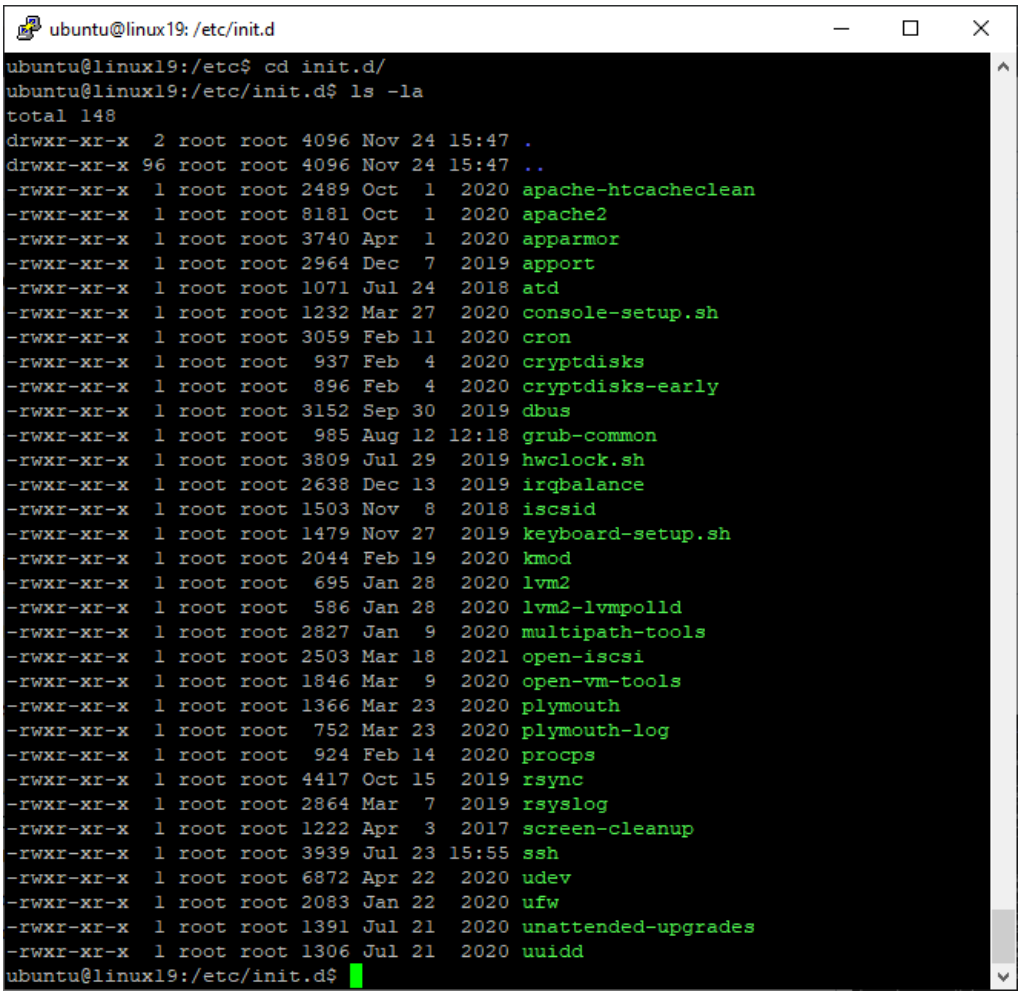
- Install apache web server with apt install apache2 and test that you can access your server with a web browser



- Listen TCP/80 (web) traffic in your server with tcpdump and test if you can see the inbound TCP SYN segments after you try to access your server with a web browser: tcpdump -n -i YOUR_NETWORK_ADAPTER_NAME_HERE tcp port 80



- Study what is runlevel? done
- Study what is systemd? done
- Check and study what are the files in /etc/init.d/ directory?



- What is your server's runlevel now?

```
ubuntu@linux19:/etc/init.d$ who -r

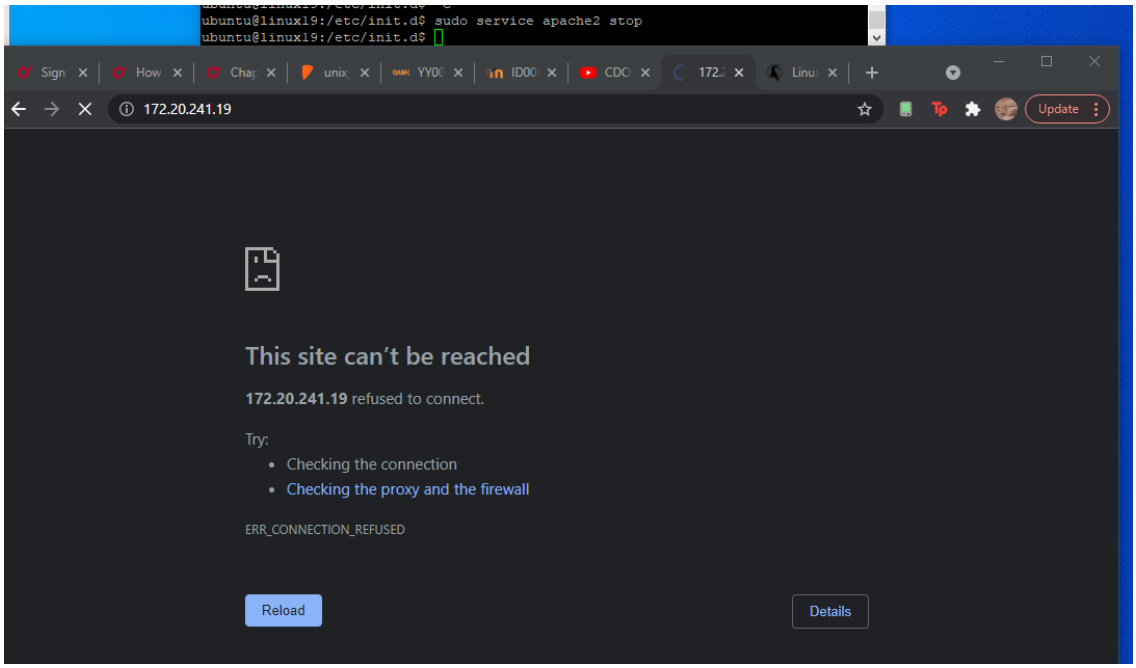
         run-level 5  2021-11-15 08:33
```

- Study but don't do: What is runlevel 6? What is the purpose of init 6 command? How would you do the same with systemd?

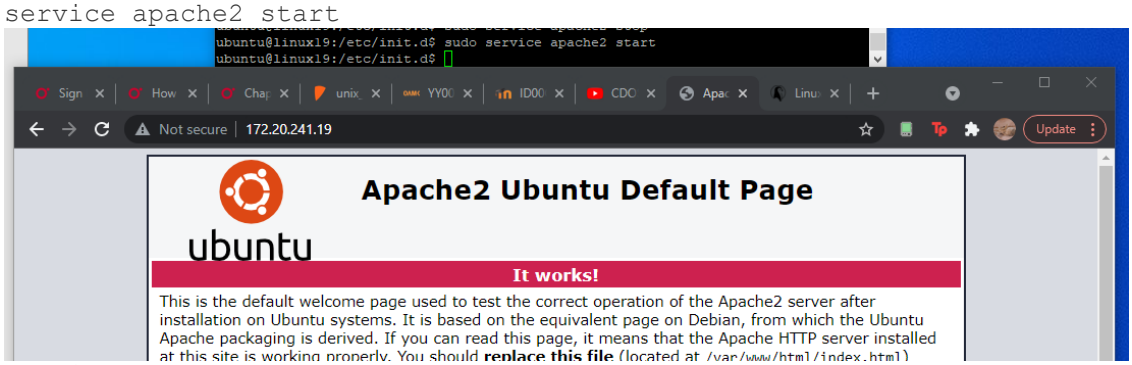
runlevel 6 quickly reboots the device. init 6 runs shutdown scripts first, therefore doing a cleaner reboot. rebooting can be done via the systemd UI systemctl. to reboot with systemctl: "$ sudo systemctl start reboot.target"

- Test these service management commands with your web server and use web browser to verify the operation whether the server is running or not:

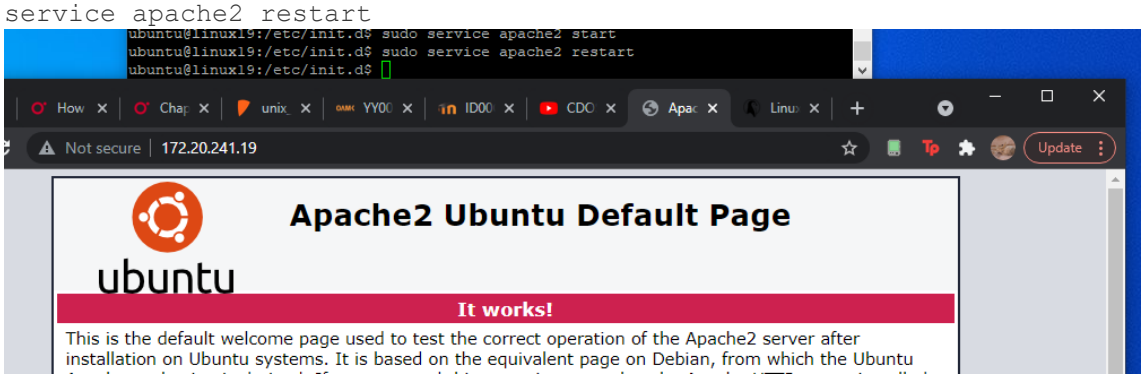
```
service apache2 stop
```




```
service apache2 start
```




```
service apache2 restart
```

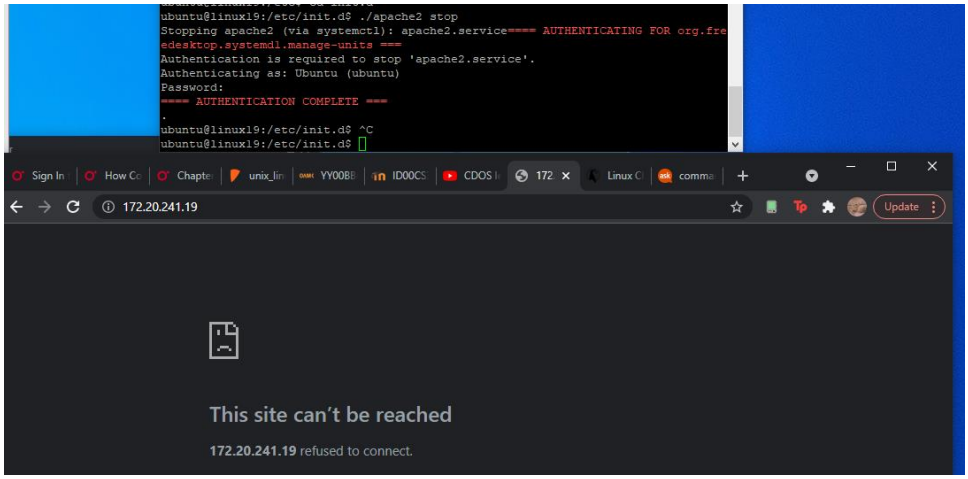


- Test these service management commands with your web server and use web browser to verify the operation whether the server is running or not:

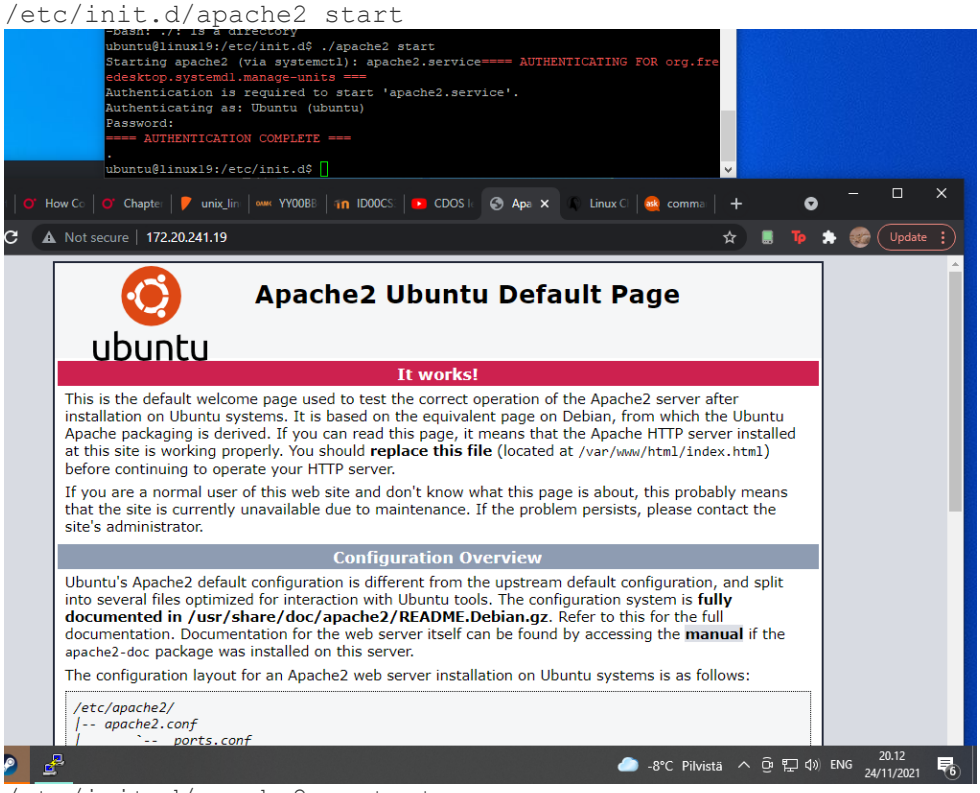
```
/etc/init.d/apache2 stop
```

```
ubuntu@linux19:/etc/init.d$ ./apache2 stop
Stopping apache2 (via systemctl): apache2.service==== AUTHENTICATING FOR
org.freedesktop.systemd1.manage-units ===
Authentication is required to stop 'apache2.service'.
Authenticating as: Ubuntu (ubuntu)
Password:
==== AUTHENTICATION COMPLETE ===
```

.



./ /etc/init.d/apache2 start



/etc/init.d/apache2 restart

```
ubuntu@linux19:/etc/init.d$ ./apache2 restart
Restarting apache2 (via systemctl): apache2.service==== AUTHENTICATING FOR
org.freedesktop.systemd1.manage-units ===
Authentication is required to restart 'apache2.service'.
Authenticating as: Ubuntu (ubuntu)
Password:
polkit-agent-helper-1: pam_authenticate failed: Authentication failure
==== AUTHENTICATION FAILED ===
Failed to restart apache2.service: Access denied
See system logs and 'systemctl status apache2.service' for details.
 failed!
ubuntu@linux19:/etc/init.d$ sudo ./apache2 restart
Restarting apache2 (via systemctl): apache2.service.
```

- Same with systemd management. Try and explain:

```
journalctl | tail -20
systemctl restart apache2
```

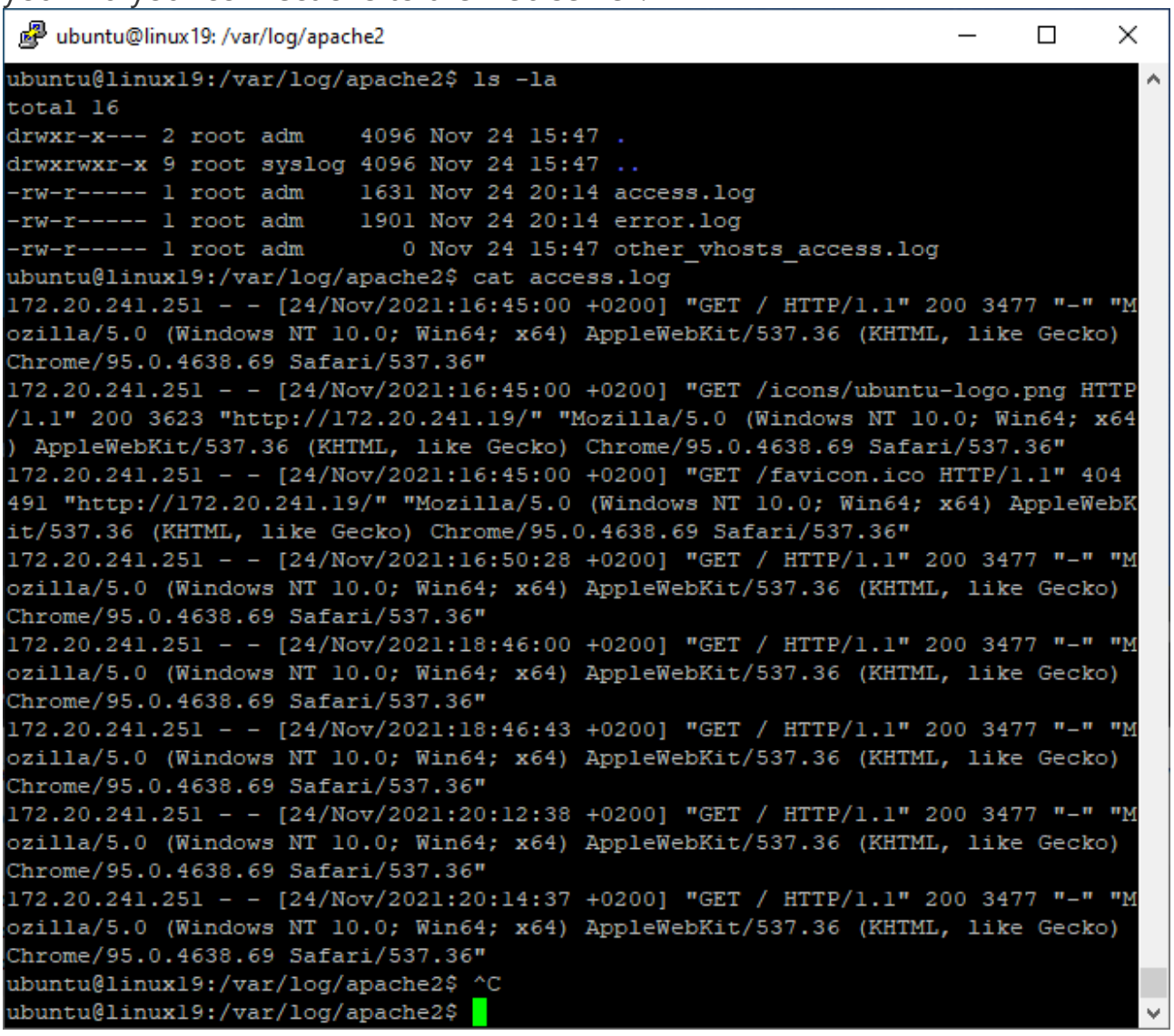
makes a log out of last 20 lines of information on apache2 being restarted (history)

```
journalctl | tail -20
systemctl stop apache2
```

a log of 20 lines of apache2 being stopped

```
systemctl start apache2
systemctl
```

- Check Apache access.log file contents in /var/log/apache2/ directory. Can you find your connections to the web server?



Week 3

Question 1: Nnnn

- Describe these CPU, computer architecture and computing related terms and concepts shortly:
    - RISC vs CISC

      RISC: more simple instructions, emphasis on HW, requires more RAM

      CISC: more complex instructions, SW emphasis, one instruction will do whole task

    - Single-board computer (SBC)

  e.g. raspberry pi, all necessary pieces of computer on single circuit board.

    - GPU

  electronic circuit to show images at a faster pace in a frame buffer for final show on for example PCs or phones.

    - CPU/SoC

  CPU: processor, executes the instructions defined in coded programs.

  SoC: many modern CPUs are on ICs, if they contain other computer components they are possibly called SoCs.

    - x86

  microprocessors based on Intel 8086 and 8088, possibility of accessing multiple data segments at same interval. x86 improves efficiency of platforms.

    - MIPS

  RISC which edits the user mode architecture and influenced upcoming RISC architectures a lot.

    - ARM

cheap and power efficient, making them great for e.g. smartphones and laptops. advanced RISC machine.

- AVR

microcontrollers, modified 8-bit RISC single-chip. apps as embedded systems, made popular in hobby usage by Arduino.

- MOS 6502

8-bit microprocessor which, by its exponentially cheaper price, made video game consoles spread fast and brought them to many homes thanks to the low price of the MOS tech 6502.

- CPU registers

fast-access location for the CPU with usually only little storage

- Opcode

operation code, the part of the machine language instruction that specifies what operation will be performed.

- Illegal opcode

instruction to a CPU not mentioned in any official documentation released by the CPU's designer or manufacturer but still has an effect

- Bytecode

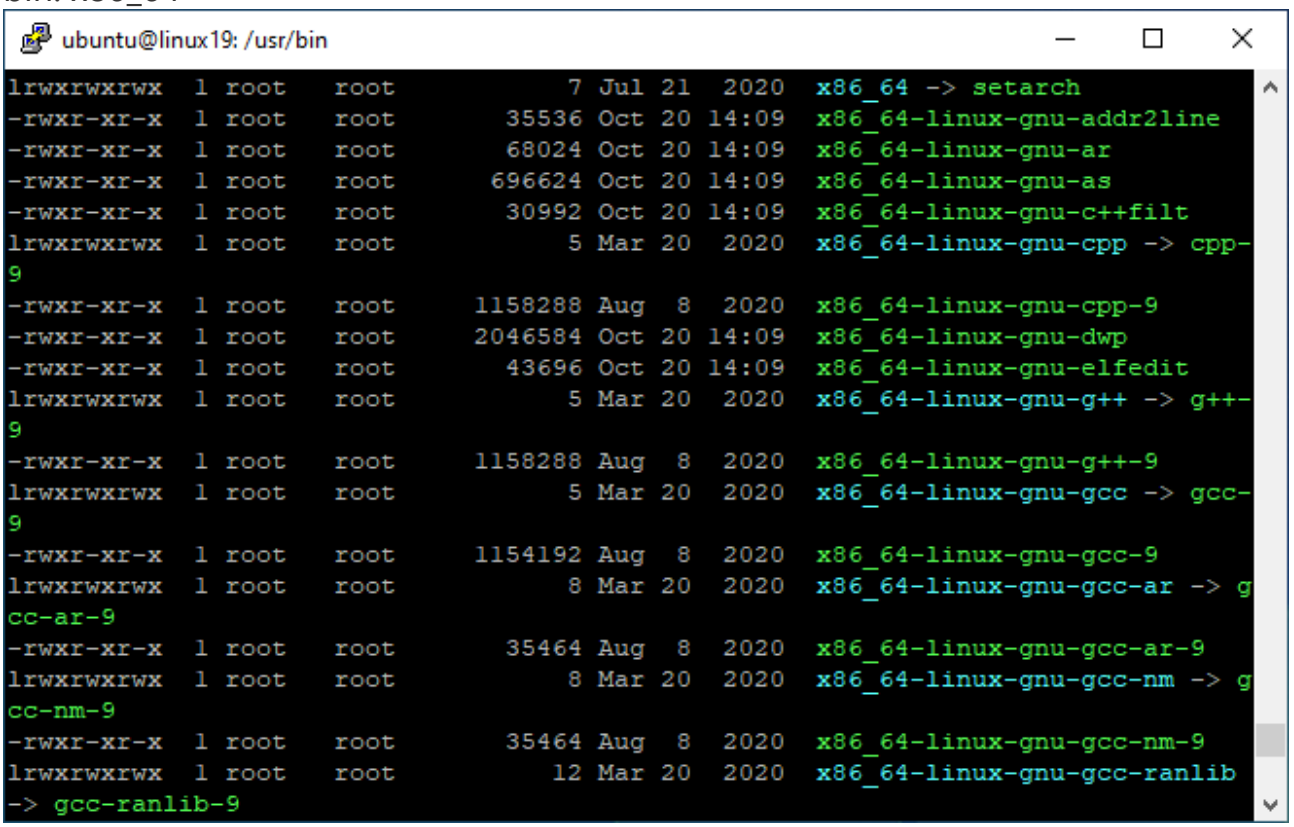
code usually processed like SW by VM program, acts like an assembler

- F00F bug

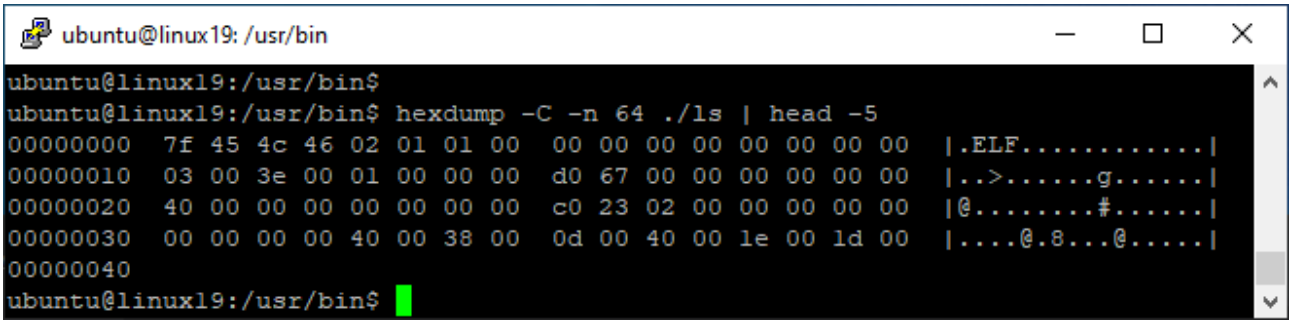
design flaw in many Pentium PCUs, can result to processor not working until rebooted.

- Use Linux file command to determine the details of /usr/bin/ls and /usr/lib/sudo/sudoers.so
  - What is the CPU architecture it was compiled to?

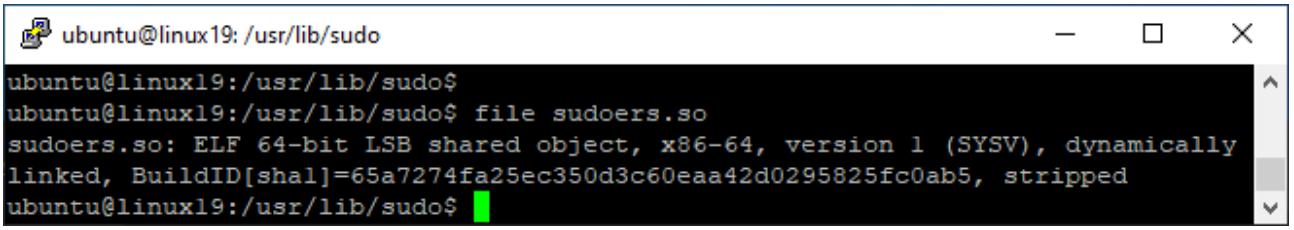
bin: x86_64



- What is the executable file format (or Extensible Linking Format) of /usr/bin/ls?



- What kind of file is /usr/lib/sudo/sudoers.so?



- CPUs and your smart phone:
  - What is the CPU/SoC of your smart phone?

1.7GHz octa-core (2x2GHz + 6x1.7GHz)

- What is the cache size(s) of the CPU/SoC?

L1: 64 KB

L2: 256 KB

L3: 1 MB

- What is the maximum clock speed it operates?

2GHz

- Who manufactured it?

Samsung electronics

- What is the CPU architecture used? How many bits?

2x 2 GHz – Kryo 460 Gold (Cortex-A76)

6x 1.7 GHz – Kryo 460 Silver (Cortex-A55)

- Describe these operating system and computing related terms and concepts shortly:
  - What is the difference between kernel and operating system?

  Kernel is a central part of an OS, a bridge between the SW and HW.

  - Operating system device driver

  provides a software interface to hardware device

  - Monolithic kernel

  A big kernel containing lots of information, static, linked modules are already included in kernel

  - Microkernel

  Dynamically loads modules

- Linux kernel module

Statically or dynamically linkable into kernel

- x86 privilege rings (protection rings)

controls how much resources available to program, 0-3; ring 3 is in contact with kernel itself

- BIOS

basic input output system, controls system from boot until OS takes control

- Bootloader

program which boots computer

- Library

Executable files can link and use libraries either dynamically or statically

- System call

OS Kernel function call programmatic way, simple commands

- Object file

Source file is compiled to object file, not directly operable

- Compiler

Turns statements into code the computer understands

- Interpreter

CLI between OS and user

- Linker

Links object files together and makes them executable

- Dynamic linker

Can be called upon whenever, even during execution

- Emulator

"simulator", simulates other devices e.g., phone

- TempleOS

Lightweight OS developed by schizophrenic programmer Terry A. Davis. Biblical, "God's Temple" OS.

- Contiki

very networked OS focusing on low-power IoT devices i.e., street lights turning on.

- FreeRTOS

real-time kernel for computer systems.

- RIOT

open-source OS for IoT devices.

- Zephyr

secure and safe RTOS for networked devices.

- OpenBSD

free multi-platform open-source UNIX-like OS

- FreeBSD

quite same as openBSD, just has access to more 3rd party apps

- Study x86 instruction listing and answer:
  - What is NOP instruction?

    no operation instruction

  - What is the x86 opcode for NOP instruction?

0x90

- What is JMP instruction?

  unconditional jump, 0xE9 to 0xEB, 0xFF/4, 0xFF/5; transfers flow of execution to change PC

- Read this blog post: https://embeddedbits.org/reverse-engineering-router-firmware-with-binwalk/ and answer:
  - What is Binwalk?

    SW which enables analyzing (and therefore reverse engineering) firmware.

  - What is U-boot?

    bootloader,

  - What is BusyBox?

    combines many common UNIX utilities to single executable, (multi-call binary)

  - What is the CPU architecture of Linux kernel which was extracted from the firmware?

    multi-call binary

Week 4

- Firewall tasks as root user:
  - Download this simple firewall example to your server and rename it to firewall.bash. Move that file to the /etc directory and set permissions to 700

```
ubuntu@linux19:/etc$ sudo wget -O firewall.bash
https://tl.oamk.fi/cdos/dl/firewall.txt

--2021-12-08 14:21:30--  https://tl.oamk.fi/cdos/dl/firewall.txt

Resolving tl.oamk.fi (tl.oamk.fi)... 193.167.100.28

Connecting to tl.oamk.fi (tl.oamk.fi)|193.167.100.28|:443... connected.

HTTP request sent, awaiting response... 200 OK

Length: 1878 (1.8K) [text/plain]

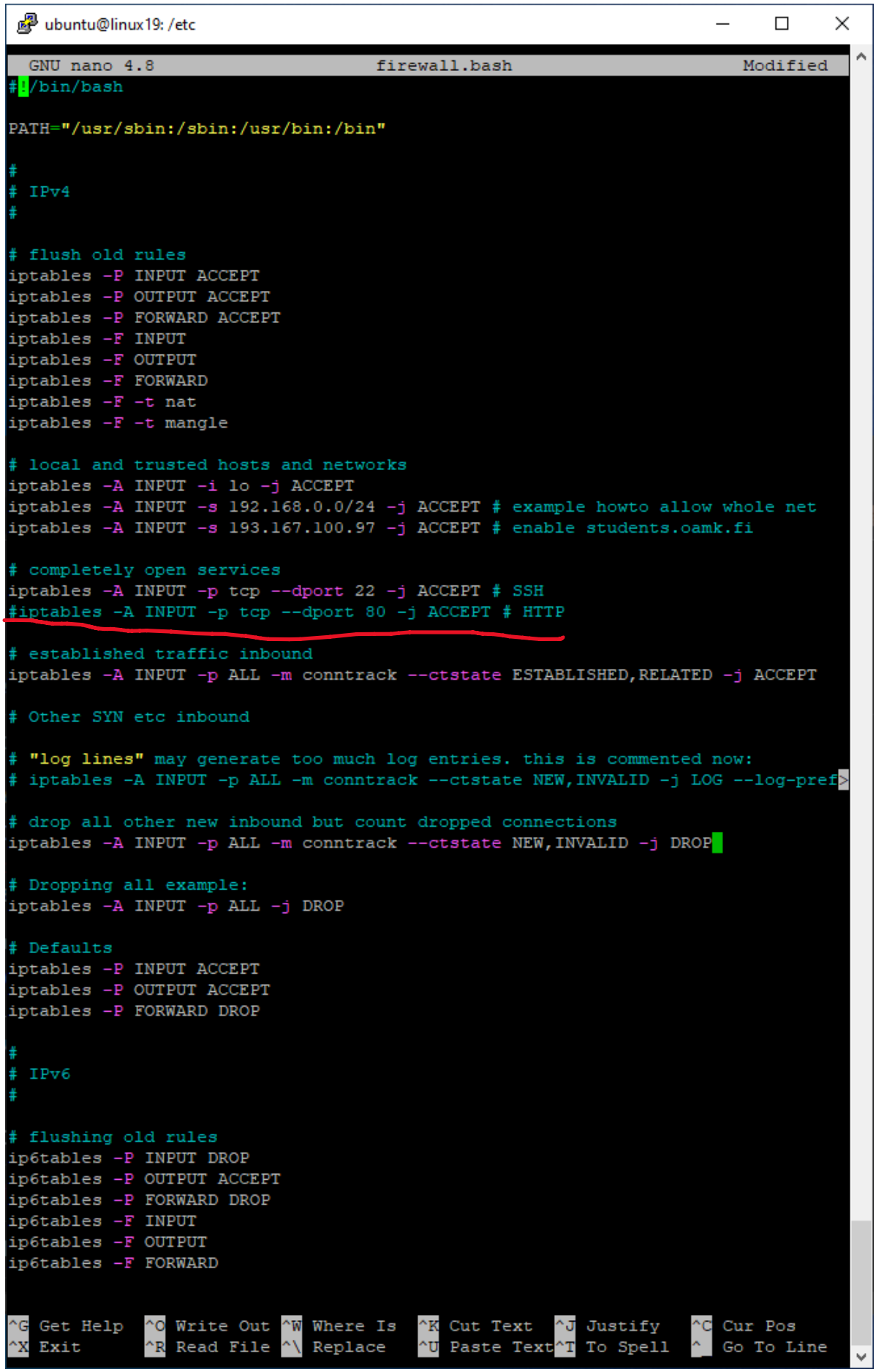
Saving to: 'firewall.bash'


firewall.bash       100%[===================>]   1.83K  --.-KB/s    in 0s


2021-12-08 14:21:30 (47.4 MB/s) - 'firewall.bash' saved [1878/1878]

ubuntu@linux19:/etc$ sudo chmod 700 firewall.bash
```

  - Study the contents and logic of the script (done)
  - Run the script as a root user and use iptables -L -n -v to see the packet counters
  - Comment out the line allowing inbound TCP/80 traffic and rerun the firewall script


ubuntu@linux19: /etc
GNU nano 4.8 firewall.bash Modified
#!/bin/bash
PATH="/usr/sbin:/sbin:/usr/bin:/bin"
#
IPv4
#
flush old rules
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -F INPUT
iptables -F OUTPUT
iptables -F FORWARD
iptables -F -t nat
iptables -F -t mangle
local and trusted hosts and networks
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -s 192.168.0.0/24 -j ACCEPT # example howto allow whole net
iptables -A INPUT -s 193.167.100.97 -j ACCEPT # enable students.oamk.fi
completely open services
iptables -A INPUT -p tcp --dport 22 -j ACCEPT # SSH
#iptables -A INPUT -p tcp --dport 80 -j ACCEPT # HTTP
established traffic inbound
iptables -A INPUT -p ALL -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
Other SYN etc inbound
"log lines" may generate too much log entries. this is commented now:
iptables -A INPUT -p ALL -m conntrack --ctstate NEW,INVALID -j LOG --log-pref
drop all other new inbound but count dropped connections
iptables -A INPUT -p ALL -m conntrack --ctstate NEW,INVALID -j DROP
Dropping all example:
iptables -A INPUT -p ALL -j DROP
Defaults
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP
#
IPv6
#
flushing old rules
ip6tables -P INPUT DROP
ip6tables -P OUTPUT ACCEPT
ip6tables -P FORWARD DROP
ip6tables -F INPUT
ip6tables -F OUTPUT
ip6tables -F FORWARD
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line

    - Enable logging lines in your firewall script and try to access your web server again (you shouldn't be able to access the web service anymore with web browser) and use tail -f /var/log/kern.log to follow linux kernel log file
    - Enable inbound TCP/80 again by removing the comment. Check that you can access your web server again with browser
    - Try to ping (IPv4) your server. It should not answer when the host firewall is now denying the new and unknown traffic. Modify your firewall script to allow inbound ICMP protocol traffic from all IPv4 addresses. Rerun your firewall script. Try to ping your server again.
- Study and explain shortly following commands and concepts:
    - sh, tcsh, bash, zsh

      sh: CL interpreter, executes commands

      tcsh: CL shell based on C shell

      bash: supports sh features and more

      zsh: sh basis with some features from bash

    - screen and tmux

      screen: push running apps to background and pull forward when you want to see them

      tmux: allows multitasking in terminal window

    - ps

      process status: shows info about currently running processes

    - jobs

      shows status of jobs started in the current terminal window

    - fg, bg

      fg: foreground = process which currently is in terminal window (shell)

      bg: background = suspends process and sends to background

- top, htop

top: lets users monitor processes and system resource usage

htop: monitor the system's vital resources or server's processes in real time (“top on steroids” :D)

- nice, renice

nice: execute a program/process with modified scheduling priority

renice: change the scheduling priority of an already running process

- su, sudo

su: switches to root user acc (needs pw)

sudo: executes command as root user

- sleep

  suspends program for specified time

- xargs

  converts standard line input to CL execution

- nohup

  no hangup = ignores HUP signal

- kill

  terminate processes manually

- pkill, killall

  pkill: terminate processess based on attributes and names etc.

  killall: killing any running process on the system based on a given name

- w, who

who: shows who are logged in

w: who is logged on and what they are doing and more additional data *who* doesn't show

- write, wall

write: send a message to another user to terminal

wall: displays a message on the terminals of all logged-in users

- aliases

define new commands by substituting a string for the first token of a simple command.

- source, .bashrc

  source: r + x content of file

  .bashrc: script file that's executed when a user logs in, hidden

- shell build-in variables, export

- How and when you start new shells? How to exit a shell?

  by executing bash or other sh ending operations/programs you start new shells. I exit shells by typing ctrl+d or ctrl+c depending on what I'm aiming to do

- Think reasons when endless while loops may be useful to run processes?

  when recording data for long times manually, multiple choice tasks, like menus with many options on what to do.

With your personal Linux host or with students.oamk.fi:

- Add shell alias "diskusage" to your shell startup-files (example .bashrc). Alias should print only current disk usage of your home directory.

```
ubuntu@linux19:/$ sudo gedit ~/.bashrc
Unable to init server: Could not connect: Connection refused

(gedit:14697): Gtk-WARNING **: 18:33:16.455: cannot open display:
ubuntu@linux19:/$
```

- Create shell alias “pp” which requires one parameter and will print all running processes with that name. Usage example:

ubuntu@linux19:/$ alias pp='ps -C'

```
tkorpela$ pp sleep \
root 21109  0.0  0.1  4084  556 pts/8  S  20:02   0:00 sleep 100 \
root 21111  0.0  0.1  4084  556 pts/8  S  20:03   0:00 sleep 100 \
root 21113  0.0  0.1  3684  556 pts/8  S  20:03   0:00 grep sleep \
```

- Which directories are currently in your PATH variable?

```
ubuntu@linux19:/$ echo $PATH

/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/
local/games:/snap/bin
```

- How do you start process directly into background when entering a command?

enter the command followed by “&”

- Start few sleep 60 - processes (one minute idle loop) to the background. How can you find and terminate them all with one-liner? Try not to use pkill, killall or xargs -commands.

```
ubuntu@linux19:~$ sleep 60 &
[4] 14903
ubuntu@linux19:~$ sleep 60 &
[5] 14904
ubuntu@linux19:~$ sleep 60 &
[6] 14905
ubuntu@linux19:~$ jobs
[3]   Running                 sleep 60 &
[4]   Running                 sleep 60 &
[5]-  Running                 sleep 60 &
[6]+  Running                 sleep 60 &
ubuntu@linux19:~$ kill -9 `jobs -p`
[3]   Killed                  sleep 60
[4]   Killed                  sleep 60
[5]-  Killed                  sleep 60
[6]+  Killed                  sleep 60
ubuntu@linux19:~$
```

- How would you do the previous killing task with xargs?

```
ubuntu@linux19:~$ sleep 60 &
[1] 14925
ubuntu@linux19:~$ sleep 60 &
[2] 14926
ubuntu@linux19:~$ sleep 60 &
[3] 14927
ubuntu@linux19:~$ sleep 60 &
[4] 14928
ubuntu@linux19:~$ jobs
[1]   Running                 sleep 60 &
[2]   Running                 sleep 60 &
[3]-  Running                 sleep 60 &
[4]+  Running                 sleep 60 &
ubuntu@linux19:~$ jobs -p | xargs kill
[1]   Terminated              sleep 60
[2]   Terminated              sleep 60
[3]-  Terminated              sleep 60
[4]+  Terminated              sleep 60
ubuntu@linux19:~$
```

- Start one 1000 second sleep to the foreground.

  ubuntu@linux19:~$ sleep 1000

- How do you suspend it?
- How do you list current jobs?

```
ubuntu@linux19:~$ sleep 1000 &
[1] 14945
ubuntu@linux19:~$ fg %1
sleep 1000
^Z
[1]+  Stopped                 sleep 1000
ubuntu@linux19:~$ jobs
[1]+  Stopped                 sleep 1000
ubuntu@linux19:~$
```

- How do you get previous sleep process back to foreground?

  ubuntu@linux19:~$ jobs

  [1]+ Running sleep 1000 &

  ubuntu@linux19:~$ fg %1

- Suspend process again and send it to background.

```
ubuntu@linux19:~$ kill -CONT %1
```

```
ubuntu@linux19:~$ jobs -p | xargs kill

kill: (14945): No such process

[1]+  Terminated              sleep 1000
```

- Kill previous sleep process from background.
- What is the difference between kill -9 and kill -1?

kill -1: All processes with a PID larger than 1 react.

kill -9: targets unresponsive processes which do not react to kill commands normally

- Delete unnecessary files created in this practice.

Week 5

- Study and explain shortly following commands and concepts:
    - cat, tac

cat: prints the lines in a file

tac: does the same but in reverse order

    - grep / egrep

grep: search file(s) for a pattern, often combined with other commands.

egrep: “extended” version, more features but slightly different

    - wc

      wordcount, plainly counts the words in a file but can be modified e.g. by adding -l to count lines

    - sort

      sort files by different parameters such as size or name

    - cut

      remove selected columns or other parts from file(s)

    - awk

      process and edit data

    - sed

      automatically edit files (e.g. replace things)

    - tr

      translate, replace patterns or characters by others.

    - expand, unexpand

expand: makes tabs in file(s) to spaces

unexpand: reverses this

- uniq

  removes duplicate lines in a file

- head

  represents start of file (first lines), head -n -5 prints 5 first lines etc.

- tail

  represents end of file

- echo

  repeats back what was written after it or the prints the file or can be modified even further

- join

  combines 2 files, usually line by line into spreadsheet style 2-columned approach

- paste

  parallel merging, kind of like paste but also multiple files

- tee

  saves printed standard input to file in addition to just printing it.

- nl

  numbers the lines in a file

With your personal Linux host or with students.oamk.fi:

- Use word counter and piping to count how many files or directories are in /usr/bin -directory?

```
ubuntu@linux19:/usr/bin$ echo * | wc
      1    1288   13277
```

- Use wget to download this irclog.txt and answers to these questions:
  - How many lines are in the file?

ubuntu@linux19:/irc$ wc -l irclog.txt

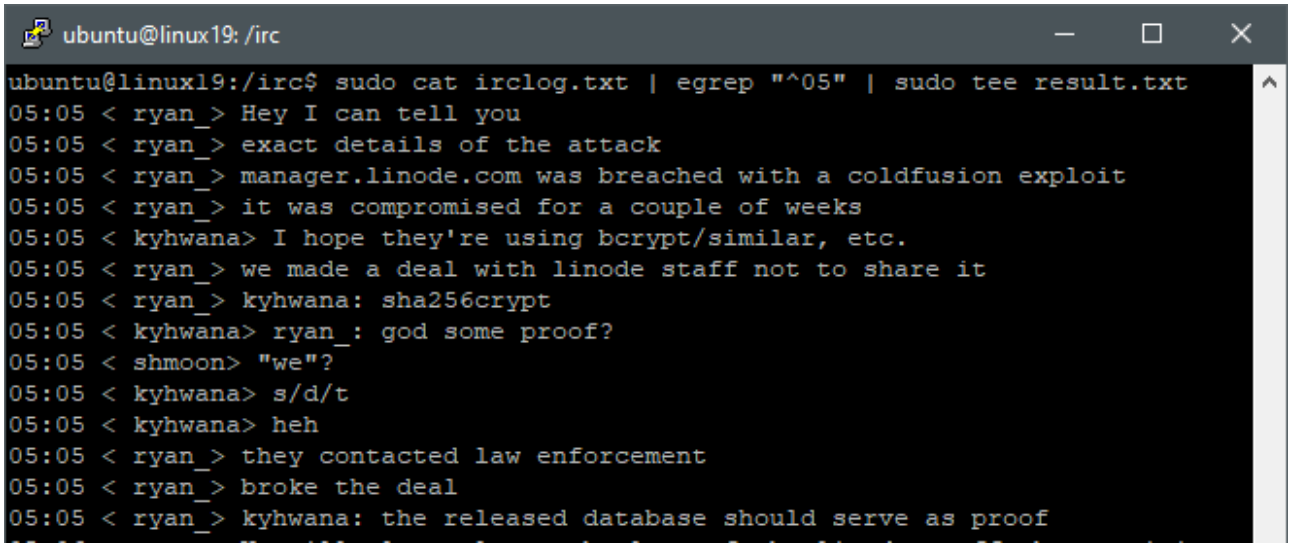
244 irclog.txt

  - How many characters are in the file?

ubuntu@linux19:/irc$ wc -m irclog.txt

16341 irclog.txt

  - List only lines where the timestamp starts with 05 and save the output to a file called result.txt

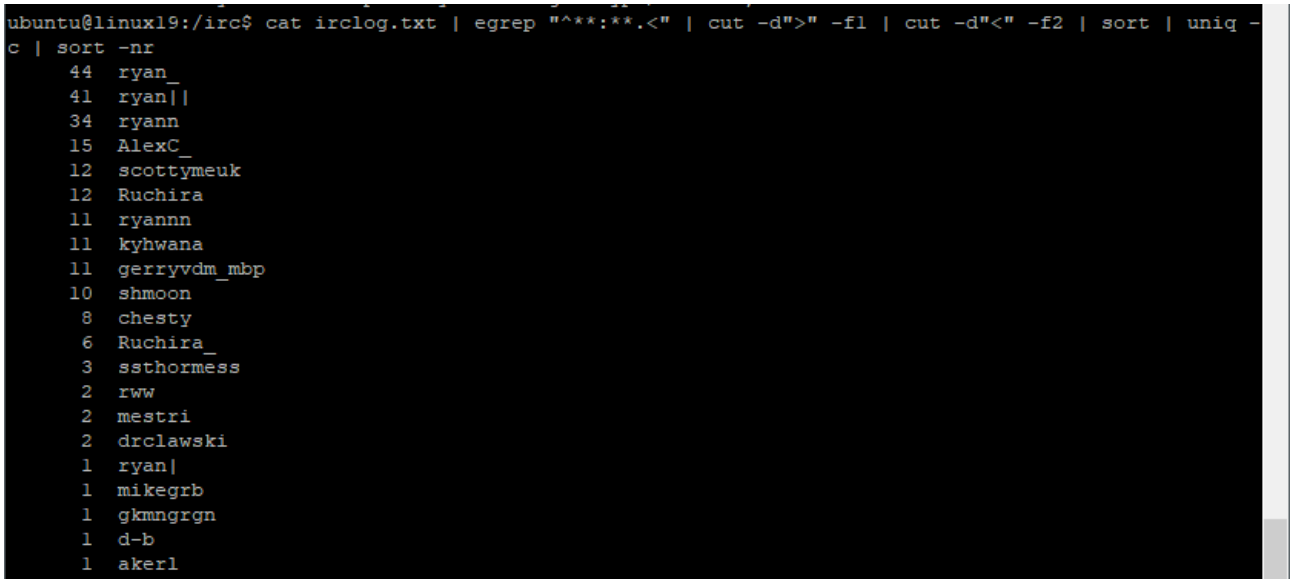



```
ubuntu@linux19:/irc$ ls -la
total 40
drwxr-xr-x  2 root root  4096 Dec 13 17:13 .
drwxr-xr-x 21 root root  4096 Dec 12 20:32 ..
-rw-r--r--  1 root root 16341 Oct 19 15:32 irclog.txt
-rw-r--r--  1 root root 12873 Dec 13 17:13 result.txt
```

- Print result.txt in reverse order

ubuntu@linux19:/irc$ tac result.txt

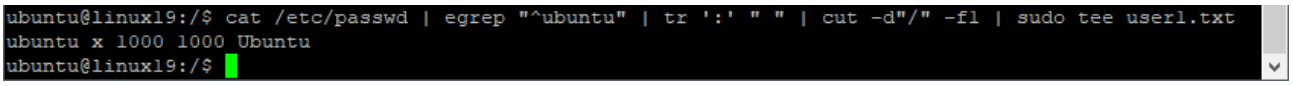
- Create numerical statistics from the irclog.txt file: How many lines each nickname wrote. Use only those lines where someone actually said something and ignore the all other lines



- List only 5 largest files from /usr/bin -directory. (Starting from largest file.)

```
ubuntu@linux19:/usr/bin$ ls -la -S | head -6
total 152156
-rwxr-xr-x  1 root   root    22443760 Sep  9 17:34 snap
-rwxr-xr-x  1 root   root     5490488 Sep 28 19:10 python3.8
-rwxr-xr-x  1 root   root     5410280 Aug  3 11:53 gnome-control-center
-rwxr-xr-x  2 root   root     3478464 Oct 19  2020 perlcd
-rwxr-xr-x  2 root   root     3478464 Oct 19  2020 perl5.30.0
ubuntu@linux19:/usr/bin$
```

- Print only usernames, UID and GID numbers from /etc/passwd -file. Replace all colons with a whitespace. Redirect output to file a “users.txt” in your home directory.



- Tip: In this example line from /etc/passwd the UID = 101 and GID = 50:

```
username:x:101:50:Teemu Korpela:/home/tkorpela:/bin/bash
```

- Use text editor nano to create a points.txt file to your home directory with following content. This list presents first names and some game scores. Who has most points, wins
    - List contents of points.txt in alphabetic order to STDOUT

```
ubuntu@linux19:/$ sudo cat points.txt | sort

Erkki:7

Esko:2

Jaska:5

Juha-Pekka:6

Matti:8

Mika:3

Teemu:4

Timo:1
```

    - List contents of file on to STDOUT, but now order is score based. List only best three players with most points

```
ubuntu@linux19:/$ cat points.txt | cut -d":" -f2 | sort -r -n | head -3

8

7

6
```

    - How do you list only player names and filter all other data

```
ubuntu@linux19:/$ cat points.txt | cut -d":" -f1

Teemu

Matti

Juha-Pekka

Timo

Mika

Esko

Jaska

Erkki
```

- List only first three characters from the beginning of each line of points.txt

```
ubuntu@linux19:/$ cat points.txt | cut -b -3

Tee

Mat

Juh

Tim

Mik

Esk

Jas

Erk
```

- List points.txt but translate all characters to upper-case

```
ubuntu@linux19:/$ cat points.txt | tr "[:lower:]" "[:upper:]"

TEEMU:4

MATTI:8

JUHA-PEKKA:6

TIMO:1
```

```
MIKA:3

JASKA:5

ERKKI:7

ESKO:2
```

- List points.txt so that points are printed before names

```
ubuntu@linux19:/home$ cat points.txt | tr ":" " " | awk '{print $2,$1}' | tr " "
":"

4:Teemu

8:Matti

6:Juha-Pekka

1:Timo

3:Mika

2:Esko

5:Jaska

7:Erkki
```

- Sort points.txt in alphabetic order and add line numbers in front of lines

```
ubuntu@linux19:/home$ sort points.txt | cat -n

     1  Erkki:7

     2  Esko:2

     3  Jaska:5

     4  Juha-Pekka:6

     5  Matti:8

     6  Mika:3

     7  Teemu:4

     8  Timo:1
```

points.txt file:

```
Teemu:4
Matti:8
Juha-Pekka:6
Timo:1
Mika:3
Esko:2
Jaska:5
Erkki:7
```

- How do you list last 5 lines from /etc/passwd file?

```
ubuntu@linux19:/etc$ cat passwd | tail -5
```

- How do you list first 5 lines from /etc/passwd file?

```
ubuntu@linux19:/etc$ cat passwd | head -5
```

- What does tail -f file do?

updates constantly, dynamically shows tail files

- Fetch current weather in Oulu with lynx (TIP: if there is no lynx, install it with: sudo apt install lynx). The command to download Oulu's weather data is: lynx -dump https://weather.willab.fi/weather.html

- Filter the output so that only temperature is displayed and nothing else

```
ubuntu@linux19:/$ lynx -dump http://weather.willab.fi/weather.html | grep "°"
| head -1

   -0.8 °C
```

- Use wget to get stock market textfile

```
ubuntu@linux19:/stonks$ sudo wget https://tl.oamk.fi/cdos/dl/stocks.txt

--2021-12-14 22:09:31--  https://tl.oamk.fi/cdos/dl/stocks.txt

Resolving tl.oamk.fi (tl.oamk.fi)... 193.167.100.28

Connecting to tl.oamk.fi (tl.oamk.fi)|193.167.100.28|:443... connected.

HTTP request sent, awaiting response... 200 OK

Length: 1239 (1.2K) [text/plain]

Saving to: ‘stocks.txt’


stocks.txt             100%[=========================>]   1.21K  --.-KB/s    in
0s


2021-12-14 22:09:31 (154 MB/s) - ‘stocks.txt’ saved [1239/1239]


ubuntu@linux19:/stonks$
```

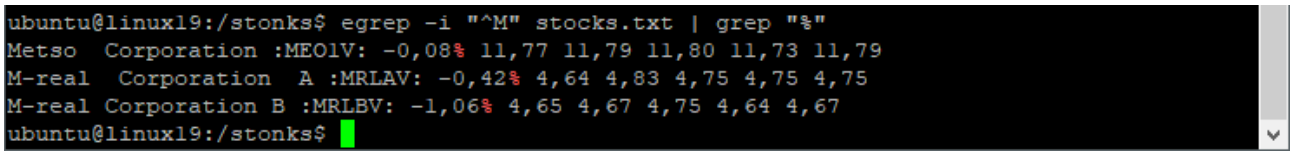
Example line and explanation from file:

```
Name                    code    change buy  sell lowest highest last
Fiskars  Corporation :FISAS: -0,36% 8,35 8,39  8,44   8,37   8,37
```

- Use grep (or egrep) and regular expressions to list only companies with “I” anywhere in in code part.

```
ubuntu@linux19:/stonks$ grep "i" stocks.txt
Metalliteollisuus
Componenta  Corporation :CTH1V: +2,06% 5,90 5,99 5,95 5,89 5,95
Fiskars  Corporation :FISAS: -0,36% 8,35 8,39 8,44 8,37 8,37
Kone  Corporation B :KONBS: -0,67% 60,92 61,00 62,01 60,73 60,99
Metso  Corporation :MEO1V: -0,08% 11,77 11,79 11,80 11,73 11,79
Nordic Aluminium Plc :NOA1V: -0,04% 9,32 9,49 0,00 0,00 9,49
Rautaruukki  Corporation :RTRKS: +0,24% 8,46 8,50 8,52 8,42
Wärtsilä  Corporation  A  :WRTAV: +3,47% 16,82 17,00 17,00 16,74 17,00
Wärtsilä  Corporation  B  :WRTBV: +1,59% 17,20 17,21 17,29 16,93 17,21
Metsäteollisuus
M-real  Corporation  A :MRLAV: -0,42% 4,64 4,83 4,75 4,75 4,75
M-real Corporation B :MRLBV: -1,06% 4,65 4,67 4,75 4,64 4,67
Stromsdal Corporation B :STMBS: +0,43% 2,00 2,06 0,00 0,00 2,10
UPM-Kymmene  Corporation  :UPM1V: -0,66% 16,59 16,60 16,80 16,54 16,59
ubuntu@linux19:/stonks$
```

- List (only) company names and stock values starting with character “M”.



Output should be:

```
Metso  Corporation :MEO1V: -0,08% 11,77 11,79 11,80 11,73 11,79
M-real  Corporation  A :MRLAV: -0,42% 4,64 4,83 4,75 4,75 4,75
M-real Corporation B :MRLBV: -1,06% 4,65 4,67 4,75 4,64 4,67
```

- Print line only if the company name begins with a character “R” and last stock value is 8,xx

```
ubuntu@linux19:/stonks$ cat stocks.txt | egrep -i "^R" stocks.txt | grep '.8,[0-
9][0-9]$'

Rautaruukki  Corporation :RTRKS: +0,24% 8,46 8,50 8,52 8,42

Rocla Oyj :ROC1V: -0,60% 8,20 8,25 8,25 8,20 8,25
```

Output should be:

```
Rautaruukki  Corporation :RTRKS: +0,24% 8,46 8,50 8,52 8,42
Rocla Oyj :ROC1V: -0,60% 8,20 8,25 8,25 8,20 8,25
```

- List all companies except the names starting with characters “R” or “W”

```
ubuntu@linux19:/stonks$ cat stocks.txt | egrep -v '^R|^W' | grep ":"

Componenta  Corporation :CTH1V: +2,06% 5,90 5,99 5,95 5,89 5,95

Fiskars  Corporation :FISAS: -0,36% 8,35 8,39 8,44 8,37 8,37

KCI  Konecranes  Plc :KCI1V: +0,06% 34,10 34,16 34,20 34,05 34,17

Kone  Corporation B :KONBS: -0,67% 60,92 61,00 62,01 60,73 60,99

Metso  Corporation :MEO1V: -0,08% 11,77 11,79 11,80 11,73 11,79

Nordic Aluminium Plc :NOA1V: -0,04% 9,32 9,49 0,00 0,00 9,49

Outokumpu  Oyj  :OUT1V: +0,98% 13,36 13,37 13,42 13,27 13,36

Ponsse Oyj 1 :PON1V:   +0,66% 15,16 15,20 15,25 15,13 15,20

M-real  Corporation  A :MRLAV: -0,42% 4,64 4,83 4,75 4,75 4,75
```

```
M-real Corporation B :MRLBV: -1,06% 4,65 4,67 4,75 4,64 4,67

Stora  Enso Oyj A :STEAV: +1,31% 11,50 11,58 11,58 11,53 11,58

Stora  Enso Oyj R :STERV: -1,04% 11,37 11,38 11,49 11,34 11,38

Stromsdal Corporation B :STMBS: +0,43% 2,00 2,06 0,00 0,00 2,10

UPM-Kymmene  Corporation  :UPM1V: -0,66% 16,59 16,60 16,80 16,54 16,59
```

- List only those stocks which have positive change value (i.e. +xx,xx%) in the list

```
ubuntu@linux19:/stonks$ cat stocks.txt | grep "+"

Componenta  Corporation :CTH1V: +2,06% 5,90 5,99 5,95 5,89 5,95

KCI  Konecranes  Plc :KCI1V: +0,06% 34,10 34,16 34,20 34,05 34,17

Outokumpu  Oyj  :OUT1V: +0,98% 13,36 13,37 13,42 13,27 13,36

Ponsse Oyj 1 :PON1V:   +0,66% 15,16 15,20 15,25 15,13 15,20

Rautaruukki  Corporation :RTRKS: +0,24% 8,46 8,50 8,52 8,42

Wärtsilä  Corporation  A  :WRTAV: +3,47% 16,82 17,00 17,00 16,74 17,00

Wärtsilä  Corporation  B  :WRTBV: +1,59% 17,20 17,21 17,29 16,93 17,21

Stora  Enso Oyj A :STEAV: +1,31% 11,50 11,58 11,58 11,53 11,58

Stromsdal Corporation B :STMBS: +0,43% 2,00 2,06 0,00 0,00 2,10
```

- Get nimipaivat.txt (finnish name days) textfile from here nimipaivat.txt)
- From nimipaivat.txt, find out how many names start with a letter A and end to a letter i?

```
root@linux19:/nimipaivat# egrep "^A" nimipaivat.txt | grep "i.[0-9]" | wc -l
34
```

- How can you convert previous names to lower-case?

```
root@linux19:/nimipaivat# cat nimipaivat.txt | egrep "^A" nimipaivat.txt | grep
"i.[0-9]" | tr "[:upper:]" "[:lower:]"
```

- From previous names, who are celebrating in December?

```
root@linux19:/nimipaivat# cat nimipaivat.txt | egrep "^A" nimipaivat.txt | grep
"i.[0-9]" | grep "12.$"

Airi 4.12.

Anneli 9.12.

Anni 9.12.

Annikki 9.12.

Auli 16.12.

Aulikki 16.12.

Aatami 24.12.
```

- From all names in nimipaivat.txt, search those who celebrate either 1st, 2nd or 3rd day in any month.

```
root@linux19:/nimipaivat# cat .ipaivat.txt | grep '[a-z].1.[0-9]\|[a-z].3.[0-
9]\|[a-z].3.[0-9]'

Linnea 3.8.

Maire 1.8.

Meri 3.12.

Nea 3.8.

Orvokki 3.6.

Outi 3.5.

Pulmu 1.4.

Raita 1.4.

Riitta 1.2.

Soila 3.9.

Soile 3.9.

Soili 3.9.

Valpuri 1.5.

Vanamo 3.8.
```

```
Vappu 1.5.

Vellamo 3.12.

Viola 3.6.

Aaro 1.7.

Aaron 1.7.

Alpi 1.3.

Alpo 1.3.

Alvi 1.3.

Arvo 3.7.

Elmer 3.1.

Elmeri 3.1.

Elmo 3.1.

Kauko 3.3.

Lyly 1.11.

Nikodemus 1.6.

Oskari 1.12.

Pirkka 1.9.

Pyry 1.11.

Raimo 3.10.

Raine 1.10.

Rainer 1.10.

Rauno 1.10.

Sampo 3.4.

Teemu 1.6.

Terho 3.11.

Valo 3.2.
```

- Use lynx -dump “url” to print webpage to STDOUT. Filter output so that you will get the current Lotto numbers, but nothing more from the

webpage. Lotto numbers are available here: http://www.yle.fi/tekstitv/txt/P471_01.html

```
root@linux19:/lotto# lynx -dump https://yle.fi/tekstitv/txt/471_0001.htm | head
-14 | tail -4 | tr -d [A-Z] | tr -d [:blank:] | tr -d '\t'| tr -d "Ä" | tr -d
':'| tr ',' '\n'| sed '9d'4

14

16

20

24

30

33

22

21
```

- Delete unnecessary files created in this practice.

HTTP access to XML:

- Use Gnu tools or Cmder's Curl and Grep (and maybe other command line tools) to create a one-liner, which downloads the XML file and parses current temperature from VTT's weather station. One-liner must print only the current temperature in Oulu and nothing else. Command line one-liner and output should look something like this:
- ```
  curl -s -L http://weather.willab.fi/weather.xml |
  _replace_with_your_commands_options_and_code_
  12.3
  ```

  ```
  root@linux19:/lotto# curl -s -L http://weather.willab.fi/weather.xml |
  egrep "^<tempnow" | cut -d">" -f2 | cut -d"<" -f1
  -0.9
  ```

- Combine these two files to a single file with command line Gnu text tools. The first file has timestamps and the second file has IP addresses. Use : as delimeter between columns. Output should look something like this:

```
root@linux19:/combo# paste -d ":" firstfile.txt secondfile.txt
```
```

- ...
- Sat Apr 11 11:03:42 2020:185.176.27.26
- Sat Apr 11 11:03:43 2020:188.26.0.66
- Sat Apr 11 11:04:15 2020:185.176.27.34
- Sat Apr 11 11:04:57 2020:87.251.74.250
- Sat Apr 11 11:05:00 2020:94.102.52.57
  ...

Week 6

- Study and use this simple incremental / full backup example script. You need to create proper directories first.
- For automatic backups do as a root:
    - Use wget to download the backup script example
    - Move the downloaded file to /etc/cron.daily/ and rename it to backup

    root@linux19:/increment# mv backup.txt /etc/cron.daily

    - Set permissions to 700 (root as owner)

    root@linux19:/etc/cron.daily# chmod 700 backup.txt

    - Create directories /mnt/backup /mnt/backup/full and /mnt/backup/increment

```
root@linux19:/# mkdir /mnt/backup
root@linux19:/# cd /mnt/backup
root@linux19:/mnt/backup# mkdir full
root@linux19:/mnt/backup# mkdir increment
root@linux19:/mnt/backup# ls -la
total 16
drwxr-xr-x 4 root root 4096 Dec 15 13:33 .
drwxr-xr-x 3 root root 4096 Dec 15 13:33 ..
drwxr-xr-x 2 root root 4096 Dec 15 13:33 full
drwxr-xr-x 2 root root 4096 Dec 15 13:33 increment
root@linux19:/mnt/backup#
```

    - Run the backup script from command line and check that it worked

root@linux19:/etc/cron.daily# chmod +x backup.txt

root@linux19:/etc/cron.daily# ./backup.txt

    - Uncomment initial delay lines from the script to create a random delay before the backup script executes all those heavy disk IO (find and tar) backup operations

mydelay=$(echo $[$RANDOM%3000+1])

sleep $mydelay

- Create some temporary test directory to /tmp and copy one of those smaller incremental backup files there

root@linux19:/mnt/backup/increment# cp increment_home_2021-Dec-15.tar.gz /tmp/smalldir/

- Use tar to list contents of the copied package

```
root@linux19:/tmp/smalldir# tar -ztvf increment_home_2021-Dec-15.tar.gz

-rw------- ubuntu/ubuntu     1 2021-12-15 13:18
home/ubuntu/.config/pulse/97215da5226f44f992a154ed833d50b0-default-sink

-rw------- ubuntu/ubuntu     1 2021-12-15 13:18
home/ubuntu/.config/pulse/97215da5226f44f992a154ed833d50b0-default-source

-rw------- ubuntu/ubuntu 35200 2021-12-15 02:05 home/ubuntu/.bash_history

-rw-rw-r-- ubuntu/ubuntu   165 2021-12-14 22:09 home/ubuntu/.wget-hsts

-rw-r--r-- root/root        66 2021-12-14 17:42 home/points.txt
```

- Use tar / gzip to decompress package and check that you managed to extract all the files from the package

```
root@linux19:/tmp/smalldir# gzip -d increment_home_2021-Dec-15.tar.gz

root@linux19:/tmp/smalldir# tar xvf increment_home_2021-Dec-15.tar

home/ubuntu/.config/pulse/97215da5226f44f992a154ed833d50b0-default-sink

home/ubuntu/.config/pulse/97215da5226f44f992a154ed833d50b0-default-source

home/ubuntu/.bash_history

home/ubuntu/.wget-hsts
```

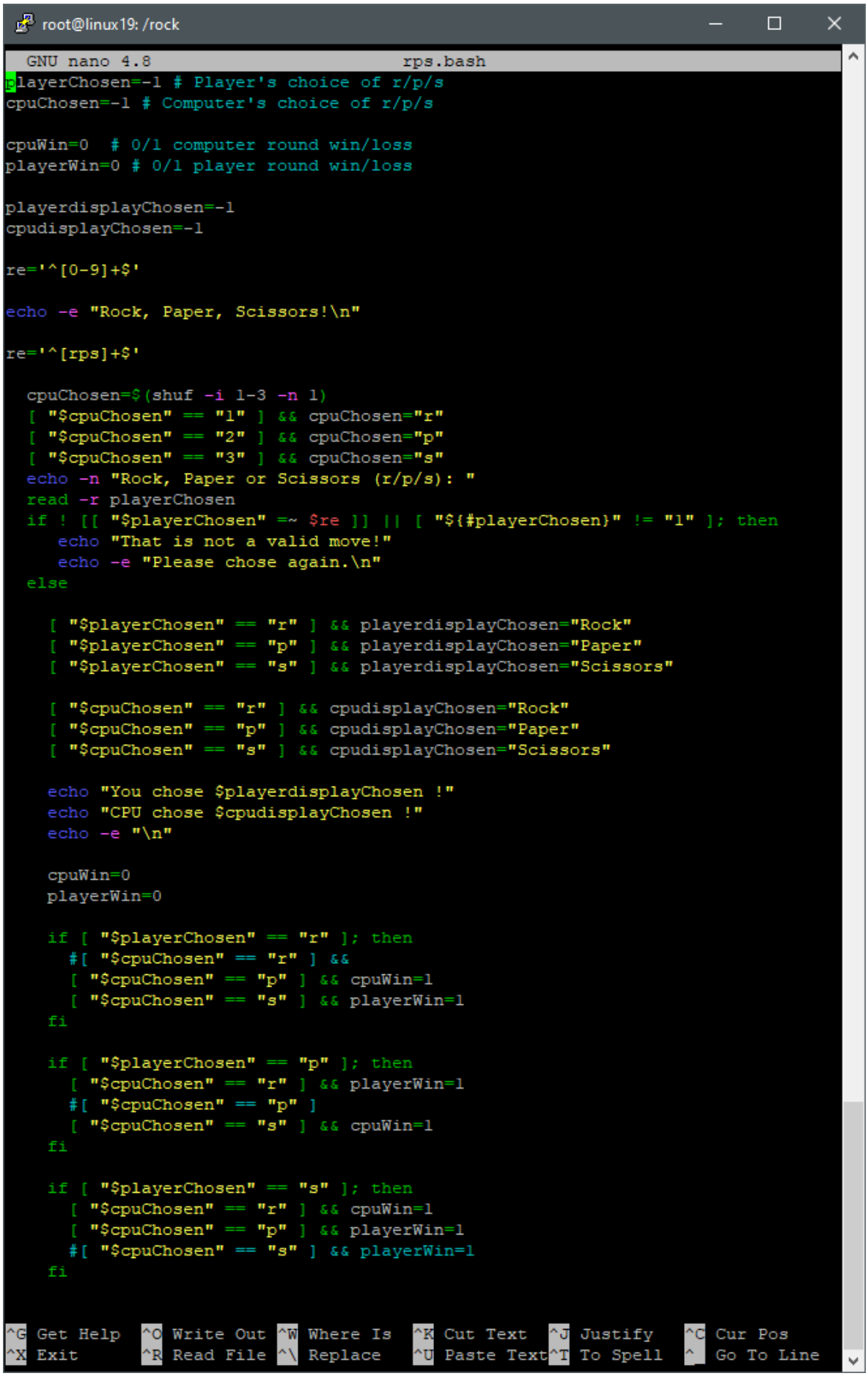
home/Create “Rock Scissor Paper”-game with Bash. Script will prompt user to pick either Rock, Scissor or Paper. Then script will randomise one option (computer player’s selection) and return results. Rules are: Rock wins

scissor. Paper wins rock. Scissor wins paper. Tip: Bash build-in $RANDOM variable returns random numbers. For example, numbers 0-9 would be:

```
echo $[RANDOM%10]
```

More advanced PRNG using /dev/urandom:

```
echo $[$(od -vAn -N2 -tu4 < /dev/urandom)%10]
```


root@linux19: /rock
GNU nano 4.8 rps.bash
playerChosen=-1 # Player's choice of r/p/s
cpuChosen=-1 # Computer's choice of r/p/s
cpuWin=0 # 0/1 computer round win/loss
playerWin=0 # 0/1 player round win/loss
playerdisplayChosen=-1
cpudisplayChosen=-1
re='^[0-9]+$'
echo -e "Rock, Paper, Scissors!\n"
re='^[rps]+$'
cpuChosen=$(shuf -i 1-3 -n 1)
["$cpuChosen" == "1"] && cpuChosen="r"
["$cpuChosen" == "2"] && cpuChosen="p"
["$cpuChosen" == "3"] && cpuChosen="s"
echo -n "Rock, Paper or Scissors (r/p/s): "
read -r playerChosen
if ! [["$playerChosen" =~ $re]] || ["${#playerChosen}" != "1"]; then
echo "That is not a valid move!"
echo -e "Please chose again.\n"
else
["$playerChosen" == "r"] && playerdisplayChosen="Rock"
["$playerChosen" == "p"] && playerdisplayChosen="Paper"
["$playerChosen" == "s"] && playerdisplayChosen="Scissors"
["$cpuChosen" == "r"] && cpudisplayChosen="Rock"
["$cpuChosen" == "p"] && cpudisplayChosen="Paper"
["$cpuChosen" == "s"] && cpudisplayChosen="Scissors"
echo "You chose $playerdisplayChosen !"
echo "CPU chose $cpudisplayChosen !"
echo -e "\n"
cpuWin=0
playerWin=0
if ["$playerChosen" == "r"]; then
#["$cpuChosen" == "r"] &&
["$cpuChosen" == "p"] && cpuWin=1
["$cpuChosen" == "s"] && playerWin=1
fi
if ["$playerChosen" == "p"]; then
["$cpuChosen" == "r"] && playerWin=1
#["$cpuChosen" == "p"]
["$cpuChosen" == "s"] && cpuWin=1
fi
if ["$playerChosen" == "s"]; then
["$cpuChosen" == "r"] && cpuWin=1
["$cpuChosen" == "p"] && playerWin=1
#["$cpuChosen" == "s"] && playerWin=1
fi
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line

```
    if [ "$cpuWin" == "0" ] && [ "$playerWin" == "0" ]; then
      echo "It's a draw!"
    elif [ "$playerWin" == 1 ]; then
      echo "You win!"
    elif [ "$cpuWin" == 1 ]; then
      echo "Computer wins :("
    fi

  fi

exit

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To Line
```

```
root@linux19: /rock
root@linux19:/rock# ./rps.bash
Rock, Paper, Scissors!

Rock, Paper or Scissors (r/p/s): p
You chose Paper !
CPU chose Rock !


You win!
root@linux19:/rock# ./rps.bash
Rock, Paper, Scissors!

Rock, Paper or Scissors (r/p/s): s
You chose Scissors !
CPU chose Paper !


You win!
root@linux19:/rock# ./rps.bash
Rock, Paper, Scissors!

Rock, Paper or Scissors (r/p/s): r
You chose Rock !
CPU chose Rock !


It's a draw!
root@linux19:/rock#
```

- Create a welcoming script which will check current time and will echo welcome message if time is:

```
nano greeting.bash

nano view:

time=$ (date +”%T”)
```

```
echo “Current time: $time”

hour= $ (date +”%H”)

if [ $hour -ge 6 -a $hour -lt 12 ]

then

          greet=”Good morning, $USER”

elif [ $hour -ge 12 -a $hour -lt 18 ]

then

          greet=”Good day, $USER”


elif [ $hour -ge 18 -a $hour -lt 22 ]

then

          greet=”Good evening, $USER”

else

          greet=”Good night, $USER”

fi


#to display the correct message:

echo $greet
```

Week 7

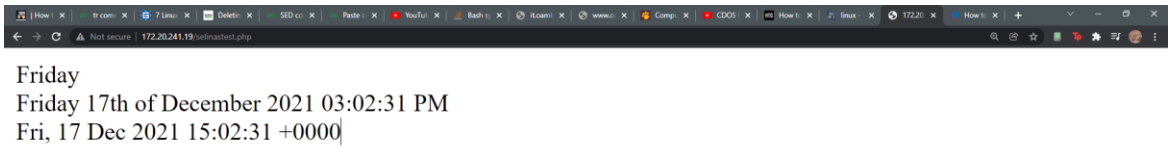
- Install PHP support to your Apache web server

```
ubuntu@linux19:/var/www/html$ php -v
PHP 8.0.13 (cli) (built: Nov 22 2021 09:50:43) ( NTS )
Copyright (c) The PHP Group
Zend Engine v4.0.13, Copyright (c) Zend Technologies
    with Zend OPcache v8.0.13, Copyright (c), by Zend Technologies
ubuntu@linux19:/var/www/html$
```

  - Add this example PHP script under the web server document root (/var/www/html)

root@linux19:/var/www/html# nano selinastest.php

  - Test that your server is executing the PHP script when requesting it with a web browser



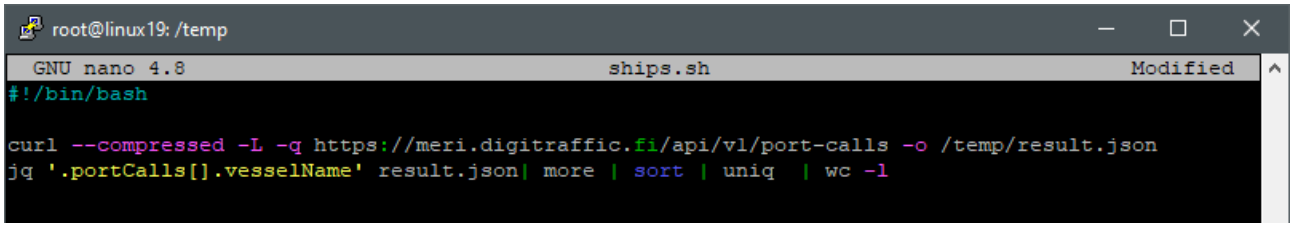
Friday
Friday 17th of December 2021 03:02:31 PM
Fri, 17 Dec 2021 15:02:31 +0000

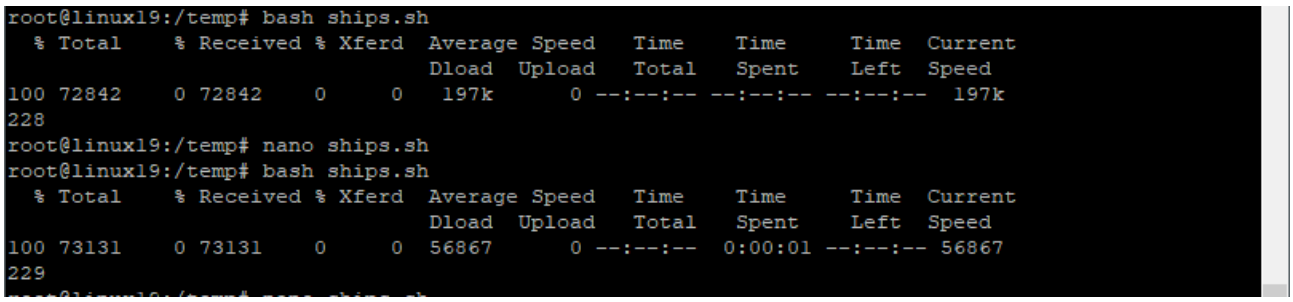
  - Example script printing few date function outputs:

```
<?php
date_default_timezone_set('UTC');
echo date("l");
echo ("<br>");
echo date('l jS \of F Y h:i:s A');
echo ("<br>");
echo date(DATE_RFC2822);
?>
```

- Create a Bash script which will fetch and process data from [marine traffic API]

- Script should download the JSON-file from marine traffic portcalls API and print how many ships are currently there? (Search vesselName from the JSON)





- Command line example with curl command: curl --compressed -L https://meri.digitraffic.fi/api/v1/port-calls -o /tmp/result.json
- Filter the /tmp/result.json file data with jq or with GNU text utilities such as sed, awk, cut, grep etc.