



CYBERSECURITY SEMINARS Google.org, 1st Cycle

Activity 2K: Linux Machine Communication

Full Name: Klajdi Cami

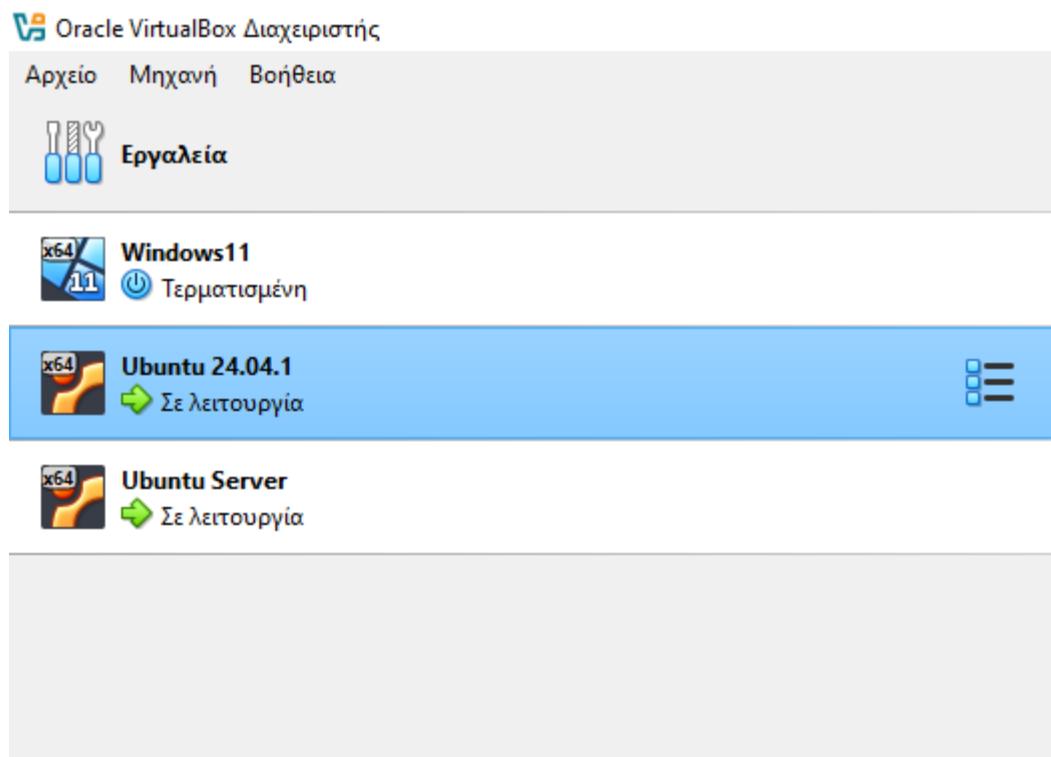
Part A: Machine Communication

Communication Settings

[1] In this part, you will properly configure the two machines to communicate with each other. Take a screenshot for every step.

Steps:

First, having installed the second Virtual Machine **Linux-Ubuntu** named **Ubuntu Server**:



Let's open a terminal on both machines, check, and install **net-tools** using the following command:

sudo apt install net-tools

```

Ubuntu 24.04.1 [εξ αρνητη] - Oracle VirtualBox
Aργος Μηχανή Πρόσωπο Επαγγελματικός Βοηθός
Feb 28 2017
klauss@klauss-VirtualBox:~$ ifconfig
Command 'ifconfig' not found, but can be installed with:
  sudo apt install net-tools
[sudo] password for klauss:
Reading package lists... done
Building dependency tree... done
Reading state information... done
The following packages were automatically installed and are no longer required:
  liblvm1f64 python3.netifaces
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  net-tools
0 upgraded, 1 newly installed, 0 to remove and 216 not upgraded.
Need to get 204 kB of archives.
After this operation, 811 kB of additional disk space will be used.
Get:1 http://gr.archive.ubuntu.com/ubuntu/noble/main amd64 net-tools amd64 2.10.0-1ubuntu4 [204 kB]
Fetched 204 kB in (250 kB/s)
Selecting previously unselected package net-tools.
(Reading database ... 149124 files and directories currently installed.)
Preparing to unpack .../net-tools_2.10.0-1ubuntu4_amd64.deb ...
Unpacking net-tools (2.10.0-1ubuntu4) ...
Setting up net-tools (2.10.0-1ubuntu4) ...
Processing triggers for man-db (2.12.0-4build2) ...
klauss@klauss-VirtualBox:~$ 

Ubuntu Server [εξ αρνητη] - Oracle VirtualBox
Aργος Μηχανή Πρόσωπο Επαγγελματικός Βοηθός
Feb 28 2017
klauss@klauss-VirtualBox:~$ ifconfig
Command 'ifconfig' not found, but can be installed with:
  sudo apt install net-tools
[sudo] password for klauss:
Reading package lists... done
Building dependency tree... done
Reading state information... done
The following NEW packages will be installed:
  net-tools
0 upgraded, 1 newly installed, 0 to remove and 216 not upgraded.
Need to get 204 kB of archives.
After this operation, 811 kB of additional disk space will be used.
Get:1 http://gr.archive.ubuntu.com/ubuntu/noble/main amd64 net-tools amd64 2.10.0-1ubuntu4 [204 kB]
Fetched 204 kB in (250 kB/s)
Selecting previously unselected package net-tools.
(Reading database ... 149057 files and directories currently installed.)
Preparing to unpack .../net-tools_2.10.0-1ubuntu4_amd64.deb ...
Unpacking net-tools (2.10.0-1ubuntu4) ...
Setting up net-tools (2.10.0-1ubuntu4) ...
Processing triggers for man-db (2.12.0-4build2) ...
klauss@klauss-VirtualBox:~$ 

```

We run the **ifconfig** command again and observe that both virtual machines have the same IP address, meaning they cannot communicate with each other.

```

Ubuntu 24.04.1 [εξ αρνητη] - Oracle VirtualBox
Aργος Μηχανή Πρόσωπο Επαγγελματικός Βοηθός
Feb 28 2019
klauss@klauss-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
  inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
    inet6 fe80::a00:27ff:fe4d:1363  prefixlen 64  scopidel 0x8<global>
      inet6 fe80::a00:27ff:fe4d:1363  prefixlen 64  scopidel 0x20<link>
      inet6 fd00::a00:27ff:fe4d:1363  prefixlen 64  scopidel 0x8<global>
      ether 00:00:27:4d:13:63  txqueuelen 1000  (Ethernet)
        RX packets 746  bytes 610958 (610.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 489  bytes 60591 (60.5 KB)
        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  scopidel 0x8<global>
      loop  txqueuelen 1000  (Local Loopback)
        RX packets 148  bytes 13767 (13.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 148  bytes 13767 (13.7 KB)
        TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
klauss@klauss-VirtualBox:~$ 

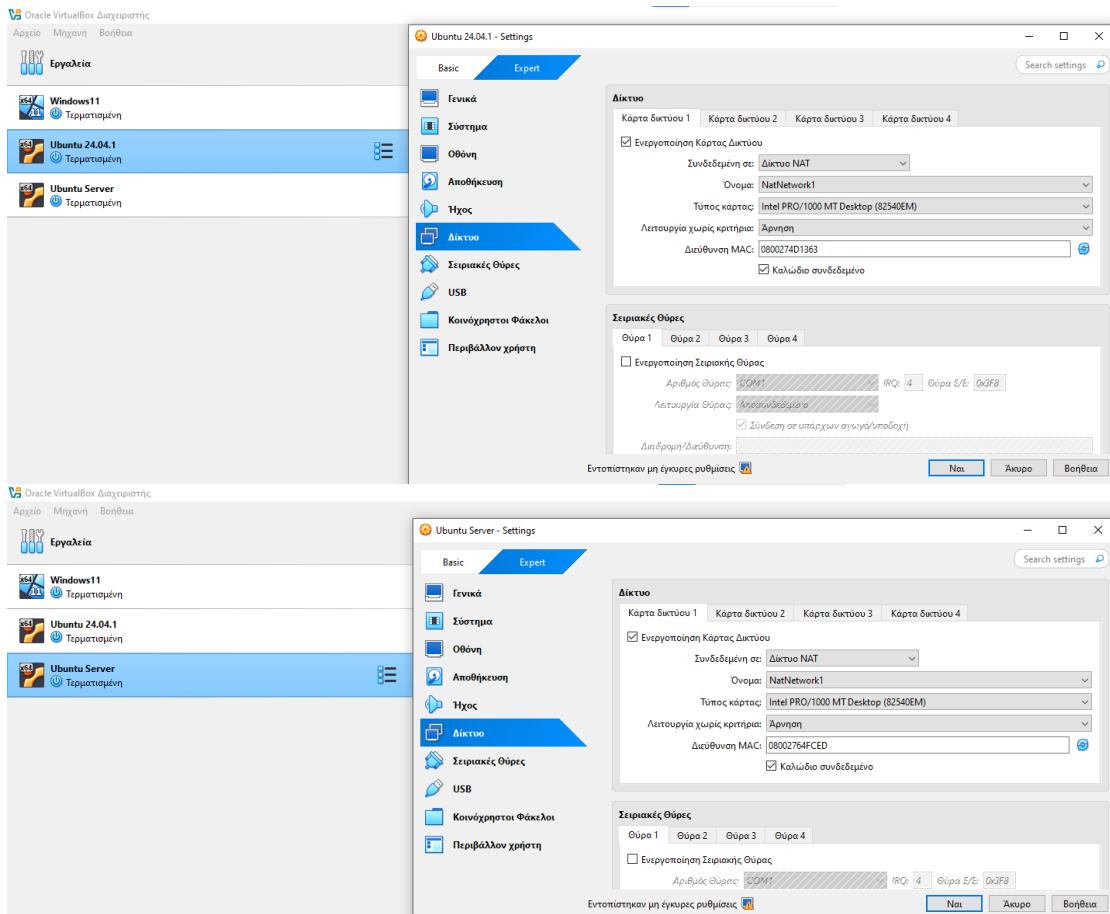
Ubuntu Server [εξ αρνητη] - Oracle VirtualBox
Aργος Μηχανή Πρόσωπο Επαγγελματικός Βοηθός
Feb 28 2019
klauss@klauss-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
  inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
    inet6 fd00::a00:27ff:fe4d:1363  prefixlen 64  scopidel 0x8<global>
      inet6 fe80::a00:27ff:fe4d:1363  prefixlen 64  scopidel 0x20<link>
      ether 00:00:27:4d:13:63  txqueuelen 1000  (Ethernet)
        RX packets 48827  bytes 61055241 (61.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10030  bytes 651383 (651.3 KB)
        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  scopidel 0x8<global>
            loop  txqueuelen 1000  (Local Loopback)
              RX packets 201  bytes 19892 (19.0 KB)
              RX errors 0  dropped 0  overruns 0  frame 0
              TX packets 201  bytes 19892 (19.0 KB)
              TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
klauss@klauss-VirtualBox:~$ 

```

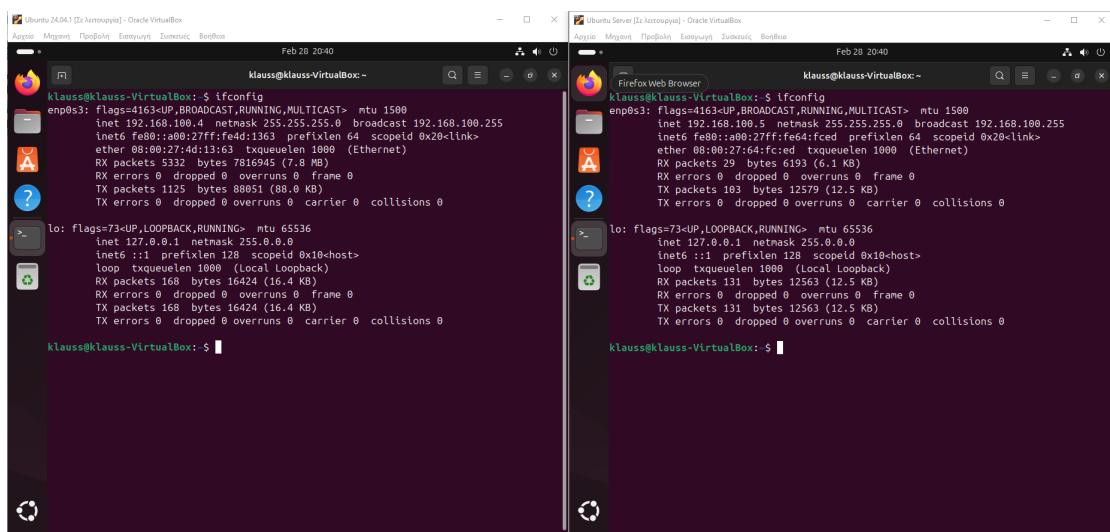
We shut down both virtual machines and navigate to **Tools -> Network -> Nat Networks**, where we create a new network.



Then, we add this network to both virtual machines.

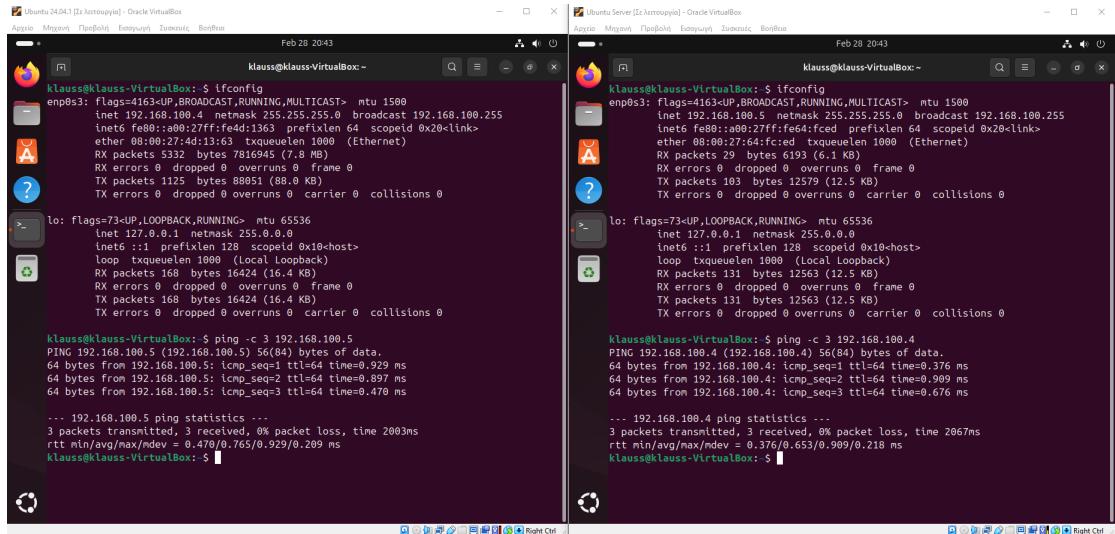


We restart the machines and run the **ifconfig** command again in both terminals. We observe that they have been assigned different IP addresses.



We confirm the connectivity between the two virtual machines by executing the following command:

ping -c 3 <IP_ADDRESS>:



The image shows two side-by-side terminal windows from the Ubuntu 24.04.1 LTS version running in Oracle VM VirtualBox. Both windows have a title bar 'Ubuntu 24.04.1 [2x Aeronaut] - Oracle VirtualBox' and a status bar at the bottom.

Left Terminal (Client):

```
klauss@klauss-VirtualBox: ~
```

```
klauuss@klauss-VirtualBox: $ ifconfig
```

```
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.100.5 brd 255.255.255.0 broadcast 192.168.100.255
        netmask 255.255.255.0
    inet6 fe80::a00:27ff:fe4d:1363 brd fe80::ff:fe4d:1363/64
        scopeid 0x20<link>
            ether 08:00:27:4d:13:63 txqueuelen 1000 (Ethernet)
            RX packets 5332 bytes 7816945 (7.8 MB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 1125 bytes 88051 (88.0 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 brd 255.0.0.0
        netmask 255.0.0.0
    inet6 ::1 brd ::1
        scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 168 bytes 16424 (16.4 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 168 bytes 16424 (16.4 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

klauss@klauss-VirtualBox: $ ping -c 3 192.168.100.5
PING 192.168.100.5 (192.168.100.5) 56(84) bytes of data.
64 bytes from 192.168.100.5: icmp_seq=1 ttl=64 time=0.929 ms
64 bytes from 192.168.100.5: icmp_seq=2 ttl=64 time=0.897 ms
64 bytes from 192.168.100.5: icmp_seq=3 ttl=64 time=0.478 ms

--- 192.168.100.5 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.476/0.765/0.929/0.209 ms
klauss@klauss-VirtualBox: $
```

Right Terminal (Server):

```
klauss@klauss-VirtualBox: ~
```

```
klauuss@klauss-VirtualBox: $ ifconfig
```

```
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.100.5 brd 255.255.255.0 broadcast 192.168.100.255
        netmask 255.255.255.0
    inet6 fe80::a00:27ff:fe4d:1363 brd fe80::ff:fe4d:1363/64
        scopeid 0x20<link>
            ether 08:00:27:4d:13:63 txqueuelen 1000 (Ethernet)
            RX packets 29 bytes 6193 (6.1 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 103 bytes 12579 (12.5 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 brd 255.0.0.0
        netmask 255.0.0.0
    inet6 ::1 brd ::1
        scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 131 bytes 12563 (12.5 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 131 bytes 12563 (12.5 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

klauss@klauss-VirtualBox: $ ping -c 3 192.168.100.4
PING 192.168.100.4 (192.168.100.4) 56(84) bytes of data.
64 bytes from 192.168.100.4: icmp_seq=1 ttl=64 time=0.376 ms
64 bytes from 192.168.100.4: icmp_seq=2 ttl=64 time=0.399 ms
64 bytes from 192.168.100.4: icmp_seq=3 ttl=64 time=0.676 ms

--- 192.168.100.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2067ms
rtt min/avg/max/mdev = 0.376/0.653/0.999/0.218 ms
klauss@klauss-VirtualBox: $
```

Part B: Setting Up FTP/SCP Servers

[1] In this part, you will configure the processes to enable machine-to-machine communication using **FTP** and **SCP**.

Steps:

First, we install and update the **vsftpd** application on the **Ubuntu Server** (the virtual machine that will function as the server) using the following command:

```
sudo apt-get update && sudo apt-get install vsftpd
```

A screenshot of a terminal window titled "Ubuntu Server [Σε λειτουργία] - Oracle VirtualBox". The terminal shows the command "sudo apt-get update && sudo apt-get install vsftpd" being run. The output lists numerous package downloads from the "ubuntu noble" and "security" repositories, including "InRelease" files and various "main", "restricted", and "universe" packages for amd64 architecture.

```
klauss@klauss-VirtualBox:~$ sudo apt-get update && sudo apt-get install vsftpd
[sudo] password for klauss:
Get:1 http://gr.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:3 http://gr.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://gr.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [641 kB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [122 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [8,972 B]
Get:8 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [667 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [131 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:11 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [815 kB]
Get:12 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [174 kB]
Get:13 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [51.9 kB]
Get:14 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [19.4 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [4,308 B]
Get:16 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:17 http://gr.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [891 kB]
Get:18 http://gr.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [201 kB]
Get:19 http://gr.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [151 kB]
```

We activate the service using the following command:

```
sudo systemctl enable vsftpd
```

A screenshot of a terminal window titled "Ubuntu Server [Σε λειτουργία] - Oracle VirtualBox". The terminal shows the command "sudo systemctl enable vsftpd" being run. The output indicates that the service is being synchronized with the SysV service script and then executed via the systemd-sysv-install command.

```
klauss@klauss-VirtualBox:~$ sudo systemctl enable vsftpd
Synchronizing state of vsftpd.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable vsftpd
klauss@klauss-VirtualBox:~$
```

We modify the **vsftpd** service settings by running the following command:

```
sudo pico /etc/vsftpd.conf
```

Then, in the editor, we remove the comment (#) from the following line:

```
write_enable=YES
```

```
GNU nano 7.2 /etc/vsftpd.conf
# capabilities.

#
#
# Run standalone?  vsftpd can run either from an inetd or as a standalone
# daemon started from an initscript.
listen=NO

#
# This directive enables listening on IPv6 sockets. By default, listening
# on the IPv6 "any" address (::) will accept connections from both IPv6
# and IPv4 clients. It is not necessary to listen on *both* IPv4 and IPv6
# sockets. If you want that (perhaps because you want to listen on specific
# addresses) then you must run two copies of vsftpd with two configuration
# files.
listen_ipv6=YES

#
# Allow anonymous FTP? (Disabled by default).
anonymous_enable=NO

#
# Uncomment this to allow local users to log in.
local_enable=YES

#
# Uncomment this to enable any form of FTP write command.
write_enable=YES

#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
#local_umask=022

#
# Uncomment this to allow the anonymous FTP user to upload files. This only
# [ Wrote 155 lines ]
```

We allow access to the **FTP Server** by executing the following command:

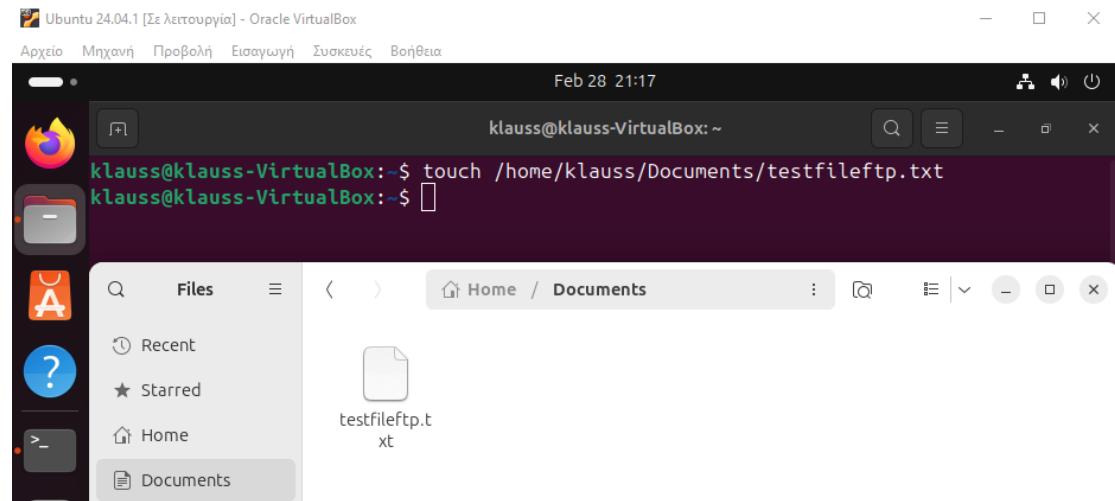
```
sudo ufw allow 21/tcp
```

Then, we observe that the access rules related to the server have been updated.

```
klauss@klauss-VirtualBox:~$ sudo ufw allow 21/tcp
Rules updated
Rules updated (v6)
klauss@klauss-VirtualBox:~$
```

We create a **.txt** file on the **client machine** and place it in the following directory:

home/<user_name directory>/Documents

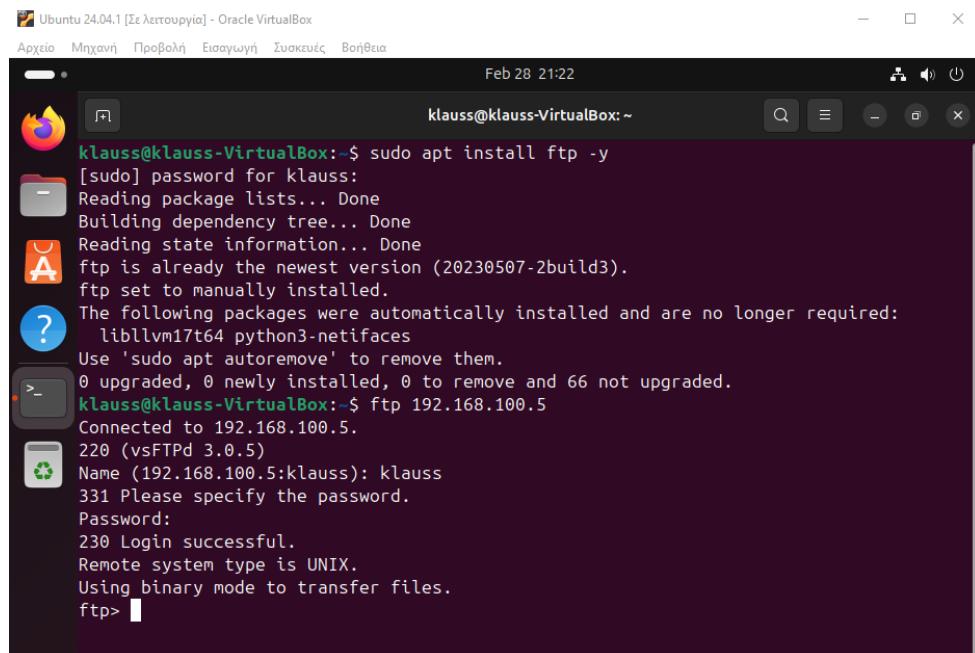


On the **Linux machine acting as the FTP client**, we install the FTP client by executing the following command:

```
sudo apt install ftp -y
```

To connect to the **FTP Server** that we set up earlier, we run the following command from the **second Linux machine (client)**:

```
ftp <SERVER'S IP>
```

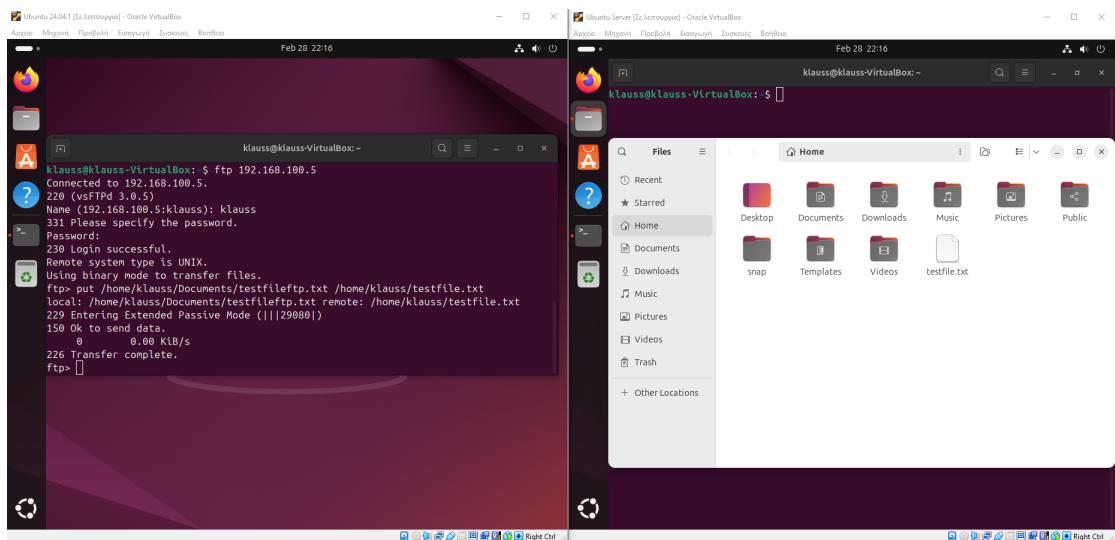


To **upload files** from the client to the server, follow these steps:

Run the following command:

```
put /home/<user_name directory>/Documents/<filename> /home/<user_name directory>/<filename>
```

Now, the file **testfileftp.txt** is successfully transferred to the machine acting as the **FTP Server**.



Activity 1:

The file created in the previous step, located in **/home/<user_name>**, should be downloaded by the **client** and placed in the directory **/home/<user_name>/Downloads**.

Answer:

To achieve this, we go to the **client** machine and, while connected to the **FTP server** (ftp 192.168.100.5), execute the following commands:

```
cd /home/klauss/
```

```
get testfile.txt /home/klauss/Downloads/testfile.txt
```

The screenshot shows a desktop environment with a terminal window and a file manager. The terminal window (top) displays an FTP session where a file named 'testfile.txt' is transferred from a local directory to a remote server at 192.168.100.5. The file manager (bottom) shows the transferred file in the 'Downloads' folder.

```
klauss@klauss-VirtualBox:~$ ftp 192.168.100.5
Connected to 192.168.100.5.
220 (vsFTPd 3.0.5)
Name (192.168.100.5:klauss): klauss
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd /home/klauss/
250 Directory successfully changed.
ftp> get testfile.txt /home/klauss/Downloads/testfile.txt
local: /home/klauss/Downloads/testfile.txt remote: testfile.txt
229 Entering Extended Passive Mode (|||31577|)
150 Opening BINARY mode data connection for testfile.txt (0 bytes).
          0      0.00 KiB/s
226 Transfer complete.
ftp>
```

Files

- Recent
- Starred
- Home
- Documents
- Downloads
- Music
- Pictures

testfile.txt

File Transfer with SCP

First, we install and update the **OpenSSH server** on the **Ubuntu Server** by executing the following command:

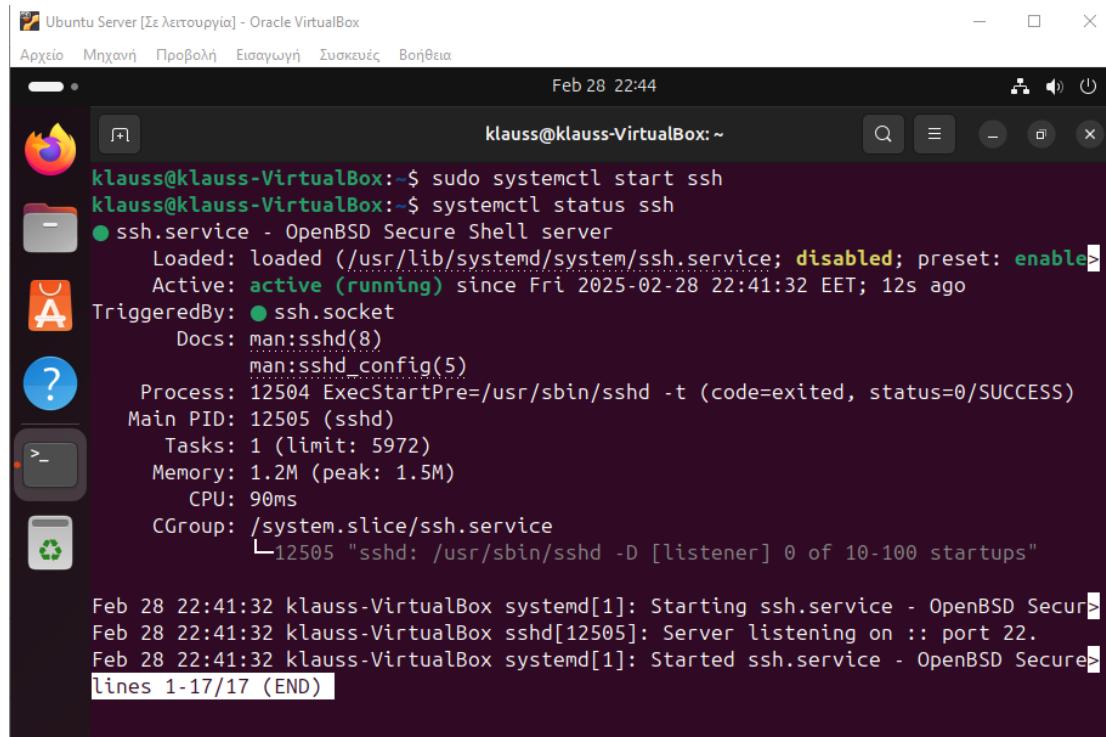
```
sudo apt-get install openssh-server
```

The screenshot shows a terminal window on an Ubuntu Server where the 'openssh-server' package is being installed via 'apt-get'. The terminal output shows the process of reading package lists, building dependency trees, and installing additional packages like 'ncurses-term', 'openssh-sftp-server', and 'ssh-import-id'. It also lists suggested packages like 'molly-guard' and 'monkeysphere'. The user is prompted to confirm the installation.

```
klauss@klauss-VirtualBox:~$ sudo apt-get install openssh-server
[sudo] password for klauss:
Reading package lists...
Building dependency tree...
Reading state information...
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere ssh-askpass
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 upgraded, 4 newly installed, 0 to remove and 219 not upgraded.
Need to get 832 kB of archives.
After this operation, 6,751 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

We activate and check if the **OpenSSH service** is running by executing the following commands:

```
sudo systemctl start ssh  
systemctl status ssh
```



The screenshot shows a terminal window titled "Ubuntu Server [Σε λειτουργία] - Oracle VirtualBox". The window contains the following text:

```
klauss@klauss-VirtualBox:~$ sudo systemctl start ssh  
klauss@klauss-VirtualBox:~$ systemctl status ssh  
● ssh.service - OpenBSD Secure Shell server  
  Loaded: loaded (/usr/lib/systemd/system/ssh.service; disabled; preset: enabled)  
  Active: active (running) since Fri 2025-02-28 22:41:32 EET; 12s ago  
    TriggeredBy: ● ssh.socket  
      Docs: man:sshd(8)  
            man:sshd_config(5)  
    Process: 12504 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)  
   Main PID: 12505 (sshd)  
     Tasks: 1 (limit: 5972)  
    Memory: 1.2M (peak: 1.5M)  
       CPU: 90ms  
      CGroup: /system.slice/ssh.service  
             └─12505 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"  
  
Feb 28 22:41:32 klauss-VirtualBox systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...  
Feb 28 22:41:32 klauss-VirtualBox sshd[12505]: Server listening on :: port 22.  
Feb 28 22:41:32 klauss-VirtualBox systemd[1]: Started ssh.service - OpenBSD Secure Shell server.  
[lines 1-17/17 (END)]
```

We connect to the **server via SSH** from the second **Linux machine (client)** by executing the following command:

```
ssh <SERVER'S IP>
```

The screenshot shows a terminal window titled "Ubuntu 24.04.1 [Σε λειτουργία] - Oracle VirtualBox". The window title bar includes the Greek text "Αρχείο Μηχανή Προβολή Εισαγωγή Συσκευές Βοήθεια". The terminal window has a dark background and displays the following text:

```
klauss@klauss-VirtualBox:~$ ssh 192.168.100.5
The authenticity of host '192.168.100.5 (192.168.100.5)' can't be established.
ED25519 key fingerprint is SHA256:zNlUjV23SDmUGweo8ICSYySlymIgT7j0MLegVbnhNc.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? Y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.100.5' (ED25519) to the list of known hosts.
klauss@192.168.100.5's password:
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.11.0-17-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

● Expanded Security Maintenance for Applications is not enabled.

 212 updates can be applied immediately.
 To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

On the **client side**, we observe that we are now inside the **server's directory**. The file **testfile.txt** is **not present** on the **client machine**. We can disconnect from the **remote server** by executing the following command: exit

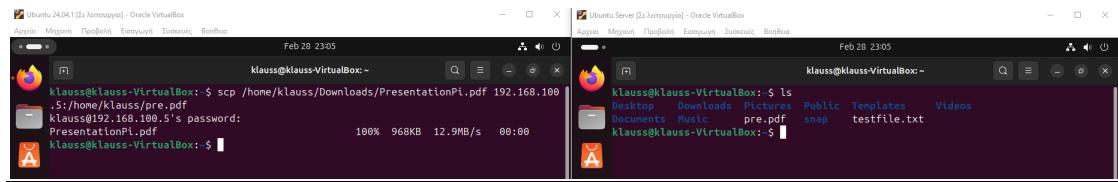
The screenshot shows a terminal window with a dark background. It displays the following text:

```
klauss@klauss-VirtualBox:~$ ls
Desktop Downloads Pictures snap      testfile.txt
Documents Music   Public   Templates Videos
klauss@klauss-VirtualBox:~$ exit
logout
Connection to 192.168.100.5 closed.
klauss@klauss-VirtualBox:~$
```

To send the file "**PresentationPi.pdf**" to the **server**, we execute the following command on the **client machine**:

```
scp <source_path> <Server's IP>:<destination_path>
```

To confirm that the file has successfully arrived on the **server**, we switch to the **server's terminal** and run: **ls**



Activity 2:

Download the file “**CyberSecurity Economics for Emerging Markets**” from eLearning to the **server**. Then, from the **client side**, download the file using **SCP**.

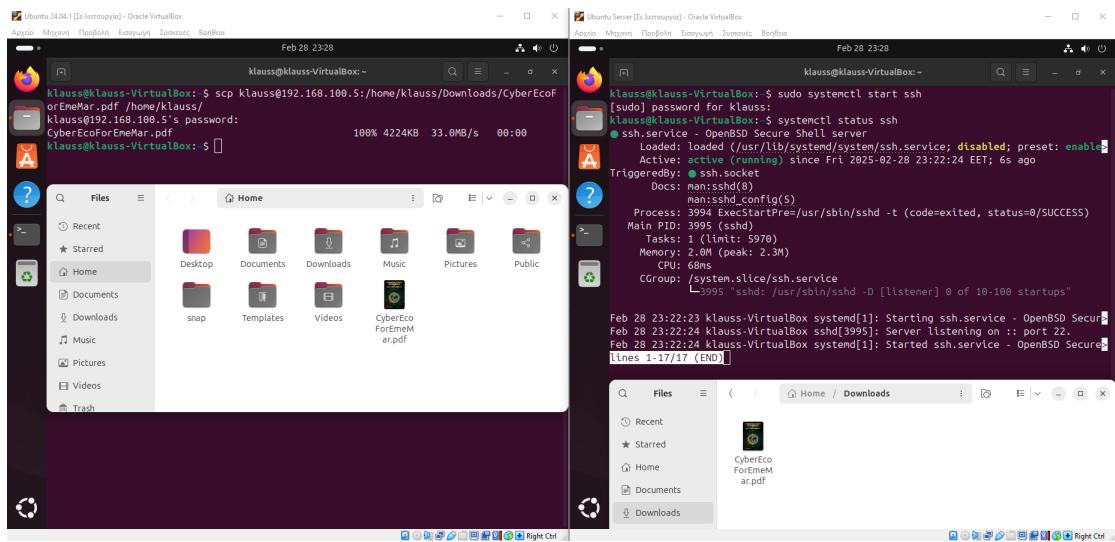
Answer:

After downloading the file to the **server**, on the **client side**, we execute the following command:

```
scp <username>@<Server's IP>:/home/<user_name>  
directory>/Downloads/CyberEcoForEmeMar.pdf /home/<local_user>/
```

Where:

- **<username>** → The username on the **server** (e.g., user).
- **<Server's IP>** → The **IP address** of the **server**.
- **/home/<user_name>/Downloads/CyberEcoForEmeMar.pdf** → The **file path** on the **server**.
- **/home/<local_user>/** → The **local folder** where the file will be saved on the **client**.



[2] Encrypt a Directory Using PGP

Step 1: Create a Directory

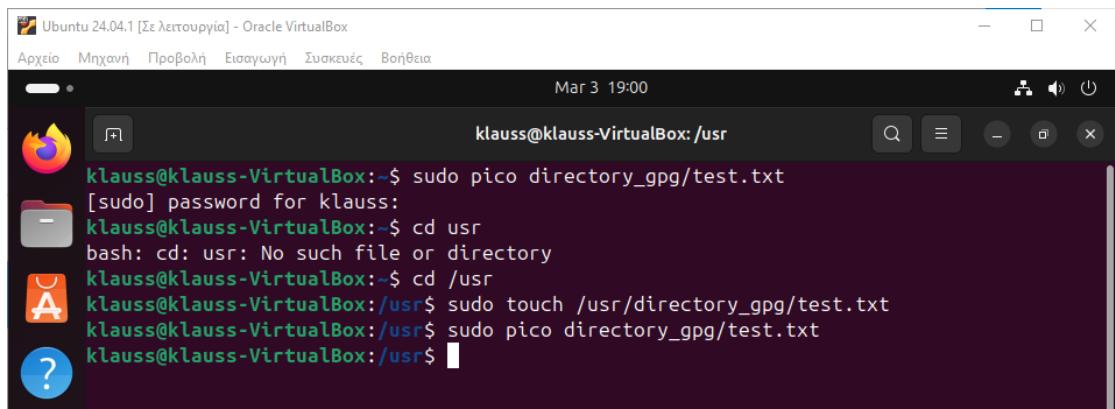
First, we create a directory named **directory_gpg** as the **root user** in the **/usr/** path using the following command:

```
sudo mkdir /usr/directory_gpg
```

Next, we create a **test file** inside **directory_gpg** by executing:

```
sudo touch /usr/directory_gpg/test.txt
```

We open the file **test.txt** and write the message "**This is a file to be encrypted!!**".



To convert the **directory** into a **file**, we use the following command:

```
sudo tar czf my_directory_gpg.tar.gz directory_gpg/
```



```
klauss@klauss-VirtualBox:/usr$ sudo tar czf my_directory_gpg.tar.gz directory_gpg/
klauss@klauss-VirtualBox:/usr$ ls
bin    dept2      games   lib    libexec  my_directory_gpg.tar.gz  share
dept1  directory_gpg  include  lib64  local    sbin                src
klauss@klauss-VirtualBox:/usr$
```

To encrypt the file **my_directory_gpg.tar.gz**, we execute the following command and choose a password (we must remember it because it is the key required for decryption).

```
sudo gpg - -symmetric my_directory_gpg.tar.gz
```



```
klauss@klauss-VirtualBox:/usr$ sudo gpg --symmetric my_directory_gpg.tar.gz
gpg: directory '/root/.gnupg' created
gpg: keybox '/root/.gnupg/pubring.kbx' created
klauss@klauss-VirtualBox:/usr$ ls
bin    dept2      games   lib    libexec  my_directory_gpg.tar.gz  sbin   src
dept1  directory_gpg  include  lib64  local    my_directory_gpg.tar.gz.gpg  share
klauss@klauss-VirtualBox:/usr$
```

The encrypted file looks something like this...



```
klauss@klauss-VirtualBox:/usr$ sudo cat my_directory_gpg.tar.gz.gpg
-----BEGIN PGP MESSAGE-----
Version: GnuPG v2.2.14 (Ubuntu)
Comment: https://www.gnupg.org

k=-----END PGP MESSAGE-----
klauss@klauss-VirtualBox:/usr$
```

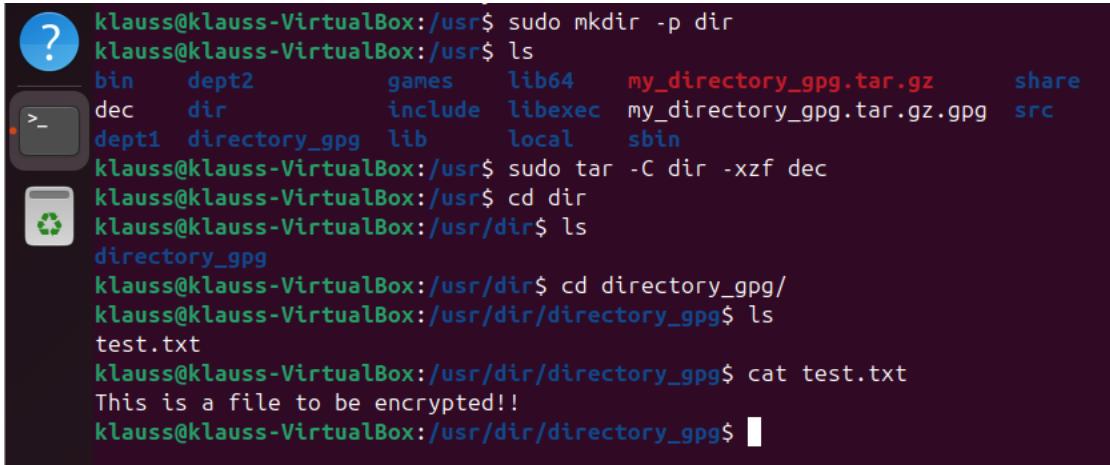
To decrypt, we execute the following command and enter the key we set in the previous step.

```
sudo gpg - -output <desired_name_of_decrypted_file> -d
my_directory_gpg.tar.gz.gpg
```



```
Ce~klauss@klauss-VirtualBox:/usr$ sudo gpg --output dec --decrypt my_directory_gpg.tar.
gpg
gpg: AES256.CFB encrypted data
gpg: encrypted with 1 passphrase
```

In our case, no password was requested because it was stored on the machine. Additionally, the file is an **archive file**, which means that to view and verify the correct functionality of the **encryption/decryption**, we must first extract it. This is done using the following command:



```
klauss@klauss-VirtualBox:/usr$ sudo mkdir -p dir
klauss@klauss-VirtualBox:/usr$ ls
bin    dept2      games   lib64   my_directory_gpg.tar.gz      share
dec    dir        include libexec my_directory_gpg.tar.gz.gpg  src
dept1 directory_gpg lib    local   sbin
klauss@klauss-VirtualBox:/usr$ sudo tar -C dir -xzf dec
klauss@klauss-VirtualBox:/usr$ cd dir
klauss@klauss-VirtualBox:/usr/dir$ ls
directory_gpg
klauss@klauss-VirtualBox:/usr/dir$ cd directory_gpg/
klauss@klauss-VirtualBox:/usr/dir/directory_gpg$ ls
test.txt
klauss@klauss-VirtualBox:/usr/dir/directory_gpg$ cat test.txt
This is a file to be encrypted!!
klauss@klauss-VirtualBox:/usr/dir/directory_gpg$
```

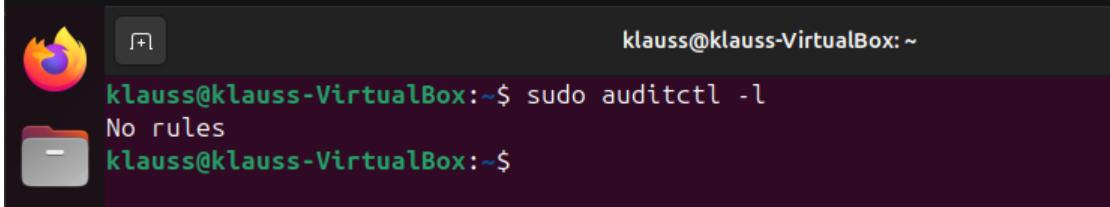
[2] Configure auditd for Event Logging

First, we install the **Auditd** application using the following command:

```
sudo apt-get install auditd
```

Then, we check if it was installed correctly and verify that no event logging rules have been set by executing:

```
sudo auditctl -l
```



```
klauss@klauss-VirtualBox:~$ sudo auditctl -l
No rules
klauss@klauss-VirtualBox:~$
```

We create a file using the following command:

```
touch /tmp/test_auditd.txt
```

Next, we grant full access permissions to the file, making it freely accessible to all users, by executing:

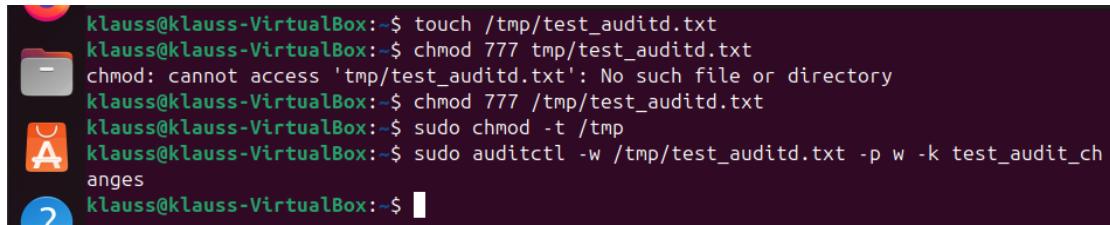
```
chmod 777 /tmp/test_auditd.txt
```

We then allow modifications to the **/tmp** directory using the following command:

```
sudo chmod +t /tmp
```

Finally, we set up a **monitoring rule** for the file using:

```
sudo auditctl -w /tmp/test_auditd.txt -p w -k test_audit_changes_changes
```



```
klauss@klauss-VirtualBox:~$ touch /tmp/test_auditd.txt
klauss@klauss-VirtualBox:~$ chmod 777 /tmp/test_auditd.txt
chmod: cannot access '/tmp/test_auditd.txt': No such file or directory
klauss@klauss-VirtualBox:~$ chmod 777 /tmp/test_auditd.txt
klauss@klauss-VirtualBox:~$ sudo chmod -t /tmp
klauss@klauss-VirtualBox:~$ sudo auditctl -w /tmp/test_auditd.txt -p w -k test_audit_ch
anges
klauss@klauss-VirtualBox:~$
```

Using the **aureport** command, we get an overview of the logs recorded by the **audit** tool on our system.

By executing:

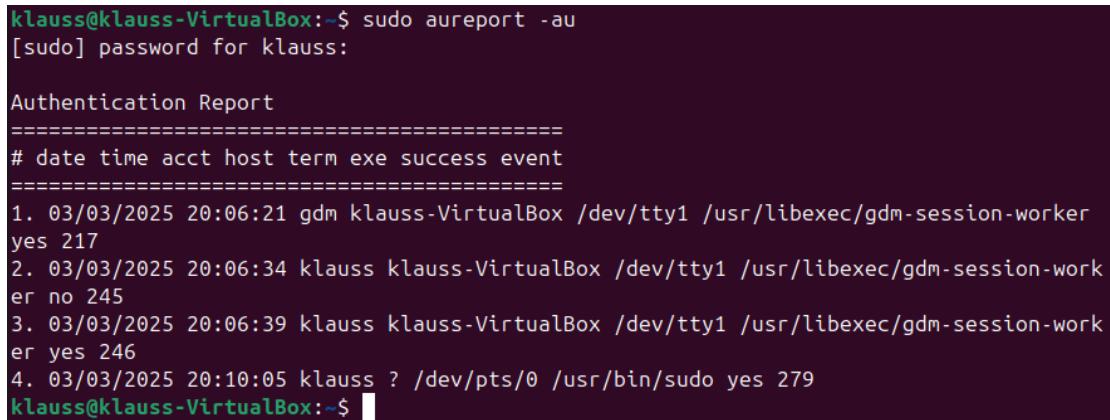
```
sudo aureport -au
```

we obtain a report of **all authentication-related events** on our system.

For example, we log out of our account and attempt to log back in, making **one incorrect password attempt** before entering the **correct password**.

Then, we run:

```
sudo aureport -au and observe that both login attempts have been recorded.
```



```
klauss@klauss-VirtualBox:~$ sudo aureport -au
[sudo] password for klauss:

Authentication Report
=====
# date time acct host term exe success event
=====
1. 03/03/2025 20:06:21 gdm klauss-VirtualBox /dev/tty1 /usr/libexec/gdm-session-worker yes 217
2. 03/03/2025 20:06:34 klauss klauss-VirtualBox /dev/tty1 /usr/libexec/gdm-session-worker no 245
3. 03/03/2025 20:06:39 klauss klauss-VirtualBox /dev/tty1 /usr/libexec/gdm-session-worker yes 246
4. 03/03/2025 20:10:05 klauss ? /dev/pts/0 /usr/bin/sudo yes 279
klauss@klauss-VirtualBox:~$
```

Generally, the **rules set in auditd** are deleted after the system shuts down. To make these **rules permanent**, we need to **modify** the following configuration file:

```
/etc/audit/rules.d/audit.rules
```

We execute the command:

```
sudo pico /etc/audit/rules.d/audit.rules
```

and add the rule we want to **persist** at the end of the file:

```
-w /tmp/test_audited.txt -p w -k test_audit_changes
```

```
klauss@klauss-VirtualBox:~$ sudo pico /etc/audit/rules.d/audit.rules
klauss@klauss-VirtualBox:~$ 

GNU nano 7.2                               /etc/audit/rules.d/audit.rules
## First rule - delete all
-D

## Increase the buffers to survive stress events.
## Make this bigger for busy systems
-b 8192

## This determine how long to wait in burst of events
--backlog_wait_time 60000

## Set failure mode to syslog
-f 1

-w /tmp/test_audited.txt -p w -k test_audit_changes
```

Part C: Creating a Backup

[1] Backup for Immediate Recovery

You will define a **directory** and use **Rsync** to back it up between the two machines for immediate recovery in case of an attack or data loss.

Answer:

First, we install **Rsync** (a fast and flexible command-line tool for synchronizing files and directories between two locations. It ensures fast file transfer by only copying the differences between the source and destination) using the following command:

```
sudo apt install rsync
```

```
klauss@klauss-VirtualBox:~$ sudo apt install rsync
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
rsync is already the newest version (3.2.7-1ubuntu1.2).
rsync set to manually installed.
The following packages were automatically installed and are no longer required:
  liblvm17t64 python3-netifaces
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 40 not upgraded.
klauss@klauss-VirtualBox:~$
```

Next, we create the directory that we want to **back up** on the **Desktop** and name it “**Vasi**” by executing the following command:

```
klauss@klauss-VirtualBox:~$ cd Desktop
klauss@klauss-VirtualBox:~/Desktop$ mkdir Vasi
klauss@klauss-VirtualBox:~/Desktop$ ls
Vasi
klauss@klauss-VirtualBox:~/Desktop$
```

Next, we create **10 text files** inside the “**Vasi**” directory by executing the following command:

```
touch /Vasi/file{1..10}.txt
```

To **confirm** that the files have been created, we list the contents of the directory:

```
ls
```

```
klauss@klauss-VirtualBox:~/Desktop$ touch Vasi/file{1..10}.txt
klauss@klauss-VirtualBox:~/Desktop$ cd Vasi/
klauss@klauss-VirtualBox:~/Desktop/Vasi$ ls
file10.txt  file2.txt  file4.txt  file6.txt  file8.txt
file1.txt   file3.txt  file5.txt  file7.txt  file9.txt
klauss@klauss-VirtualBox:~/Desktop/Vasi$
```

We create a second directory “**Vasi_cp**” and synchronize/copy the “**Vasi**” directory to “**Vasi_cp**” by executing the following command. Then, we confirm the result.

```
klauss@klauss-VirtualBox:~/Desktop$ mkdir Vasi_cp
klauss@klauss-VirtualBox:~/Desktop$ rsync -a Vasi Vasi_cp
klauss@klauss-VirtualBox:~/Desktop$ ls Vasi_cp/
Vasi
klauss@klauss-VirtualBox:~/Desktop$ ls Vasi_cp/Vasi/
file10.txt  file2.txt  file4.txt  file6.txt  file8.txt
file1.txt   file3.txt  file5.txt  file7.txt  file9.txt
klauss@klauss-VirtualBox:~/Desktop$
```

The backup can be set to run automatically at the frequency we desire. We add **5 .pdf files** to the **Vasi** directory by executing the following command:

```
touch Vasi/file{1..5}.pdf
```

```

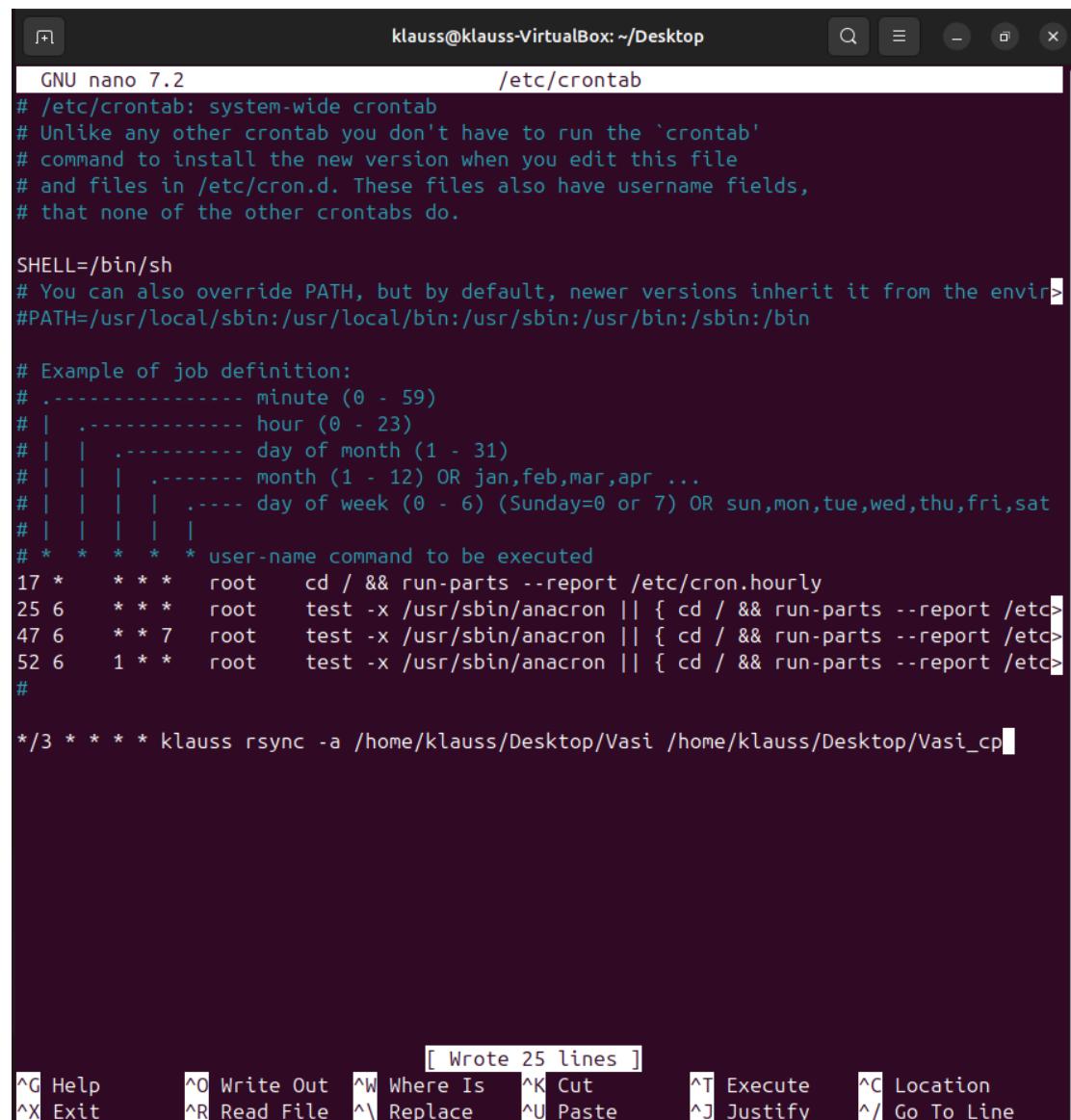
klauss@klauss-VirtualBox:~/Desktop$ touch Vasi/file{1..5}.pdf
klauss@klauss-VirtualBox:~/Desktop$ cd Vasi/
klauss@klauss-VirtualBox:~/Desktop/Vasi$ ls
file10.txt  file2.pdf  file3.txt  file5.pdf  file7.txt
file1.pdf   file2.txt  file4.pdf  file5.txt  file8.txt
file1.txt   file3.pdf  file4.txt  file6.txt  file9.txt
klauss@klauss-VirtualBox:~/Desktop/Vasi$ █

```

To set the backup frequency, we need to modify the configuration file by executing the following command:

sudo pico /etc/crontab

With the rule shown at the end of the file, the backup is performed every **3 minutes**.



```

GNU nano 7.2          /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the 'crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
# You can also override PATH, but by default, newer versions inherit it from the envir>
#PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin

# Example of job definition:
# ----- minute (0 - 59)
# | ----- hour (0 - 23)
# | | ----- day of month (1 - 31)
# | | | ----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | ----- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | |
# * * * * user-name command to be executed
17 *    * * *    root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *    root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc>
47 6    * * 7    root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc>
52 6    1 * *    root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc>
#
*/3 * * * * klauss rsync -a /home/klauss/Desktop/Vasi /home/klauss/Desktop/Vasi_cp█

[ Wrote 25 lines ]
^O Help      ^O Write Out  ^W Where Is  ^K Cut      ^T Execute  ^C Location
^X Exit      ^R Read File  ^\ Replace   ^U Paste    ^J Justify  ^/ Go To Line

```

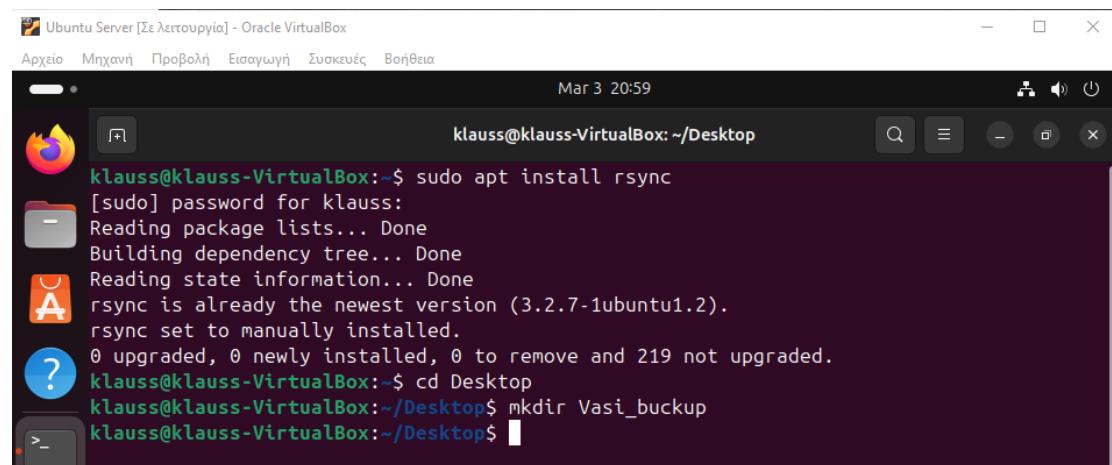
The backup can be set to run automatically at the frequency we desire for specific file types. In this scenario, we want to back up only the **.pdf** files located in the **Vasi** directory.

```
klauss@klauss-VirtualBox:~$ rsync -av --exclude '*.txt' /home/klauss/Desktop/Vasi/ /home/klauss/Desktop/Vasi_pdf
sending incremental file list
created directory /home/klauss/Desktop/Vasi_pdf
./
file1.pdf
file2.pdf
file3.pdf
file4.pdf
file5.pdf

sent 351 bytes received 166 bytes 1,034.00 bytes/sec
total size is 0 speedup is 0.00
klauss@klauss-VirtualBox:~$
```

Scenario: We want to keep a backup of the “**Vasi**” directory on the second machine we have set up, which functions as a server.

Step 1: On the machine functioning as a server, we create a directory “**Vasi_backup**” on the desktop, where the backup will be stored.



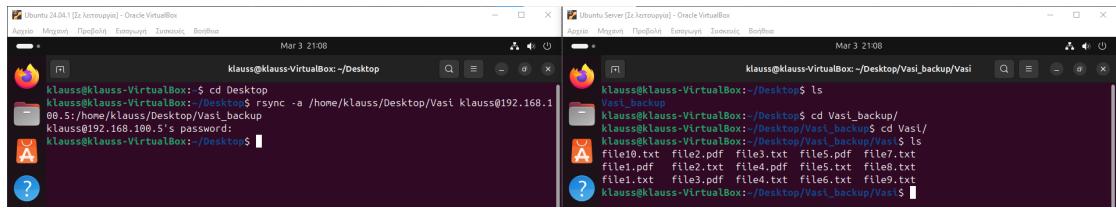
A screenshot of a terminal window titled "Ubuntu Server [Σε λειτουργία] - Oracle VirtualBox". The window shows a command-line interface with the following text:

```
klauss@klauss-VirtualBox:~$ sudo apt install rsync
[sudo] password for klauss:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
rsync is already the newest version (3.2.7-1ubuntu1.2).
rsync set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 219 not upgraded.
klauss@klauss-VirtualBox:~$ cd Desktop
klauss@klauss-VirtualBox:~/Desktop$ mkdir Vasi_backup
klauss@klauss-VirtualBox:~/Desktop$
```

Step 2: From the first machine, which contains the “**Vasi**” directory that we want to back up, we execute the following command:

```
Rsync -a /home/klauss/Desktop/Vasi
klauss@192.168.100.5:/home/klauss/Desktop/Vasi_backup
```

And we confirm that the backup was successful by typing the following in the **server's terminal**:



The image shows two side-by-side terminal windows from Oracle VirtualBox. Both windows have a title bar with the text "Ubuntu 24.04.1 [2x Xfce] - Oracle VirtualBox". The left window's title bar also includes "Mar 3 21:08". The right window's title bar also includes "Mar 3 21:08".

The left terminal window contains the following text:

```
klauss@klauss-VirtualBox:~$ cd Desktop
klauss@klauss-VirtualBox:~/Desktop$ rsync -a /home/klauss/Desktop/Vasi klauss@192.168.1.00.5:/home/klauss/Desktop/Vasi_backup
klauss@192.168.100.5's password:
klauss@klauss-VirtualBox:~/Desktop$
```

The right terminal window contains the following text:

```
klauss@klauss-VirtualBox:~$ ls
Vasi_backup
klauss@klauss-VirtualBox:~/Desktop$ cd Vasi_backup/
klauss@klauss-VirtualBox:~/Desktop/Vasi_backup$ cd Vasi/
klauss@klauss-VirtualBox:~/Desktop/Vasi_backup/Vasi$ ls
file1.pdf  file1.txt  file2.pdf  file2.txt  file3.pdf  file3.txt  file4.pdf  file4.txt  file5.pdf  file5.txt  file6.pdf  file6.txt  file7.pdf  file7.txt  file8.pdf  file8.txt  file9.pdf  file9.txt
klauss@klauss-VirtualBox:~/Desktop/Vasi_backup/Vasi$
```