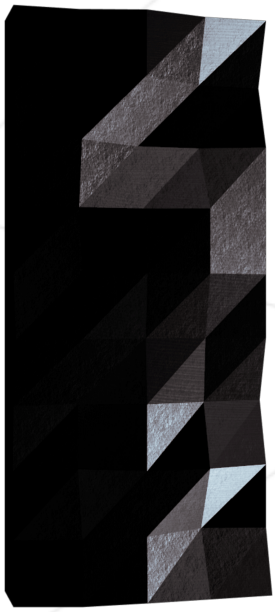


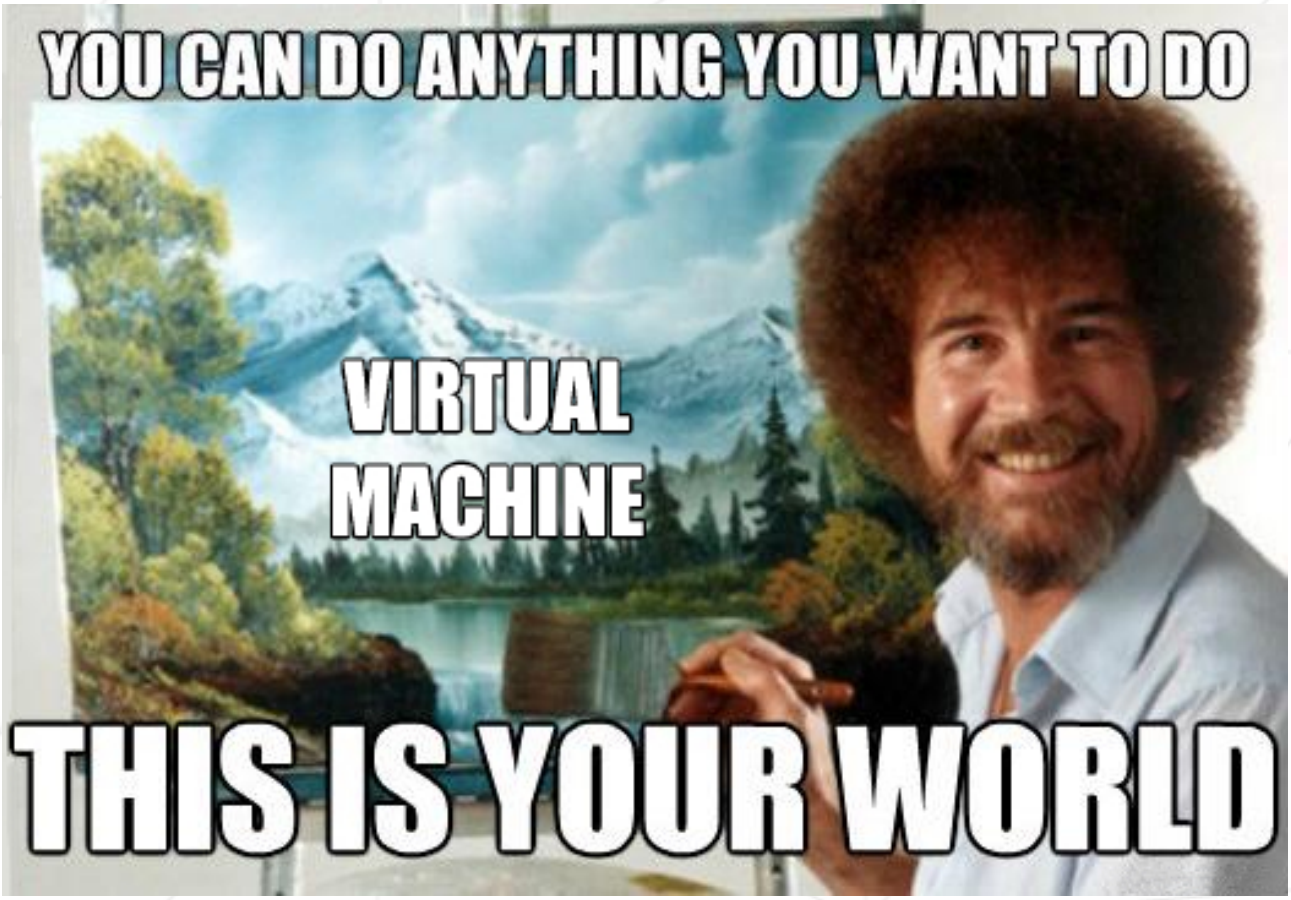
Born2beRoot

Summary: This document is a System Administration related exercise.

Contents

Chapter I

Preamble



Chapter II

Introduction

This project aims to introduce you to the wonderful world of virtualization.

You will create your first machine in `VirtualBox` (or `UTM` if you can't use `VirtualBox`) under specific instructions. Then, at the end of this project, you will be able to set up your own operating system while implementing strict rules.

Chapter III

General guidelines

- The use of `VirtualBox` (or `UTM` if you can't use `VirtualBox`) is mandatory.
- You only have to turn in a `signature.txt` file at the root of your repository. You must paste in it the signature of your machine's virtual disk. Go to Submission and peer-evaluation for more information.

Chapter IV

Mandatory part

This project consists of having you set up your first server by following specific rules.

Since it is a matter of setting up a server, you will install the minimum of services. For this reason, a graphical interface is of no use here. It is therefore forbidden to install X.org or any other equivalent graphics server. Otherwise, your grade will be 0.

You must choose as an operating system either the latest stable version of `Debian` (no testing/unstable), or the latest stable version of `CentOS`. `Debian` is highly recommended if you are new to system administration.

Setting up CentOS is quite complex. Therefore, you don't have to set up KDump. However, SELinux must be running at startup and its configuration has to be adapted for the project's needs. AppArmor for Debian must be running at startup too.

You must create at least 2 encrypted partitions using `LVM`. Below is an example of the expected partitioning:

```
wil@wil:~$ lsblk
NAME                  MAJ:MIN RM  SIZE RO TYPE  MOUNTPOINT
sda                     8:0    0    8G  0 disk
├─sda1                  8:1    0  487M  0 part  /boot
├─sda2                  8:2    0    1K  0 part
└─sda5                  8:5    0  7.5G  0 part
  └─sda5_crypt        254:0    0  7.5G  0 crypt
    ├─wil--vg-root    254:1    0  2.8G  0 lvm   /
    ├─wil--vg-swap_1  254:2    0  976M  0 lvm   [SWAP]
    └─wil--vg-home    254:3    0  3.8G  0 lvm   /home
sr0                    11:0    1 1024M  0 rom
wil@wil:~$ _
```

During the defense, you will be asked a few questions about the operating system you chose. For instance, you should know the differences between aptitude and apt, or what SELinux or AppArmor is. In short, understand what you use!

A SSH service will be running on port 4242 only. For security reasons, it must not be possible to connect using SSH as root.

The use of SSH will be tested during the defense by setting up a new account. You must therefore understand how it works.

You have to configure your operating system with the UFW firewall and thus leave only port 4242 open.

Your firewall must be active when you launch your virtual machine. For CentOS, you have to use UFW instead of the default firewall. To install it, you will probably need DNF.

- The hostname of your virtual machine must be your login ending with 42 (e.g., wil42). You will have to modify this hostname during your evaluation.
- You have to implement a strong password policy.
- You have to install and configure sudo following strict rules.
- In addition to the root user, a user with your login as username has to be present.
- This user has to belong to the user42 and sudo groups.

During the defense, you will have to create a new user and assign it to a group.

To set up a strong password policy, you have to comply with the following requirements:

- Your password has to expire every 30 days.
- The minimum number of days allowed before the modification of a password will be set to 2.
- The user has to receive a warning message 7 days before their password expires.
- Your password must be at least 10 characters long. It must contain an uppercase letter and a number. Also, it must not contain more than 3 consecutive identical characters.

- The password must not include the name of the user.
- The password must have at least 7 characters that are not part of the former password.
- Of course, your root password has to comply with this policy.

`After setting up your configuration files, you will have to change all the passwords of the accounts present on the virtual machine, including the root account.`

To set up a strong configuration for your `sudo` group, you have to comply with the following requirements:

- Authentication using `sudo` has to be limited to 3 attempts in the event of an incorrect password.
- A custom message of your choice has to be displayed if an error due to a wrong password occurs when using `sudo`.
- Each action using `sudo` has to be archived, both inputs and outputs. The log file has to be saved in the `/var/log/sudo/` folder.
- The `TTY` mode has to be enabled for security reasons.
- For security reasons too, the paths that can be used by `sudo` must be restricted. Example:
  `/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin`

Finally, you have to create a simple script called `monitoring.sh`. It must be developed in `bash`.

At server startup, the script will display some information (listed below) on all terminals every 10 minutes (take a look at `wall`). The banner is optional. No error must be visible.

Your script must always be able to display the following information:

- The architecture of your operating system and its kernel version.
- The number of physical processors.
- The number of virtual processors.
- The current available RAM on your server and its utilization rate as a percentage.
- The current available memory on your server and its utilization rate as a percentage.
- The current utilization rate of your processors as a percentage.
- The date and time of the last reboot.
- Whether LVM is active or not.
- The number of active connections.
- The number of users using the server.
- The IPv4 address of your server and its MAC (Media Access Control) address.
- The number of commands executed with the `sudo` program.

During the defense, you will be asked to explain how this script works. You will also have to interrupt it without modifying it. Take a look at cron.

This is an example of how the script is expected to work:

```
Broadcast message from root@wil (tty1) (Sun Apr 25 15:45:00 2021):

    #Architecture: Linux wil 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64 GNU/Linux
    #CPU physical : 1
    #vCPU : 1
    #Memory Usage: 74/987MB (7.50%)
    #Disk Usage: 1009/2Gb (39%)
    #CPU load: 6.7%
    #Last boot: 2021-04-25 14:45
    #LVM use: yes
    #Connexions TCP : 1 ESTABLISHED
    #User log: 1
    #Network: IP 10.0.2.15 (08:00:27:51:9b:a5)
    #Sudo : 42 cmd
```

Below are two commands you can use to check some of the subject's requirements:

For `CentOS`:

```
[root@wil ~]# head -n 2  /etc/os-release
NAME="CentOS Linux"
VERSION="8"
[root@wil ~]# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      32
[root@wil ~]# ss -tunlp
Netid State  Recv-Q  Send-Q   Local Address:Port   Peer Address:Port
tcp   LISTEN 0       128           0.0.0.0:4242        0.0.0.0:*     users:(("sshd",pid=822,fd=5))
tcp   LISTEN 0       128              [::]:4242           [::]:*     users:(("sshd",pid=822,fd=7))
[root@wil ~]# ufw status
Status: active

To                         Action      From
--                         ------      ----
4242                       ALLOW       Anywhere
4242 (v6)                  ALLOW       Anywhere (v6)

[root@wil ~]# _
```

For `Debian`:

```
root@wil:~# head -n 2 /etc/os-release
PRETTY_NAME="Debian GNU/Linux 10 (buster)"
NAME="Debian GNU/Linux"
root@wil:/home/wil# /usr/sbin/aa-status
apparmor module is loaded.
root@wil:/home/wil# ss -tunlp
Netid State  Recv-Q Send-Q Local Address:Port  Peer Address:Port

tcp   LISTEN 0      128          0.0.0.0:4242       0.0.0.0:*     users:(("sshd",pid=523,fd=3))
tcp   LISTEN 0      128             [::]:4242          [::]:*     users:(("sshd",pid=523,fd=4))
root@wil:/home/wil# /usr/sbin/ufw status
Status: active

To                         Action      From
--                         ------      ----
4242                       ALLOW       Anywhere
4242 (v6)                  ALLOW       Anywhere (v6)
```

Chapter V

Bonus part

Bonus list:

- Set up partitions correctly so you get a structure similar to the one below:

```
# lsblk
NAME                    MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
sda                       8:0    0  30.8G  0 disk
├─sda1                    8:1    0   500M  0 part  /boot
├─sda2                    8:2    0     1K  0 part
└─sda5                    8:5    0  30.3G  0 part
  └─sda5_crypt          254:0    0  30.3G  0 crypt
    ├─LVMGroup-root     254:1    0    10G  0 lvm   /
    ├─LVMGroup-swap     254:2    0   2.3G  0 lvm   [SWAP]
    ├─LVMGroup-home     254:3    0     5G  0 lvm   /home
    ├─LVMGroup-var      254:4    0     3G  0 lvm   /var
    ├─LVMGroup-srv      254:5    0     3G  0 lvm   /srv
    ├─LVMGroup-tmp      254:6    0     3G  0 lvm   /tmp
    └─LVMGroup-var--log 254:7    0     4G  0 lvm   /var/log
sr0                      11:0    1  1024M  0 rom
```

- Set up a functional WordPress website with the following services: lighttpd, MariaDB, and PHP.
- Set up a service of your choice that you think is useful (NGINX / Apache2 excluded!). During the defense, you will have to justify your choice.

To complete the bonus part, you have the possibility to set up extra services. In this case, you may open more ports to suit your needs. Of course, the UFW rules has to be adapted accordingly.

The bonus part will only be assessed if the mandatory part is PERFECT. Perfect means the mandatory part has been integrally done and works without malfunctioning. If you have not passed ALL the mandatory requirements, your bonus part will not be evaluated at all.

Chapter VI

Submission and peer-evaluation

You only have to turn in a `signature.txt` file at the root of your `Git` repository. You must paste in it the signature of your machine's virtual disk. To get this signature, you first have to open the default installation folder (it is the folder where your VMs are saved):

- Windows: `%HOMEDRIVE%%HOMEPATH%\VirtualBox VMs\`
- Linux: `~/VirtualBox VMs/`
- MacM1: `~/Library/Containers/com.utmapp.UTM/Data/Documents/`
- MacOS: `~/VirtualBox VMs/`

Then, retrieve the signature from the `".vdi"` file (or `".qcow2` for `UTM`'users) of your virtual machine in `sha1` format. Below are 4 command examples for a `centos_serv.vdi` file:

- Windows: `certUtil -hashfile centos_serv.vdi sha1`
- Linux: `sha1sum centos_serv.vdi`
- For Mac M1: `shasum Centos.utm/Images/disk-0.qcow2`
- MacOS: `shasum centos_serv.vdi`

This is an example of what kind of output you will get:

- `6e657c4619944be17df3c31faa030c25e43e40af`

Please note that your virtual machine's signature may be altered after your first evaluation. To solve this problem, you can duplicate your virtual machine or use save state.

It is of course FORBIDDEN to turn in your virtual machine in your Git repository. During the defense, the signature of the signature.txt file will be compared with the one of your virtual machine. If the two of them are not identical, your grade will be 0.