Lab Session 2 (L2)

Secure Deployment of Virtualized Environments

Part 1 – Deploying Secure VMs

Objectives

The objectives of L2 are:

- To practice the configuration and deployment of a secure virtual machines (VM), applying measures explained in guidelines and recommendations available in the reference documentation links.
- To understand the weaknesses and vulnerabilities that each applied measure mitigate/cancel.
- To play with a penetration testing (pentesting) engine in order to test the applied measures and evaluate the security of the deployed VM.
- In addition, practice with the implementation of measures that create intended vulnerabilities in the VM, which is can be exploited by the pentesting engine.

Statement

Preliminaries

In this lab, we will create/use VMs using as virtual machine manager (hypervisor) the well-known Oracle VirtualBox environment. If you do not have it installed yet, please download it from this link:

https://www.oracle.com/virtualization/technologies/vm/downloads/virtualbox-downloads.html#vbox

In addition, please install the Oracle VM VirtualBox Extension Pack (you can find it in the same link)

Moreover, please download the following lightweight Debian distribution: https://cdimage.debian.org/cdimage/archive/12.9.0/amd64/iso-cd/debian-12.9.0-amd64-netinst.iso (we will use this as ISO image of the secure VM to create).

Finally, Download Kali Linux LiveCD (Kali 2025.3 release live image), that we will use for penetration testing purposes:

https://www.kali.org/get-kali/#kali-live

Create a document named **L2_D1** to report the activities of each of the following tasks.

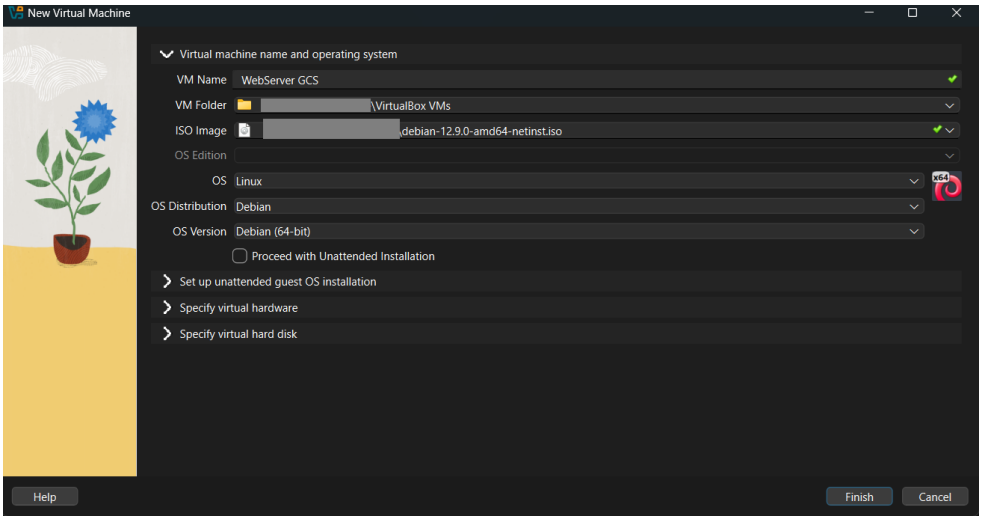
Task 0: Creating an initial VM

First of all, we we want to instantiate a VM running Ubuntu Server OS that will run (in a future) a web server. In other words, it must be configured to have Internet Access.

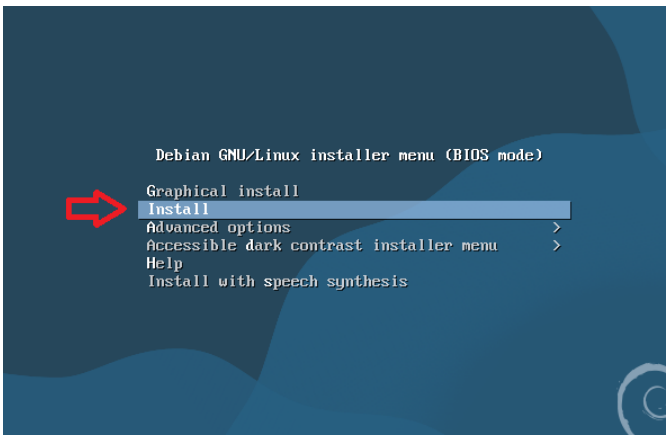
Please follow the next steps in order:

Note: Please, read all the steps before proceeding with the installation

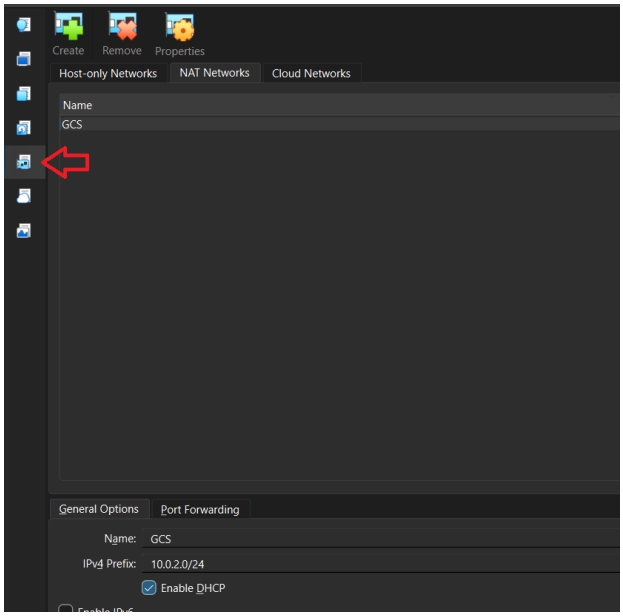
1) Create in VBOX a new VM with name "WebServerGCS", with 1 CPU, 2 GB RAM, and with the Debian ISO image (**don't forget to uncheck the "Proceed with Unattended Installation" option**):



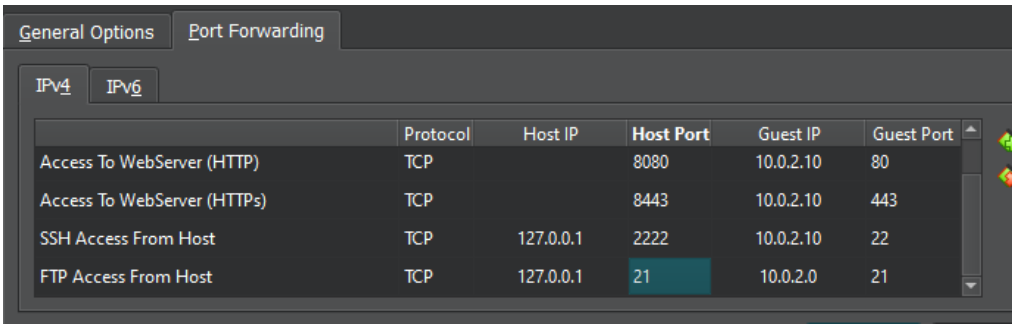
2) Install Debian in Standard mode (no graphical install – see image in next page), configuring the following parameters:
   - `Hostname: webserver`
   - `User: gcs`
   - `Password: sistemes`
   - `Root Password: sistemes`
   - `Partition disks: Guided – use entire disk – sda | All files in one partition`
   - `Software selection: SSH Server + Standard System utilities`
     - `Don't forget to` **`uncheck the Debian desktop environment`**`)`



3) Create a NAT network named GCS with 10.0.2.0/24 and DHCP

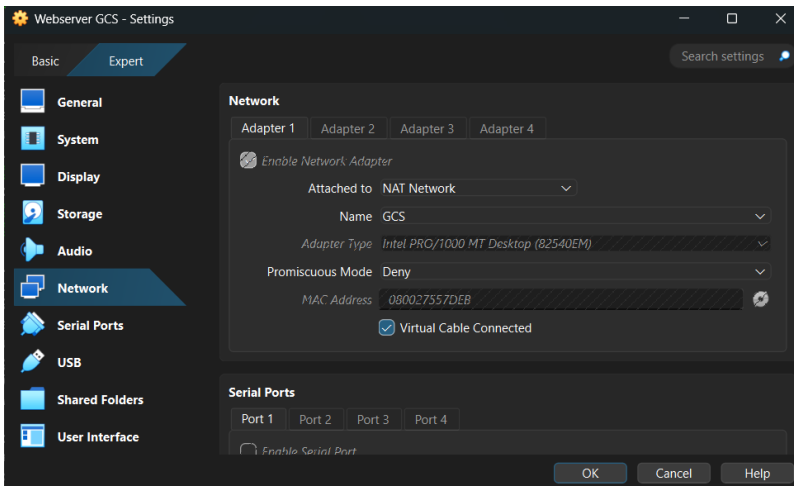


Configure the following port forwarding rules for the network:



	Protocol	Host IP	Host Port	Guest IP	Guest Port
Access To WebServer (HTTP)	TCP		8080	10.0.2.10	80
Access To WebServer (HTTPs)	TCP		8443	10.0.2.10	443
SSH Access From Host	TCP	127.0.0.1	2222	10.0.2.10	22
FTP Access From Host	TCP	127.0.0.1	21	10.0.2.0	21

4) Configure the VM to use the NAT Network we just created:



5) Access the VM again and assign a static IP (10.0.2.10/24), replacing /etc/network/interfaces with this configuration:

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
        address 10.0.2.10
        netmask 255.255.255.0
        gateway 10.0.2.1
        dns-nameservers 1.1.1.1 8.8.8.8
# This is an autoconfigured IPv6 interface
#iface enp0s3 inet6 auto
```

6) Restart networking and check connectivity:

```
#> systemctl restart networking.service
#> ip a
#> ping www.google.es
```

7) Go to your host machine and verify that you can access guest VM via ssh:

```
#> ssh gcs@localhost -p 2222
```

```
PS                    ssh gcs@localhost -p 2222
The authenticity of host '[localhost]:2222 ([127.0.0.1]:2222)' can't be established.
ED25519 key fingerprint is SHA256:j8d2xqu5SccDCFXkA6DRtQKfbJel72Jr4kf9Ac4041c.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[localhost]:2222' (ED25519) to the list of known hosts.
gcs@localhost's password:
Linux webserver 6.1.0-26-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.112-1 (2024-09-30) x86_64

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Oct 23 19:31:31 2024
gcs@webserver:~$
```

Task 1: Enhancing VM security (3p)

(This task starts from the basic VM obtained after doing Task 0. Solve questions in the deliverable)

Using the VM created in the previous step, now your mission is to improve it in terms of security. To this aim, you need to decide the best actions and practices to improve and enhance overall VM security. Be free to use any measure/policy. We suggest to follow the guidelines for secure VM deployment recommended by VirtualBox: https://www.virtualbox.org/manual/ch13.html. You can play with VM configuration parameters (touching VBOX VM configuration) as well as configuration of guest SO processes.

Among different measures, you should consider aspects such as:

- Proper configuration of VM networking mode
- Secure remote access
- Proper configuration of users and privileges
- Control/limit communication between host and guest OS's
- Proper configuration of virtual devices (units) and peripherals
- Proper configuration of ports and guest SO processes
- Use of encryption mechanisms

Note: Explain the measures you consider most relevant. It is not mandatory to apply all the measures, but try to implement at least two of them.

[Q1] For each of the applied measures/actions you tried, include a table in L2_D1 with the following template:

Measure *Name of the measure*
Description *Brief description of the measure*
Vulnerability/weakness *Describe which vulnerability or weakness this measure aims at mitigating or eliminating*
Already Implemented? *Yes, if VM of Task 0 already had that measure (No, otherwise)*
Steps/procedure (if not implemented) *Detail of the steps to follow (can include captures, screenshots)*
Means of verification *Explain the actions that could be performed to check whether the measure is effective for the proposed security improvement*
Additional comments (if any) *e.g., assumptions that you made to define a vulnerability, missing steps that have not been implemented due to some situation, etc*

For reducing the number of tables to include in the deliverable, you can aggregate sequences of actions under the same measure name (if they are needed for mitigating a common set of vulnerabilities/weaknesses).

Task 2: Adding a Vulnerable Service (3p)

(This task starts from the enhanced VM obtained after doing Task 1.Solve questions in the deliverable)

The objective of this task is to deploy a vulnerable service on the enhanced VM. To do this aim, first we execute the following commands, **as root**, to instantiate a vulnerable web server.

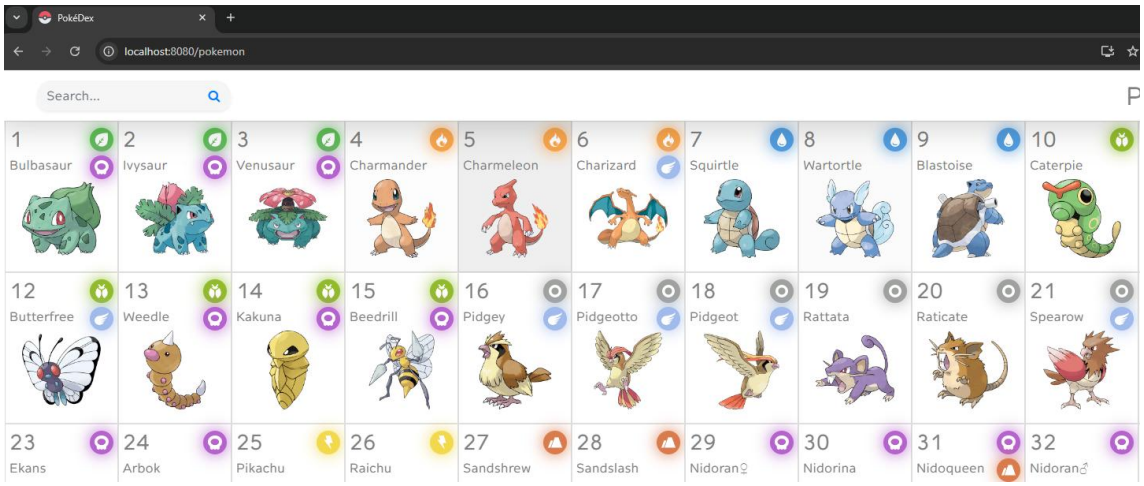
```
cd /root
apt install -y curl
curl -s https://raw.githubusercontent.com/TheMatrix97/Insecure-
WebStack/refs/tags/2.0.0/script.sh | bash
```

[Q2] Is this way of installing software secure? Why? Do you usually use this type of commands and scripts to install/configure software?

Now, download and extract the static website we want to use as example:

```
wget https://github.com/TheMatrix97/pokedex-angular-
app/releases/download/refs%2Fheads%2Fmaster/dist.tar.gz
tar -zxvf dist.tar.gz
cp -r dist/pokedex/* /var/www/html/ && rm -rf dist/ dist.tar.gz
```

Try that you can access from the host, using a browser to “http://localhost:8080”:



Task 3: Exploiting Vulnerabilities (4p)

(This task starts from the enhanced VM with vulnerable service obtained after doing Task 2. Solve questions in the deliverable)

The objective of this task is to exploit some specific vulnerabilities of the created virtualized environment. To do this aim, first we need to create a new VM that will host the pentesting tool (Kali). Follow the steps:

1) Turn on the webserver VM

2) Create a new virtual machine called “Kali” with the ISO of Kali Live CD. Select “Skip Unattended Installation”. Allocate 2 CPU and 2GB of RAM to this VM
3) Configure the same NAT network (10.0.2.0/24) than the target VM
4) Run the machine, open a terminal, and configure keymap properly:

```
setxkbmap es
```

5) Verify that you can reach from Kali VM the webserver VM:

```
ping 10.0.2.10
curl -I http://10.0.2.10
```

[Q3] Find out which command you need to execute in Kali VM to discover which services of webserver VM are exposed and reachable. Put the result (snapshot) of the command and analyse the obtained results. Did you detect some potential vulnerability at this stage?

Now let us find the version of the applications exposed in the open ports you found.

[Q4] Find out the command to do that and list of obtained versions

Our goal now is to find an outdated and (potentially) vulnerable application. To do that, you can take advantage of the Metaexploit Framework, to look for public exploits, using the *msfconsole* command, in order to search exploits associated to the services from the terminal:

```
msfconsole
search apache 2.4.60  <- This is the example for Apache 2.4.60
application
```

[Q5] Did you find a service with potential vulnerabilities related with the version? Which one?

You can now try to find an exploit of that vulnerability. In the *msfconsole* you can get info of the exploit by running:

```
info exploit/selected_exploit
```

[Q6] Did you find any exploit? Explain what the exploit does and from which vulnerability it takes advantage

After identifying the exploit, you can run it in the target webserver host:

```
use exploit/selected_exploit
Set RHOSTS 10.0.2.10
run
```

[Q7] Did the exploit successfully achieve the objective? What information you get? What is the impact of this exploit?

As you may already know, incorrect webserver configurations can expose directories and files containing secrets, which could be exploited by attackers. Is our webserver exposing any secrets?

[Q8] Perform a Brute-Force attack to list directories and files that could contain sensitive information. Did you find any secrets? Explain the procedure and include any secrets you find in the lab report

Hint: Use the wordlist located in the following path to perform the attack:

` /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt`

Finally, verify that exposed services are not vulnerable to attacks exploiting weak credentials.

[Q9] Perform a Brute-Force attack targeting authentication mechanisms on exposed services. Did you find any passwords? Explain the procedure you followed

Hint: Use the wordlist located in the following path to perform the attack
` /usr/share/wordlists/wfuzz/general/common.txt `

Hint 2: Try to find passwords for commonly used usernames (“admin”, “root”, “mysql” …etc)

Delivery

Don’t forget to save your progress on the L2 Delivery. More information about the delivery will be provided in the next sessions.

Supporting Material

https://www.kali.org/

https://owasp.org/www-project-top-ten/

https://docs.metasploit.com/