Rapport ECF VirtualLab Environment

Benjamin NAMUR

July 3, 2024

Contents

1 Introduction

L'objectif de ce projet est de développer une solution de laboratoire virtuelle dédiée au test d'une infrastructure virtualisée exploitant Proxmox VE. Cette solution impliquera le déploiement de cinq machines virtuelles, comprenant deux systèmes Windows et trois systèmes Linux. De plus, nous mettrons en place des solutions spécifiques pour faciliter l'accès à distance ainsi que l'automatisation du déploiement des machines virtuelles..

En résumé, cette documentation detaille l'approche méthodique utilisée pour atteindre les objectifs fixés sur cet ECF, offrant ainsi une documentation complète pour les étapes nécessaires à la réussite de ce challenge.

2 Les Outils

Nos premiers outils sont **VirtualBox** et **ProxMox VE 8.2**.

Proxmox est une plateforme de virtualisation, elle permet le déploiement et la gestion de machines virtuelles et de conteneurs.

1- Choix des VMs qui seront installées sur le laboratoire :

- Parrot-Security 6.1
- ZorinOS 17.1
- Ubuntu 24.04
- Deux Windows 10

2- Pour l'accès à distance des machines virtuelles :

- Apache Guacamole

Apache Guacamole est une passerelle de bureau à distance sans client. Il prend en charge des protocoles standards comme VNC, RDP et SSH.

Il faut que cet accès à distance soit fiable et sécurisée, nous verront cela plus loin dans la documentation.

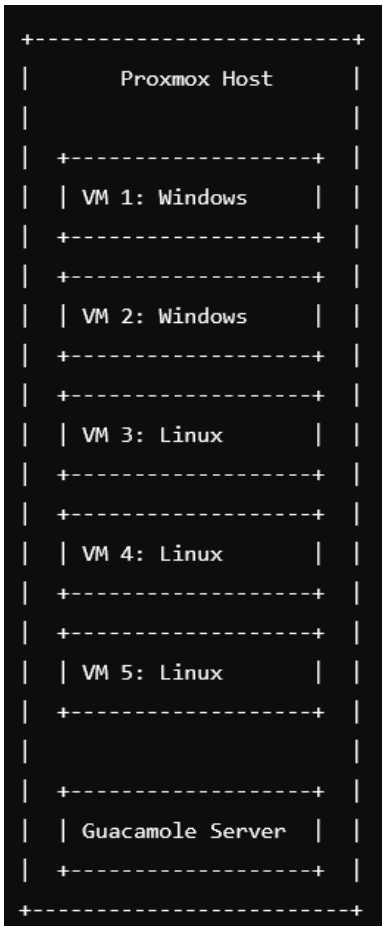
3- Pour le choix de l'automatisation du déploiement de l'infrastructure virtuelle nous utiliseront l'outil :

- Vagrant

Vagrant permet la création et la configuration d'environnements de développementlégers, reproductibles et portables.

Avec nos outils bien différenciés, nous pouvons maintenant passer au processus de mise en place. Nous commençons par faire un schéma pour visualiser à quoi devrait ressembler notre infrastructure.

3 Schéma


Proxmox Host
VM 1: Windows
VM 2: Windows
VM 3: Linux
VM 4: Linux
VM 5: Linux
Guacamole Server

4 Configuration

4.1 Installation de Proxmox VE

- Télécharger Proxmox VE

- Suivre les instructions d'installation pour configurer notre hôte Proxmox.
On s'assure que notre matériel est compatible et que la connexion réseau est correctement configurée.

Durant l'installation, un premier problème vient se présenter. Sur Windows 11, il y a une incompatibilité entre Hyper-V et la virtualisation VT-r.

Pour régler le problème, il faut d'abord activer les paramètres de virtualisation dans le BIOS de l'ordinateur. Ensuite, désactivez l'isolation du noyau de Windows 11 ainsi que ses dépendances virtuelles.

```
Touche Win + R

On tape : "control"

On cherche : "Programme"
- Active ou desactive des fonctionnalites Windows

Decocher :
- Hyper-V,
- Plateforme de l'hyperviseur Windows
- Plateforme de Machine Virtuellle

Reboot de la machine
```

En suivant ces étapes, le problème avec VT-r ne devrait plus avoir lieu.

Pour la configuration du **réseau** sous Proxmox :

On gère l'interface réseau avec la commande **nano**, dans mon cas pour une adresse IP non statique :

```
nano /etc/network/interfaces

auto lo
iface lo inet loopback

auto vmbr0
iface vmbr0 inet dhcp
    bridge_ports eth0
    bridge_stp off
    bridge_fd 0
    dns-nameservers 8.8.8.8 8.8.4.4
```

Suivi de la modification du **nameserver** pour le **DNS**

```
nano /etc/resolv.conf

nameserver 8.8.8.8
nameserver 8.8.4.4
```

Pour savoir si les modifications fonctionnent, on fait deux nouvelles commandes :

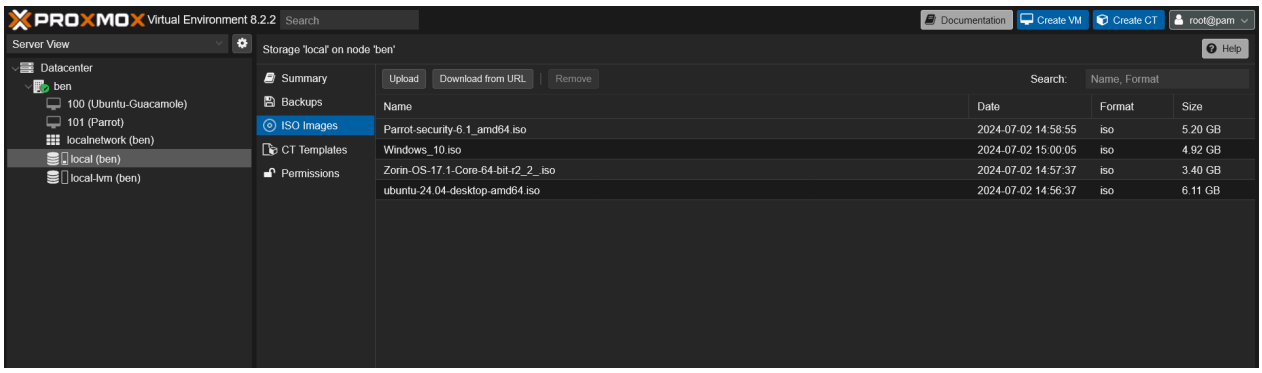
```
nslookup google.com
ping -c 4 google.com
```

```
root@ben:~# ip a
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 noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master vmbr0 state UP group default qlen 1000
    link/ether 08:00:27:df:43:1a brd ff:ff:ff:ff:ff:ff
3: vmbr0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 08:00:27:df:43:1a brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.115/24 brd 192.168.1.255 scope global dynamic vmbr0
       valid_lft 86098sec preferred_lft 86098sec
    inet6 fe80::a00:27ff:fedf:431a/64 scope link
       valid_lft forever preferred_lft forever
root@ben:~# nslookup google.com
Server:         192.168.1.254
Address:        192.168.1.254#53

Non-authoritative answer:
Name:   google.com
Address: 142.250.179.110
Name:   google.com
Address: 2a00:1450:4007:80d::200e

root@ben:~# ping -c google.com
ping: invalid argument: 'google.com'
root@ben:~# ping -c 4  google.com
PING google.com (142.250.75.238) 56(84) bytes of data.
64 bytes from par10s41-in-f14.1e100.net (142.250.75.238): icmp_seq=1 ttl=115 time=18.1 ms
64 bytes from par10s41-in-f14.1e100.net (142.250.75.238): icmp_seq=2 ttl=115 time=18.1 ms
64 bytes from par10s41-in-f14.1e100.net (142.250.75.238): icmp_seq=3 ttl=115 time=17.8 ms
64 bytes from par10s41-in-f14.1e100.net (142.250.75.238): icmp_seq=4 ttl=115 time=17.2 ms

--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3173ms
rtt min/avg/max/mdev = 17.172/17.790/18.121/0.375 ms
root@ben:~#
```

Si on a un retour de ping, on est bon.

4.2 Upload et Installation des machines virtuelles

De retour sur Proxmox avec la configuration réseau à jour, on peut maintenant uploader nos ISO de Linux et Windows qui vont nous permettre l'installation.



Pour la configuration des VMs Linux et Windows, il n'y a rien à faire, il faut juste suivre les instructions.

La connexion réseau se fait directement en **Bridge**, aucune manipulation de plus. On pourrait aller plus loin en créant des **VLANs** et connexion **bridge** spécifique pour chaque VM.

4.3 Installation de Guacamole sur une VM Ubuntu

C'est une partie de tutoriel, trouvé sur différents forums, je n'ai pas eu le temps de finir la configurations. Aucune idée de si ça fonctionne.

Apache Guacamole a besoin de dépendances pour son bon fonctionnement, ici c'est la configuration pour avoir accès au VMs via RDP. Il faut penser également à actvier le RDP sur les VMs installées :

- Tomcat
- MySQL/MariaDB
- Guacd
- Client web

4.3.1 Installer les dépendances

```
sudo apt-get update
sudo apt-get install libcairo2-dev libjpeg62-turbo-dev libpng-dev libtool-
   bin libossp-uuid-dev \
libavcodec-dev libavformat-dev libavutil-dev libswscale-dev freerdp2-dev
   libpango1.0-dev \
libssh2-1-dev libtelnet-dev libvncserver-dev libpulse-dev libssl-dev
   libvorbis-dev \
libwebp-dev wget
```

4.3.2 Télécharger et compiler Guacamole Server

```
wget https://apache.org/dyn/closer.lua/guacamole/1.4.0/source/guacamole-
   server-1.4.0.tar.gz
tar -xzf guacamole-server-1.4.0.tar.gz
cd guacamole-server-1.4.0
./configure --with-init-dir=/etc/init.d
make
sudo make install
sudo ldconfig
sudo systemctl enable guacd
sudo systemctl start guacd
```

4.4 Installation de Guacamole Client Web

4.4.1 Tomcat

```
sudo apt-get install tomcat9
```

4.4.2 Téléchargement et déploiement du fichier .war de Guacamole

```
wget https://apache.org/dyn/closer.lua/guacamole/1.4.0/binary/guacamole
   -1.4.0.war
sudo mv guacamole-1.4.0.war /var/lib/tomcat9/webapps/guacamole.war
sudo systemctl restart tomcat9
```

4.5 Configurer Guacamole

4.5.1 Créer le répertoire de configuration

```
sudo mkdir /etc/guacamole
```

4.5.2 Configurer le fichier guacamole.properties

Créez et éditez le fichier **/etc/guacamole/guacamole.properties**

```
sudo nano /etc/guacamole/guacamole.properties
```

On ajoute les lignes suivantes :

```
guacd-hostname: localhost
guacd-port: 4822
```

4.5.3 Configurer l'authentification

Il faut télécharger et déployer le fichier mysql-connector-java pour l'authentification via une base de données MySQL, ou configurer un fichier user-mapping.xml pour une authentification simple.

Pour une authentification simple :

Créez le fichier **/etc/guacamole/user-mapping.xml**

```
sudo nano /etc/guacamole/user-mapping.xml
```

Puis on ajoute les lignes suivantes :

```
<user-mapping>
    <authorize username="guacadmin" password="guacadmin">
        <connection name="My RDP Connection">
            <protocol>rdp</protocol>
            <param name="hostname">YOUR_VM_IP_ADDRESS</param>
            <param name="port">3389</param>
            <param name="username">YOUR_VM_USERNAME</param>
            <param name="password">YOUR_VM_PASSWORD</param>
        </connection>
    </authorize>
</user-mapping>
```

4.6 Guacamole

On ouvre le navigateur web et on accéde à Guacamole via http://192.168.0.100:8080/guacamole.

5 Sécurité

Pour la sécurité, il faut utiliser le protocole HTTPS avec certificat, un pare-feu peut-être également essentiel pour restreindre l'accès uniquement aux adresses IP autorisées

6 Vagrant

Il faut installer Vagrant sur la machine qui gère l'environnement.

On crée un Vagrantfile pour définir la configuration de nos VMs

On installe un plugin pour proxmox

```
$ vagrant plugin install vagrant-proxmox
```

COPIER COLLER, aucune idée de si ça fonctionne.

```
Vagrant.configure("2") do |config|
  config.vm.provider :proxmox do |proxmox|
    proxmox.endpoint = 'https://proxmox.example.com/api2/json'
    proxmox.user_name =
    proxmox.password = 'Metabief'
    proxmox.vm_name = 'Parrot-security'
    proxmox.template = 'local:vztmpl/ubuntu-20.04-standard_20.04-1_amd64.
       tar.gz'
    proxmox.memory = 2048
    proxmox.cores = 2
  end
  config.vm.provision "shell", inline: <<-SHELL
    sudo apt-get update
    sudo apt-get install -y nginx
  SHELL
end
```

7 CONCLUSION

En conclusion, très gros projet et vraiment intéressant, malheureusement beaucoup trop de problèmes de configuration avec les différents réseaux, les différentes machines.

Puis un manque de temps énorme pour pouvoir finir cet ECF.