

Adiministracion de Servidores

Práctica 2 Firewall

Gabriel Fernando Sánchez Reina

José Manuel Morales García

1 Sobre esta práctica

En el presente documento se describe la entrega de ejercicios de la práctica de Administración de servidores sobre redes. Las prácticas se realizarán por parejas, teniendo que ser conocedores los miembros del equipo de todos los ejercicios realizados.

La entrega consistirá en un documento PDF donde se explique cada uno de los pasos seguidos para la correcta resolución de los enunciados, así como los ficheros Vagrant.

2 Instalación de máquinas virtuales mediante Vagrant (2 puntos)

En esta primera parte vamos a crear el entorno de trabajo, consistente en tres máquinas virtuales, pertenecientes a una misma red privada.

Las máquinas se tendrán que crear a partir de un único fichero Vagrant.

1. VM1, con IP 192.168.2.101
2. VM2, con IP 192.168.2.102
3. VM3, con IP 192.168.2.103

Las máquinas tendrán la siguiente configuración:

- *nmpp tiene que estar instalado en todos.*
- *iptables en todas las máquinas (debería estar instalado por defecto).*
- *ufw en la máquina VM1 (debería de estar instalado por defecto).*
- *fwbuilder La instalación de los paquetes se deberá realizar mediante la provisión de Vagrant.*

Primero, se abre una terminal en la carpeta donde se van a establecer las maquinas virtuales, e inicializamos un Vagrantfile con “vagrant init hashicorp/precise64“. Luego lo abrimos, y lo dejamos de la siguiente forma:

```
Vagrant.configure("2") do |config|

  config.vm.provision :shell, path: "bootstrap.sh"

  config.vm.define "vm1" do |vm1|
    vm1.vm.box="hashicorp/precise64"
    vm1.vm.hostname="vm1"
    vm1.vm.network "private_network", ip: "192.168.2.101"
  end

  config.vm.define "vm2" do |vm2|
    vm2.vm.box="hashicorp/precise64"
    vm2.vm.hostname="vm2"
    vm2.vm.network "private_network", ip: "192.168.2.102"
  end

  config.vm.define "vm3" do |vm3|
    vm3.vm.box="hashicorp/precise64"
    vm3.vm.hostname="vm3"
    vm3.vm.network "private_network", ip: "192.168.2.103"
  end

end
```

Ahora, en la misma carpeta escribimos el archivo de aprovisionamiento “bootstrap.sh”, y dentro escribimos los comandos “apt-get update” y “apt-get install -y nmap”, de forma que todas las máquinas lo tengan; ufw e iptables vienen por defecto.

```
#!/usr/bin/env bash
```

```
apt-get update  
apt-get install -y nmap
```

Luego ejecutamos “vagrant up”, y deberíamos ver que las 3 máquinas se inicializan correctamente.

Luego, abrimos 3 terminales independientes, y nos conectamos en cada una a una máquina con “vagrant ssh vmX” con X=1, 2 y 3 respectivamente.

3 Visibilidad de las máquinas (1 punto)

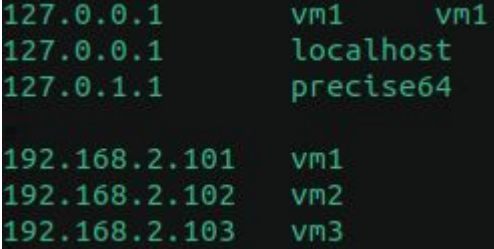
Para los distintos ejercicios, se identifica a las máquinas como VM1, VM2 y VM3. Por comodidad, es recomendable poder usar nombres en las reglas. Para ello, se puede añadir en /etc/hosts una línea asociando un nombre y una IP con la siguiente sintaxis

<IP><Nombre><Alias>

como:

192.168.1.101 vm1

En cada máquina, editamos el archivo hosts con “sudo nano /etc/hosts”, y añadimos los 3 nombres con su ip respectiva, de forma que acabe de forma similar a la imagen.



```
127.0.0.1    vm1      vm1  
127.0.0.1    localhost  
127.0.1.1    precise64  
  
192.168.2.101 vm1  
192.168.2.102 vm2  
192.168.2.103 vm3
```

```
vagrant@vm1:~$ ping vm1
PING vm1 (127.0.0.1) 56(84) bytes of data.
64 bytes from vm1 (127.0.0.1): icmp_req=1 ttl=64 time=0.046 ms
64 bytes from vm1 (127.0.0.1): icmp_req=2 ttl=64 time=0.059 ms
64 bytes from vm1 (127.0.0.1): icmp_req=3 ttl=64 time=0.061 ms
64 bytes from vm1 (127.0.0.1): icmp_req=4 ttl=64 time=0.053 ms
64 bytes from vm1 (127.0.0.1): icmp_req=5 ttl=64 time=0.067 ms
^C
--- vm1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3998ms
rtt min/avg/max/mdev = 0.046/0.057/0.067/0.008 ms
vagrant@vm1:~$ nmap vm1
Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-20 12:04 UTC
Nmap scan report for vm1 (127.0.0.1)
Host is up (0.00031s latency).
Hostname vm1 resolves to 2 IPs. Only scanned 127.0.0.1
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
Nmap done: 1 IP address (1 host up) scanned in 0.05 seconds
vagrant@vm1:~$
```

4 Configuraciones iptables

En cada ejercicio, se deberá comprobar el acceso antes y después de aplicar las reglas del cortafuegos.

4.1 Primeras pruebas (1 punto)

En este ejercicio se pide testear VM1 desde VM2, realizando los siguientes ejercicios:

1. Desde VM2 comprobar los puertos que VM1 tiene abiertos.

Para ver los puertos, ejecutamos “nmap vm1”

2. Prohibir el acceso por ssh.

SSH se ejecuta sobre el puerto 22, basta con bloquearlo con

“sudo iptables -A INPUT -p tcp --dport 22 -j DROP”

3. Responde a las siguientes preguntas:

- ¿Qué ha pasado?
- ¿Puedo crear una nueva conexión?
- ¿La consola sigue funcionando?

La terminal ha quedado bloqueada y no responde. Al forzar su cierre e intentar volver a conectarse por ssh, no llega a conectarse. Al intentar apagar la máquina virtual, no consigue cerrarla correctamente, y fuerza el apagado.

```
neo@Dawnmoon:~/Escritorio/Administracion de servidores/P2$ sudo vagrant halt vm1
[sudo] contraseña para neo:
==> vm1: Attempting graceful shutdown of VM...
vm1: Guest communication could not be established! This is usually because
vm1: SSH is not running, the authentication information was changed,
vm1: or some other networking issue. Vagrant will force halt, if
vm1: capable.
==> vm1: Forcing shutdown of VM...
neo@Dawnmoon:~/Escritorio/Administracion de servidores/P2$
```

Podemos ver en la imagen que la máquina virtual no consigue transmitir la orden de apagado y fuerza a esta a apagarse. Como no se guardaron los cambios en iptables al reiniciarla esta todo en el estado previo al bloqueo.

4.2 Configuración Mínima (2 puntos)

En los ejercicios siguientes, siempre debe de partir de esta configuración.

- Permitir conexiones locales.
"sudo iptables -A INPUT -i lo -j ACCEPT"
- Permitir conexiones ya establecidas.
"sudo iptables -A INPUT -m state --state ESTABLISHED -j ACCEPT"
- Políticas por defecto de rechazar en input.
"sudo iptables -A INPUT -j DROP"

Veremos que en las conexiones permitidas, el nmap las mostrará como "closed", debido a que no hay ningún servicio escuchando. Las conexiones rechazadas por el cortafuegos aparecerán como "filtered".

```
vagrant@vm2:~$ nmap -p 25478 vm1

Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-27 11:46 UTC
Nmap scan report for vm1 (192.168.2.101)
Host is up (0.00059s latency).
PORT      STATE      SERVICE
25478/tcp  filtered  unknown
Nmap done: 1 IP address (1 host up) scanned in 1.49 seconds
```

Puede verse que mantener las conexiones ya establecidas funciona porque al denegar todas las demás conexiones, en esta ocasión no se queda bloqueada la terminal al rechazar SSH.

4.3 Configurando servidor web completo (2 puntos)

Configurar VM1 para que tenga la configuración de un servidor web, permitiendo: • Todos se conecten a los puertos http y https.

- Conexión únicamente por parte de VM2 al servidor ftp.

- Configurar VM1 para que sólo se pueda conectar localmente a mysql.

- Permitir conexiones a puertos 80 y 443 (http y https) desde cualquier IP:

“sudo iptables -A INPUT -p tcp --match multiport --dports 80,443 -j ACCEPT”

```
vagrant@vm2:~$ nmap -p 80 vm1
Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-27 10:47 UTC
Nmap scan report for vm1 (192.168.2.101)
Host is up (0.00079s latency).
PORT      STATE SERVICE
80/tcp    closed http
Nmap done: 1 IP address (1 host up) scanned in 1.50 seconds
vagrant@vm2:~$ nmap -p 443 vm1
Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-27 10:48 UTC
Nmap scan report for vm1 (192.168.2.101)
Host is up (0.00039s latency).
PORT      STATE SERVICE
443/tcp   filtered https
Nmap done: 1 IP address (1 host up) scanned in 1.51 seconds
vagrant@vm2:~$
```

- Permitir conexiones ftp solo por parte de vm2 (puertos 20 y 21):

“sudo iptables -A INPUT -p tcp -s 192.168.2.102 --match multiport --dports 20,21 -j ACCEPT”


```
vagrant@vm2:~$ nmap -p 20 vm1

Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-27 10:58 UTC
Nmap scan report for vm1 (192.168.2.101)
Host is up (0.00059s latency).
PORT      STATE SERVICE
20/tcp    closed ftp-data

Nmap done: 1 IP address (1 host up) scanned in 1.29 seconds
vagrant@vm2:~$ nmap -p 20 vm1

Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-27 10:59 UTC
Nmap scan report for vm1 (192.168.2.101)
Host is up (0.00046s latency).
PORT      STATE SERVICE
20/tcp    closed ftp-data

Nmap done: 1 IP address (1 host up) scanned in 1.24 seconds
vagrant@vm2:~$
```

```
vagrant@vm3:~$ nmap -p 20 vm1

Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-27 11:02 UTC
Nmap scan report for vm1 (192.168.2.101)
Host is up (0.00039s latency).
PORT      STATE SERVICE
20/tcp    filtered ftp-data

Nmap done: 1 IP address (1 host up) scanned in 1.50 seconds
vagrant@vm3:~$ nmap -p 21 vm1

Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-27 11:02 UTC
Nmap scan report for vm1 (192.168.2.101)
Host is up (0.00042s latency).
PORT      STATE SERVICE
21/tcp    filtered ftp

Nmap done: 1 IP address (1 host up) scanned in 1.58 seconds
vagrant@vm3:~$
```

- Permitir solo conexiones locales a mySQL (puerto 3306):

“sudo iptables -A INPUT -i lo -p tcp --dport 3306 -j ACCEPT”

```
vagrant@vm1: ~  
Archivo  Editar  Ver  Buscar  Terminal  Ayuda  
vagrant@vm1:~$ nmap -p 3306 vm1  
  
Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-27 11:04 UTC  
Nmap scan report for vm1 (127.0.0.1)  
Host is up (0.000069s latency).  
Hostname vm1 resolves to 2 IPs. Only scanned 127.0.0.1  
PORT      STATE SERVICE  
3306/tcp  closed mysql  
  
Nmap done: 1 IP address (1 host up) scanned in 0.04 seconds  
vagrant@vm1:~$  
  
vagrant@vm3: ~  
Archivo  Editar  Ver  Buscar  Terminal  Ayuda  
vagrant@vm3:~$ nmap -p 3306 vm1  
  
Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-27 11:04 UTC  
Nmap scan report for vm1 (192.168.2.101)  
Host is up (0.00039s latency).  
PORT      STATE SERVICE  
3306/tcp  filtered mysql  
  
Nmap done: 1 IP address (1 host up) scanned in 1.73 seconds  
vagrant@vm3:~$
```

4.4 Poniendo excepciones (1 punto)

Permitir conectar a VM1 desde VM2 y VM3 el acceso a los puertos desde 1:1000, con la excepción de que VM2 no puede conectar por FTP.

Para permitir un rango de puertos pero impedir una excepción concreta, se escribe la excepción como primera regla y luego la regla permitir el rango, de forma que al seguir el orden descarte siempre la que queremos.

- Negar a vm2 conectarse por ftp:

```
"sudo iptables -A INPUT -p tcp -s 192.168.2.102 --match multiport --dports 20,21 -j DROP"
```

- Permitir todo lo demas a vm2 y vm3 en el rango:

```
"sudo iptables -A INPUT -s 192.168.2.102 -p tcp --dport 1:1000 -j ACCEPT"
```

```
"sudo iptables -A INPUT -s 192.168.2.103 -p tcp --dport 1:1000 -j ACCEPT"
```

```
vagrant@vm2:~$ sudo nmap -p 500 vm1
Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-27 12:11 UTC
Nmap scan report for vm1 (192.168.2.101)
Host is up (0.00075s latency).
PORT      STATE SERVICE
500/tcp   closed isakmp
MAC Address: 08:00:27:E1:A6:A2 (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 0.39 seconds
vagrant@vm2:~$ sudo nmap -p 20 vm1
Starting Nmap 5.21 ( http://nmap.org ) at 2019-03-27 12:11 UTC
Nmap scan report for vm1 (192.168.2.101)
Host is up (0.00034s latency).
PORT      STATE SERVICE
20/tcp    filtered ftp-data
MAC Address: 08:00:27:E1:A6:A2 (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 0.32 seconds
vagrant@vm2:~$
```

5 UFW (1 punto)

Los números de los puertos pueden también cambiarse por los protocolos, por ejemplo “ufw allow http” en vez de 80.

- Permitir conexiones a puertos 80 y 443 (http y https) desde cualquier IP:

```
“sudo ufw allow 80”
```

```
“sudo ufw allow 443”
```

- Permitir conexiones ftp solo por parte de vm2 (puertos 20 y 21):

```
“sudo ufw allow from 192.168.2.102 to any port 20”
```

```
“sudo ufw allow from 192.168.2.102 to any port 21”
```

- Permitir sólo conexiones locales a mySQL (puerto 3306):

```
“sudo ufw allow from 127.0.0.1 to any port 3306”
```

Los resultados son iguales que el 4.3