



Universidade do Minho  
Escola de Engenharia

# MESTRADO INTEGRADO EM ENGENHARIA DE TELECOMUNICAÇÕES E INFORMÁTICA

SEGURANÇA EM REDES DE COMPUTADORES

## PRÁTICAS COM FIREWALLS (IPTABLES)

TRABALHO PRÁTICO Nº5

Grupo 9:

Cláudia Cristiana de Amorim Dias - A78232

David José Ressurreição Alves - A79625

Guimarães, 10 de Maio de 2019

# Índice

<b>1</b>	<b>Introdução</b>	<b>4</b>
<b>2</b>	<b>Tarefa 1</b>	<b>5</b>
2.1	Teste dos serviços activados . . . . .	5
2.2	Execução do comando “ <i>iptables -L -v</i> ” . . . . .	8
2.3	Execução do comando “ <i>iptables-save &gt; iptables.dump</i> ” . . . . .	9
2.4	Desativação da <i>firewall</i> . . . . .	10
<b>3</b>	<b>Tarefa 2</b>	<b>11</b>
3.1	Verificação de conectividade . . . . .	11
3.2	Execução do comando “ <i>nmap -sS &lt;ip_address&gt;</i> ” . . . . .	12
3.3	Execução do comando “ <i>w3m http://&lt;ip_address&gt;</i> ” . . . . .	13
3.4	Execução do comando “ <i>ftp &lt;ip_address&gt;</i> ” . . . . .	14
3.5	Execução do comando “ <i>ssh &lt;ip_address&gt;</i> ” . . . . .	14
<b>4</b>	<b>Tarefa 3</b>	<b>15</b>
4.1	Configuração personalizada da <i>firewall</i> . . . . .	15
4.2	Execução do comando “ <i>iptables -L -v</i> ” . . . . .	17
4.3	Execução do comando <i>ping&lt;ip_address&gt;</i> . . . . .	18
4.4	Execução do comando “ <i>w3m http://&lt;ip_address&gt;</i> ” . . . . .	18
4.5	Execução do comando “ <i>ftp &lt;ip_address&gt;</i> ” . . . . .	19
4.6	Execução do comando “ <i>nmap -sS &lt;ip_address&gt;</i> ” . . . . .	19
4.7	Execução do comando “ <i>iptables -L -v</i> ” . . . . .	20
4.8	Configuração recorrendo a interfaces gráficas . . . . .	21

## Lista de Figuras

1	Verificação dos serviços inicializados. . . . .	5
2	Página exemplo do servidor HTTP. . . . .	6
3	Teste do servidor FTP usando o <i>browser</i> . . . . .	6
4	Teste do serviço SSH usando o terminal. . . . .	7
5	Teste do servidor FTP usando o terminal. . . . .	7
6	Políticas automaticamente definidas pela <i>firewall</i> . . . . .	8
7	Execução do comando para guardar as políticas atuais num ficheiro. . . . .	9
8	Execução do comando para verificar políticas atuais. . . . .	10
9	Execução do comando para verificar a conectividade entre cliente e servidor. . . . .	11
10	Execução do comando “ <i>nmap -sS</i> ”. . . . .	12
11	Execução do comando “ <i>nmap -sV</i> ”. . . . .	12
12	Execução do comando “ <i>nmap -O</i> ”. . . . .	13
13	Execução do comando “ <i>w3m http://10.0.2.7</i> ”. . . . .	13
14	Execução do comando “ <i>ftp 10.0.2.7</i> ”. . . . .	14
15	Execução do comando “ <i>ssh 10.0.2.7</i> ”. . . . .	14
16	Ativação do serviço FTP na <i>firewall</i> . . . . .	15
17	Ativação do serviço SSH na <i>firewall</i> . . . . .	15
18	Ativação do serviço HTTP na <i>firewall</i> . . . . .	16
19	Rejeição de pedidos "Echo Request" pela <i>firewall</i> . . . . .	16
20	Políticas de segurança personalizadas pela <i>firewall</i> . . . . .	17
21	Execução do comando “ <i>ping 10.0.2.7</i> ”. . . . .	18
22	Execução do comando “ <i>w3m http://10.0.2.7</i> ”. . . . .	18
23	Execução do comando “ <i>ftp 10.0.2.7</i> ”. . . . .	19
24	Execução do comando “ <i>nmap -sS 10.0.2.7</i> ”. . . . .	19
25	Execução do comando “ <i>iptables -L -v</i> ”. . . . .	20
26	Configuração da <i>firewall</i> “SRC-G9”. . . . .	21
27	Configuração da interface “eth0”. . . . .	21
28	Configuração da “Rede NAT”. . . . .	22
29	Regras definidas para a <i>firewall</i> “SRC-G9”. . . . .	22
30	Execução do comando “ <i>iptables -L -v</i> ”. . . . .	23
31	Execução do comando <i>ping</i> do cliente para o servidor. . . . .	24


32	Leitura do ficheiro de <i>log</i> do servidor. . . . .	24
33	Teste de funcionamento do serviço HTTP. . . . .	25
34	Teste de funcionamento do serviço FTP. . . . .	25
35	Teste de funcionamento do serviço SSH. . . . .	25

# 1. Introdução

No âmbito da realização do trabalho prático proposto na unidade curricular de Segurança em Redes de Computadores, sobre tecnologias para segurança em redes, serve o presente relatório para mostrar os resultados da elaboração do mesmo. Para a realização deste trabalho prático foram utilizados os sistemas operativos Kali Linux (como cliente), e o CentOS, versão 6.10 (como servidor).

## 2. Tarefa 1

### 2.1. Teste dos serviços activados

Depois de, tal como pedido, se terem ativado e iniciado os serviços HTTP, FTP, SSH no servidor, procedemos ao teste dos mesmos. 

Para isso, executando o comando `"netstat -l"`, verificámos que os mesmos estavam listados, tal como se pode ver na figura 1.

```
[osboxes@localhost ~]$ netstat -l
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         Stat
e
tcp        0      0 *:sunrpc                *:*                     LIST
EN
tcp        0      0 *:46386                 *:*                     LIST
EN
tcp        0      0 *:ftp                   *:*                     LIST
EN
tcp        0      0 *:ssh                   *:*                     LIST
EN
tcp        0      0 localhost.localdomain:ipp *:*                     LIST
EN
tcp        0      0 localhost.localdomain:smtp *:*                     LIST
EN
tcp        0      0 *:33184                 *:*                     LIST
EN
tcp        0      0 *:sunrpc                *:*                     LIST
EN
tcp        0      0 *:http                  *:*                     LIST
EN
tcp        0      0 *:ssh                   *:*                     LIST
EN
tcp        0      0 localhost6.localdomain6:ipp *:*                     LIST
EN
udp        0      0 *:bootpc                *:*
```

Figura 1: Verificação dos serviços inicializados.

De seguida, usando o *browser* Mozilla Firefox, presente por defeito no CentOS, verificou-se o correto funcionamento do servidor HTTP, tal como se pode ver na figura 2, na qual é apresentada uma página de teste, definida automaticamente pelo CentOS.

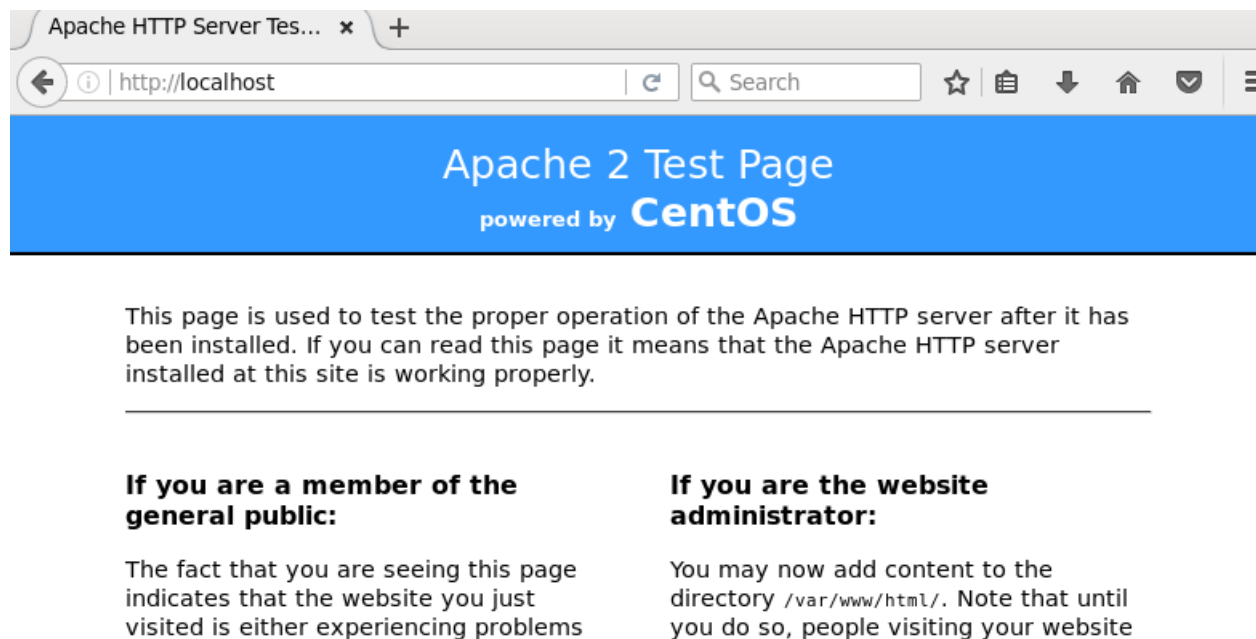


Figura 2: Página exemplo do servidor HTTP.

Depois, para o serviço FTP, usando novamente o mesmo *browser*, procedemos ao teste do mesmo, tal como mostra a figura 3, na qual se pode observar os ficheiros contidos no servidor FTP.

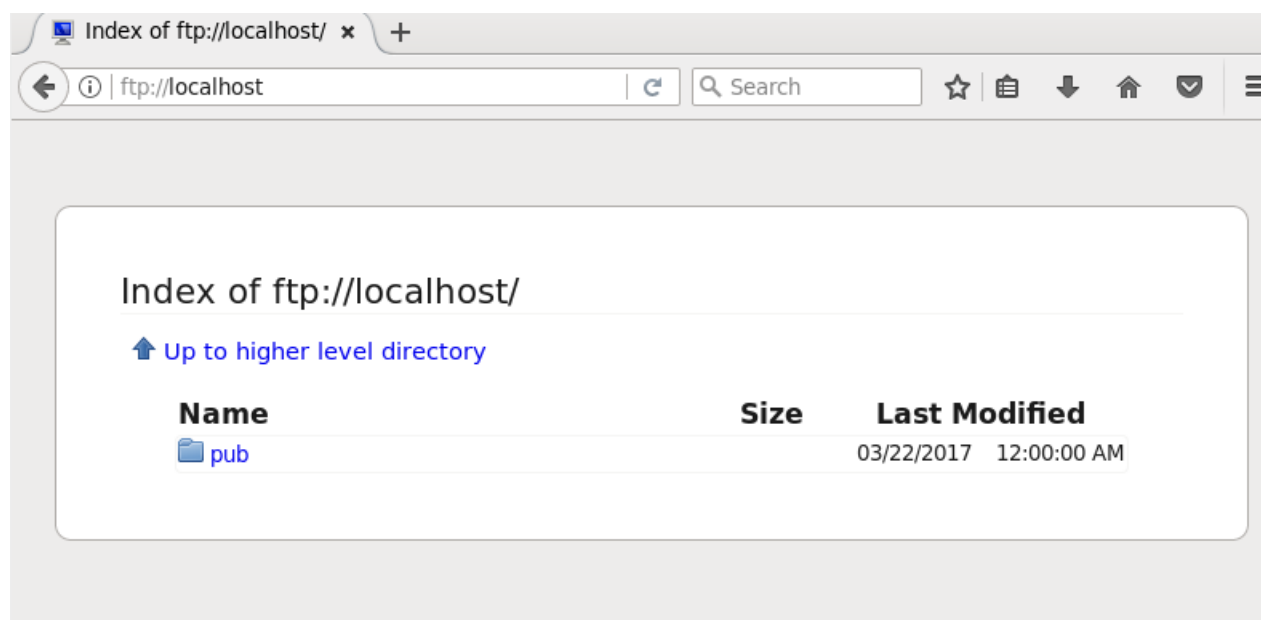


Figura 3: Teste do servidor FTP usando o *browser*.

Por fim, procedemos ao acesso dos serviços FTP e SSH através do terminal, como mostram as figuras 4 e 5, respetivamente.

```
[root@localhost ~]# ftp localhost
Connected to localhost (127.0.0.1).
220 (vsFTPd 2.2.2)
Name (localhost:root): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
227 Entering Passive Mode (127,0,0,1,143,151).
150 Here comes the directory listing.
drwxr-xr-x  2 0      0              4096 Mar 22  2017 pub
226 Directory send OK.
ftp> ^\Quit (core dumped)
[root@localhost ~]#
```

Figura 4: Teste do serviço SSH usando o terminal.

```
[root@localhost ~]# ssh localhost
root@localhost's password:
Last login: Sun May  5 15:10:28 2019 from localhost.localdomain
[root@localhost ~]# ls
anaconda-ks.cfg  Documents  Music      post-install  Public  Videos
Desktop          Downloads  Pictures   post-install.log  Templates
[root@localhost ~]# exit
logout
Connection to localhost closed.
[root@localhost ~]#
```

Figura 5: Teste do servidor FTP usando o terminal.



## 2.2. Execução do comando “*iptables -L -v*”



Após se ter ativando a *firewall* do servidor usando o comando “*system-config-firewall-tui*”, para observar as definições criadas pela mesma, recorrendo ao comando “*iptables -L -v*”, observámos que a *firewall* definiu as seguintes políticas que são apresentadas na figura 6.

```
[root@localhost ~]# iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination          state
   0    0 ACCEPT     all  --  any    any    anywhere            anywhere            state RELATED,ESTABLISHED
   0    0 ACCEPT     icmp --  any    any    anywhere            anywhere
   0    0 ACCEPT     all  --  lo     any    anywhere            anywhere
   0    0 ACCEPT     tcp  --  any    any    anywhere            anywhere            state NEW tcp dpt:ssh
   0    0 REJECT     all  --  any    any    anywhere            anywhere            reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination          reject-with
   0    0 REJECT     all  --  any    any    anywhere            anywhere            icmp-host-prohibited

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
```

Figura 6: Políticas automaticamente definidas pela *firewall*.

Segundo o que se observa da figura, pode-se concluir que a *firewall* rejeita todos os pedidos que chegam ao servidor para aceder a qualquer serviço, excepto o serviço SSH, sendo este o único serviço que a *firewall* confia e aceita todos os pedidos feitos ao servidor relacionados com o mesmo (isto é uma predefinição da *firewall* do CentOS).

Mais especificamente, para cada cadeia, as regras foram:



- **INPUT:**

- Podemos observar que estão ativos serviços como o ICMP (que vai permitir fazer *ping* entre o servidor e o cliente), no qual são aceites todos os pedidos que chegam à *firewall*;
- Observa-se também que, todos os pedidos SSH, através do protocolo TCP, são aceites.
- Para todos os restantes serviços, é rejeitado qualquer pedido de acesso, enviando aquando desse pedido, uma mensagem ("reject-with-icmp-host-prohibited").

- **FORWARD:**

- Aqui observámos que a *firewall* está configurada para não permitir que o servidor faça *routing* em qualquer tipo de serviço, ou seja o servidor não consegue reencaminhar os pacotes que recebe e que seriam destinados a outros *hosts*.

- **OUTPUT:**

- Aqui a *firewall* não tem nenhuma regra específica, sendo que o servidor não tem nenhuma restrição em fazer pedidos de qualquer serviço.

Em termos de segurança, não se poderá dizer que tenha um nível elevado ou complexo de segurança, até porque existem serviços que estão ativos e que podem ser acedidos por qualquer cliente que se conecte ao servidor, mesmo que este não pertença a uma rede "confiável" como por exemplo, a rede local do servidor, isto poderá fazer com que qualquer cliente através da rede Internet se possa conectar ao servidor, por SSH (que é o serviço que está ativo), e explorar vulnerabilidades de este serviço. Além disso, a *firewall* não proíbe que qualquer cliente possa fazer *ping* ao servidor, podendo originar ataques DDoS por parte de clientes.

## 2.3. Execução do comando “iptables-save > iptables.dump”

Por razões de segurança, tal como pedido, guardámos as políticas definidas pela *firewall* num ficheiro com o nome "iptables.dump", tal como mostra a figura 7.

```
[root@localhost /]# iptables-save > iptables.dump
[root@localhost /]# ls
bin  dev  home  lib  lost+found  misc  net  proc  sbin  srv  var
boot  etc  iptables.dump  lib64  media  mnt  opt  root  selinux  sys  usr
[root@localhost /]# cat iptables.dump
# Generated by iptables-save v1.4.7 on Thu May  2 16:43:35 2019
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [8523:519839]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
# Completed on Thu May  2 16:43:35 2019
```

Figura 7: Execução do comando para guardar as políticas atuais num ficheiro.

## 2.4. Desativação da *firewall*

Tal como indicado no enunciado, foi-nos pedido que desativássemos a *firewall* e observássemos as alterações. Após o termos feito, executando o comando “*iptables -L -v*” pudemos observar o que está presente na figura 8.

```
[root@localhost /]# system-config-firewall-tui
[root@localhost /]# sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target      prot opt in      out     source         destination
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target      prot opt in      out     source         destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target      prot opt in      out     source         destination
[root@localhost /]#
```

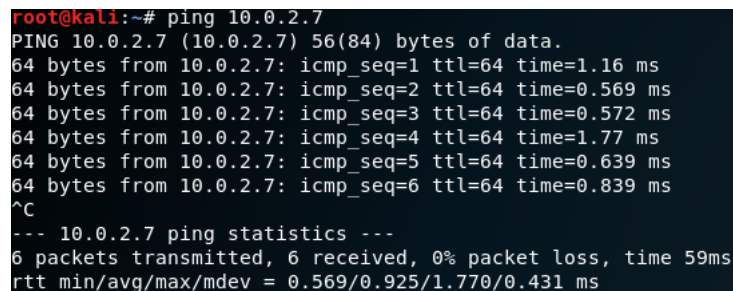
Figura 8: Execução do comando para verificar políticas atuais.

De acordo com estas novas regras, não existe nenhum serviço no qual é rejeitado o seu acesso, sendo estas regras menos seguras, e isto faz com que o servidor se torne menos seguro uma vez que não há um controlo rigoroso de quem pode aceder aos serviços, nem de que serviços específicos podem ser acedidos (simplesmente podem-se aceder a todos os serviços). No final desta tarefa voltámos, tal como indicado no enunciado, a ativar a *firewall*.

## 3. Tarefa 2

### 3.1. Verificação de conectividade

Usando uma máquina virtual Kali Linux como cliente, e estando esta na mesma rede NAT do servidor, procedemos à verificação da conectividade entre cliente e servidor, executando para isso, no cliente, o comando "ping 10.0.2.7", sendo "10.0.2.7" o IP do servidor. Tal como mostra a figura 9, conseguiu-se obter pacotes provenientes do servidor, verificando-se assim com sucesso a comunicação entre cliente e servidor.



```
root@kali:~# ping 10.0.2.7
PING 10.0.2.7 (10.0.2.7) 56(84) bytes of data.
64 bytes from 10.0.2.7: icmp_seq=1 ttl=64 time=1.16 ms
64 bytes from 10.0.2.7: icmp_seq=2 ttl=64 time=0.569 ms
64 bytes from 10.0.2.7: icmp_seq=3 ttl=64 time=0.572 ms
64 bytes from 10.0.2.7: icmp_seq=4 ttl=64 time=1.77 ms
64 bytes from 10.0.2.7: icmp_seq=5 ttl=64 time=0.639 ms
64 bytes from 10.0.2.7: icmp_seq=6 ttl=64 time=0.839 ms
^C
--- 10.0.2.7 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 59ms
rtt min/avg/max/mdev = 0.569/0.925/1.770/0.431 ms
```

Figura 9: Execução do comando para verificar a conectividade entre cliente e servidor.

### 3.2. Execução do comando “*nmap -sS <ip\_address>*”

Tal como pedido, executámos o comando “*nmap -sS*”, este comando permite-nos analisar as portas do servidor, mostrando quais aquelas que se encontram abertas. Neste caso em concreto, tal como se vê na figura 10, existe apenas uma porta aberta (22), sendo esta por defeito, a porta do serviço SSH, o que significa que a *firewall* do CentOS, por defeito, ao ser ativada com as configurações padrão deixa possível de ser acedido remotamente o serviço SSH.

```
root@kali:~# nmap -sS 10.0.2.7
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-05 11:03 EDT
Nmap scan report for 10.0.2.7
Host is up (0.0014s latency).
Not shown: 999 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
MAC Address: 08:00:27:72:F7:03 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 9.63 seconds
```

Figura 10: Execução do comando “ *nmap -sS* ”.

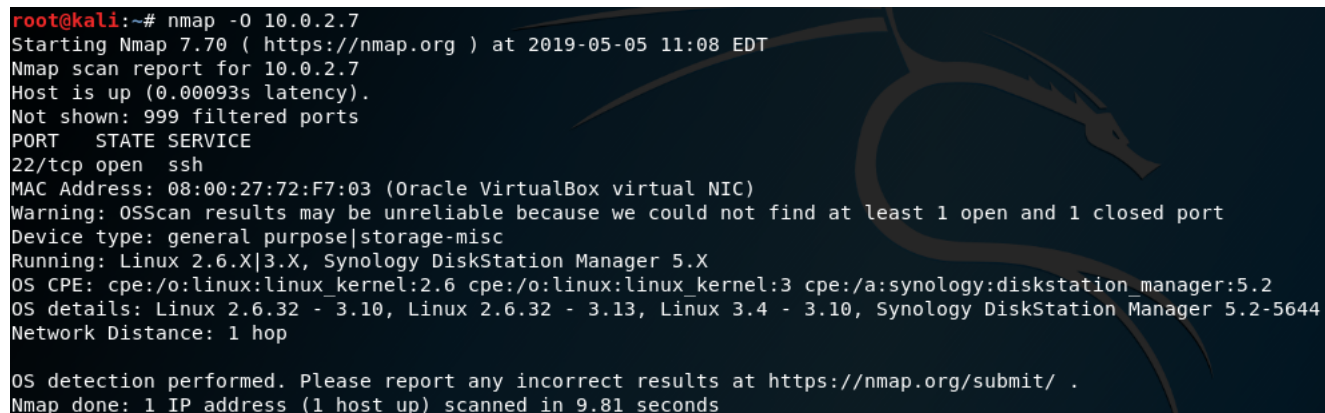
Adicionalmente, a fim de explorar as funcionalidades da ferramenta nmap, executámos o comando “*nmap -sV*”, o qual nos deu informações sobre a versão dos serviços que usam as portas abertas, tal como se pode ver na figura 11.

```
root@kali:~# nmap -sV 10.0.2.7
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-05 11:05 EDT
Nmap scan report for 10.0.2.7
Host is up (0.00065s latency).
Not shown: 999 filtered ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 5.3 (protocol 2.0)
MAC Address: 08:00:27:72:F7:03 (Oracle VirtualBox virtual NIC)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.11 seconds
```

Figura 11: Execução do comando “ *nmap -sV* ”.

Também executámos o comando "nmap -O", de modo a que pudéssemos saber qual o sistema operativo presente no servidor, tal como mostra a figura 12.



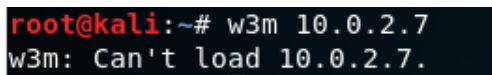
```
root@kali:~# nmap -O 10.0.2.7
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-05 11:08 EDT
Nmap scan report for 10.0.2.7
Host is up (0.00093s latency).
Not shown: 999 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
MAC Address: 08:00:27:72:F7:03 (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose|storage-misc
Running: Linux 2.6.X|3.X, Synology DiskStation Manager 5.X
OS CPE: cpe:/o:linux:linux_kernel:2.6 cpe:/o:linux:linux_kernel:3 cpe:/a:synology:diskstation_manager:5.2
OS details: Linux 2.6.32 - 3.10, Linux 2.6.32 - 3.13, Linux 3.4 - 3.10, Synology DiskStation Manager 5.2-5644
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.81 seconds
```

Figura 12: Execução do comando “*nmap -O*” .

### 3.3. Execução do comando “*w3m http://<ip\_address>*”

Para verificar o estado de funcionamento do serviço HTTP, foi-nos pedido para instalar no cliente o "w3m", um *browser* textual, e de seguida executando o comando “*w3m http://10.0.2.7*”, pudemos observar que o serviço encontra-se bloqueado pela *firewall*, pelo que não conseguimos obter nenhuma página do servidor, tal como mostra a figura 13.



```
root@kali:~# w3m 10.0.2.7
w3m: Can't load 10.0.2.7.
```

Figura 13: Execução do comando “*w3m http://10.0.2.7*” .

### 3.4. Execução do comando “*ftp <ip\_address>*”

Tentámos de seguida aceder ao serviço FTP, mas não obtemos resposta do servidor, verificando assim que a *firewall* bloqueia os pedidos de acesso ao mesmo.

```
root@kali:~# ftp 10.0.2.7
ftp: connect: No route to host
```

Figura 14: Execução do comando “*ftp 10.0.2.7*” .

### 3.5. Execução do comando “*ssh <ip\_address>*”

Por fim tentámos aceder ao serviço SSH do servidor, e conseguimos fazê-lo com sucesso, bastando para isso colocar a *password* do utilizador "root", e conseguimos visualizar os documentos presentes no servidor, tal como mostra a figura 15, e o que confirma que a *firewall* do CentOS por defeito não bloqueia o acesso ao serviço SSH.

```
root@kali:~# ssh 10.0.2.7
The authenticity of host '10.0.2.7 (10.0.2.7)' can't be established.
RSA key fingerprint is SHA256:YrwJEEuykyGL+ltokoYwldUXeMp2Yg7PVNlqXE+nh+E.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.2.7' (RSA) to the list of known hosts.
root@10.0.2.7's password:
Last login: Sun May  5 15:10:59 2019 from localhost.localdomain
[root@localhost ~]# ls
anaconda-ks.cfg  Desktop  Documents  Downloads  iptables.dump  Music  Pictures  post-install  post-install.log  Public  Templates  Videos
[root@localhost ~]# exit
logout
Connection to 10.0.2.7 closed.
```

Figura 15: Execução do comando “*ssh 10.0.2.7*” .

## 4. Tarefa 3

### 4.1. Configuração personalizada da *firewall*

De modo a personalizar a *firewall* do servidor para aceitar tráfego FTP, SSH e HTTP definimos estes serviços como sendo de confiança, encontrando-se ilustrado nas figuras 16, 17, 18.

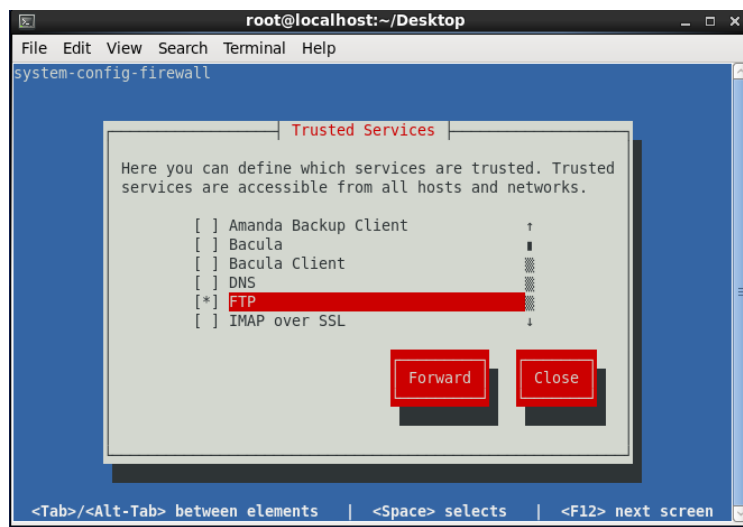


Figura 16: Ativação do serviço FTP na *firewall*.

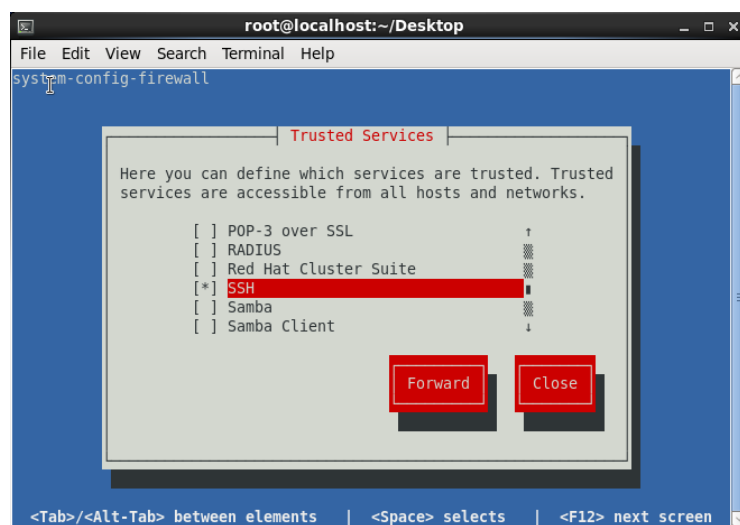


Figura 17: Ativação do serviço SSH na *firewall*.



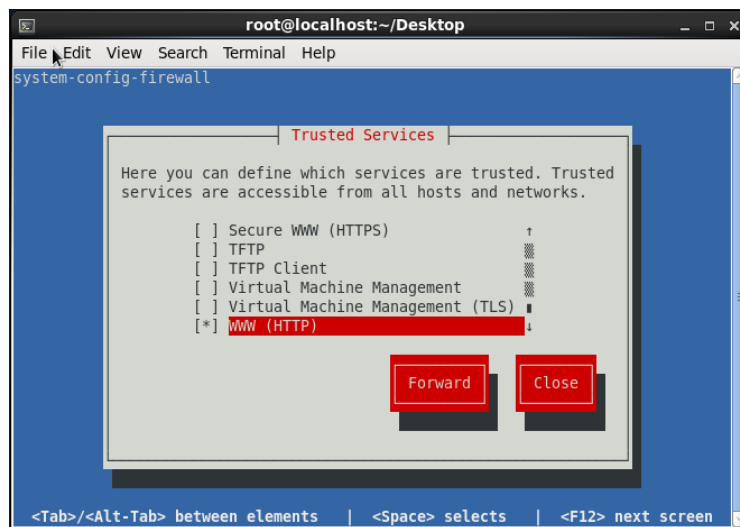


Figura 18: Ativação do serviço HTTP na *firewall*.

Na figura 19 existem várias opções de filtragens do protocolo ICMP, sendo neste caso rejeitados apenas pacotes do tipo "Echo Request" impedindo que os clientes efetuem "*ping*" para o servidor.

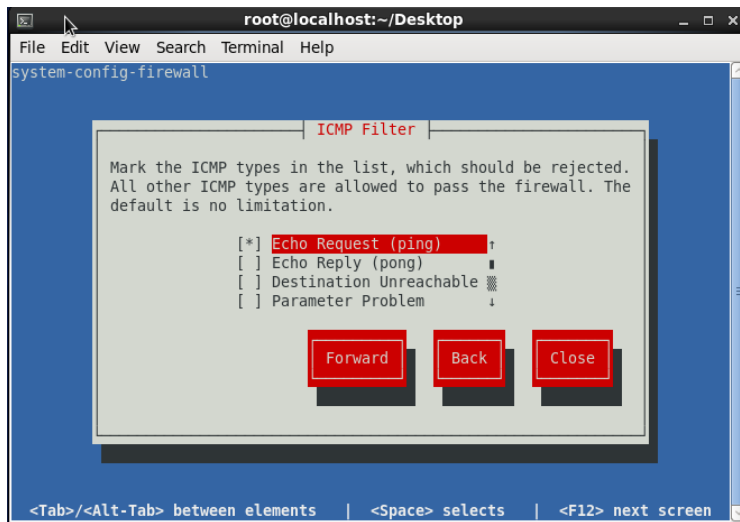


Figura 19: Rejeição de pedidos "Echo Request" pela *firewall*.

## 4.2. Execução do comando “*iptables -L -v*”

Uma vez configurada a *firewall* é possível observar através do “*iptables*” as novas regras resultantes desta configuração. Como é ilustrado na figura 20, o servidor aceita tráfego dos 3 protocolos (HTTP, SSH e FTP) mas rejeita pacotes ICMP “Echo Request” enviando uma mensagem de erro.

```
[root@localhost ~]# iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target      prot opt in      out     source       destination
 65 5596 ACCEPT     all  --  any    any    anywhere     anywhere
state RELATED,ESTABLISHED
 7 588 REJECT     icmp --  any    any    anywhere     anywhere
icmp echo-request reject-with icmp-host-prohibited
 0 0 ACCEPT     icmp --  any    any    anywhere     anywhere
 2 256 ACCEPT     all  --  lo     any    anywhere     anywhere
 0 0 ACCEPT     tcp  --  any    any    anywhere     anywhere
state NEW tcp dpt:ssh
 0 0 ACCEPT     tcp  --  any    any    anywhere     anywhere
state NEW tcp dpt:http
 0 0 ACCEPT     tcp  --  any    any    anywhere     anywhere
state NEW tcp dpt:ftp
 0 0 REJECT     all  --  any    any    anywhere     anywhere
reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target      prot opt in      out     source       destination
 0 0 REJECT     all  --  any    any    anywhere     anywhere
reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT 74 packets, 6636 bytes)
pkts bytes target      prot opt in      out     source       destination
```

Figura 20: Políticas de segurança personalizadas pela *firewall*.

Para cada cadeia de acesso, as políticas de acesso são:

- **INPUT:**

- É ilustrado que o servidor aceita tráfego SSH, FTP e HTTP permitindo que o servidor aceite pedidos destes serviços;
- O servidor rejeita todo o tráfego correspondente a pacotes ICMP “Echo Request” com uma mensagem do tipo “host-prohibited”;
- Todo o restante tráfego que chega à *firewall* que não é dos tipos referidos anteriormente também será rejeitado com uma mensagem do tipo “host-prohibited”;

- **FORWARD:**

- Aqui observámos que a *firewall* está configurada para não permitir que o servidor faça *routing* em qualquer tipo de serviço, ou seja o servidor não consegue reencaminhar os pacotes que recebe e que seriam destinados a outros *hosts*;

## • OUTPUT:

- A *firewall* não possui nenhuma regra específica, sendo que o servidor não tem nenhuma restrição em fazer pedidos de qualquer serviço.

### 4.3. Execução do comando *ping*<*ip\_address*>

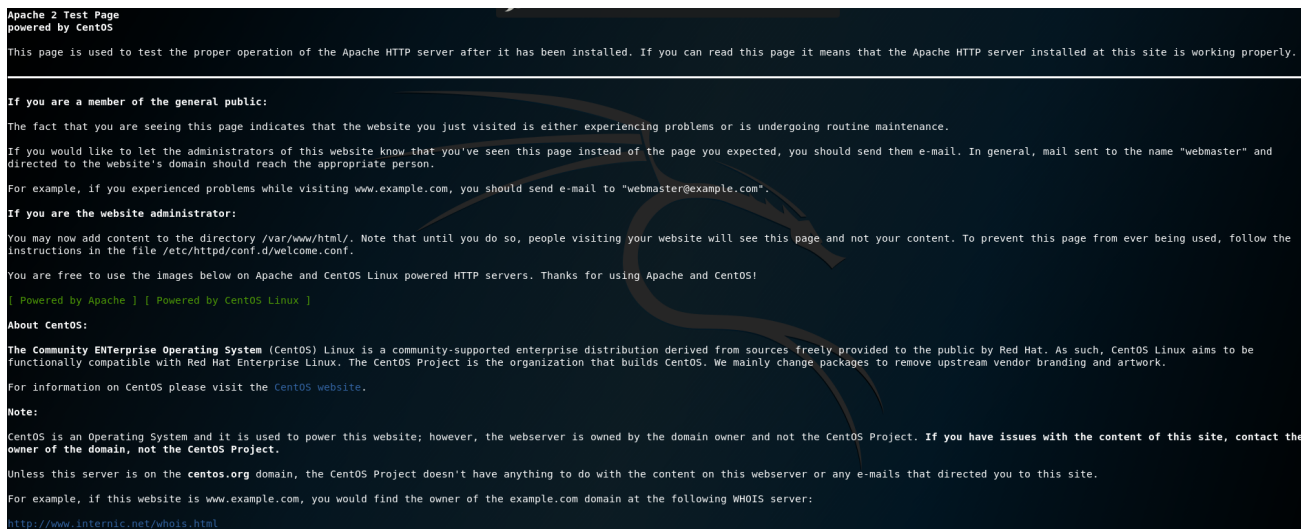
Como consequência da personalização da *firewall* definido na subsecção 4.1 é apresentado na figura 21 a impossibilidade do cliente efetuar um ping para o endereço do servidor obtendo como resposta "Destination Host Prohibited".

```
root@kali:~# ping 10.0.2.7
PING 10.0.2.7 (10.0.2.7) 56(84) bytes of data.
From 10.0.2.7 icmp_seq=1 Destination Host Prohibited
From 10.0.2.7 icmp_seq=2 Destination Host Prohibited
From 10.0.2.7 icmp_seq=3 Destination Host Prohibited
From 10.0.2.7 icmp_seq=4 Destination Host Prohibited
From 10.0.2.7 icmp_seq=5 Destination Host Prohibited
^C
--- 10.0.2.7 ping statistics ---
5 packets transmitted, 0 received, +5 errors, 100% packet loss, time 93ms
```

Figura 21: Execução do comando “*ping 10.0.2.7*”

### 4.4. Execução do comando “*w3m http://<ip\_address>*”

Uma vez que a *firewall* foi configurada para aceitar pedidos HTTP, o cliente já tem acesso à página do servidor.



```
Apache 2 Test Page
powered by CentOS

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page it means that the Apache HTTP server installed at this site is working properly.

If you are a member of the general public:

The fact that you are seeing this page indicates that the website you just visited is either experiencing problems or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

If you are the website administrator:

You may now add content to the directory /var/www/html/. Note that until you do so, people visiting your website will see this page and not your content. To prevent this page from ever being used, follow the instructions in the file /etc/httpd/conf.d/welcome.conf.

You are free to use the images below on Apache and CentOS Linux powered HTTP servers. Thanks for using Apache and CentOS!

[ Powered by Apache ] [ Powered by CentOS Linux ]

About CentOS:

The Community Enterprise Operating System (CentOS) Linux is a community-supported enterprise distribution derived from sources freely provided to the public by Red Hat. As such, CentOS Linux aims to be functionally compatible with Red Hat Enterprise Linux. The CentOS Project is the organization that builds CentOS. We mainly change packages to remove upstream vendor branding and artwork.

For information on CentOS please visit the CentOS website.

Note:

CentOS is an Operating System and it is used to power this website; however, the webserver is owned by the domain owner and not the CentOS Project. If you have issues with the content of this site, contact the owner of the domain, not the CentOS Project.

Unless this server is on the centos.org domain, the CentOS Project doesn't have anything to do with the content on this webserver or any e-mails that directed you to this site.

For example, if this website is www.example.com, you would find the owner of the example.com domain at the following WHOIS server:

http://www.internic.net/whois.html
```

Figura 22: Execução do comando “*w3m http://10.0.2.7*”

## 4.5. Execução do comando “*ftp <ip\_address>*”

Como a *firewall* também foi configurada para aceitar pedidos do serviço FTP, é comprovado através da figura 23 que o servidor está a conceder os mesmos.

```
root@kali:~# ftp 10.0.2.7
Connected to 10.0.2.7.
220 (vsFTPd 2.2.2)
Name (10.0.2.7:root): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> exit
221 Goodbye.
```

Figura 23: Execução do comando “*ftp 10.0.2.7*”

## 4.6. Execução do comando “*nmap -sS <ip\_address>*”.

Uma vez que a *firewall* tem novos serviços ativos, efetuando o “*nmap -sS 10.0.2.7*” é possível visualizar as portas que estão abertas assim como a que serviço correspondem.

```
root@kali:~# nmap -sS 10.0.2.7
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-09 10:58 EDT
Nmap scan report for 10.0.2.7
Host is up (0.0012s latency).
Not shown: 997 filtered ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 08:00:27:72:F7:03 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 7.57 seconds
```

Figura 24: Execução do comando “*nmap -sS 10.0.2.7*”.

## 4.7. Execução do comando “*iptables -L -v*”

É possível observar na figura 25 que os valores dos pacotes trocados em cada cadeia foi alterado como consequência dos pedidos efetuados anteriormente ao servidor.

```
[root@localhost ~]# iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source    destination
 104 10331 ACCEPT    all  --  any    any    anywhere  anywhere
    7   588 REJECT    icmp --  any    any    anywhere  anywhere
state RELATED,ESTABLISHED
reject-with icmp-echo
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source    destination
    0     0 ACCEPT    icmp --  any    any    anywhere  anywhere
    2   256 ACCEPT    all  --  lo     any    anywhere  anywhere
    1    44 ACCEPT    tcp  --  any    any    anywhere  anywhere
    2   104 ACCEPT    tcp  --  any    any    anywhere  anywhere
    2   104 ACCEPT    tcp  --  any    any    anywhere  anywhere
state NEW tcp dpt:ssh
state NEW tcp dpt:http
state NEW tcp dpt:ftp
reject-with icmp-host-prohibited
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source    destination
    0     0 REJECT    all  --  any    any    anywhere  anywhere
reject-with icmp-host-prohibited
Chain OUTPUT (policy ACCEPT 127 packets, 17477 bytes)
 pkts bytes target    prot opt in     out     source    destination
```

Figura 25: Execução do comando “*iptables -L -v*”.

Analisando as cadeias que sofreram alterações:

- **INPUT:**

- Como era expectável, nos serviços que foram ativados e testados o valor dos pacotes aumentou comparativamente com a figura 20 devendo-se ao facto do servidor ter recebido pedidos por parte do cliente para aceder a esses serviços;
- Relativamente ao aumento considerável do número de pacotes enviados ao servidor que não correspondem aos serviços definidos e consequentemente que o servidor tem de rejeitar, isto deve-se ao comando "nmap" que efetua pedidos de diversos serviços de modo a diagnosticar quais os que estão ativados pela *firewall*;

- **OUTPUT:**

- O número de pacotes que saíram do servidor aumentou uma vez que estes representam respostas do mesmo a pedidos do cliente.

## 4.8. Configuração recorrendo a interfaces gráficas

Para aumentar o nosso conhecimento sobre o funcionamento e configuração de *firewalls*, tal como sugerido no enunciado do problema, recorreremos a uma interface gráfica que nos permite aplicar múltiplas funcionalidades presentes também no comando *iptables*, mas de uma forma mais intuitiva.

O programa que usámos para a configuração da *firewall* no servidor, foi o *fwbuilder*.

Após a leitura de alguma documentação sobre o funcionamento do *fwbuilder*, ficámos a perceber melhor a sua forma de atuar. De seguida iremos mostrar algumas das configurações por nós definidas.

Primeiro estabelecemos as definições gerais para a nossa *firewall*, à qual foi dado o nome de "SRC-G9", tal como se pode observar na seguinte figura.

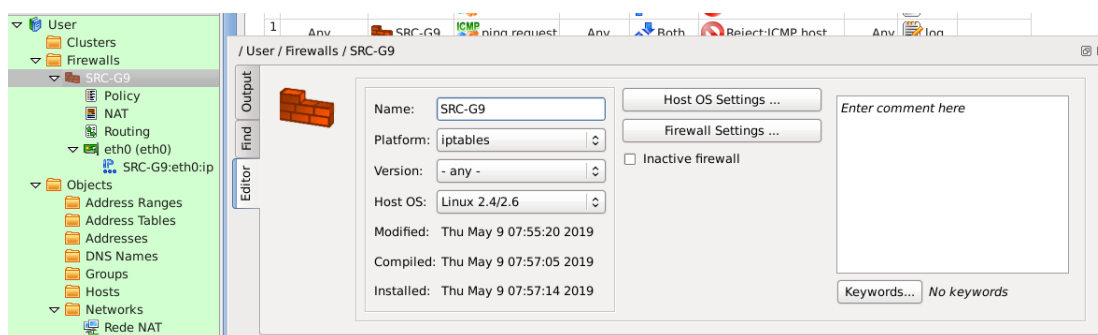


Figura 26: Configuração da *firewall* "SRC-G9".

Também definimos o endereço IP da interface do servidor que se liga à rede local.

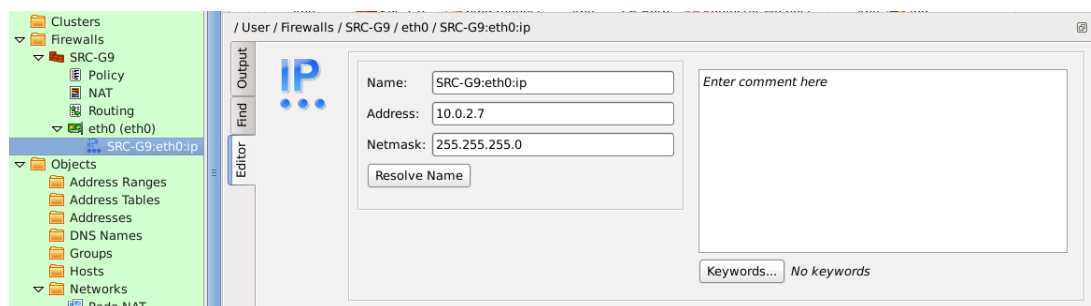


Figura 27: Configuração da interface "eth0".

De seguida definimos a configuração de uma rede interna/local à qual demos o nome de "Rede NAT", como se pode ver na seguinte figura.

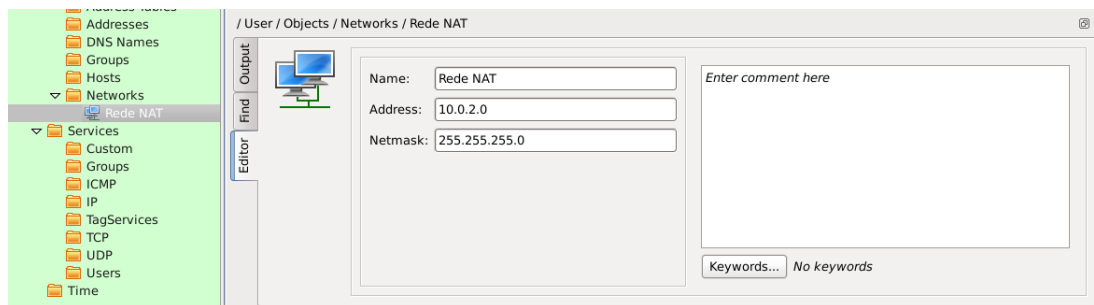


Figura 28: Configuração da “Rede NAT”.

Após isto, demos passo à definição de regras para a nossa *firewall*, tal como mostra a figura 29.

Tal como pretendido no enunciado, limitámos o acesso aos serviços de SSH, FTP e HTTP, para apenas poderem ser acedidos por clientes da "Rede NAT". Além disso também ativámos a opção de *log*, (que também era pedida no enunciado), bem como proibimos que um cliente possa fazer ping ao servidor, respondendo este, uma mensagem "ICMP host-prohibited". Por fim proibimos o acesso, por parte de clientes que não pertençam à "Rede NAT", a qualquer serviço, sendo que nem se sequer lhes é enviada qualquer mensagem de erro, (para aumentar a segurança), tendo por isso sido usada a opção "Deny".

	Source	Destination	Service	Interface	Direction	Action	Time	Options	Comment
0	SRC-G9	Any	ICMP ping reply	Any	Both	Reject:ICMP host ...	Any	log	Rejeita dar resposta a ping.
1	Any	SRC-G9	ICMP ping request	Any	Both	Reject:ICMP host ...	Any	log	Rejeita qualquer pedido de ping.
2	Rede NAT	SRC-G9	ftp	Any	Both	Accept	Any	log	Aceita apenas pedidos provenientes da Rede NAT
3	Rede NAT	SRC-G9	http	Any	Both	Accept	Any	log	Aceita apenas pedidos provenientes da Rede NAT
4	Rede NAT	SRC-G9	ssh	Any	Both	Accept	Any	log	Aceita apenas pedidos provenientes da Rede NAT
5	Any	SRC-G9	Any	Any	Both	Deny	Any	log	Pedidos provenientes de fora da Rede NAT são negados.

Figura 29: Regras definidas para a *firewall* “SRC-G9”.

Posto isto, compilámos e instalámos esta *firewall* no nosso servidor, sendo que de seguida executámos o comando "*iptables -L -v*" para observar as alterações efetuadas, e obtivemos o resultado presente na figura 30.

```
Chain INPUT (policy DROP 0 packets, 0 bytes)
pkts bytes target      prot opt in     out     source            destination        state
86 8571 ACCEPT      all  --  any    any     anywhere          anywhere            state RELATED,ESTABLISHED
0 0 RULE_0       icmp --  any    any     10.0.2.7          anywhere            icmp type 0 code 0
12 1008 RULE_1       icmp --  any    any     anywhere          anywhere            icmp type 8 code 0
2 120 RULE_2       tcp  --  any    any     10.0.2.0/24       anywhere            tcp dpt:ftp state NEW
1 60 RULE_3       tcp  --  any    any     10.0.2.0/24       anywhere            tcp dpt:http state NEW
1 60 RULE_4       tcp  --  any    any     10.0.2.0/24       anywhere            tcp dpt:ssh state NEW
7 4032 RULE_5       all  --  any    any     anywhere          anywhere

Chain FORWARD (policy DROP 0 packets, 0 bytes)
pkts bytes target      prot opt in     out     source            destination        state
0 0 ACCEPT      all  --  any    any     anywhere          anywhere            state RELATED,ESTABLISHED

Chain OUTPUT (policy DROP 935 packets, 126K bytes)
pkts bytes target      prot opt in     out     source            destination        state
92 14969 ACCEPT    all  --  any    any     anywhere          anywhere            state RELATED,ESTABLISHED
0 0 RULE_0       icmp --  any    any     anywhere          anywhere            icmp type 0 code 0
0 0 RULE_1       icmp --  any    any     anywhere          10.0.2.7            icmp type 8 code 0
0 0 RULE_5       all  --  any    any     anywhere          10.0.2.7

Chain RULE_0 (2 references)
pkts bytes target      prot opt in     out     source            destination        LOG level info prefix `RULE 0 -- REJECT '
0 0 LOG          all  --  any    any     anywhere          anywhere            reject-with icmp-host-unreachable
0 0 REJECT       all  --  any    any     anywhere          anywhere

Chain RULE_1 (2 references)
pkts bytes target      prot opt in     out     source            destination        LOG level info prefix `RULE 1 -- REJECT '
12 1008 LOG          all  --  any    any     anywhere          anywhere            reject-with icmp-host-prohibited
12 1008 REJECT    all  --  any    any     anywhere          anywhere

Chain RULE_2 (1 references)
pkts bytes target      prot opt in     out     source            destination        LOG level info prefix `RULE 2 -- ACCEPT '
2 120 LOG          all  --  any    any     anywhere          anywhere
2 120 ACCEPT     all  --  any    any     anywhere          anywhere

Chain RULE_3 (1 references)
pkts bytes target      prot opt in     out     source            destination        LOG level info prefix `RULE 3 -- ACCEPT '
1 60 LOG          all  --  any    any     anywhere          anywhere
1 60 ACCEPT     all  --  any    any     anywhere          anywhere

Chain RULE_4 (1 references)
pkts bytes target      prot opt in     out     source            destination        LOG level info prefix `RULE 4 -- ACCEPT '
1 60 LOG          all  --  any    any     anywhere          anywhere
1 60 ACCEPT     all  --  any    any     anywhere          anywhere

Chain RULE_5 (2 references)
pkts bytes target      prot opt in     out     source            destination        LOG level info prefix `RULE 5 -- DENY '
7 4032 LOG          all  --  any    any     anywhere          anywhere
7 4032 DROP      all  --  any    any     anywhere          anywhere
```

Figura 30: Execução do comando "*iptables -L -v*".

Tal como anteriormente, verificámos também todos os serviços a partir do cliente Kali Linux.

Começando por verificar que o cliente não consegue fazer *ping* para o servidor, tal como mostra a figura 31.



```

root@kali:~# ping 10.0.2.7
PING 10.0.2.7 (10.0.2.7) 56(84) bytes of data.
From 10.0.2.7 icmp_seq=1 Destination Host Prohibited
From 10.0.2.7 icmp_seq=2 Destination Host Prohibited
From 10.0.2.7 icmp_seq=3 Destination Host Prohibited
From 10.0.2.7 icmp_seq=4 Destination Host Prohibited
From 10.0.2.7 icmp_seq=5 Destination Host Prohibited
From 10.0.2.7 icmp_seq=6 Destination Host Prohibited
From 10.0.2.7 icmp_seq=7 Destination Host Prohibited
From 10.0.2.7 icmp_seq=8 Destination Host Prohibited
From 10.0.2.7 icmp_seq=9 Destination Host Prohibited
From 10.0.2.7 icmp_seq=10 Destination Host Prohibited
0/10 packets, 100% loss
From 10.0.2.7 icmp_seq=11 Destination Host Prohibited
^C
--- 10.0.2.7 ping statistics ---
11 packets transmitted, 0 received, +11 errors, 100% packet loss, time 102ms

```

Figura 31: Execução do comando *ping* do cliente para o servidor.

Sendo que agora podemos observar o registo desta tentativa de *ping*, num ficheiro de *log* presente no servidor, e o respetivo envio da mensagem de "Reject" para o cliente, tal como mostra a seguinte figura.

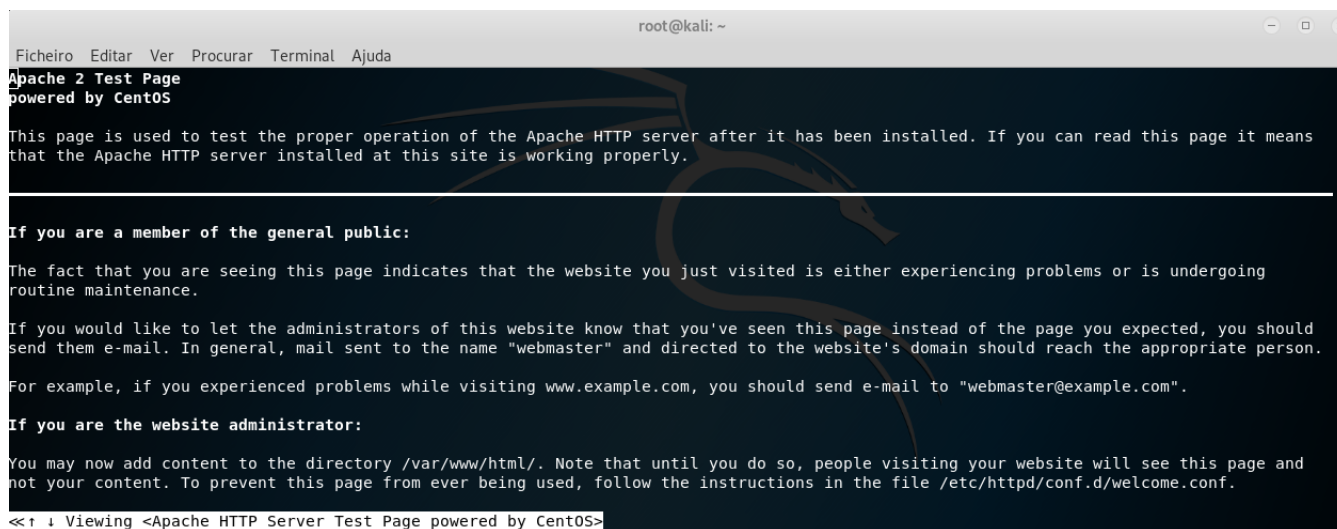
```

May  9 15:07:07 localhost kernel: RULE 1 -- REJECT IN=eth0 OUT= MAC=08:00:27:72:f7:03:08:00:27:f7:ba:ec:08:00 SRC=10.0.2.6 DST=10.0.2.7 LEN=84 TOS=0x00 PREC=
0x00 TTL=64 ID=60810 DF PROTO=ICMP TYPE=8 CODE=0 ID=12445 SEQ=1
May  9 15:07:08 localhost kernel: RULE 1 -- REJECT IN=eth0 OUT= MAC=08:00:27:72:f7:03:08:00:27:f7:ba:ec:08:00 SRC=10.0.2.6 DST=10.0.2.7 LEN=84 TOS=0x00 PREC=
0x00 TTL=64 ID=60866 DF PROTO=ICMP TYPE=8 CODE=0 ID=12445 SEQ=2
May  9 15:07:09 localhost kernel: RULE 1 -- REJECT IN=eth0 OUT= MAC=08:00:27:72:f7:03:08:00:27:f7:ba:ec:08:00 SRC=10.0.2.6 DST=10.0.2.7 LEN=84 TOS=0x00 PREC=
0x00 TTL=64 ID=60986 DF PROTO=ICMP TYPE=8 CODE=0 ID=12445 SEQ=3
May  9 15:07:10 localhost kernel: RULE 1 -- REJECT IN=eth0 OUT= MAC=08:00:27:72:f7:03:08:00:27:f7:ba:ec:08:00 SRC=10.0.2.6 DST=10.0.2.7 LEN=84 TOS=0x00 PREC=
0x00 TTL=64 ID=61229 DF PROTO=ICMP TYPE=8 CODE=0 ID=12445 SEQ=4
May  9 15:07:11 localhost kernel: RULE 1 -- REJECT IN=eth0 OUT= MAC=08:00:27:72:f7:03:08:00:27:f7:ba:ec:08:00 SRC=10.0.2.6 DST=10.0.2.7 LEN=84 TOS=0x00 PREC=
0x00 TTL=64 ID=61319 DF PROTO=ICMP TYPE=8 CODE=0 ID=12445 SEQ=5
May  9 15:07:12 localhost kernel: RULE 1 -- REJECT IN=eth0 OUT= MAC=08:00:27:72:f7:03:08:00:27:f7:ba:ec:08:00 SRC=10.0.2.6 DST=10.0.2.7 LEN=84 TOS=0x00 PREC=
0x00 TTL=64 ID=61533 DF PROTO=ICMP TYPE=8 CODE=0 ID=12445 SEQ=6
May  9 15:07:13 localhost kernel: RULE 1 -- REJECT IN=eth0 OUT= MAC=08:00:27:72:f7:03:08:00:27:f7:ba:ec:08:00 SRC=10.0.2.6 DST=10.0.2.7 LEN=84 TOS=0x00 PREC=
0x00 TTL=64 ID=61725 DF PROTO=ICMP TYPE=8 CODE=0 ID=12445 SEQ=7
May  9 15:07:14 localhost kernel: RULE 1 -- REJECT IN=eth0 OUT= MAC=08:00:27:72:f7:03:08:00:27:f7:ba:ec:08:00 SRC=10.0.2.6 DST=10.0.2.7 LEN=84 TOS=0x00 PREC=
0x00 TTL=64 ID=61767 DF PROTO=ICMP TYPE=8 CODE=0 ID=12445 SEQ=8
May  9 15:07:15 localhost kernel: RULE 1 -- REJECT IN=eth0 OUT= MAC=08:00:27:72:f7:03:08:00:27:f7:ba:ec:08:00 SRC=10.0.2.6 DST=10.0.2.7 LEN=84 TOS=0x00 PREC=
0x00 TTL=64 ID=61933 DF PROTO=ICMP TYPE=8 CODE=0 ID=12445 SEQ=9
May  9 15:07:16 localhost kernel: RULE 1 -- REJECT IN=eth0 OUT= MAC=08:00:27:72:f7:03:08:00:27:f7:ba:ec:08:00 SRC=10.0.2.6 DST=10.0.2.7 LEN=84 TOS=0x00 PREC=
0x00 TTL=64 ID=62034 DF PROTO=ICMP TYPE=8 CODE=0 ID=12445 SEQ=10
May  9 15:07:17 localhost kernel: RULE 1 -- REJECT IN=eth0 OUT= MAC=08:00:27:72:f7:03:08:00:27:f7:ba:ec:08:00 SRC=10.0.2.6 DST=10.0.2.7 LEN=84 TOS=0x00 PREC=
0x00 TTL=64 ID=62054 DF PROTO=ICMP TYPE=8 CODE=0 ID=12445 SEQ=11

```

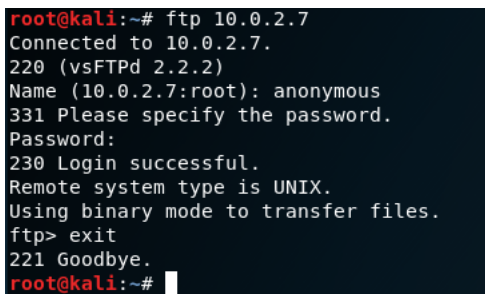
Figura 32: Leitura do ficheiro de *log* do servidor.

Por fim, verificámos se os serviços HTTP, FTP e SSH estavam disponíveis, e tal como mostram as figuras 33, 34 e 35, os mesmos estavam a funcionar corretamente.



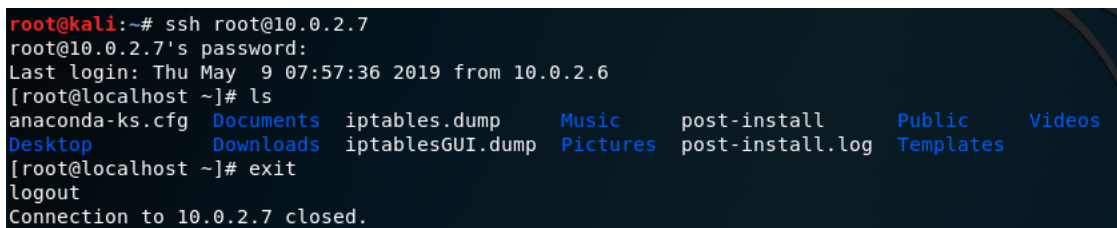
```
root@kali: ~  
Ficheiro Editar Ver Procurar Terminal Ajuda  
Apache 2 Test Page  
powered by CentOS  
This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page it means  
that the Apache HTTP server installed at this site is working properly.  
  
If you are a member of the general public:  
The fact that you are seeing this page indicates that the website you just visited is either experiencing problems or is undergoing  
routine maintenance.  
If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should  
send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.  
For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".  
  
If you are the website administrator:  
You may now add content to the directory /var/www/html/. Note that until you do so, people visiting your website will see this page and  
not your content. To prevent this page from ever being used, follow the instructions in the file /etc/httpd/conf.d/welcome.conf.  
<< ↑ Viewing <Apache HTTP Server Test Page powered by CentOS>
```

Figura 33: Teste de funcionamento do serviço HTTP.



```
root@kali:~# ftp 10.0.2.7  
Connected to 10.0.2.7.  
220 (vsFTPD 2.2.2)  
Name (10.0.2.7:root): anonymous  
331 Please specify the password.  
Password:  
230 Login successful.  
Remote system type is UNIX.  
Using binary mode to transfer files.  
ftp> exit  
221 Goodbye.  
root@kali:~#
```

Figura 34: Teste de funcionamento do serviço FTP.



```
root@kali:~# ssh root@10.0.2.7  
root@10.0.2.7's password:  
Last login: Thu May 9 07:57:36 2019 from 10.0.2.6  
[root@localhost ~]# ls  
anaconda-ks.cfg  Documents  iptables.dump  Music  post-install  Public  Videos  
Desktop  Downloads  iptablesGUI.dump  Pictures  post-install.log  Templates  
[root@localhost ~]# exit  
logout  
Connection to 10.0.2.7 closed.
```

Figura 35: Teste de funcionamento do serviço SSH.