Segurança Informática

Relatório Terceiro Projecto, Grupo 001

Francisco Pires, nº 44314
Pedro Neves, nº 45787
Tiago Maurício, nº 45105

22 de Maio de 2016

Capítulo 1

iptables

1.1 Regras aplicadas

As seguintes regras foram aplicadas por esta ordem antes de serem feitos os testes:

1. ligações já estabelecida devem ser aceites

   ```
   $ sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED  -j ACCEPT
   ```

   ```
   $ sudo iptables -A OUTPUT -m state --state ESTABLISHED,RELATED  -j ACCEPT
   ```

2. A máquina responde a pings apenas com origem nas máquinas da sua sub-rede local (com máscara 255.255.254.0)

   ```
   $ sudo iptables -A INPUT -s 10.101.148.182/23 -p icmp -j ACCEPT
   ```

3. Aceita ligações de clientes com qualquer origem para o servidor *myWhats*

   ```
   $ sudo iptables -A INPUT -p tcp --dport 23457 -j ACCEPT
   ```

4. Aceita ligações ssh apenas da máquina gcc

   ```
   $ sudo iptables -A INPUT -s gcc.alunos.di.fc.ul.pt -p tcp  --dport ssh
   -j ACCEPT
   ```

5. A máquina apenas pode fazer ping à máquina gcc.

   ```
   $ sudo iptables -A OUTPUT -d gcc.alunos.di.fc.ul.pt -p icmp -j ACCEPT
   ```

6. Não impedir o acesso às maquinas descritas no enunciado, para o bom funcionamento das máquinas do laboratório.

   ```
   $ sudo iptables -A INPUT -d "10.101.253.11,  10.101.253.12, 10.101.253.13,
   10.121.53.14, 10.121.53.15,  10.101.53.16, 10.101.249.63,  10.101.85.6,
   10.101.85.138,  10.101.85.18, 10.101.148.1,  10.101.85.134" -j ACCEPT
   ```

   ```
   $ sudo iptables -A OUTPUT -s "10.101.253.11,  10.101.253.12, 10.101.253.13,
   10.121.53.14, 10.121.53.15,  10.101.53.16, 10.101.249.63,  10.101.85.6,
   10.101.85.138,  10.101.85.18, 10.101.148.1,  10.101.85.134" -j ACCEPT
   ```

7. Não filtrar o tráfego do dispositivo de loopback

   ```
   $ sudo iptables -A INPUT -i lo -j ACCEPT
   ```

   ```
   $ sudo iptables -A OUTPUT -o lo -j ACCEPT
   ```

8. Não permitir mais nenhuma ligação na table *input* e *output* para alem das acima definidas

   ```
   $ sudo iptables -A INPUT -j DROP
   ```

   ```
   $ sudo iptables -A OUTPUT -j DROP
   ```

1.2 Resultado das regras aplicadas (iptables -L)

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            \
      state RELATED,ESTABLISHED
ACCEPT     icmp --  10.101.148.0/23      anywhere
ACCEPT     tcp  --  anywhere             anywhere             \
     tcp dpt:23457
ACCEPT     tcp  --  gcc.alunos.di.fc.ul.pt  anywhere          \
       tcp dpt:ssh
ACCEPT     all  --  anywhere             10.101.253.11
ACCEPT     all  --  anywhere             10.101.253.12
ACCEPT     all  --  anywhere             10.101.253.13
ACCEPT     all  --  anywhere             10.121.53.14
ACCEPT     all  --  anywhere             10.121.53.15
ACCEPT     all  --  anywhere             10.101.53.16
ACCEPT     all  --  anywhere             10.101.249.63
ACCEPT     all  --  anywhere             iate.alunos.di.fc.ul.pt
ACCEPT     all  --  anywhere             falua.di.fc.ul.pt
ACCEPT     all  --  anywhere             nemo.alunos.di.fc.ul.pt
ACCEPT     all  --  anywhere             submarino.alunos.di.fc.ul.pt
ACCEPT     all  --  anywhere             farol.alunos.di.fc.ul.pt
ACCEPT     all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere
```

```
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            \
    state RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere             gcc.alunos.di.fc.ul.pt
ACCEPT     all  --  anywhere             10.101.253.11
ACCEPT     all  --  anywhere             10.101.253.12
ACCEPT     all  --  anywhere             10.101.253.13
ACCEPT     all  --  anywhere             10.121.53.14
ACCEPT     all  --  anywhere             10.121.53.15
ACCEPT     all  --  anywhere             10.101.53.16
ACCEPT     all  --  anywhere             10.101.249.63
ACCEPT     all  --  anywhere             iate.di.fc.ul.pt
ACCEPT     all  --  anywhere             falua.di.fc.ul.pt
ACCEPT     all  --  anywhere             nemo.alunos.di.fc.ul.pt
ACCEPT     all  --  anywhere             submarino.alunos.di.fc.ul.pt
ACCEPT     all  --  anywhere             farol.alunos.di.fc.ul.pt
ACCEPT     all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere
```

1.3 Testes de ligações estabelecidas

1.3.1 Teste da Regra nº 2

Realizamos um ping de outra maquina do laboratório para testar esta regra:

Comando: `$ ping -c 3 10.101.148.4`
Resultado:

```
PING 10.101.148.4 (10.101.148.4) 56(84) bytes of data.
64 bytes from 10.101.148.4 (10.101.85.18): icmp_seq=1 ttl=64 time=0.242
64 bytes from 10.101.148.4 (10.101.85.18): icmp_seq=2 ttl=64 time=0.269
64 bytes from 10.101.148.4 (10.101.85.18): icmp_seq=3 ttl=64 time=0.265

--- 10.101.148.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.242/0.258/0.269/0.022 ms
```

1.3.2 Teste da Regra nº 3

Por uma questão de simplicidade, utilizados o modulo `SimpleHTTPServer` do python para criar um servidor HTTP na mesma porta onde era suposto correr o servidor *myWhats*, e o telnet para fazer um pedido a simular o cliente. Referimos que as respostas por parte dos módulos só servem para verificar que a porta esta aberta e é possível aceder.

Comando no servidor: `python -m SimpleHTTPServer 23457`
Resultado:

```
Serving HTTP on 0.0.0.0 port 23457 ...
10.101.148.182 - - [14/May/2016 17:44:47] code 400, message Bad request syntax
('GET')
10.101.148.182 - - [14/May/2016 17:44:47] 'GET' 400 -
```

Comando no cliente: `telnet 10.101.148.183 23457`
Resultado:

```
Trying 10.101.148.183...
Connected to 10.101.148.183.
...
```

1.3.3 Teste da Regra nº 4

Não é possível testar esta regra, visto que os alunos não tem autorização para usar o ssh na maquina gcc. Por isso, usamos os nossos computados pessoais numa rede local para testar a regra.

ip maquina local: 192.168.1.66
ip maquina remota (gcc): 192.168.1.82

Alteração da regra para:

```
$ sudo iptables -A INPUT -s 192.168.1.82 -p tcp  --dport ssh -j ACCEPT
$ sudo iptables -A INPUT -j DROP
```

Comando para teste: `ssh 192.168.1.66`

```
sherby@sherby-desktop:~$ ssh 192.168.1.66
sherby@192.168.1.66's password:
Welcome to Ubuntu 16.04 LTS (GNU/Linux 4.4.0-22-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

0 packages can be updated.
0 updates are security updates.

Last login: Sat May 14 22:18:26 2016 from 192.168.1.82
-> ~
```

1.3.4 Teste da Regra nº 5

Realizamos um ping da maquina gcc para testar esta regra:

Comando: `$ ping -c 3 gcc.alunos.di.fc.ul.pt`
Resultado:

```
PING gcc.alunos.di.fc.ul.pt (10.101.151.5) 56(84) bytes of data.
64 bytes from gcc.alunos.di.fc.ul.pt (10.101.151.5): icmp_seq=1 \
        ttl=64 time=0.341 ms
64 bytes from gcc.alunos.di.fc.ul.pt (10.101.151.5): icmp_seq=2 \
        ttl=64 time=0.403 ms
64 bytes from gcc.alunos.di.fc.ul.pt (10.101.151.5): icmp_seq=3 \
        ttl=64 time=0.333 ms

--- gcc.alunos.di.fc.ul.pt ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.333/0.359/0.403/0.031 ms
```

1.4 Testes de ligações não estabelecidas

Estes testes visam mostrar que as regras do iptables não estabelecem ligações não definidas.

1. Outras portas para verificar se os pacotes são filtrados.

   Comando no servidor: `python -m SimpleHTTPServer 3467`

   Resultado:

   ```
   Serving HTTP on 0.0.0.0 port 3467 ...
   ```

   Comando no cliente: `telnet 10.101.148.166 3467`

   Resultado:

   ```
   Trying 10.101.148.166...
   ```

2. Outros pings não são permitidos

   Comando: `ping -c 3 10.101.148.161`

   ```
   PING 10.101.148.161 (10.101.148.161) 56(84) bytes of data.
   ping: sendmsg: Operation not permitted
   ping: sendmsg: Operation not permitted
   ping: sendmsg: Operation not permitted

   --- 10.101.148.161 ping statistics ---
   3 packets transmitted, 0 received, 100% packet loss, time 2016ms
   ```

Capítulo 2

snort

2.1 Regras aplicadas

As seguintes regras foram escritas no ficheiro *snort.config* por esta ordem, antes de serem feitos os testes:

1. Alerta para pacotes tcp cujo ip de destino seja o da maquina de teste, e que o porto de destino esteja entre 1 e 2048.

```
alert tcp any any -> 10.101.148.166 1:2048 \
(msg:"varrimento de portos"; sid:420; rev:0;)
```

2. Alerta para pacotes tcp cujo ip de destino seja o da maquina de teste, e que o porto de destino seja o 23457

```
alert tcp any any -> 10.101.148.166 23457 \
(msg: "tentativa de descobrir a password"; sid:666;rev:0;)
```

3. Filtro aplicado à primeira regra - Notifica **apenas** 1 vez a cada 2 minutos

```
event_filter gen_id 1, sig_id 420, \
type both, track by_dst, count 3, seconds 120
```

4. Filtro aplicado à segunda regra - Notifica 3 vezes a cada 30 segundos

```
event_filter gen_id 1, sig_id 666, \
type threshold, track by_src, count 3, seconds 30
```

2.2 Testes

2.2.1 Na maquina a correr *snort*:

1. Teste da primeira regra e filtro:

   ```
   Commencing packet processing (pid=3575) 05/22-16:26:08.832675 [**] [1:420:0]
   varrimento de portos [**] [Priority:  0]  TCP 10.101.148.161:45246 ->
   10.101.148.166:3
   ```

2. Teste da segunda regra e filtro

   ```
   Commencing packet processing (pid=3564) 05/22-16:24:35.084257 [**] [1:666:0]
   tentativa de descobrir a password [**] [Priority:  0] TCP 10.101.148.161:35699
   -> 10.101.148.166:23457
   ```

2.2.2 Na maquina remota:

1. Teste da primeira regra e filtro:

   Comando: `netcat -zv 10.101.148.166 23457`

   ```
   Connection to 10.101.148.166 23457 port [tcp/*] succeeded!
   ```

2. Teste da segunda regra e filtro

   Comando: `netcat -zv 10.101.148.166 1-2048`

   ```
   Connection to 10.101.148.166 port 1 (tcp) failed: \
   Connection refused
   ...
   Connection to 10.101.148.166 port 2048 (tcp) failed: \
   Connection refused
   ```