Relatório Iptables

Segurança e Confiabilidade, 2018/2019

Grupo 51:

João Marques nº49038

João David nº49448

Luís Moreira nº49531

Índice

Iptables

Neste relatório será explicado o ficheiro iptable-config.sh, que corresponde a um script que contém todas as regras iptables.

Para correr o ficheiro, visto que é uma script, basta correr a seguinte linha:

```
./iptable-config.sh
```

Nota: O script pode não ser possível executar direto pela consola, se isso acontecer, é só ir ao directório do script, carregar com a tecla direita do rato nele, ir à tab "Permissions" e no fim existe a opção "Allow this file to run as a program", ativa-se essa opção e a script já deve aparecer como executável.

O ficheiro é o seguinte:

```
#!/bin/bash

#Listar todas as regras com mais informação: sudo iptables --list -v

LOCAL_IP="10.101.148.209"

#clean Chains
sudo /sbin/iptables -F

#loopback nao filtrado
sudo /sbin/iptables -A INPUT -i lo -j ACCEPT
sudo /sbin/iptables -A OUTPUT -o lo -j ACCEPT

#ligacao estabelecida continua sendo aceite
sudo /sbin/iptables -A INPUT -m state --state ESTABLISHED,RELATED -j
ACCEPT
sudo /sbin/iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j
ACCEPT

#DCs
sudo /sbin/iptables -A INPUT -s 10.121.52.14 -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.121.52.14 -s $LOCAL_IP -j ACCEPT

sudo /sbin/iptables -A INPUT -s 10.121.52.15 -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.121.52.15 -s $LOCAL_IP -j ACCEPT

sudo /sbin/iptables -A INPUT -s 10.121.52.16 -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.121.52.16 -s $LOCAL_IP -j ACCEPT

#Storage
```

```
sudo /sbin/iptables -A INPUT -s 10.121.72.23 -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.121.72.23 -s $LOCAL_IP -j ACCEPT

#Iate/Falua
sudo /sbin/iptables -A INPUT -s 10.101.85.6 -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.101.85.6 -s $LOCAL_IP -j ACCEPT

sudo /sbin/iptables -A INPUT -s 10.101.85.138 -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.101.85.138 -s $LOCAL_IP -j ACCEPT

#Luna
sudo /sbin/iptables -A INPUT -s 10.101.85.24 -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.101.85.24 -s $LOCAL_IP -j ACCEPT

#Gateway
sudo /sbin/iptables -A INPUT -s 10.101.148.1 -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.101.148.1 -s $LOCAL_IP -j ACCEPT

#Proxy
sudo /sbin/iptables -A INPUT -s 10.101.85.134 -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.101.85.134 -s $LOCAL_IP -j ACCEPT

#Ping - gcc
sudo /sbin/iptables -A INPUT -s 10.101.151.5 -p icmp -d $LOCAL_IP -j
ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.101.151.5 -p icmp -s $LOCAL_IP -j
ACCEPT

#Mask 255.255.254.0

#Ping - para as maquinas locais max.2 por segundo
sudo /sbin/iptables -A INPUT -p icmp -m icmp --icmp-type 8 -s
10.101.148.0/23 -j ACCEPT
sudo /sbin/iptables -A OUTPUT -p icmp -m icmp --icmp-type 8 -d
10.101.148.0/23 -m limit --limit 2/s -j ACCEPT

#SSH - maquinas
sudo /sbin/iptables -A INPUT -s 10.101.148.0/23 -p tcp -d $LOCAL_IP --
dport 22 -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.101.148.0/23 -p tcp -s $LOCAL_IP --
dport 22 -j ACCEPT
```

A script em si, começa com a seguinte linha, `LOCAL_IP="10.101.148.60"`, esta linha serve para colocar o IP da máquina local para que as regras de iptables seguintes sejam aplicadas corretamente na máquina corrente.

Comandos iptables

Para aceder e alterar o ficheiro das iptables ao qual o SO recorre, temos de usar o seguinte comando:`sudo /sbin/iptables`. O que isto faz é aceder ao ficheiro iptables na diretoria sbin.

O sudo é um comando que permite recorrer à conta administrador, root, da máquina corrente e é necessárias permissões para correr estes comandos de iptables.

Fazemos um flush, **-F** nos comandos de iptables, logo ao começar a script. Isto para apagar todas as regras iniciais da máquina e poder colocar as que definimos. Conseguimos isto com a seguinte linha: `sudo /sbin/iptables -F`

Dispositivo de Loopback

As seguintes linhas,

```
sudo /sbin/iptables -A INPUT -i lo -j ACCEPT

sudo /sbin/iptables -A OUTPUT -o lo -j ACCEPT
```

servem para não filtrar a informação que usa a interface de rede **lo**, esta interface é a responsável por dar o ip de localhost à máquina, e por aqui que a máquina comunica com ela mesma. Esta interface serve para inúmeras coisas, como testes e desenvolvimentos de aplicações, entre outros, e é usada por várias aplicações, não devendo ser bloqueada.

Nestes comandos são usados o -A que é um target (command), que faz append da regra seguinte ao ficheiro iptables. A regra é constituída por **INPUT**, numa das linhas, e **OUTPUT** na outra, isto deve-se ao facto de que a primeira linha verifica se o pacote recebido, **INPUT**, é um pacote da interface lo, e isto é verificado com o match **-i** que é uma abreviatura para interface, ou seja ele verifica se o pacote de interface vêm da interface **lo,** com a seguinte parte do comando **INPUT -i lo**. Se o pacote pertencer à interface é feito um jump, **-j**, para o **ACCEPT**, e o pacote passa pelas regras e é recebido pelo SO. Por outro lado, o **OUTPUT** verifica se um pacote está para ser enviado para a rede e o **-o** verifica se esse pacote é um pacote de interface que vai sair para o ip/interface designado a seguir, por exemplo a parte do comando **OUTPUT -o lo** verifica se um pacote de interface está para sair para a interface **lo** e se sim há um jump, **-j**, novamente para o **ACCEPT** e permite que o pacote passe o SO.

Ligações Estabelecidas

As seguintes linhas,

```
sudo /sbin/iptables -A INPUT -m state --state ESTABLISHED,RELATED
-j ACCEPT
sudo /sbin/iptables -A OUTPUT -m state --state ESTABLISHED,RELATED
-j ACCEPT
```

da mesma forma que o comando explicado anterior o **-A** serve para fazer append das regras ao ficheiro iptables, o **INPUT** designa os pacotes que estão para entrar e o **OUTPUT** designa os pacotes que estão para sair. Neste caso é verificado com a match, **-m**, o estado da ligação, **--state**, e os comandos verificam se o estado da ligação está já estabelecida, **ESTABLISHED** e **RELATED**, se sim fazem jump para **ACCEPT**, tanto no **OUTPUT** como no **INPUT** porque a comunicação é bidirecional.

Acessos

Todas os acessos são dadas com os comandos:

```
sudo /sbin/iptables -A INPUT -s “IP” -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d “IP” -s $LOCAL_IP -j ACCEPT
```

Em que “IP” é o ip da máquina com que queremos comunicar, o **-A**, o **INPUT** e o **OUTPUT** já foram explicados tal como o **-j** e o **ACCEPT**. O que resta o **-d**, o **-s** e o **$LOCAL_IP**.

O **$LOCAL_IP** não é nada mais do que a variável explicada no ínicio deste relatório, que é o que representa o ip da máquina local, mas não é o localhost. O **-d** indica o destinatário do pacote a receber ou a ser enviado. Por exemplo, **OUTPUT -d “IP”** indica que o pacote a analisar é para ser enviado e o destinatário desse pacote é a máquina com o endereço ip **“IP”**. O **-s** não é nada mais nada menos que o source, ou seja, a máquina que enviou o pacote. Desta forma o exemplo completo, **INPUT -s “IP” -d $LOCAL_IP**, é explicado com: o pacote que está a ser analisado foi recebido da máquina com ip **“IP”** e o seu destinatário é a máquina local com ip **$LOCAL_IP**. E se o ip fonte do pacote corresponder ao ip **“IP”** e o ip destino corresponder ao **$LOCAL_IP**, o pacote é recebido e passa para o SO.

Existem duas vertentes destes comandos usados, uma com ip’s fixos e outra com máscara de rede.

IP fixo:

```
sudo /sbin/iptables -A INPUT -s 10.101.85.134 -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.101.85.134 -s $LOCAL_IP -j ACCEPT
```

Máscara de rede:

```
sudo /sbin/iptables -A INPUT -s 10.101.148.0/23 -p tcp -d $LOCAL_IP --
dport 22 -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.101.148.0/23 -p tcp -s $LOCAL_IP --
dport 22 -j ACCEPT
```

Neste exemplo de máscara de rede, é usado a máscara de 23 bits fixos e 9 variáveis, e isto gera uma sub-rede e a máquina só pode comunicar com outros ip's de máquinas que estejam dentro dessa sub-rede dada pela máscara a partir de uma operação lógica AND. Temos também a verificação do protocolo, **-p**, nesta ligação que neste caso verificamos se é **tcp** e só apenas ligações por tcp podem ser executadas. Por fim, verificamos se a porta a ser usada é a porta 22 com o **–dport**, corresponde à maneira de verificar se a ligação tcp está encaminhada para a porta 22 da máquina corrente. Se sim, sabemos que a ligação é uma ligação ssh, pois é o único protocolo que usa esta porta e está reservada para ele.

Pings

Começando pelos primeiros comandos de ping, permissão para a máquina gcc executar pings para a nossa máquina servidor. Sendo estes commandos os seguintes:

```
sudo /sbin/iptables -A INPUT -s 10.101.151.5 -p icmp -d $LOCAL_IP -j ACCEPT
sudo /sbin/iptables -A OUTPUT -d 10.101.151.5 -p icmp -s $LOCAL_IP -j ACCEPT
```

O que isto indica é que tal como no exemplo da máscara de rede, apenas serão aceites pings da máquina com o ip 10.101.151.5. Isto é verificado pelas tags **-s** e **-d** e o protocolo de pings é verificado com **-p icmp**, sendo o icmp o protocolo de pings.

Outra regra para os pings é dada pelos comandos:

```
sudo /sbin/iptables -A INPUT -p icmp -m icmp --icmp-type 8 -s
10.101.148.0/23 -j ACCEPT
sudo /sbin/iptables -A OUTPUT -p icmp -m icmp --icmp-type 8 -d
10.101.148.0/23 -m limit --limit 2/s -j ACCEPT
```

Tal como os comandos anteriormente usados, as tags **-A INPUT/OUTPUT**, indicam que a regra que vai ser colocada no final do ficheiro, e que o pacote vai ser verificado de maneira diferentes, à saída (**OUTPUT**) e à entrada (**INPUT**).

É verificado a seguir o protocolo com o **-p** e verificamos se o protocolo é o **icmp**, sendo este o protocolo de ping. O **–icmp-type** permite que definemos o tipo de ping que queremos que chegue ou que entre, neste caso queremos pings do tipo “echo” no qual não tem códigos, e que apenas volta para o receptor se for bem sucedido a chegar ao destino. Já foi explicado o **-d** e o **-s**, no qual define o destino do pacote e este destino é apenas para os ip’s dentro da sub-rede dada por 10.101.148.0/23.

Por fim, isto leva-nos ao **-m limit** que define que os ping vão ser limitados e o limite é dado pela tag **--limit 2/s** que limita os pings a dois por segundo. Sempre que houver um ping e ainda não tenha sido atingido o limite de 2 pings por segundo a regra faz jump, **-j**, para o **ACCEPT** e o ping é enviado.

Testes e Resultados

Foram executados os seguintes testes:

1. Pings a todas as máquinas a que não podemos restringir o acesso, como os DC's, Storage, etc;
2. Pings às máquinas da sub-rede 10.101.148.0/23;
3. Ping ao gcc;
4. Correr o MsgServer juntamente com as iptables aplicadas.

Começando pelos testes nº1, como na regra não está definida nenhum protocolo é possível comunicar por qualquer protocolo incluindo o icmp (protocolo do ping), desta forma "pingámos" as máquinas para saber se estas nos respondiam e comunicavam connosco. Obtivemos os seguintes resultados:

Sucesso: DC's, Iate, Luna, Gateway, gcc, máquinas da sub-rede

Insucesso: Storage, Falua, Proxy

Não temos nada a dizer aos sucessos pois eram os resultados esperados, porém os insucessos podem estar relacionados com as próprias firewalls ou os roteamentos da rede que protegem as máquinas correspondentes.

Um detalhe importante, para se saber se a regra que permite o acesso das máquinas da sub-rede "pingarem-se" estava a funcionar, rejeitámos os pacotes de ping da sub-rede em vez de os aceitar. Com isto a nossa máquina ao correr as iptables tornou-se unreachable, o que torna a regra um sucesso.

Para suportar os testes e os resultados usamos o comando **sudo iptables -L -v** e o que isto nos dá, é a lista de todas as regras que correm na máquina e verificámos que apenas as regras estabelecidas no ficheiro iptables-config.sh estavam a ser executadas.