Practical Assignment #1

João Neto – 2023234004

Vasco Alves – 2022228207

22 de março de 2026

Conteúdo

1 Introduction

O objetivo principal deste trabalho era aprender IPTables e como configurar um com o Suricata um sistema de filtração e deteção de ataques. Para esse fim, foi simulado um sistema dividido em três redes e um router para conectar-las. As três redes são a DMZ (23.214.219.128/25, enp0s8), Internal network (192.168.10.0/24, enp0s9) e Internet (87.248.214.0/24, enp0s10).

```
1 Rede,Interface,Gama IP
2 DMZ,enp0s8,23.214.219.128/25
3 Internal,enp0s9,192.168.10.0/24
4 Internet,enp0s10,87.248.214.0/24
```

As três redes tem varios serviços, o DMZ tem dns(23.214.219.130), mail(23.214.219.134), vpn-gw(23.214.219.133), www(23.214.219.132) e smpt(23.214.219.131). A Internal network tem ftp(192.168.10.2), datastore(192.168.10.3) e clientes (nos testes os clientes tem ip 192.168.10.4, mas está configurado para dar para qualquer edereço). Por fim a rede Internet tem dns2 (87.248.214.99) e eden (87.248.214.100), existe também outros serviços (87.248.214.98). Para facilitar a recriação deste sistema foi criado 4 ficheiros .sh (um para cada rede e o router), e disponibilizamos os ficheiros suricata.rules e suricata.yaml, para o suricata que estiver ligado ao Router. Os ficheiros .sh vão ter comandos para configurar o sistema para este exercicio.

2 Firewall

2.1 Packet fileter without NAT

O policy que foi escolhido foi:

```
1 iptables -P INPUT DROP
2 iptables -P FORWARD DROP
3 iptables -P OUTPUT ACCEPT
```

Foi escolhido porque é mais facil dar DROP a todos os pacotes que não foi criado regras do que criar uma regra de DROP para todos os protocolos e possibilidades, o OUTPUT ficou para ACCEPT porque não existe razão para dar DROP dos pacotes que estamos a enviar neste trabalho. Para o router conseguir resolver DNS requests e para aceitar coneções SSH da rede interna ou da VPN gateway foi utilizado estes comandos:

```
1 sudo iptables -A INPUT -i enp0s10 -p udp --dport 53 -j ACCEPT
2 sudo iptables -A INPUT -i enp0s9 -p tcp --dport 22 -j ACCEPT
3 sudo iptables -A INPUT -i enp0s8 -s 23.214.219.133 -p tcp --
    dport 22 -j ACCEPT
```

Para conseguirmos a confirguração pedida entre redes foi utilizado estes commandos:

```
1 sudo iptables -A FORWARD -i enp0s8 -o enp0s10 -s
      23.214.219.130 -p udp --dport 53 -j ACCEPT
2 sudo iptables -A FORWARD -i enp0s9 -o enp0s8 -d
      23.214.219.130 -p udp --dport 53 -j ACCEPT
3 sudo iptables -A FORWARD -i enp0s8 -o enp0s10 -s
      23.214.219.130 -p tcp --dport 53 -j ACCEPT
4 sudo iptables -A FORWARD -i enp0s9 -o enp0s8 -d
      23.214.219.131 -p tcp --dport 587 -j ACCEPT
5 sudo iptables -A FORWARD -i enp0s9 -o enp0s8 -d
      23.214.219.134 -p tcp --dport 143 -j ACCEPT
6 sudo iptables -A FORWARD -i enp0s9 -o enp0s8 -d
      23.214.219.134 -p tcp --dport 110 -j ACCEPT
7 sudo iptables -A FORWARD -i enp0s9 -o enp0s8 -d
      23.214.219.132 -p tcp --dport 80 -j ACCEPT
8 sudo iptables -A FORWARD -i enp0s9 -o enp0s8 -d
      23.214.219.132 -p tcp --dport 443 -j ACCEPT
9 sudo iptables -A FORWARD -i enp0s9 -o enp0s8 -d
      23.214.219.133 -p udp --dport 1194 -j ACCEPT
10 sudo iptables -A FORWARD -i enp0s8 -o enp0s9 -s
      23.214.219.133 -d 192.168.10.2 -j ACCEPT
11 sudo iptables -A FORWARD -i enp0s8 -o enp0s9 -s
      23.214.219.133 -d 192.168.10.3 -j ACCEPT
```

Inicialmente as implementações de respostas a forward eram especificas para cada regra isto é por exemplo:

```
1 sudo iptables -A FORWARD -o enp0s8 -i enp0s10 -p udp --dport
      53 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

No entanto isso facilmente originava confusão entre nós, então decimos utilizar estas duas regras:

```
1 sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -
      j ACCEPT
2 sudo iptables -A FORWARD -m state --state ESTABLISHED,RELATED
       -j ACCEPT
```

Neste cenario o uso destas regras faz sentido, mas pode existir outros cenarios no futuro que não queremos uma resposta, e nesse caso temos de criar as regras necessarias.

2.2 Packet filtering with NAT

Para conecções com origem/destino na internet foi utilizado DNAT/SNAT e iptables para ”esconder”o ip para a internet que querer aceder a rede interna para não terem acesso ao edereço ip e iproutes para bloquear certos pacotes de entrar, para conseguir a configuração utilizamos estes comandos:

```
1 sudo iptables -A FORWARD -i enp0s10 -o enp0s9 -d 192.168.10.2
       -p tcp --dport 21 -j ACCEPT
```

```
2 sudo iptables -A FORWARD -i enp0s9 -o enp0s10 -p tcp --sport
      20 -j ACCEPT
3 sudo iptables -t nat -A PREROUTING -s $dns2 -p tcp --dport 22
       -j DNAT --to-destination 192.168.10.3
4 sudo iptables -t nat -A PREROUTING -s $eden -p tcp --dport 22
       -j DNAT --to-destination 192.168.10.3
5 sudo iptables -t nat -A PREROUTING -i enp0s10 -p tcp --dport
      21 -j DNAT --to-destination 192.168.10.2
6 sudo iptables -A FORWARD -i enp0s10 -o enp0s9 -d 192.168.10.3
       -s $dns2 -p tcp --dport 22 -j ACCEPT
7 sudo iptables -A FORWARD -i enp0s10 -o enp0s9 -d 192.168.10.3
       -s $eden -p tcp --dport 22 -j ACCEPT
8 sudo iptables -t nat -A POSTROUTING -s 192.168.10.0/24 -o
      enp0s10 -j SNAT --to-source 87.248.214.97
9 sudo iptables -A FORWARD -i enp0s9 -o enp0s10 -p udp --dport
      53 -j ACCEPT
10 sudo iptables -A FORWARD -i enp0s9 -o enp0s10 -p tcp --dport
      80 -j ACCEPT
11 sudo iptables -A FORWARD -i enp0s9 -o enp0s10 -p tcp --dport
      443 -j ACCEPT
12 sudo iptables -A FORWARD -i enp0s9 -o enp0s10 -p tcp --sport
      21 -j ACCEPT
13 sudo iptables -A FORWARD -i enp0s9 -o enp0s10 -p tcp --dport
      21 -j ACCEPT
```

3 Intrusion Detection

As regras que utilizamos para o suricata foram estas:

```
1 drop tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"ET"; flags:
      S; threshold:type both, track by_src, count 5, seconds 60;
       classtype:attempted-recon; sid:1000001; rev:1;)
2 drop tcp any any -> any 80 (msg:"SQL injection"; content:"
      union"; nocase; content:"select"; nocase; classtype:web-
      application-attack; sid:1000002; rev:1;)
3 drop tcp any any -> any 80 (msg:"SQl injection"; content:"'or
       1=1"; nocase; classtype:web-application-attack; sid
      :1000003; rev:1;)
4 drop tcp any any -> any 80 (msg:"XSS"; content:"<script";
      nocase; classtype:web-application-attack; sid:1000004; rev
      :1;)
```

A primeira é para port scaning, a segunda e a terceira é para o caso de SQL injection, e a ultima é para XSS atacks. Também atualizamos o iptables para passar para o suricata os pacotes para analizar e bloquear com:

```
1 sudo iptables -I FORWARD -j NFQUEUE --queue-bypass
2 sudo iptables -I INPUT -j NFQUEUE --queue-bypass
```

4 Tests utilizados

Netcat foi utilizado para maior parte dos testes excepto para FTP, em que devido ás suas caracteristicas especificas, utilizamos os serviços para ter a certeza que funcionava com a nossa configuração. Utilizamos estes comandos curl para testar se eram bloqueados:

```
1 curl -i "http://23.214.219.132/index.php?id=1%20union%20
    select%201,2,3"
2 curl -i "http://23.214.219.132/login.php?user='or%201=1"
3 curl -i "http://23.214.219.132/search.php?q=<script>alert('
    XSS')</script>"
```

5 Conclusion

Ao realizar-mos este projeto aprendemos sobre a criação de scenarios em VMs, a configuração de uma firewall utilizando IPTables e a configuração de um IDS/IPS system utilizando Suricata