Assignment 3: iptables on ACME Co.

Alessandro Serpi - 1647244

8 March 2019

Contents

1 Introduction

The private network of *ACME co.* is composed by four subnetworks living in the shared address space `100.64.0.0/16`. DMZ is connected directly to the main firewall-router *mainfw* and offers services accessible from the external network, while *client* and *internal server* networks are behind a second line of defence (represented by the internal firewall-router *intfw*). The latter offers services that may be used only by hosts in the private network, whereas the former does not offer any services.

All hosts except the firewalls are already configured.

2 Evaluation of the security policy

2.1 DNS

Only internal hosts should be able to resolve internal domain names using the private DNS server. While the requirement are clear in regard to the impossibility for the server to reach the internet, recursive queries are enabled for the entire private network.

2.2 DMZ

Only external hosts should be able to connect to the service offered by server in the DMZ, namely FTP and web.

2.3 SSH

Hosts in the *client* network should be able to use the SSH protocol in the internal network, no other use is authorised.

2.4 Syslog

Internal hosts (except clients) should be able to send logging messages to the syslog server. No other use is authorised, therefore admins can access the logs only through a SSH session.

3 Policy implementation

3.1 Default policies

The default policies for `FORWARD`, `INPUT` and `OUTPUT` chains are set to `DROP` in both firewalls.

3.2 *mainfw*

3.2.1 Logging

```
-I FORWARD -j LOG --log-level 4 --log-prefix "netfilter forward drop: "
-I INPUT -j LOG --log-level 4 --log-prefix "netfilter input drop: "
-I OUTPUT -j LOG --log-level 3 --log-prefix "netfilter output drop: "
```

3.2.2 DMZ

```
-t nat -I PREROUTING -i eth0 -p tcp -m multiport --dports $FTP_PORTS -j
      ↪ DNAT --to-destination $FTP_ADDR
-I FORWARD -d $FTP_ADDR -p tcp -m multiport --dports $FTP_PORTS -j
      ↪ ACCEPT
-I FORWARD -s $FTP_ADDR -p tcp -m multiport --sports $FTP_PORTS -m
      ↪ conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

-I FORWARD -i eth0 -d $WEB_ADDR -p tcp --dport $WEB_PORT -j ACCEPT
-I FORWARD -o eth0 -s $WEB_ADDR -p tcp --sport $WEB_PORT -m conntrack
      ↪ --ctstate ESTABLISHED,RELATED -j ACCEPT
-t nat -I PREROUTING -i eth0 -p tcp --dport $WEB_PORT -j DNAT
      ↪ --to-destination $WEB_ADDR
```

Since the servers must be accessible from the outside but the `100.64.0.0/16` IP range is not routable, it is necessary to perform port forwarding.

3.2.3 Internal network

```
-I FORWARD -d $DNS_ADDR -s $DMZ_NET -p udp --dport $DNS_PORT -j ACCEPT
-I FORWARD -s $DNS_ADDR -p udp --sport $DNS_PORT -m conntrack --ctstate
      ↪ ESTABLISHED,RELATED -j ACCEPT
-I FORWARD -d $DNS_ADDR -s $DMZ_NET -p tcp --dport $DNS_PORT -j ACCEPT
-I FORWARD -s $DNS_ADDR -p tcp --sport $DNS_PORT -m conntrack --ctstate
      ↪ ESTABLISHED,RELATED -j ACCEPT

-I FORWARD -d $LOG_ADDR -s $NET -p udp --dport $LOG_PORT -j ACCEPT
-I OUTPUT -d $LOG_ADDR -p udp --dport $LOG_PORT -j ACCEPT
```

3.2.4 SSH

```
-I FORWARD -s $CLIENT_NET -p tcp --dport $SSH_PORT -j ACCEPT
-I FORWARD -d $CLIENT_NET -p tcp --sport $SSH_PORT -m conntrack
      ↪ --ctstate ESTABLISHED,RELATED -j ACCEPT
-I INPUT -s $CLIENT_NET -p tcp --dport $SSH_PORT -j ACCEPT
-I OUTPUT -d $CLIENT_NET -p tcp --sport $SSH_PORT -m conntrack --ctstate
      ↪ ESTABLISHED,RELATED -j ACCEPT
```

3.2.5 Anti-spoofing

```
-N ANTISPOOFING
-P ANTISPOOFING DROP
-I ANTISPOOFING -j LOG --log-level 3 --log-prefix
      ↪ "netfilter anti-spoofing: "

-I FORWARD -i eth0 -s $NET -j ANTISPOOFING
-I INPUT -i eth0 -s $NET -j ANTISPOOFING
-I FORWARD -i eth1 ! -s $DMZ_NET -j ANTISPOOFING
-I INPUT -i eth1 ! -s $DMZ_NET -j ANTISPOOFING
-I FORWARD -i eth2 -s $DMZ_NET -j ANTISPOOFING
-I INPUT -i eth2 -s $DMZ_NET -j ANTISPOOFING
-I FORWARD -i eth2 -s $MAINFW_ADDR -j ANTISPOOFING
-I INPUT -i eth2 - $MAINFW_ADDR -j ANTISPOOFING
```

Since previous instructions do not enforce specific interface, we must ascertain that an attacker is not using a spoofed IP address.

3.3 *intfw*

3.3.1 Internal network

```
-I FORWARD -d $DNS_ADDR -s $CLIENT_NET -p udp --dport $DNS_PORT -j
      ↪ ACCEPT
-I FORWARD -d $DNS_ADDR -s $DMZ_NET -p udp --dport $DNS_PORT -j ACCEPT
-I FORWARD -s $DNS_ADDR -p udp --sport $DNS_PORT -m conntrack --ctstate
      ↪ ESTABLISHED,RELATED -j ACCEPT
-I FORWARD -d $DNS_ADDR -s $CLIENT_NET -p tcp --dport $DNS_PORT -j
      ↪ ACCEPT
-I FORWARD -d $DNS_ADDR -s $DMZ_NET -p tcp --dport $DNS_PORT -j ACCEPT
-I FORWARD -s $DNS_ADDR -p tcp --sport $DNS_PORT -m conntrack --ctstate
      ↪ ESTABLISHED,RELATED -j ACCEPT

-I FORWARD -d $LOG_ADDR -s $NET -p udp --dport $LOG_PORT -j ACCEPT
-I OUTPUT -d $LOG_ADDR -p udp --dport $LOG_PORT -j ACCEPT
```

3.3.2 SSH

```
-I FORWARD -s $CLIENT_NET -p tcp --dport $SSH_PORT -j ACCEPT
-I FORWARD -d $CLIENT_NET -p tcp --sport $SSH_PORT -m conntrack
      ↪ --ctstate ESTABLISHED,RELATED -j ACCEPT
-I INPUT -s $CLIENT_NET -p tcp --dport $SSH_PORT -j ACCEPT
-I OUTPUT -d $CLIENT_NET -p tcp --sport $SSH_PORT -m conntrack --ctstate
      ↪ ESTABLISHED,RELATED -j ACCEPT
```

3.3.3 Anti-spoofing

```
-N ANTISPOOFING
-P ANTISPOOFING DROP
-I ANTISPOOFING -j LOG --log-level 3 --log-prefix
      ↪ "netfilter anti-spoofing: "

-I FORWARD -i eth0 -s $SERVER_NET,$CLIENT_NET,$INTFW_ADDR -j
      ↪ ANTISPOOFING
-I INPUT -i eth0 -s $SERVER_NET,$CLIENT_NET -j ANTISPOOFING
-I FORWARD -i eth1 ! -s $SERVER_NET -j ANTISPOOFING
-I INPUT -i eth1 ! -s $SERVER_NET -j ANTISPOOFING
-I FORWARD -i eth2 ! -s $CLIENT_NET -j ANTISPOOFING
-I INPUT -i eth2 ! -s $CLIENT_NET -j ANTISPOOFING
```

Since previous instructions do not enforce specific interface, we must ascertain that an attacker is not using a spoofed IP address.

4 Test of the configuration

We use the test environment integrated in Kathará. To verify the correctness of the implementation, execute `ltest --verify=user` in the lab root folder. For each host, faulty elements are listed in the `_test/diff` folder. Note that there may be some false positive.

Given that *expect* is not installed and both *ftp* and *ssh* clients can not be reliably used with simple bash scripts, testing the corresponding protocols must be done manually, checking the connectivity for each host.

5 Final remarks

Hosts in the private network are isolated from the internet (with the exception of DMZ servers, which can not initiate new connections). This is a very restrictive policy and clashes with the DNS configuration.

The Kathará test environment has proven to be unreliable, often giving false positive regarding internet availability in DNS and FTP servers. This is probably caused by a race condition between the test and the configuration scripts.