SVS Bachelor-Projekt Network Security

Blatt 5: Beschreibung der Experimentierumgebung

Louis Kobras
6658699

Utz Pöhlmann
6663579

1 Netzwerkeinstellungen

1.2

ClientVM:

IP-Adresse (`ifconfig -a`): 192.168.254.44	**Standard-Gateway** (`route -n`): 192.168.254.2	**DNS-Nameserver** (`nslookup ubuntu.com`): 10.1.1.1

RouterVM:

eth0	eth1
172.16.137.222	192.168.254.2

ServerVM:
IP-Adresse der Server-VM: 172.16.137.144

2 Absichern eines Einzelplatzrechners mit iptables (ClientVM)

2.1

Anzeigen der Firewall-Regeln mit `sudo iptables -L`; alle Regeln löschen mit `sudo iptables -F`[1]; OpenSSH-Server nach Paketquellen-Update via apt-get installiert (automatisch gestartet).

2.2

Regelwerk siehe [2.2: ClientVM-Filterregeln (S. 5)].
`iptables` säubern mit `sudo iptables -F`, einladen der Regeln aus einem Textfile mit `sudo iptables-restore < ~/iptable` (Dateiinhalt im Anhang ebenda).

2.3

- SSH-Verbindungsversuch von RouterVM mit `sudo ssh 192.168.254.44` erfolgreich
- SSH-Verbindungsversuch in die andere Richtung nicht erfolgreich (Connection refused)
- hosten eines Servers mit `netcat -l 5555` erfolgreich, Verbindung (`sudo netcat 192.168.254.44 5555`) erwartungsgemäß fehlgeschlagen
- Firefox ist bei DROP schneller als bei REJECT

2.4

Dynamische Regeln vgl. [2.4: ClientVM Stateful Filtering (S. 5)].
Man muss nicht jeden Port und jedes Protokoll einzelnd abdecken. Stateful Filter sind effizienter, da sie sich nur die Paket-Header ansehen.

[1] löscht alle Regeln nacheinander

3 Absichern eines Netzwerks (RouterVM)

3.1

Der Aufruf bedeutet (nach [1]): “Maskiere alles, was an eth0 ausgeht”.
Es wird die Adressumsetzung (NAT) aktiviert und die Schnittstelle markiert ([2]).
Source: 192.168.254.0; Maske: 24

3.2

Die Client-VM kann die Server-VM anpingen; umgekehrt geht dies nicht.
Vermutung: Die Client-VM ist von außen nicht direkt ansprechbar, da sie hinter der RouterVM versteckt ist.

3.3

Regelsatz im Anhang unter [3.3: Filterregeln (S. 5)]
ACHTUNG: Funktioniert nicht! Ab hier alle Angaben theoretische Überlegungen

3.4

Folgender Eintrag in der iptable *filter an Stelle [0] sollte den SSH-Tunnel zulassen: `-A FORWARD -d 172.16.137.144 -p tcp -port 22 -j ACCEPT`

3.5

Folgende Regeln sollte die Aufgabe erfüllen:

```
iptables -A PREROUTING -t nat -i eth0 -p tcp -dport 5022 -j DNAT -to 192.168.254.44:22
iptables -A FORWARD -p tcp -d 192.168.254.44 -dport 22 -j ACCEPT
```

Zusätzlich muss der öffentliche Port mithilfe von netcat geöffnet werden: `nc -l 5022`

3.6

.

4 SSH-Tunnel

4.1

.

4.2

.

4.3

.

4.4

.

5 OpenVPN

5.1

.

5.2

.

5.3

.

5.4

.

5.5

.

5.6

.

6 HTTP-Tunnel

6.1

.

6.2

.

6.3

.

6.4

.

6.5

.

6.6

.

Literatur

[1] www.netfilter.org/documentation/HOWTO/de/NAT-HOWTO-6.html

[2] https://wiki.ubuntuusers.de/Router/

ANHANG

2.2: ClientVM-Filterregeln

TODO: textfile

2.4: ClientVM Stateful Filtering

TODO: textfile

3.3: Filterregeln

```
1 iptables -t filter -A FORWARD -d 10.1.1.2/32 -j DROP
2 iptables -t filter -A FORWARD -d 10.0.0.0/8 -j DROP
3 iptables -t filter -A FORWARD -p udp --dport 53 --sport 53 -J ACCEPT
4 iptables -t filter -A FORWARD -i eth1 -m state --state NEW -j ACCEPT
5 iptables -t filter -A FORWARD -m state --state ESTABLISHED,RELATED -j
    ACCEPT
6 iptables -t filter -A FORWARD -p tcp -m multiport ! --ports 80,443,8080 -j
    DROP
```