Sieć

- Przekierowanie ruchu sieciowego pomiędzy kartami sieciowymi
- Skrypt tworzący reguły przekierowania dla portów mailowych
- Weryfikacja zapory sieciowej (firewalld) oraz dodanie wyjątków

Przekierowanie ruchu sieciowego pomiędzy kartami sieciowymi

Włączenie przekazywania pakietów (IP Forwarding)

Na początku należy sprawdzić konfigurację sysctl

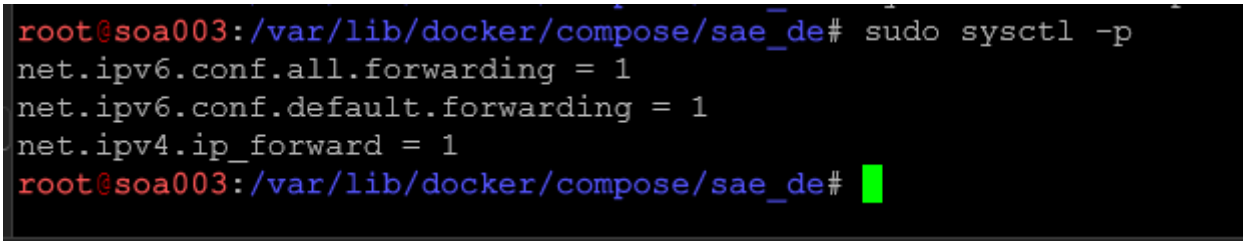
```
sudo vim /etc/sysctl.conf
```

i szukamy wpisu:

```
net.ipv4.ip_forward = 1
```

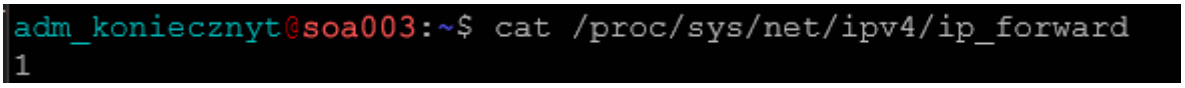
Przeładowujemy konfigurację

```
sudo sysctl -p
```



i sprawdzamy czy opcja jest aktywna:

```
cat /proc/sys/net/ipv4/ip_forward
```



Jeśli wynik to `1`, przekazywanie pakietów jest włączone.

Konfiguracja NAT (SNAT) poprzez **IPTABLES**

```
sudo iptables -t nat -I POSTROUTING -p all -s AdresacjaIPskąd ! -d AdresacjaIPdokąd -j SNAT --to-source JakimIPmaWychodzić
```

Przykład z całymi podsieciami:

```
sudo iptables -t nat -I POSTROUTING -p all -s 172.18.0.0/29 ! -d 172.18.0.0/29 -j SNAT --to-source 10.95.227.9
```

Co robi ta reguła?

- **iptables →** komenda którą nanosimy zmiany
- **-t nat →** Modyfikuje tablicę NAT.
- **-I POSTROUTING →** Wstawia regułę do łańcucha POSTROUTING, czyli po podjęciu decyzji o routingu.
- **-p all →** Dotyczy wszystkich protokołów (TCP, UDP, ICMP itp.).
- **-s 172.18.0.0/29 →** Ogranicza regułę do ruchu wychodzącego z tej podsieci.
- **! -d 172.18.0.0/29 →** Nie dotyczy ruchu wewnątrz tej samej podsieci (eliminuje NAT dla ruchu lokalnego).
- **-j SNAT --to-source 10.95.227.9 →** Zamienia źródłowy adres IP na 10.95.227.9, aby umożliwić komunikację z innymi sieciami.

Sprawdzamy czy reguła została dodana:

```
sudo iptables -t nat -L -v -n
```

```
adm_koniecznyt@soa003:~$ sudo iptables -t nat -L -v -n
Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
2408K  145M DOCKER     all  --  *      *       0.0.0.0/0            0.0.0.0/0            ADDRTYPE match

Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 MASQUERADE  all  --  *      !docker0  172.17.0.0/16        0.0.0.0/0
98403 6152K MASQUERADE  all  --  *      !sae-srs-bridge  172.18.0.0/29        0.0.0.0/0
```

Zapisujemy konfigurację:

```
sudo service iptables save
```

```
sudo iptables-save | sudo tee /etc/sysconfig/iptables
```

Skrypt tworzący reguły przekierowania dla portów mailowych

Uruchamiamy na systemie proxy:

```
#!/bin/bash

# Sposób użycia: sudo ./mail_routing.sh <docelowy_adres_IP>
#
# Przykład: sudo ./mail_routing.sh 192.168.1.138
# Sprawdzenie, czy podano adres IP

if [ -z "$1" ]; then
    echo "Użycie: $0 <docelowy_adres_IP>"
    exit 1
fi

TARGET_IP="$1"

# Lista standardowych portów pocztowych
MAIL_PORTS=(25 465 587 143 993 110 995)

echo "Usuwanie poprzednich reguł..."
iptables -t nat -F PREROUTING
iptables -F FORWARD
iptables -t nat -F POSTROUTING

echo "Dodawanie nowych reguł przekierowania dla adresu: $TARGET_IP"

# Dodawanie reguł przekierowania dla każdego portu
for PORT in "${MAIL_PORTS[@]}"; do
    echo "Przekierowanie portu $PORT na $TARGET_IP:$PORT"
```

```
    iptables -t nat -A PREROUTING -p tcp --dport $PORT -j DNAT --to-destination $TARGET_IP:$PORT
    iptables -A FORWARD -p tcp --dport $PORT -d $TARGET_IP -j ACCEPT
    iptables -t nat -A POSTROUTING -p tcp -d $TARGET_IP --dport $PORT -j MASQUERADE
done

# Zapisywanie reguł, aby przetrwały restart systemu
echo "Zapisywanie reguł iptables..."
iptables-save > /etc/sysconfig/iptables 2>/dev/null || service iptables save 2>/dev/null

echo "Konfiguracja zakończona! Sprawdzenie reguł:"
iptables -t nat -L -v -n
iptables -L -v -n
```

Przykładowa część wyniku:

```
root@lrproxyprd-vnic:~$: ./mail-routing.sh 192.168.1.138
Usuwanie poprzednich reguł...
Dodawanie nowych reguł przekierowania dla adresu: 192.168.1.138
Przekierowanie portu 25 na 192.168.1.138:25
Przekierowanie portu 465 na 192.168.1.138:465
Przekierowanie portu 587 na 192.168.1.138:587
Przekierowanie portu 143 na 192.168.1.138:143
Przekierowanie portu 993 na 192.168.1.138:993
Przekierowanie portu 110 na 192.168.1.138:110
Przekierowanie portu 995 na 192.168.1.138:995
Zapisywanie reguł iptables...
Konfiguracja zakończona! Sprawdzenie reguł:
Chain PREROUTING (policy ACCEPT 2125 packets, 420K bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DNAT       6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:25 to:192.168.1.138:25
    0     0 DNAT       6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:465 to:192.168.1.138:465
    0     0 DNAT       6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:587 to:192.168.1.138:587
    0     0 DNAT       6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:143 to:192.168.1.138:143
    0     0 DNAT       6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:993 to:192.168.1.138:993
    0     0 DNAT       6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:110 to:192.168.1.138:110
    0     0 DNAT       6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:995 to:192.168.1.138:995

Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 6898 packets, 613K bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 MASQUERADE  6    --  *      *       0.0.0.0/0            192.168.1.138        tcp dpt:25
    0     0 MASQUERADE  6    --  *      *       0.0.0.0/0            192.168.1.138        tcp dpt:465
    0     0 MASQUERADE  6    --  *      *       0.0.0.0/0            192.168.1.138        tcp dpt:587
    0     0 MASQUERADE  6    --  *      *       0.0.0.0/0            192.168.1.138        tcp dpt:143
    0     0 MASQUERADE  6    --  *      *       0.0.0.0/0            192.168.1.138        tcp dpt:993
    0     0 MASQUERADE  6    --  *      *       0.0.0.0/0            192.168.1.138        tcp dpt:110
    0     0 MASQUERADE  6    --  *      *       0.0.0.0/0            192.168.1.138        tcp dpt:995
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 20 packets, 926 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     6    --  *      *       0.0.0.0/0            192.168.1.138        tcp dpt:25
    0     0 ACCEPT     6    --  *      *       0.0.0.0/0            192.168.1.138        tcp dpt:465
```

Weryfikacja zapory sieciowej (firewalld) oraz dodanie wyjątków

> Komendy wykonujemy z CLI bash / sh z uprawnieniami root (sudo)

Sprawdzenie wszystkich reguł firewallD:

```
sudo firewall-cmd --list-all
```

Dodanie portu:

```
sudo firewall-cmd --add-port=NumerPortu/tcp --permanent
```

Dodanie portu ograniczając do jednej zone

```
sudo firewall-cmd --zone=NazwaZony --add-port=NumerPortu/tcp --permanent
```

Dodanie usługi:

```
sudo firewall-cmd --permanent --add-service=NazwaService
```

Po zmianach należy przeładować firewallD:

```
sudo firewall-cmd --reload
```

Co to jest "rich rule" w firewallD?

Rich rules to zaawansowane reguły w firewallD, które pozwalają na bardziej precyzyjne definiowanie zasad niż standardowe strefy i usługi. Umożliwiają m.in.:

- **Filtrowanie według protokołu, adresów IP, portów, interfejsów**
- **Ustalanie kierunku ruchu (input, output, forward)**
- **Logowanie i kontrolę pasma**

Przykład dodania rich-rule:

```
sudo firewall-cmd --add-rich-rule='rule protocol value="NazwaProtokołu" accept' --permanent
```

Przykład wpuszczenia ruchu **ssh** z jednego adresu IP

```
sudo firewall-cmd --add-rich-rule='rule family="ipv4" source address="192.168.1.100" service name="ssh" accept' --permanent
```

Po zmianach należy przeładować firewallD:

```
sudo firewall-cmd --reload
```