Warsztaty z Sieci komputerowych
Lista 8

1 Uwagi ogólne

Pracę rozpocznij poleceniem `netmode lab`, a następnie skonfiguruj interfejs `eth0` za pomocą protokołu DHCP (`ifup eth0`). Sprawdź, że otrzymany adres IP należy do podsieci `172.16.1.0/24`. Na dzisiejszej pracowni wszystkie komputery są podłączone interfejsami `eth0` do przełącznika, zaś interfejsy `eth1` spinają parami sąsiednie komputery.

Konfigurację zapory przeprowadza się poleceniem `iptables`; zapora jest po prostu listą reguł, które rozpatrywane są od początku do końca (kolejność ma znaczenie!). Konfiguracja przeprowadzana jest interaktywnie przez wpisanie reguł w odpowiednie miejsca listy. Wygodniej jest jednak wykorzystać następujące podejście:

1. Tworzymy plik `firewall.sh`, na którego początku znajduje się polecenie wyczyszczenia listy reguł.

2. Kolejne polecenia z pliku `firewall.sh` to wywołania programu `iptables`, dopisujące reguły *na koniec* listy.

Dzięki temu jeśli wykonamy plik `firewall.sh`, reguły zapory będą dokładnie takie, jak zapisane w tym pliku.

2 Zadania

Zadanie 1. W tym zadaniu stworzymy podstawowe reguły zapory. Utwórz plik `firewall.sh` i nadaj mu prawa do uruchamiania. Następnie wpisz do niego następujące wiersze pomijając numery i następujące po nich dwukropki (przykładowo pierwszy wiersz powinien być równy `#!/bin/bash`).

```
 1: #!/bin/bash
 2: set -x
 3:
 4: sudo iptables -F
 5: sudo iptables -t nat -F
 6:
 7: sudo iptables -P INPUT DROP
 8: sudo iptables -P FORWARD DROP
 9: sudo iptables -P OUTPUT ACCEPT
10:
11-15: # tu będą reguły zapory
```

```
16:
17-21: # a tu będą reguły NAT
22:
23: sudo iptables -A INPUT -j LOG --log-prefix "blokada INPUT "
24: sudo iptables -A FORWARD -j LOG --log-prefix "blokada FORWARD "
```

Wiersze 1–2 informują, że skrypt będzie skryptem powłoki `bash` i że podczas jego uruchamiania mają być wypisywane poszczególne instrukcje (ułatwia to późniejsze znajdowanie błędów). Wiersze 4–5 powodują usunięcie wszystkich reguł (z domyślnej tabeli `filter` i tabeli `nat`). Wiersze 7–9 ustawiają domyślną politykę modułów (*chains*) `INPUT` i `FORWARD` na wyrzucanie pakietów, zaś modułu `OUTPUT` na ich przyjmowanie.

Wiersze 11–15 i 17–21 wypełnimy później. Będą tam znajdować się reguły *zezwalające* na różne typy pakietów przychodzących (moduł `INPUT`) albo przechodzących przez nasz komputer (moduł `FORWARD`).

Do wierszy 22–23 dotrą zatem pakiety, które nie zostaną wpuszczone przez żadne wcześniejsze reguły. Reguły z tych wierszy powodują zapisanie informacji o takich pakietach do pliku dziennika (typowo do `/var/log/messages`).

Po wpisaniu poleceń do pliku `firewall.sh` uruchom je poleceniem

```
$> ./firewall.sh
```

A następnie wyświetl bieżące ustawienia zapory poleceniami

```
#> iptables -L -n
#> iptables -t nat -L -n
```

Powyższe polecenia należy wykonywać *po każdej* edycji pliku `firewall.sh`. (Nie będzie to zaznaczone w poniższych zadaniach).

W osobnej konsoli wpisz (i pozostaw uruchomione) polecenie

```
$> tail -f /var/log/messages
```

Będzie ono wyświetlać bieżącą końcówkę pliku dziennika, czyli w szczególności zablokowane pakiety.

Spróbuj teraz połączyć się teraz z zewnętrznym serwerem SSH poleceniem

```
$> ssh 192.168.254.254
```

Czy udaje się nawiązać połączenie? Oglądając plik dziennika wywnioskuj, że przyczyną problemów jest niewpuszczanie odpowiedzi od serwera SSH. Spróbuj wpisać też polecenie:

```
$> ssh eagle-server.example.com
```

Tym razem przyczyną powinno być niewpuszczanie odpowiedzi od serwera DNS.

Zadanie 2. Do pliku `firewall.sh` dopisz regułę wpuszczającą pakiety należące do już nawiązanych połączeń

```
11: sudo iptables -A INPUT -m state --state ESTABLISHED -j ACCEPT
```

Ponownie sprawdź połączenie z serwerem SSH poleceniem

```
$> ssh eagle-server.example.com
```

Tym razem powinno ono zadziałać. Sprawdź, czy możesz połączyć się z serwerem SSH swojego sąsiada wpisując

```
$> ssh adres_IP_sąsiada
```

Co pojawia się w Twoim pliku dziennika, a co w jego? Połączenie nie powinno się udać, gdyż próba połączenia z portem 22 będzie odrzucana. Z tego samego powodu niemożliwe powinno być także połączenie się z własnym serwerem SSH poleceniem

```
$> ssh 127.0.0.1
```

Aby to naprawić wpuść wszystkie połączenia lokalne i połączenia SSH z zewnątrz wpisując do pliku `firewall.sh` następujące wiersze.

```
12: sudo iptables -A INPUT -i lo -j ACCEPT
13: sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
```

Ponownie spróbuj połączyć się przez SSH z sąsiadem i samym sobą.

Zadanie 3. Spróbuj pingnąć samego siebie i sąsiada. Ping sąsiada powinien być odrzucany. Sprawdź, co jest dopisywane do pliku dziennika. Aby wpuścić pakiety ICMP wykorzystywane przez ping (pakiety o typie `echo-request` dodaj następujące wiersze do pliku `firewall.sh`.

```
14: sudo iptables -A INPUT -p icmp --icmp-type echo-request -j LOG
    --log-prefix "Ktos pinga! "
15: sudo iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
```

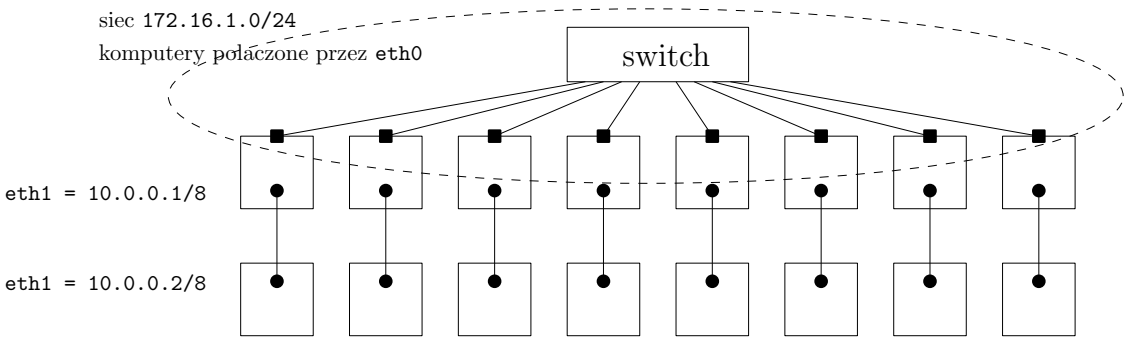
(Wiersz 14 powinien być jednym wierszem, został rozbity na dwa, żeby zmieścić się na kartce.) Wiersz 14 powoduje rejestrowanie pingów w pliku dziennika. Reguła `-j LOG` jest specyficzna: nawet jeśli zostanie zastosowana do pakietu, to następne reguły są nadal przeglądane. Sprawdź, jakie komunikaty pojawiają się w plikach dziennika (Twoim i Twojego sąsiada) jeśli pingacie się nawzajem, a jakie, kiedy pingacie swoje własne komputery. Wyświetl statystyki zapory poleceniem

```
#> iptables -L -nv
```

Ile pakietów zostało zatrzymanych a ile przepuszczonych przez poszczególne reguły? W tym momencie w pliku `firewall.sh` powinny być już wypełnione wiersze 11–15.

Wykorzystaj program `nmapfe`, aby sprawdzić jakie porty są otwarte na komputerze obok (wpisz jego IP w polu *Target*). W polu *Profile* wybierz *Intense scan*. Przeczytaj wyświetlane informacje. Oglądając pakiety w Wiresharku i komunikaty o zablokowanych pakietach w pliku dziennika sprawdź, z jakimi portami usiłował połączyć się `nmapfe`.

Zadanie 4. W tym i kolejnym zadaniu wykorzystamy interfejsy `eth1`; zadania należy wykonywać w parach. Komputer należący do jednej z osób będzie *routerem NAT*, zaś drugi komputer będzie *stacją roboczą*. Dążymy do otrzymania działającej konfiguracji przedstawionej na poniższym rysunku (pierwsza warstwa to routery NAT zaś druga stacje robocze).



Na stacji roboczej nie będziemy już dodawać kolejnych poleceń do konfiguracji zapory. Należy tam wyłączyć interfejs `eth0`, a interfejsowi `eth1` przypisać adres `10.0.0.2` poleceniami

```
#> ifconfig eth0 down
#> ifconfig eth1 10.0.0.2 up
```

Na routerze NAT należy skonfigurować interfejs `eth1` przypisując mu adres `10.0.0.1` poleceniem

```
#> ifconfig eth1 10.0.0.1 up
```

Poleceniem `ping` sprawdźcie, czy komputery widzą się nawzajem przez interfejs `eth1`.

Zauważ, że adresy z sieci `172.16.1.0/24` jednoznacznie identyfikują adresy kart `eth0` routerów NAT, lecz takie same adresy `10.0.0.0/8` są przypisane do wielu komputerów. Najpierw pokażemy, do jakich problemów może to prowadzić.

Na stacji roboczej ustaw jako bramę domyślną sąsiedni router NAT poleceniem

```
#> route add default gw 10.0.0.1
```

Oznaczmy adres IP karty `eth0` sąsiedniego routera NAT przez A zaś adres IP karty `eth0` jakiegoś innego routera NAT przez B. Na stacji roboczej pingnij adresy A i B. Zaobserwuj, co pojawia sie wtedy w pliku dziennika sąsiedniego routera NAT. Okazuje, się że winne jest blokowanie ruchu przechodzącego przez router NAT. Napraw to dopisując na routerze NAT następujące wiersze do pliku `firewall.sh`.

```
17: sudo iptables -A FORWARD -m state --state ESTABLISHED -j ACCEPT
18: sudo iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT
```

Drugie z nich przepuszcza cały ruch pochodzący od stacji roboczej, zaś pierwsze wpuszcza pakiety należące do już nawiązanych połączeń.

Ponownie pingnij adresy A i B ze stacji roboczej. Sprawdźcie w Wiresharku, że w drugim przypadku odpowiedź na Twojego pinga otrzymuje stacja robocza sąsiadująca z routerem B! Dlaczego tak się dzieje? Aby temu zaradzić włączymy funkcję źródłowego NAT na routerach NAT dopisując do pliku `firewall.sh` wiersz

```
19: sudo iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 172.16.1.X
```

gdzie adres `172.16.1.X` jest adresem IP przypisanym karcie `eth0`. Na stacji roboczej ponownie pingnij adresy A i B. Na routerze NAT B, w Wiresharku zaobserwuj z jakiego adresu IP pochodzą dochodzące do niego pingi (wygląda to tak, jakby pingał go router A, choć pole TTL jest o 1 mniejsze niż normalnie). Tym razem oba pingnięcia powinny się udać. Natomiast na routerze NAT A sprawdź, że wszystkie pakiety są rejestrowane dwukrotnie w Wiresharku: przed i po podmianie adresów IP.

Zadanie 5. W poprzednich zadaniach skonfigurowaliśmy możliwość łączenia się stacji roboczych (reprezentujących lokalne sieci) z siecią `172.16.1.0/24` (reprezentującą Internet). Co jednak zrobić, jeśli zapragniemy połączyć się ze stacją roboczą (np. przez protokół SSH)?

Wystarczy w tym celu przekierować jakiś port (np. 2222) routera NAT do portu 22 stacji roboczej sąsiadującej z tym routerem wpisując do pliku `firewall.sh` na routerze NAT wiersze

```
20: sudo iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 2222
    -j DNAT --to 10.0.0.2:22
21: sudo iptables -A FORWARD -i eth0 -o eth1 -p tcp -d 10.0.0.2
    --dport 22 -j ACCEPT
```

Pierwszy wiersz powoduje zamianę docelowego adresu IP na `10.0.0.2`, zaś drugi przepuszcza ruch przez router NAT pod warunkiem, że jest on skierowany do portu 22 stacji roboczej. (Reguły modułu `PREROUTING` tabeli `nat` są rozpatrywane wcześniej niż modułu `FORWARD`, więc adresy zostały już podmienione).

Sprawdź, że połączenie z serwerem SSH stacji roboczej jest możliwe zarówno w przypadku kiedy klientem jest jakiś router NAT jak i inna stacja robocze. W obu przypadkach polecenie powinno wyglądać następująco:

```
$> ssh -p 2222 IP_routera_NAT
```

gdzie *IP_routera_NAT* jest adresem karty `eth0` routera NAT sąsiadującego ze stacją roboczą z której usługą SSH chcemy się połączyć. Obejrzyj w Wiresharku przesyłane pakiety. Zobacz też, co zapisywane jest do pliku dziennika.

Lista i materiały znajdują się pod adresem http://www.ii.uni.wroc.pl/~mbi/dyd/

Marcin Bieńkowski