Kacper Szaruch
Jan Wojciechowski

Politechnika Warszawska

Sprawozdanie z realizacji laboratorium KRI nr 7
Segment Routing

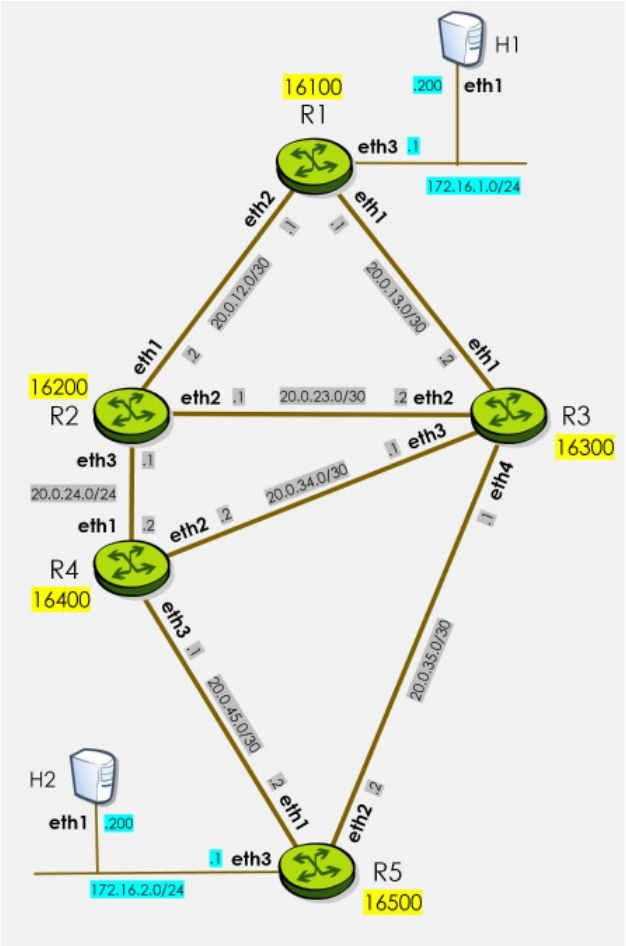
16 marca 2024

Spis treści

Wstęp

Niniejszy dokument to sprawozdanie z realizacji laboratorium w ramach przedmiotu KRI. Oświadczamy, że ta praca, stanowiąca podstawę do uznania osiągnięcia efektów uczenia się z przedmiotu KRI, została wykonana przez nas samodzielnie.

A: Przypisanie adresów IP



Rys. 1: Topologia sieci

Router	Address
R1	1.1.1.1/32
R2	2.2.2.2/32
R3	3.3.3.3/32
R4	4.4.4.4/32
R5	5.5.5.5/32

Rys. 2: Adresacja interfejsów loopback

Powyżej przedstawiona została topologia sieci oraz adresacja interfejsów loopback wykorzystywanych w ramach tego ćwiczenia laboratoryjnego.

B: Konfiguracja IS-IS

Podstawowa konfiguracja IS-IS został wykonana przez autorów laboratorium. Poniżej można zaobserwować, na jakich ścieżkach przesyłane są pakiety między hostami w sieci.

```
~/Labs/sr docker exec -it clab-sr-H1 bash
bash-5.1# ping 172.16.2.200
PING 172.16.2.200 (172.16.2.200): 56 data bytes
64 bytes from 172.16.2.200: seq=0 ttl=61 time=0.750 ms
64 bytes from 172.16.2.200: seq=1 ttl=61 time=0.117 ms
^C
--- 172.16.2.200 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.117/0.433/0.750 ms
bash-5.1# traceroute 172.16.2.200
traceroute to 172.16.2.200 (172.16.2.200), 30 hops max, 46 byte packets
 1  172.16.1.1 (172.16.1.1)  0.054 ms  0.009 ms  0.006 ms
 2  20.0.13.2 (20.0.13.2)  0.006 ms  0.010 ms  0.007 ms
 3  20.0.35.2 (20.0.35.2)  0.006 ms  0.008 ms  0.007 ms
 4  172.16.2.200 (172.16.2.200)  0.012 ms  0.012 ms  0.008 ms
bash-5.1#
```

Rys. 3: Wynik wykonania komendy *ping* i *traceroute* z **H1** na **H2**

```
~/Labs/sr docker exec -it clab-sr-H2 bash
bash-5.1# ping 172.16.1.200
PING 172.16.1.200 (172.16.1.200): 56 data bytes
64 bytes from 172.16.1.200: seq=0 ttl=61 time=0.160 ms
64 bytes from 172.16.1.200: seq=1 ttl=61 time=0.187 ms
^C
--- 172.16.1.200 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.160/0.173/0.187 ms
bash-5.1# traceroute 172.16.1.200
traceroute to 172.16.1.200 (172.16.1.200), 30 hops max, 46 byte packets
 1  172.16.2.1 (172.16.2.1)  0.012 ms  0.014 ms  0.008 ms
 2  20.0.35.1 (20.0.35.1)  0.008 ms  0.009 ms  0.006 ms
 3  20.0.13.1 (20.0.13.1)  0.005 ms  0.008 ms  0.007 ms
 4  172.16.1.200 (172.16.1.200)  0.006 ms  0.007 ms  0.006 ms
bash-5.1#
```

Rys. 4: Wynik wykonania komendy wykonania komendy *ping* i *traceroute* z **H2** na **H1**

C: Zadania do wykonania

C1: Konfiguracja Segment Routingu

Zadanie polega na skonfigurowaniu sieci przy pomocy przygotowanej przez autorów zadania przykładowej konfiguracji, a następnie sprawdzeniu, czy połączenie między hostami w sieci jest możliwe.

```
~/Labs/sr docker exec -it clab-sr-H1 bash
bash-5.1# ping 172.16.2.200
PING 172.16.2.200 (172.16.2.200): 56 data bytes
64 bytes from 172.16.2.200: seq=0 ttl=61 time=0.172 ms
64 bytes from 172.16.2.200: seq=1 ttl=61 time=0.201 ms
^C
--- 172.16.2.200 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.172/0.186/0.201 ms
```

Rys. 5: Wynik wykonania komendy *ping* z **H1** na **H2**

```
~/Labs/sr docker exec -it clab-sr-H2 bash
bash-5.1# ping 172.16.1.200
PING 172.16.1.200 (172.16.1.200): 56 data bytes
64 bytes from 172.16.1.200: seq=0 ttl=61 time=0.180 ms
64 bytes from 172.16.1.200: seq=1 ttl=61 time=0.119 ms
^C
--- 172.16.1.200 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.119/0.149/0.180 ms
```

Rys. 6: Wynik wykonania komendy *ping* z **H1** na **H2**

Na powyższy zdjęciach można zauważyć, że komunikacja jest między hostami w sieci jest możliwa.

C2: Segment Routing - Traffic Engineering

Polecenie polegało na skonfigurowaniu sieci w taki sposób, aby ruch pomiędzy hostami odbywał się na drodze przez **R2** → **R3** → **R4**. W tym celu wykorzystano komendę umożliwiającą wymuszenie ścieżki pakietów w routerze źródłowym przy użyciu etykiet MPLS.

ip route add A.B.C.D/MASK encap mpls Label1/Label2/.. via E.F.G.H

```
bash-5.1# ping 172.16.2.200
PING 172.16.2.200 (172.16.2.200): 56 data bytes
64 bytes from 172.16.2.200: seq=0 ttl=61 time=0.217 ms
64 bytes from 172.16.2.200: seq=1 ttl=61 time=0.216 ms
64 bytes from 172.16.2.200: seq=2 ttl=61 time=0.276 ms
64 bytes from 172.16.2.200: seq=3 ttl=61 time=0.261 ms
64 bytes from 172.16.2.200: seq=4 ttl=61 time=0.229 ms
^C
--- 172.16.2.200 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0.216/0.239/0.276 ms
```

Rys. 7: Wynik wykonania komendy *ping* z **H1** na **H2**

```
bash-5.1# traceroute 172.16.2.200
traceroute to 172.16.2.200 (172.16.2.200), 30 hops max, 46 byte packets
 1  172.16.1.1 (172.16.1.1)  0.010 ms  0.011 ms  0.007 ms
 2  *  *  *
 3  20.0.45.2 (20.0.45.2)  0.056 ms  0.009 ms  0.007 ms
 4  172.16.2.200 (172.16.2.200)  0.011 ms  0.009 ms  0.006 ms
bash-5.1#
```

Rys. 8: Wynik wykonania komendy *traceroute* z **H1** na **H2**

Jak można zauważyć na powyższych zdjęciach komunikacja między hostami w sieci jest dalej możliwa. Można też zaobserwować jak wygląda wynik komendy *traceroute* w przypadku kiedy w obu kierunkach została wymuszona konkretna ścieżka. Powstaje wtedy znana z poprzedniego laboratorium "dziurę". W momencie, kiedy ścieżka była by ustawiona tylko w jednym kierunku to ruch powrotny odbywał by się po najkrótszej możliwej ścieżce tak jak miało to miejsce na początku. Zachowanie takie jest domyślne dla **MPLS TE**.

```
▸ Frame 7: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface -, id 0
▸ Ethernet II, Src: aa:c1:ab:fe:eb:2c (aa:c1:ab:fe:eb:2c), Dst: aa:c1:ab:3d:df:f2 (aa:c1:ab:3d:df:f2)
▸ MultiProtocol Label Switching Header, Label: 16400, Exp: 0, S: 0, TTL: 63
▸ MultiProtocol Label Switching Header, Label: 16500, Exp: 0, S: 1, TTL: 63
▸ Internet Protocol Version 4, Src: 172.16.1.200, Dst: 172.16.2.200
▸ Internet Control Message Protocol
```

Rys. 9: Pakiet zaobserwowany na łączu **R2** → **R3**

.

Ruch po wymuszonej ścieżce odbywa się poprzez dodanie do pakietu stosu odpowiednich etykiet. Kiedy router odbiera pakiet sprawdza najwyższą etykietę, a następnie przesyła pakiet na podstawie odczytanej etykiety. Na powyższym przykładzie pakietu, który został zaobserwowany na łączu **R2** → **R3** można zauważyć jak wygląda taki stos etykiet dołączonych do pakietu. Jak widać w momencie, kiedy któraś etykieta zostanie już sprawdzona nie jest ona dołączana ponownie do pakietu, stąd w powyższym pakiecie brak etykiety o numerze 16300.