

ROUTING DYNAMICZNY

...a Linux

Łukasz Bromirski

lukasz@bromirski.net



Agenda

- **Powtórka z rozrywki: podstawy routingu IP**
- **Skąd wziąć interesujące elementy**
- **Protokoły routingu – IGP i EGP**
- **Inne zagadnienia**
- **Q & A**

POWTÓRKA Z ROZRYWKI: ROUTING IP



Routing IP

O czym mówimy?

- Routing IP to decyzja (standardowo) podejmowana na podstawie adresu **docelowego** pakietu IP
- Kernel podejmuje tą decyzję na podstawie tablicy **FIB – Forwarding Information Base**
- Aplikacje zapewniające routing dynamiczny utrzymują zwykle swoją tablicę – **RIB – Routing Information Base** – z której najlepsze wpisy eksportowane są do FIB

Routing IP

O czym mówimy?

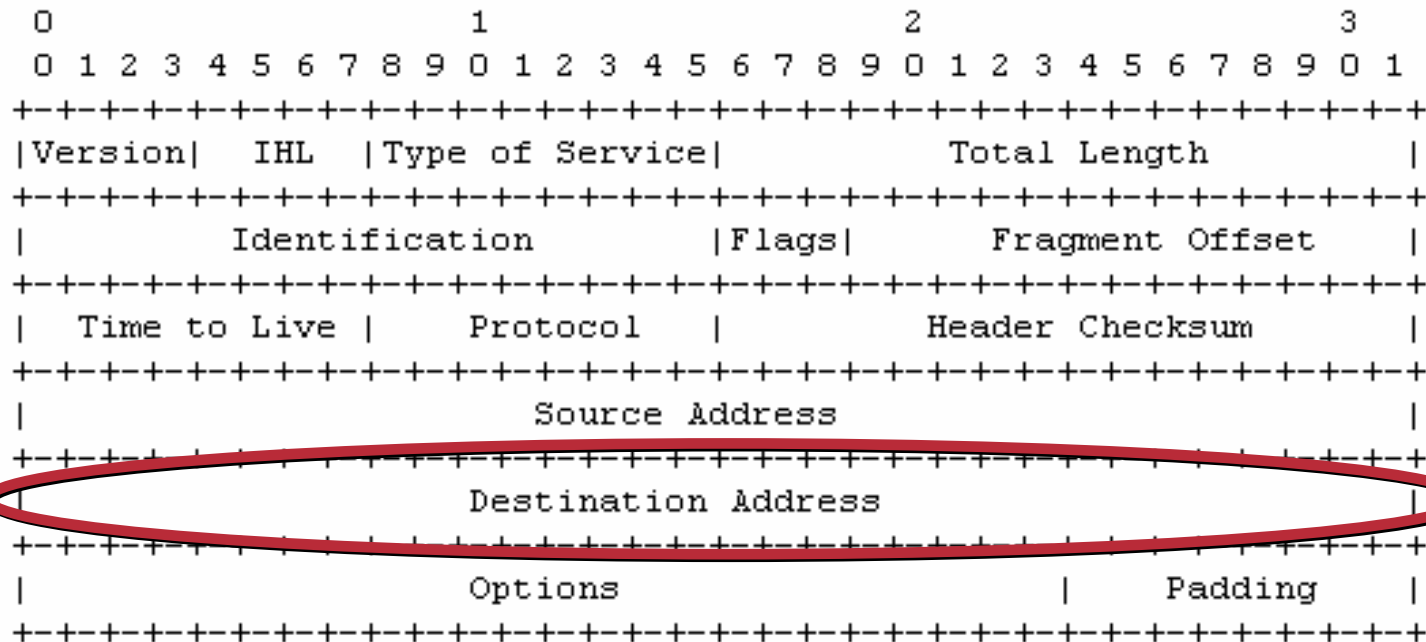
- Narzędzia systemowe wpływają na **FIB**
opcje **FIB_HASH/FIB_TRIE** w kernelach 2.6.x
- Narzędzia konkretnej aplikacji wpływają na **RIB**
właściwy dla pakietu
- Dodatkowo Linux posiada bogate opcje routingu
na podstawie adresu źródłowego
pakiet iproute2
integracja z aplikacjami zewnętrznymi – tablice, realms

Realms: <http://vcalinus.gemeni.ro/quaggarealms.html>

Routing IP

Czym zajmuje się router?

- Router otrzymuje datagramy IPv4 w postaci:



RFC 791, <http://www.ietf.org/rfc/rfc0791.txt>

Routing IP

24x7x365...

- **Router w dużym uproszczeniu cały czas wykonuje następującą pętlę:**

odbiera pakiet

jeśli nie TTL=1, adres docelowy=adres mojego interfejsu lub [...]

sprawdź, na jaki interfejs wskazuje w tablicy routingu wpis dla adresu docelowego z pakietu

jeśli wpis zawiera inny adres, rozwiąż go na prawidłowy adres następnej bramy

zmniejsz TTL o 1

wstaw pakiet do bufora wyjściowego interfejsu, który wybrałeś po znalezieniu w tablicy routingu najdokładniejszego wpisu

odbiera pakiet

...

Routing IP

Budowa FIB

- Tablica routingu (FIB) zawiera wpisy pochodzące z wielu źródeł, ale w znormalizowanej postaci
- W ogólnodostępnych, wolnych systemach operacyjnych pola obecne dla każdego wpisu to między innymi:
 - destination** – sieć lub host docelowy
 - gateway** – przez jakie next-hop IP osiągaln(a/y)
 - flags** – dodatkowe atrybuty trasy
 - use** – ile razy użyto trasy
 - if** – przez który interfejs pakiet zostanie wysłany

Routing IP

Budowa FIB

- Zawartość FIB:

```
[me@slack ~]$ ip route list
```

```
10.0.0.0/24          dev eth0  proto kernel src 10.0.0.100
169.254.0.0/16     dev eth0  scope link
default via 10.0.0.1 dev eth0
```

```
[me@slack ~]$ route -n
```

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	If
10.0.0.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
169.254.0.0	0.0.0.0	255.255.0.0	U	0	0	0	eth0
0.0.0.0	10.0.0.1	0.0.0.0	UG	0	0	0	eth0

Routing IP

Droga trasy do FIB (z 10,624 metrów)

- Protokół routingu dynamicznego poznaje trasę
- Protokół routingu dynamicznego w wyniku działania swoich algorytmów ustala, że jest to trasa najlepsza i umieszcza ją (je) w **RIB**
- Demon odpowiedzialny za interakcję z kernelem, eksportuje najlepsze trasy **z RIB** (mogą pochodzić z różnych protokołów) **do systemowego FIB**

**QUAGGA/ZEBRA vs
XORP vs
OpenOSPFd**



Quagga/Zebra (0.99.3/0.98.5)

- Quagga posiada budowę modułarną
- Proces **zebra** odpowiada za interakcje wszystkich pozostałych z kernelem (FIB) i zarządzanie RIB
- Osobne procesy odpowiedzialne za protokoły routingu
 - ripd (v1/v2), ripngd (v3 dla IPv6)
 - ospfd (v2), ospf6d (v3 dla IPv6)
 - bgpd (v4+)
 - is-is*
- Dostępne narzędzie **vtysh** do zarządzania „wszystkim jednocześnie”
- Quagga jest przygotowana do przechowywania wielu takich samych tras w RIB:

```
configure [...] --enable-multipath=X
```

XORP

(1.2)

- XORP również posiada budowę modułarną
- Router manager (**rtrmngrr**) nadzoruje pracę grupy procesów
- Dwie osobne ścieżki:
 - unicast: BGP4, RIPv1/2 i RIPvng, OSPFv2
 - multicast: PIM-SM, IGMPv1/v2, MLDv1
- Wydzielony RIB dla wszystkich protokołów
- Wydzielona FEA, pozwalająca uniezależnić się od systemu/dostępnych interfejsów
- Dostępna powłoka **xorpsd** do zarządzania
- Wiele rozmaitych problemów

OpenOSPFd

- Projekt zespołu OpenBSD

Henning Brauer, Claudio Jeker & Esen Norby

- Projekt w trakcie dopracowywania

IPv6

problemy z redystrybucją/wstrzyknięciem trasy default

- Tradycyjny zestaw narzędzi i plików:

`ospfd` – demon odpowiedzialny za protokół

`ospfctl` – narzędzie do kontroli

`/etc/ospfd.conf` - konfiguracja

OpenBGPd

- Ten sam team zespołu OpenBSD

Henning Brauer, Claudio Jeker & Esen Norby i inni

- Projekt w trakcie dopracowywania

IPv6 + różne zagadnienia z
filtrowaniem/utrzymywaniem atrybutów tras

- Podobnie jak OpenOSPFd:

bgpd – demon odpowiedzialny za protokół

bgpctl – narzędzie do kontroli

/etc/bgpd.conf - konfiguracja

Podsumowanie obecnego stanu

- **Wiele możliwości, potęgowanych możliwościami pakietu iproute2**
- **Nadal brak jasnej wizji dotyczącej zarządzaniem przez kernel różnymi źródłami informacji routingowej**
- **Ten sam problem mają wszystkie dystrybucje Linuksa oraz Free/Net/OpenBSD**

PROTOKÓŁ ROUTINGU RIP



Protokół routingu RIP

Routing Information Protocol

- **Routery wymieniają się swoimi tablicami routingu co określone odstępy czasu**

RIP standardowo co 30 sekund (z małymi różnicami)

- **Metryką trasy w protokole RIP jest ilość hopów, jaką musi pokonać pakiet, by dotrzeć do sieci/hosta**

0 = ja, 1-14 = trasa prawidłowa

15 = trasa nieosiągalna

- **Tylko RIPv2 przesyła maskę dla trasy**

...dodatkowo umożliwia uwierzytelnianie (MD5)

Protokół routingu RIP

Routing Information Protocol

```
ripd# show ip rip status
```

```
Routing Protocol is "rip"
```

```
  Sending updates every 30 seconds with +/-50%, next due in 6 seconds
```

```
  Timeout after 180 seconds, garbage collect after 120 seconds
```

```
  Outgoing update filter list for all interface is not set
```

```
  Incoming update filter list for all interface is not set
```

```
  Default redistribution metric is 1
```

```
  Redistributing:
```

```
  Default version control: send version 2, receive version 2
```

Interface	Send	Recv	Key-chain
eth0	2	2	
lo0	2	2	

```
[...]
```

Protokół routingu RIP

Routing Information Protocol

```
ripd# show ip rip status
```

```
[...]
```

```
Routing for Networks:
```

```
172.16.91.0/24
```

```
192.168.0.0/24
```

```
Routing Information Sources:
```

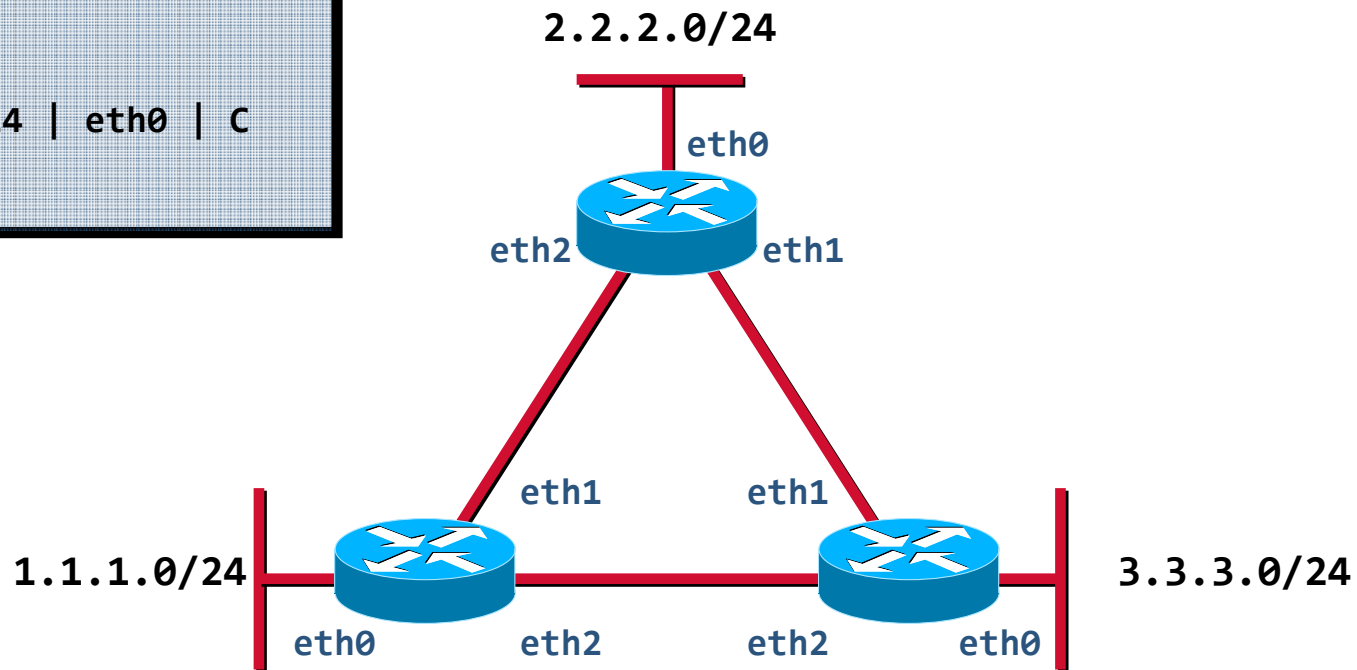
```
Gateway          BadPackets BadRoutes  Distance Last Update
```

```
Distance: (default is 120)
```

Protokół routingu RIP

Jak działa?

Router B:
2.2.2.0/24 | eth0 | C

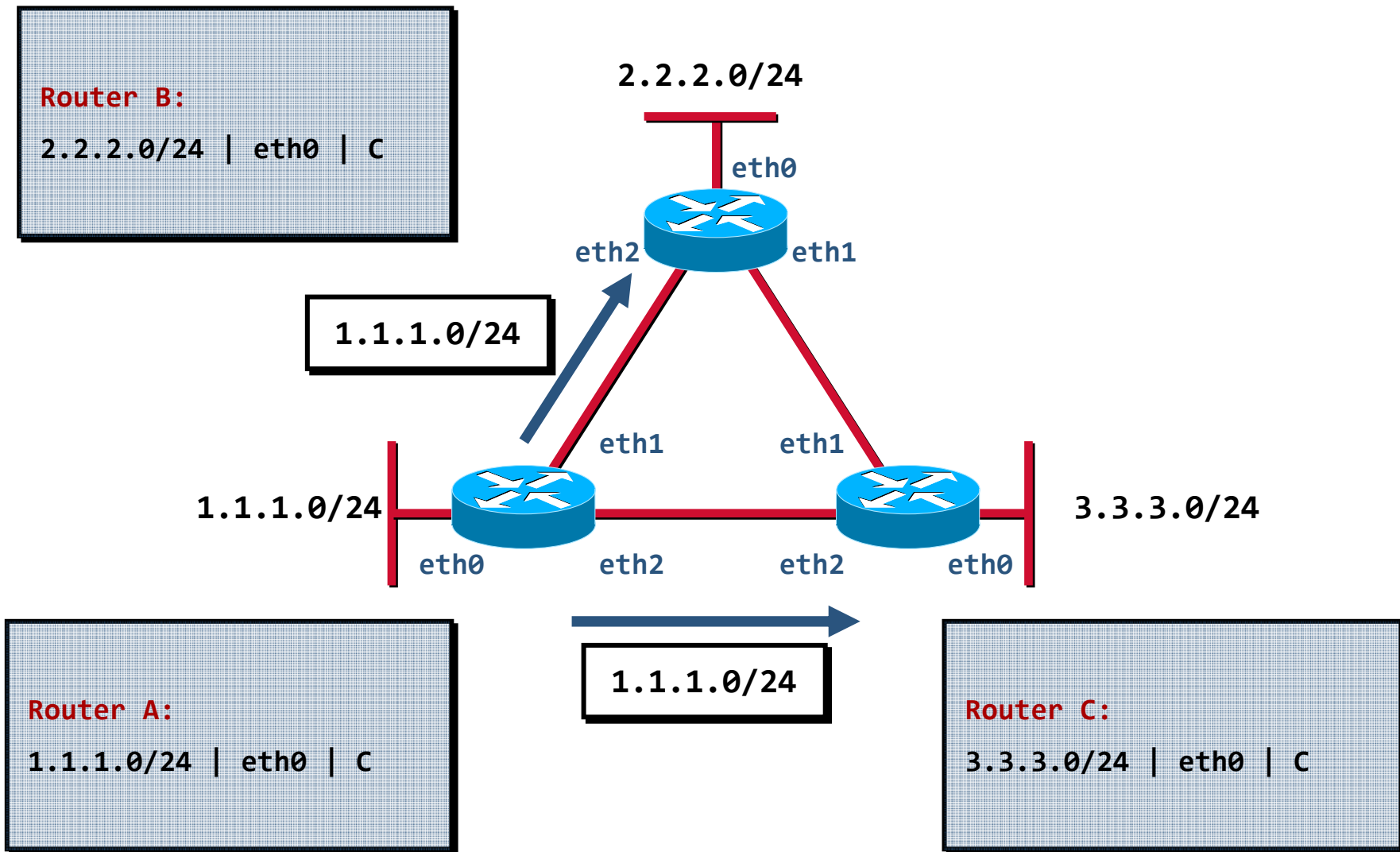


Router A:
1.1.1.0/24 | eth0 | C

Router C:
3.3.3.0/24 | eth0 | C

Protokół routingu RIP

Jak działa?



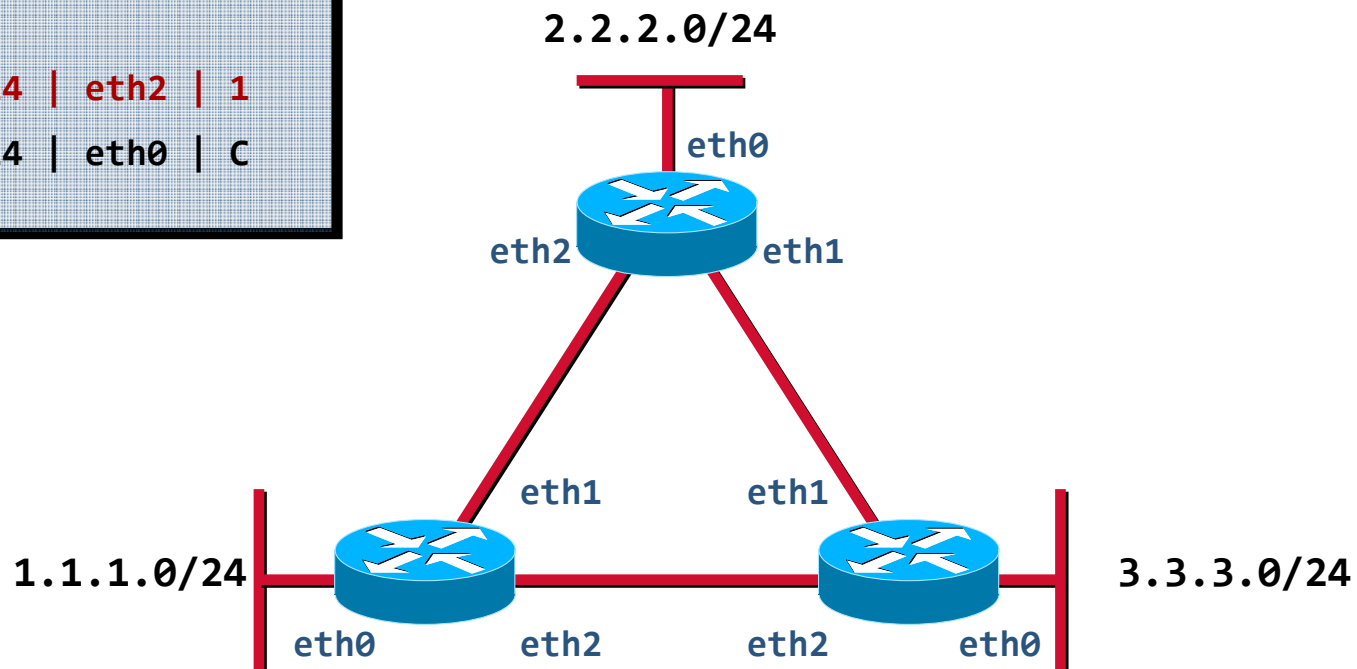
Protokół routingu RIP

Jak działa?

Router B:

1.1.1.0/24 | eth2 | 1

2.2.2.0/24 | eth0 | C



Router A:

1.1.1.0/24 | eth0 | C

Router C:

1.1.1.0/24 | eth2 | 1

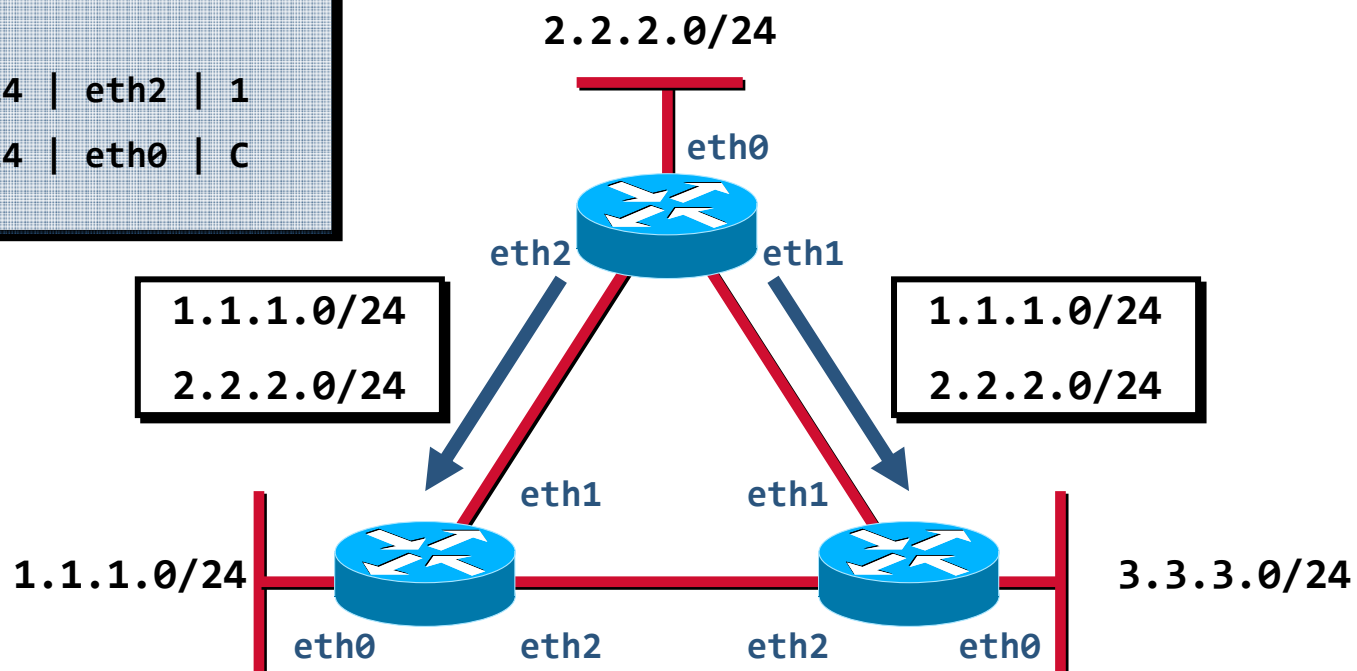
3.3.3.0/24 | eth0 | C

Protokół routingu RIP

Jak działa?

Router B:

```
1.1.1.0/24 | eth2 | 1  
2.2.2.0/24 | eth0 | C
```



Router A:

```
1.1.1.0/24 | eth0 | C
```

Router C:

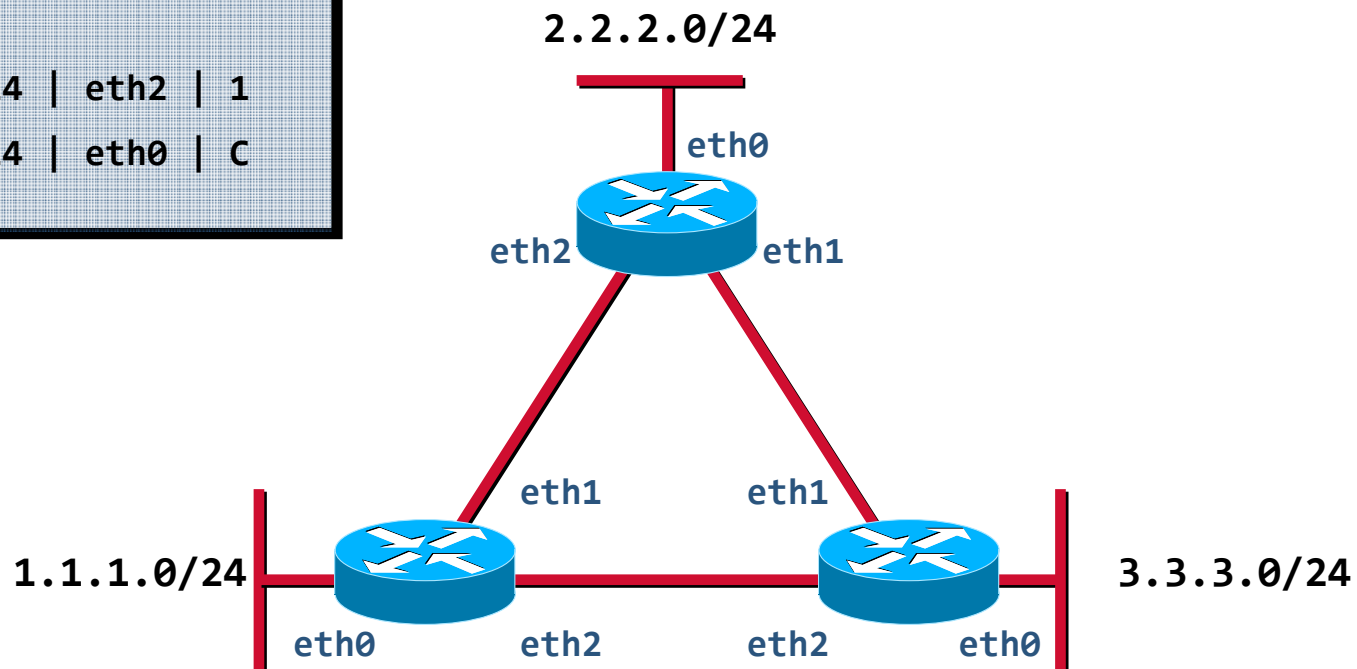
```
1.1.1.0/24 | eth2 | 1  
3.3.3.0/24 | eth0 | C
```


Protokół routingu RIP

Jak działa?

Router B:

```
1.1.1.0/24 | eth2 | 1  
2.2.2.0/24 | eth0 | C
```



Router A:

```
1.1.1.0/24 | eth0 | C  
2.2.2.0/24 | eth1 | 1
```

Router C:

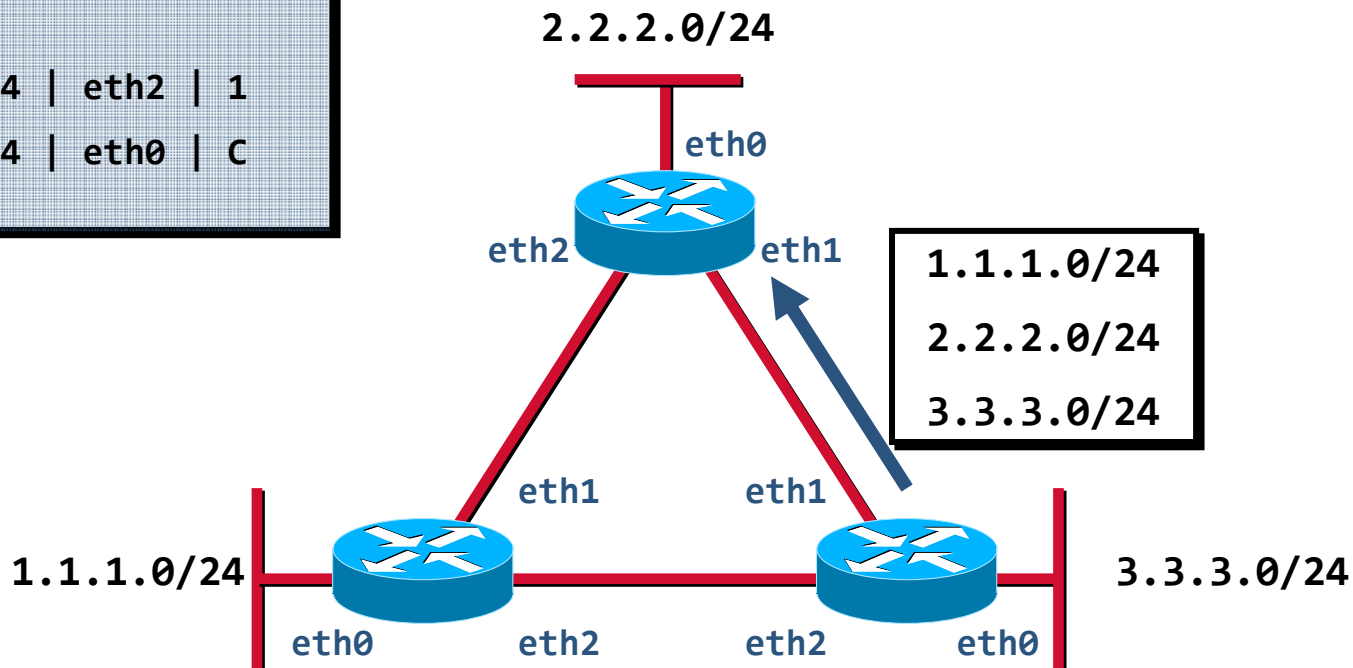
```
1.1.1.0/24 | eth2 | 1  
2.2.2.0/24 | eth1 | 1  
3.3.3.0/24 | eth0 | C
```

Protokół routingu RIP

Jak działa?

Router B:

```
1.1.1.0/24 | eth2 | 1  
2.2.2.0/24 | eth0 | C
```



Router A:

```
1.1.1.0/24 | eth0 | C  
2.2.2.0/24 | eth1 | 1
```

```
1.1.1.0/24  
2.2.2.0/24  
3.3.3.0/24
```

Router C:

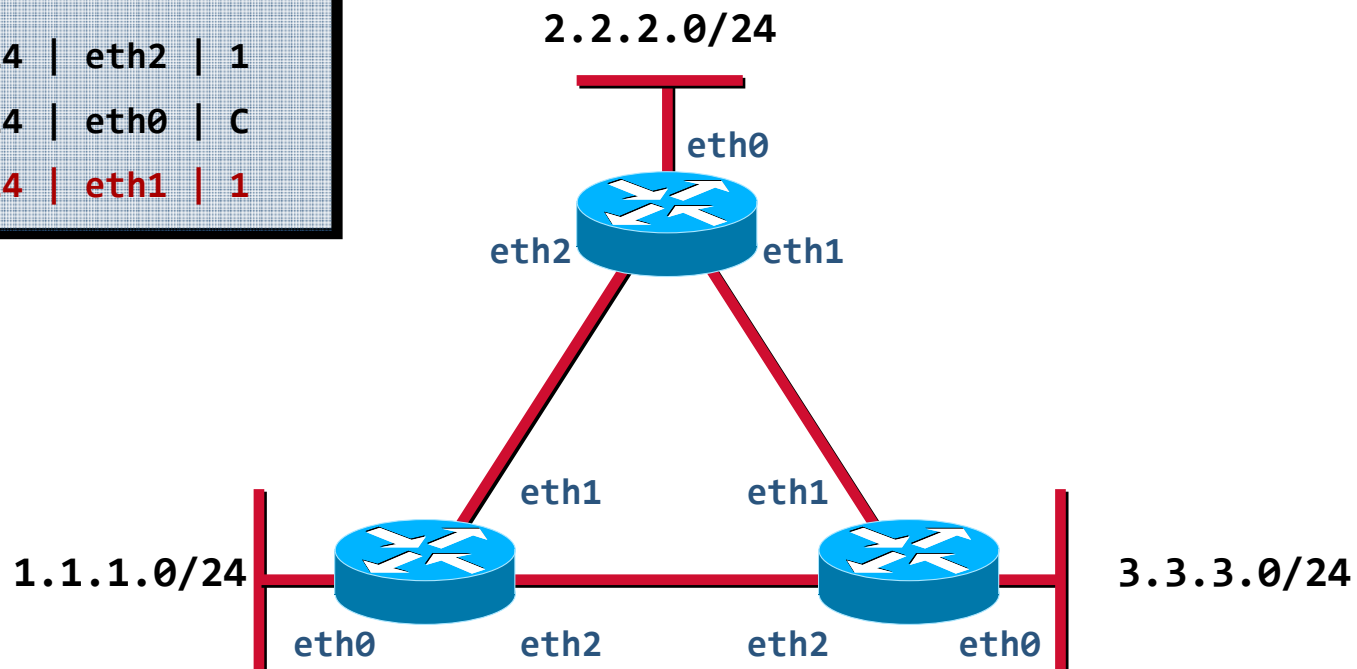
```
1.1.1.0/24 | eth2 | 1  
2.2.2.0/24 | eth1 | 1  
3.3.3.0/24 | eth0 | C
```

Protokół routingu RIP

Jak działa?

Router B:

1.1.1.0/24	eth2	1
2.2.2.0/24	eth0	C
3.3.3.0/24	eth1	1



Router A:

1.1.1.0/24	eth0	C
2.2.2.0/24	eth1	1
3.3.3.0/24	eth2	1

Router C:

1.1.1.0/24	eth2	1
2.2.2.0/24	eth1	1
3.3.3.0/24	eth0	C

Protokół routingu RIP

Kiedy zastosować?

- Najlepiej **nie stosować** – są lepsze, efektywniejsze i o większych możliwościach również dostępne na licencjach BSD/GPL/etc.
- Czasami wymaga tego obecność prostej „zamkniętej” bramki w sieci
 - zwykle obsługują tylko RIP
 - uwaga na problemy ze zgodnością
 - na większość da się załadować Linuxa/BSD....☺
- Doskonały protokół routingu do nauczenia się, jak działają protokoły routingu
 - bardzo (lub relatywnie) prosty debugging

PROTOKÓŁ ROUTINGU OSPF



Protokół OSPF

Historia

- Tworzony od 1987 przez IETF
- OSPFv2 opublikowana w RFC 1247 w 1991
- Ostatnia wersja dla IPv4 - OSPFv2 – RFC2328
- Wersja obsługująca IPv6 – OSPFv3 – RFC2740
- Metryką jest koszt trasy
 - 100Mbit/s = 10
 - 1Gbit/s = 1
- Szybka konwergencja i relatywnie małe wymagania
 - hierarchiczny podział na obszary

Protokół routingu OSPF

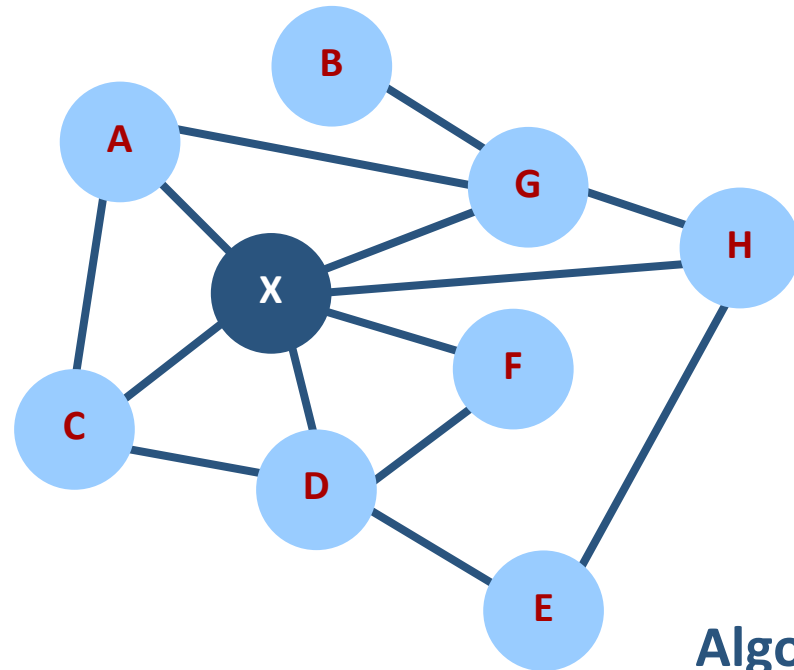
Jak działa?

- **OSPF posługuje się hierarchiczną strukturą sieci:**
 - obszar backbone (area 0)
 - obszary podłączone różnych typów
- **Każdy z obszarów musi być połączony do obszaru 0**
 - jeśli nie może – przez link wirtualny
- **Routery identyfikowane są za pomocą **router-id****
 - najwyższy adres IP ze wszystkich interfejsów
 - pierwszeństwo mają interfejsy loopback – użyj ich!

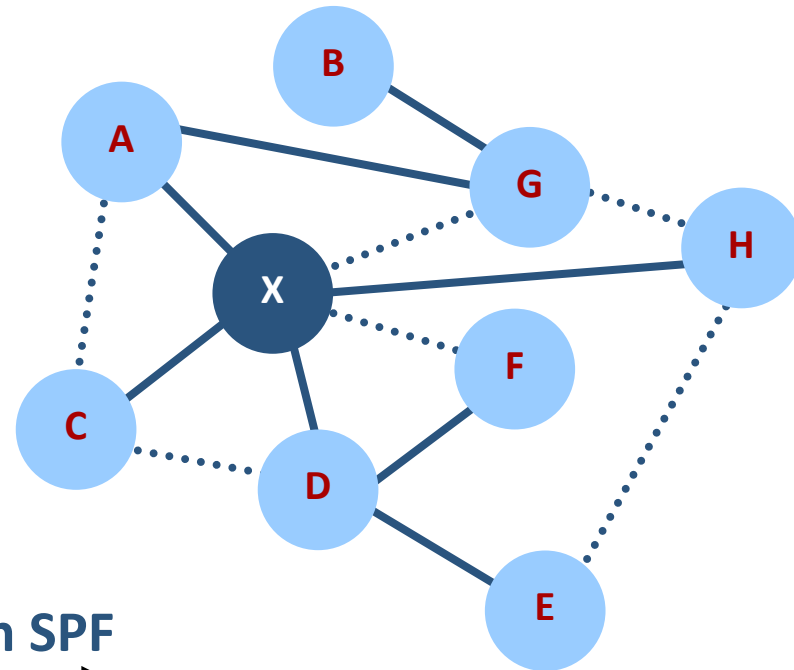
Protokół routingu OSPF

Jak działa?

Baza topologii (łącze-stan)



Drzewo najkrótszych ścieżek



Algorytm SPF



Protokół routingu OSPF

Jak działa?

- Routery wymieniają się **LSA** – Link State Advertisement
- W zależności od obszaru i roli routera, zestaw LSA może być różny

Typ	Nazwa LSA
1	Router
2	Network
3	Summary Network
4	Summary ASBR
5	External
7	NSSA

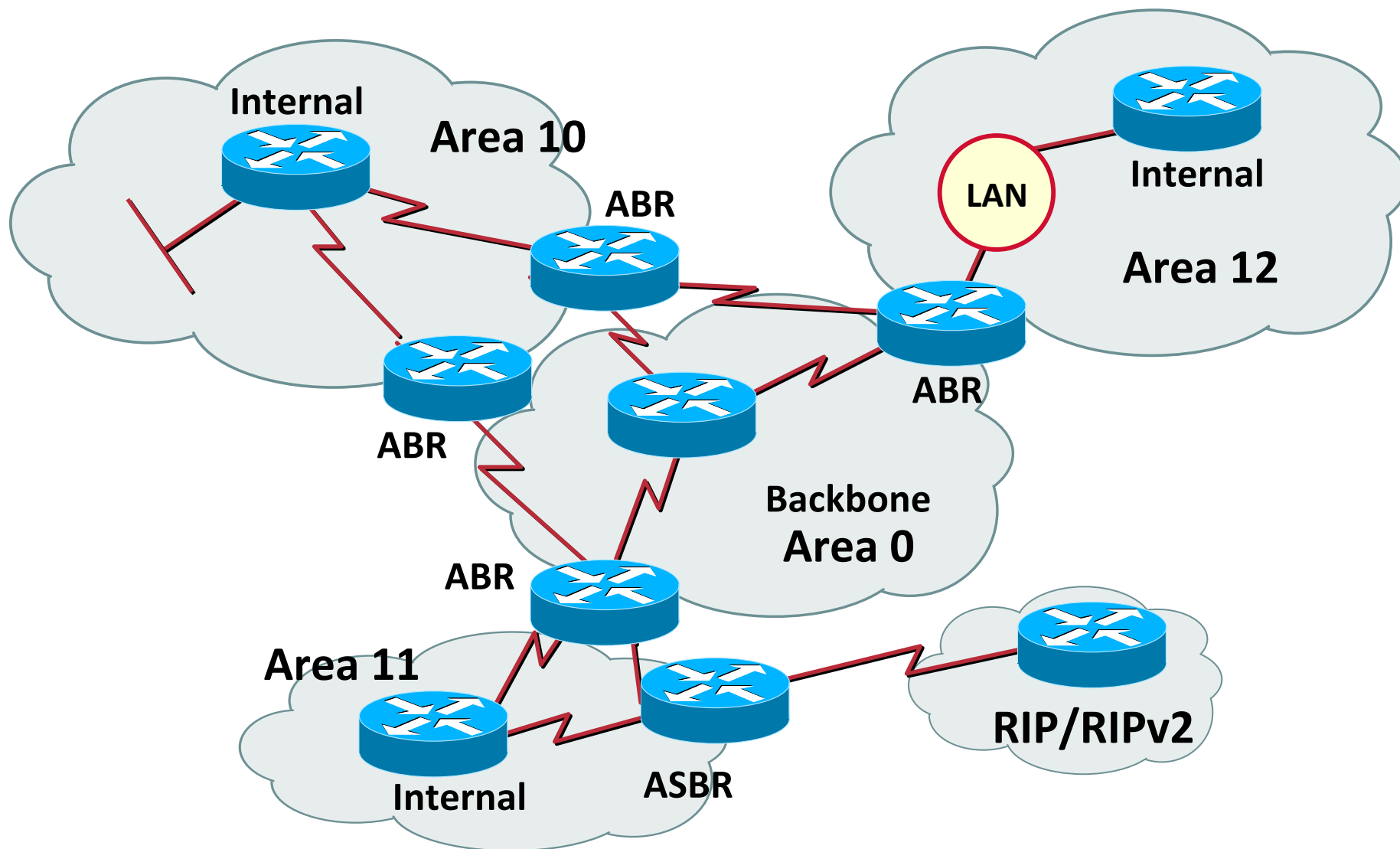
Protokół routingu OSPF

Algorytm SPF

- Router utrzymuje osobną bazę topologii dla każdego obszaru, do którego należy
- Routery w tym samym obszarze posiadają **tą samą** bazę topologii
- SPF działa osobno dla każdego obszaru
- Flooding LSA odbywa się tylko w granicach obszaru

Protokół routingu OSPF

Przykład topologii



Protokół OSPF

Rodzaje łączy

- **Broadcast**

np. Ethernet

wybór DR i BDR dla segmentu

- **Non-Broadcast Multi-Access (NBMA)**

np. Frame Relay

konieczność wskazania sąsiadów poleceniem neighbour

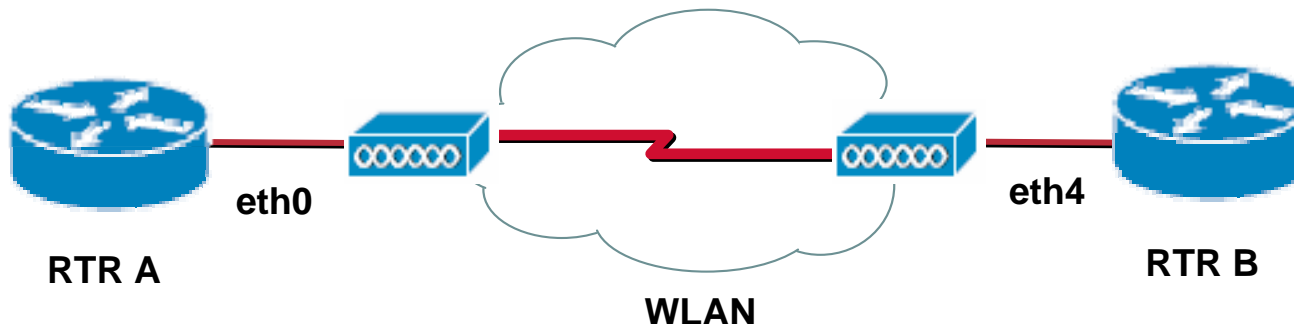
- **Point-to-Point, Point-to-Multipoint, Point-to-Multipoint Non-Broadcast**

Protokół OSPF

Kiedy point-to-point?

- Zalecane w przypadku sieci używających jako warstwy transportowej 802.11:

oszczędzamy czas potrzebny na elekcję DR/BDR



UWAGA: koszt trasy przez AP jest równy wynegocjowanej z AP prędkości na interfejsie – zwykle 100Mbit/s (koszt 10). Jeśli posiadasz inne, równoległe i szybsze łącze warto zastanowić się nad zmianą (zwiększeniem) kosztu przez ten interfejs!

Protokół OSPF

Konfiguracja point-to-point

```
interface eth1
  ip ospf network point-to-point
  ip ospf cost 20 ! zawyżenie kosztu do odpowiadającego łączy 50Mbit/s
                  ! aby inne łączy o realnej przepustowości 100Mbit/s
                  ! było atrakcyjniejsze
q-ospfd# show ip ospf interface eth1
eth1 is up
  Internet Address 192.168.50.1/30, Broadcast 192.168.50.3, Area 0.0.0.50
  Router ID 172.16.254.10, Network Type POINTOPOINT, Cost: 20
  Transmit Delay is 1 sec, State Point-To-Point, Priority 1
  No designated router on this network
  No backup designated router on this network
  [...]
  Neighbor Count is 1, Adjacent neighbor count is 1
```

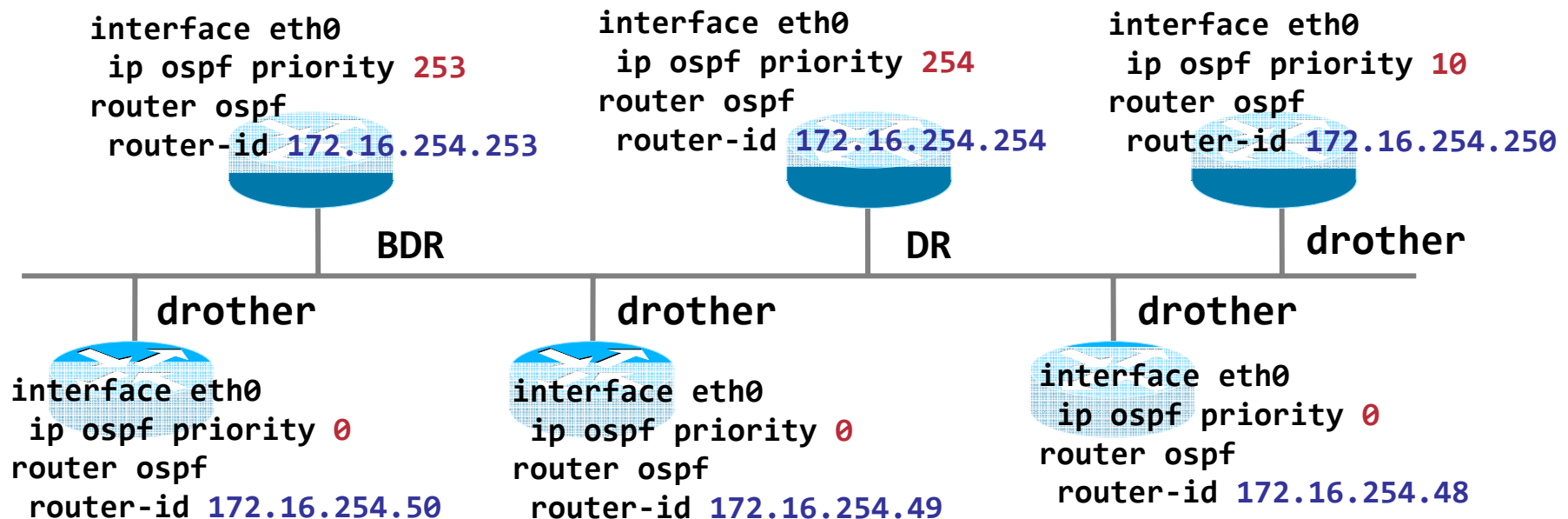
Protokół OSPF

Kiedy broadcast?

- We wszystkich topologiach, w których wiele routerów łączy wspólny segment Ethernet

priority routera (0-254, 254 najwyższy, 0 – nie zostanie DR)

wyższe router-id (zalecane stabilne rozplanowanie numeracji interfejsów loopback!)



Protokół OSPF

Konfiguracja broadcast

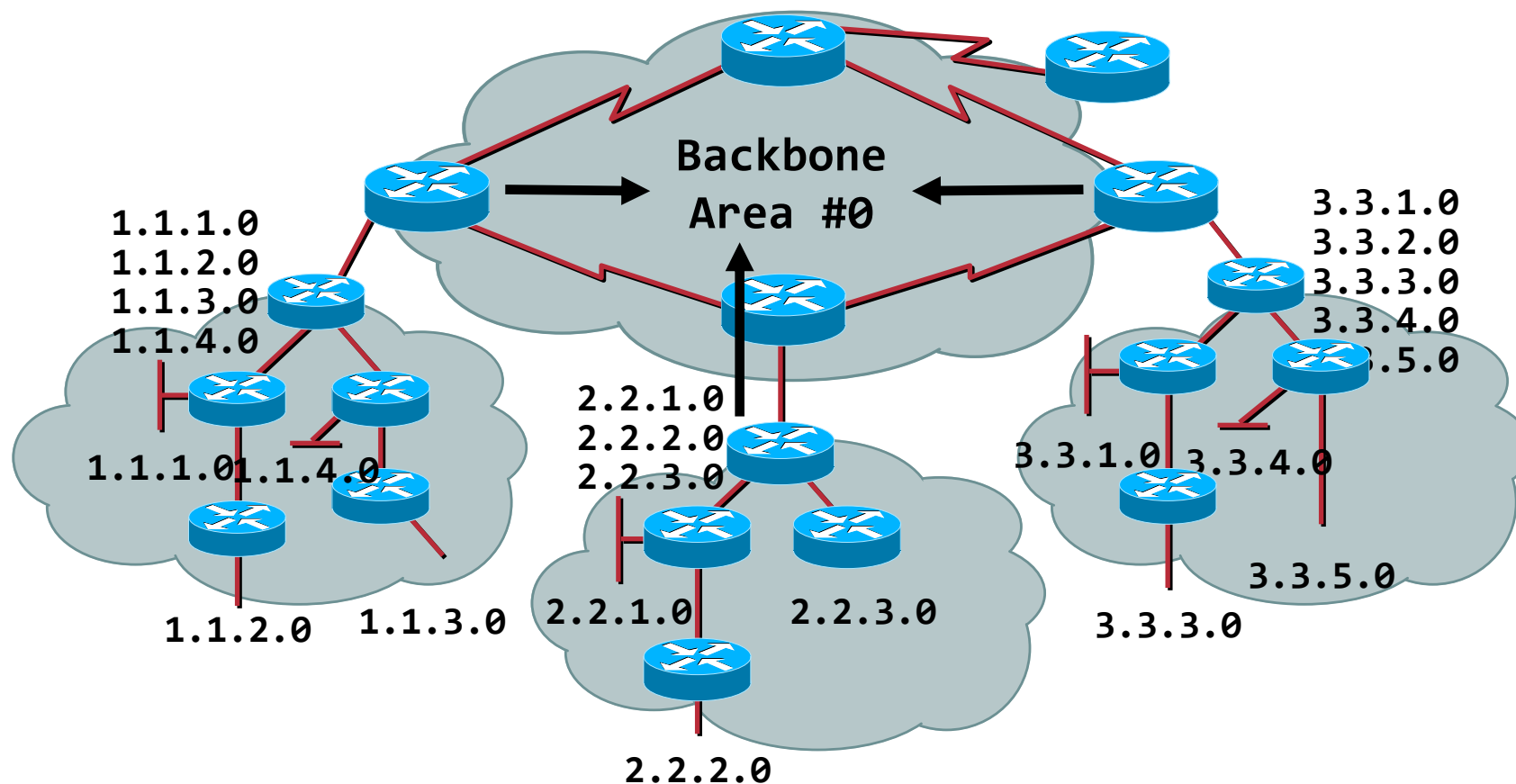
```
interface eth0
  ip ospf network broadcast ! domyślnie dla interfejsów Ethernet

q-ospfd# show ip ospf interface eth0

eth0 is up
[...]
Internet Address 192.168.0.100/24, Broadcast 192.168.0.255, Area 0.0.0.0
Router ID 172.16.254.10, Network Type BROADCAST, Cost: 10
Transmit Delay is 1 sec, State DROther, Priority 1
Designated Router (ID) 172.16.254.201, Interface Address 192.168.0.201
Backup Designated Router (ID) 172.16.254.200, Interface Address 192.168.0.200
Multicast group memberships: OSPFAllRouters
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
  Hello due in 00:00:04
Neighbor Count is 2, Adjacent neighbor count is 2
```


Protokół routingu OSPF

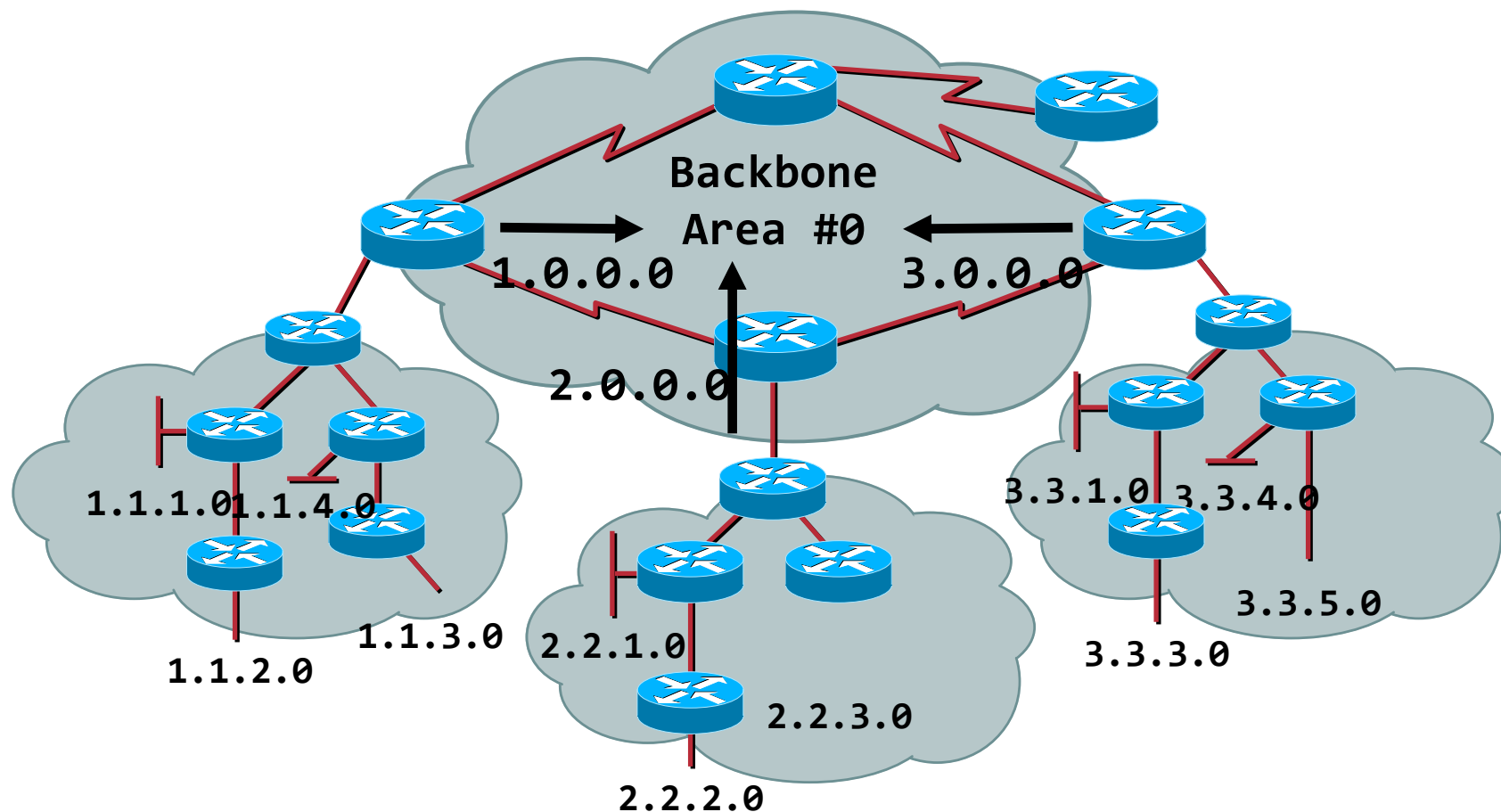
Sumaryzacja - brak



- Wiele prefiksów w Area 0
- Dodatkowe obciążenie dla algorytmu SPF i routerów backbone

Protokół routingu OSPF

Sumaryzacja - poprawnie



- Rozgłaszamy tylko summary LSA
- Zmiany stanów łącz nie propagują się pomiędzy obszarami

Protokół routingu OSPF

Sumaryzacja - konfiguracja

```
router ospf
```

```
ospf router-id 172.16.254.11
```

```
network 192.168.0.0/24 area 0.0.0.0
```

```
network 192.168.50.0/30 area 0.0.0.50
```

```
area 0.0.0.50 range 172.16.10.0/24
```

Protokół routingu OSPF

Scenariusz #1 – OSPF i wstrzyknięcie trasy domyślnej - Quagga

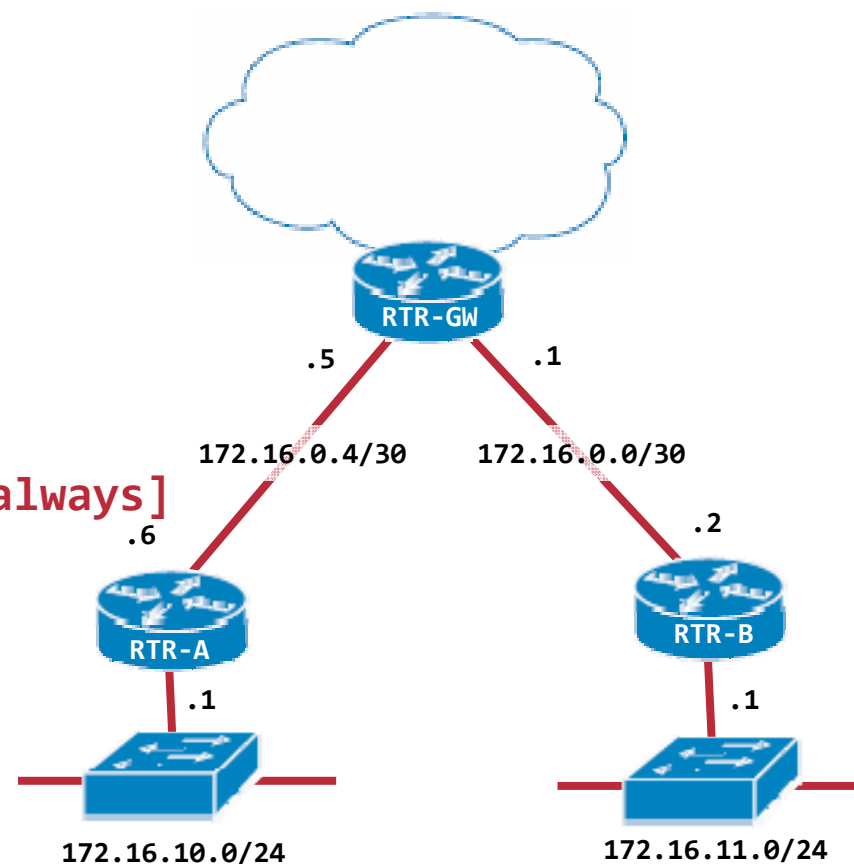
- Na RTR-GW:
rozgłosić default

```
router ospf
```

```
network 172.16.0.0/30 area 10
```

```
network 172.16.0.8/30 area 11
```

```
default-information originate [always]
```



Protokół routingu OSPF

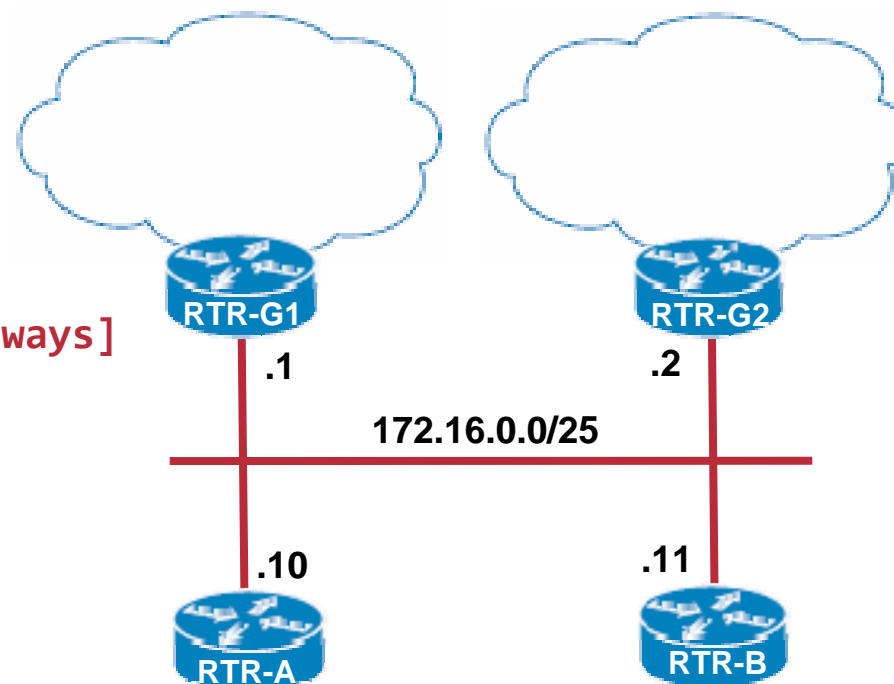
Scenariusz #2 – dwa OSPF ASBR

- Na RTR-G1:
rozgłosić default

```
router ospf  
network 172.16.0.0/25 area 0  
default-information originate [always]
```

- Na RTR-G2:
rozgłosić default

```
router ospf  
network 172.16.0.0/25 area 0  
default-information originate [always]
```



Protokół routingu OSPF

Scenariusz #2 – dwa OSPF ASBR

- Na RTR-A: load-balancing w oparciu o dwie trasy
- Uwaga na uRPF po stronie ISP!

```
rtra-ospfd# show ip ospf route
[...]
===== OSPF external routing table =====
N E2 0.0.0.0/0                [20/10] tag: 10
                                via 172.16.0.1, eth0
                                via 172.16.0.2, eth0
```

UWAGA: wiele tras o tej samej metryce pojawi się tylko wtedy, gdy Quagga została skompilowana z opcją **multipath**

Protokół OSPF

Problem redundancji first-hop

- **VRRP – standard (RFC3768)**

VRRPd: <http://off.net/~jme/vrrpd/>

- **UCARP – port CARP z OpenBSD**

<http://www.ucarp.org/project/ucarp>

- **Integracja aplikacji i systemu operacyjnego:**

LVS: <http://www.linuxvirtualserver.org/>

ct_sync (netfilter) + keepalived + LVS/coś innego:

<http://svn.netfilter.org/cgi-bin/viewcvs.cgi/trunk/netfilter-ha/>

http://www.netfilter.org/projects/libnetfilter_conntrack/index.html

PROTOKÓŁ ROUTINGU BGP



Protokół BGP

Wstęp i rozwinięcie

- **Protokół routingu używany do wymiany informacji o osiągalności sieci (prefiksów) pomiędzy systemami autonomicznymi**

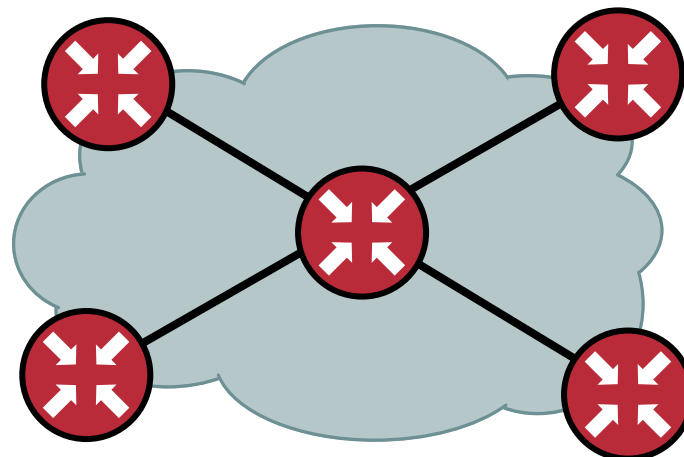
klasy EGP – Exterior Gateway Protocol

- **Do niedawna RFC1771 + rozszerzenia, od stycznia 2006 obowiązuje RFC4271:**

<http://www.ietf.org/rfc/rfc4271.txt>

Protokół BGP

System Autonomiczny

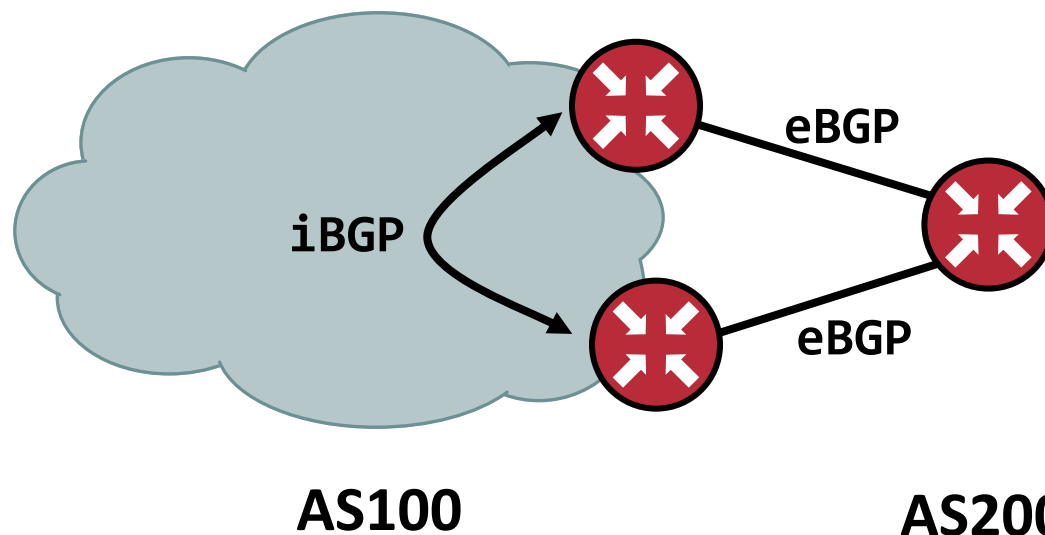


AS100

- Zwykle jedna firma/organizacja
- Zarządzana przez jedną grupę ludzi
- Jedna polityka routingu wewnętrznego i zewnętrznego
- ASN **0** i **65535** zarezerwowane, **1-64511** zarządzane centralnie (publiczne)
- **64512-65534** do prywatnego użytku

Protokół BGP

internal BGP vs external BGP

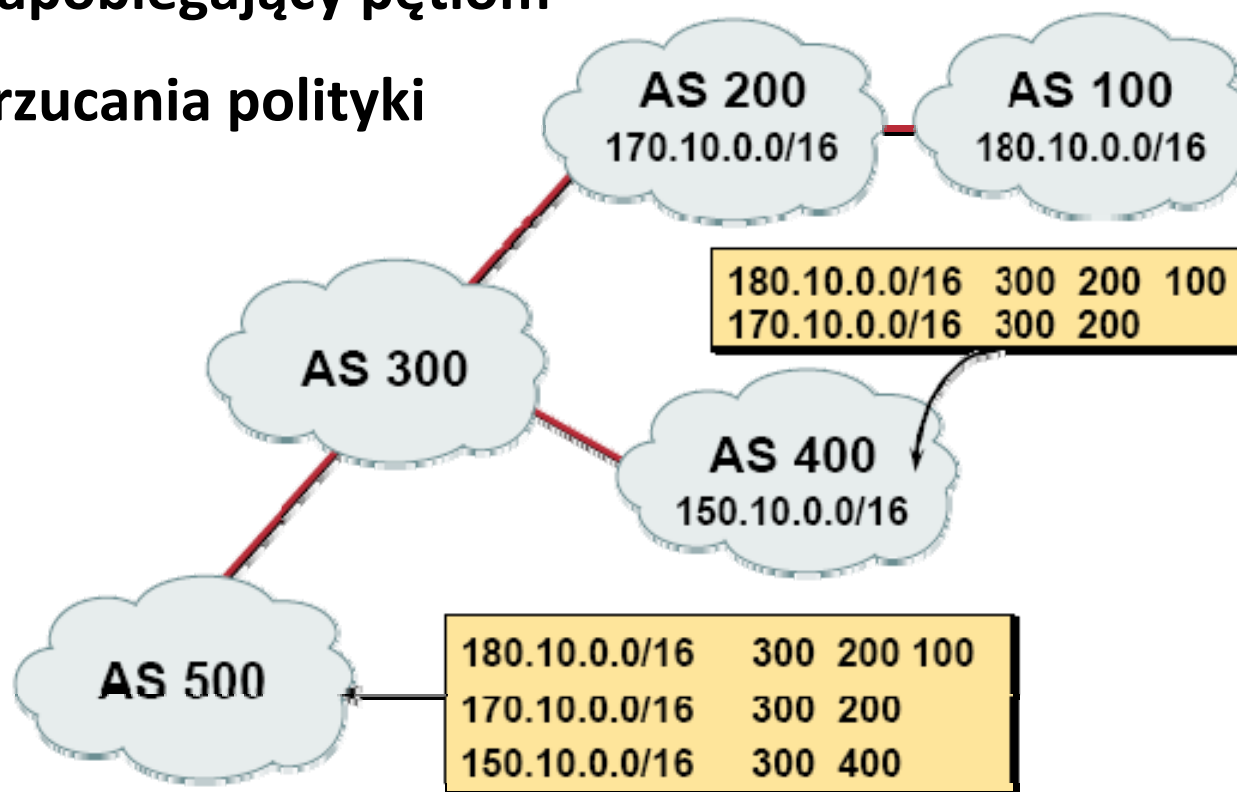


- **iBGP – sesje pomiędzy routerami w tym samym AS**
wszystkie routery wewnątrz AS muszą nawiązać sesje każdy z każdym (można to obejść przez konfederacje/klastry)
- **eBGP – sesje pomiędzy routerami w różnych AS**
domyślnie połączenie bezpośrednie, należy wprost wskazać że połączenie jest multihop

Protokół BGP

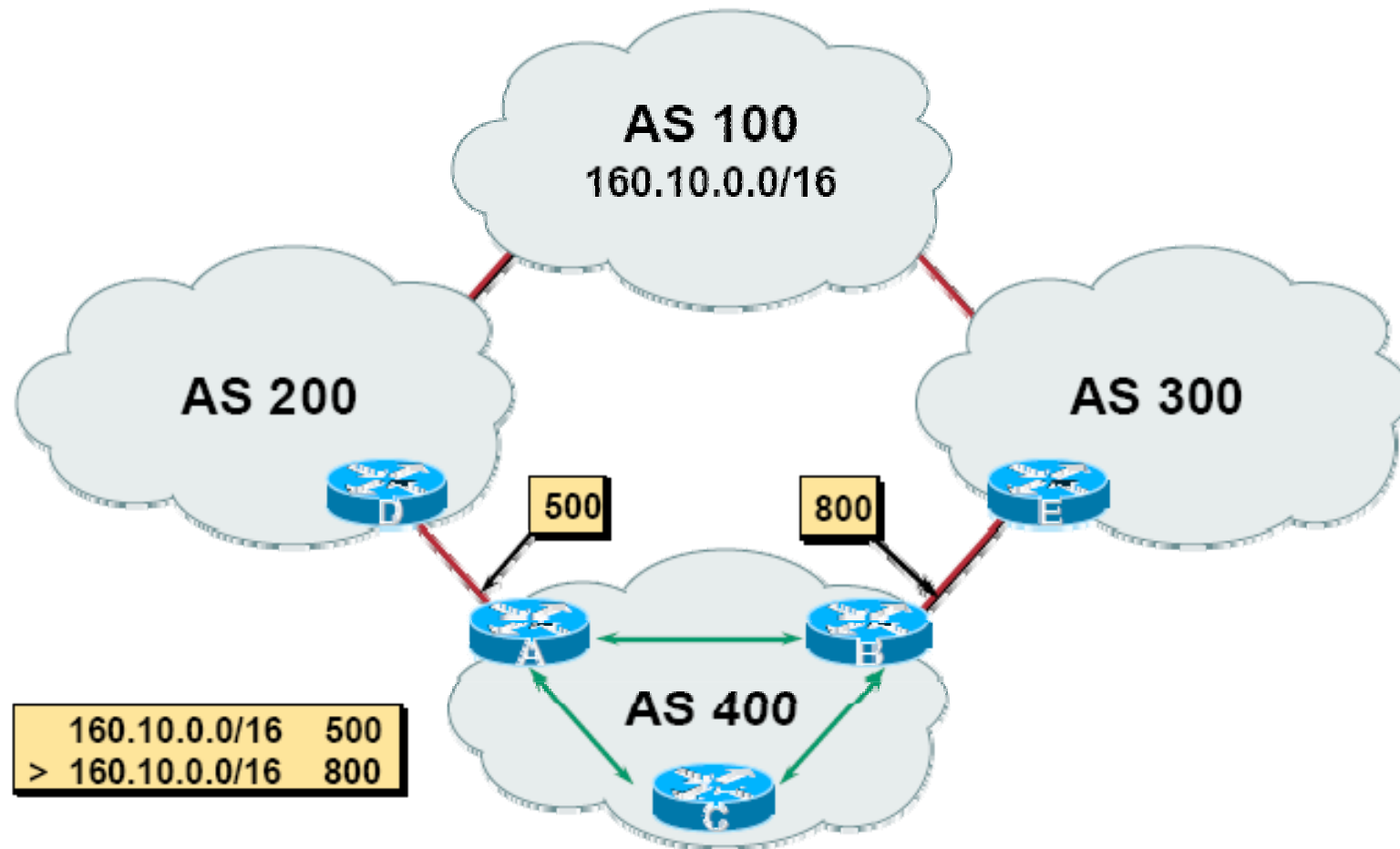
Atrybuty – AS-Path

- Trasa, jaką prefiks przeszedł
- Mechanizm zapobiegający pętlom
- Narzędzie narzucania polityki



Protokół BGP

Atrybuty – local preference



Protokół BGP

Atrybuty – community

- Opisane w RFC1997, mogą być przekazywane pomiędzy ASami (transitive) i są opcjonalne
 - 32 bitowa wartość dodatnia
- Dla wygody zapisywana jako dwie 16-bitowe wartości rozdzielone dwukropkiem
 - Standardowo **<NUMER-AS>:XXXX**
 - Na przykład: **64999:666**
- Bardzo przydatne i często spotykane w publicznych peeringach do sterowania polityką dotyczącą oznaczonego prefiksu

Protokół BGP

Atrybuty – community

- Opisane w RFC1997, używane między ASami (Community) – 32 bitowa wartość
- Dla wygody zapisywane w postaci wartości rozdzielonej kropką. Standardowo <NUM>. Na przykład: 64990
- Bardzo przydatne w konfiguracji publicznych peerów, dotycząca oznaczenia

```
Internet Partners BGP community support
e-mail contact: <bgp4@ipartners.pl>
-----
Communities to control traffic (settable by peers):

8246:2000 Do not announce to GTS CE (AS5588)
8246:2001 Prepend +1 when announcing to GTS CE
8246:2002 Prepend +2 when announcing to GTS CE
8246:2003 Prepend +3 when announcing to GTS CE

8246:2011 Prepend +1 when announcing to T-Systems
8246:2012 Prepend +2 when announcing to T-Systems
8246:2013 Prepend +3 when announcing to T-Systems

8246:2100 Do not announce to TPNET (AS5617)
8246:2101 Prepend +1 when announcing to TPNET
8246:2102 Prepend +2 when announcing to TPNET
8246:2103 Prepend +3 when announcing to TPNET

8246:2200 Do not announce to NASK (AS8308)
8246:2201 Prepend +1 when announcing to NASK
8246:2202 Prepend +2 when announcing to NASK
8246:2203 Prepend +3 when announcing to NASK

8246:2300 Do not announce to POL34 (AS8501)

8246:2400 Do not announce to Sunsite ICM (AS8664)
8246:2401 Prepend +1 when announcing to Sunsite ICM
8246:2402 Prepend +2 when announcing to Sunsite ICM
8246:2403 Prepend +3 when announcing to Sunsite ICM

8246:2411 Prepend +1 when announcing to WP
8246:2412 Prepend +2 when announcing to WP
8246:2413 Prepend +3 when announcing to WP
```

Protokół BGP

Atrybuty – community

```
Internet Partners BGP community support
e-mail contact: <bgp4@ipartners.pl>
-----
Communities to control traffic (settable by peers):

8246:2000 Do not announce to GTS CE (AS5588)
8246:2001 Prepend +1 when announcing to GTS CE
8246:2002 Prepend +2 when announcing to GTS CE
8246:2003 Prepend +3 when announcing to GTS CE

8246:2011 Prepend +1 when announcing to T-Systems
8246:2012 Prepend +2 when announcing to T-Systems
8246:2013 Prepend +3 when announcing to T-Systems

8246:2100 Do not announce to TPNET (AS5617)
8246:2101 Prepend +1 when announcing to TPNET
8246:2102 Prepend +2 when announcing to TPNET
8246:2103 Prepend +3 when announcing to TPNET

8246:2200 Do not announce to NASK (AS8308)
8246:2201 Prepend +1 when announcing to NASK
8246:2202 Prepend +2 when announcing to NASK
8246:2203 Prepend +3 when announcing to NASK

8246:2300 Do not announce to POL34 (AS8501)

8246:2400 Do not announce to Sunsite ICM (AS8664)
8246:2401 Prepend +1 when announcing to Sunsite ICM
8246:2402 Prepend +2 when announcing to Sunsite ICM
8246:2403 Prepend +3 when announcing to Sunsite ICM

8246:2411 Prepend +1 when announcing to WP
8246:2412 Prepend +2 when announcing to WP
8246:2413 Prepend +3 when announcing to WP
```


Protokół BGP

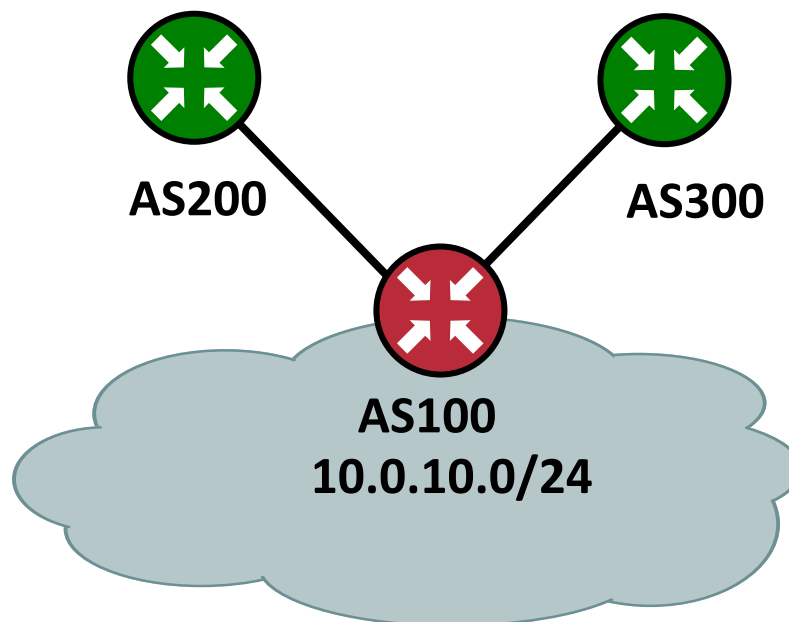
Jak BGP wybiera najlepszą trasę? (wersja skrócona)

- **Najwyższy local preference (w ramach AS)**
- **Najkrótsza ścieżka AS-Path**
- **Najniższy kod pochodzenia (Origin)**
 - IGP < EGP < incomplete
- **Najniższa wartość MED (Multi-Exit Discriminator)**
- **Lepiej trasa z eBGP niż z iBGP**
- **Najpierw trasa z niższym kosztem wg. IGP do next-hop**
- **Najniższy router-id routera BGP**
- **Najniższy adres peer'a**

(pełna lista w RFC4271)

Protokół BGP

Przykładowa konfiguracja



- Dwie pełne światowe tablice routingu (2*~186k prefiksów)
- Trasa default nie jest potrzebna (mamy wszystkie trasy jakie są w Internecie)
- Dla Quaggi potrzebne ok. 100MB RAM, dla OpenBGPD około 45-60MB

Protokół BGP

Przykładowa konfiguracja

```
router bgp 100
  bgp router-id 10.0.10.1
  network 10.0.10.0 mask 255.255.255.0
  aggregate-address 10.0.10.0 255.255.255.0 summary-only

  neighbor 172.16.10.1 remote-as 200
  neighbor 172.16.10.1 description AS200
  neighbor 172.16.10.1 version 4
  neighbor 172.16.10.1 password AJAX*PERSIL*E

  neighbor 172.16.20.1 remote-as 300
  neighbor 172.16.20.1 description AS300
  neighbor 172.16.20.1 version 4
  neighbor 172.16.20.1 password E*LISREP*XAJA
```

Protokół BGP

Przykładowa konfiguracja – świadoma polityka ruchowa

- **Wszystkie** prefiksy otrzymane z AS200 są lepsze niż inne, z domyślnym (100) local-preference

```
router bgp 100
  neighbor 172.16.10.1 remote-as 200
  neighbor 172.16.10.1 route-map KuLepszejPrzyszlosci in

route-map KuLepszejPrzyszlosci permit 10
  set local-preference 5000
```

Protokół BGP

Przykładowa konfiguracja – świadoma polityka ruchowa

- Tylko prefiks **192.168.10.0/24** lub bardziej specyficzny jest zawsze lepszy przez AS200

```
router bgp 100
  neighbor 172.16.10.1 remote-as 200
  neighbor 172.16.10.1 route-map KuLepszejPrzyszlosci in

ip prefix-list JedynaDroga permit 192.168.10.0/24 le 32

route-map KuLepszejPrzyszlosci permit 10
  match ip prefix-list JedynaDroga
  set local-preference 5000
```

Dobry opis jak działają prefix-listy:

<http://www.groupstudy.com/archives/ccielab/200404/msg00539.html>

Protokół BGP

Przykładowa konfiguracja – świadoma polityka ruchowa

- „Pogorszenie” atrakcyjności własnego prefiksu 10.0.10.0/24 przez AS200

```
router bgp 100
  neighbor 172.16.10.1 remote-as 200
  neighbor 172.16.10.1 route-map KuLepszejPrzyszlosci out

route-map KuLepszejPrzyszlosci permit 10
  set as-path prepend 100 100
```

```
AS200rtr# show ip bgp 10.0.10.0
   Network          Next Hop      LocPrf  Path
*  10.0.10.0/24     172.16.10.2   100     100 100 100 i
*>                 172.16.99.1  100     300 100 i
```

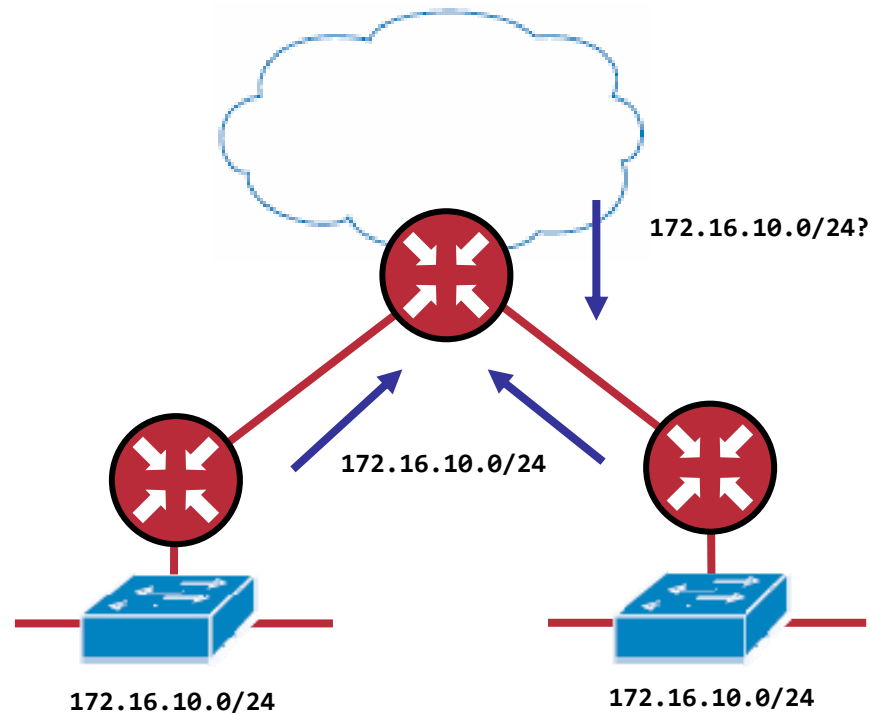
ROUTING DYNAMICZNY A INNE ZAGADNIENIA



Routing dynamiczny a inne zagadnienia

NAT

- Typowa „sieć osiedlowa” stosuje NAT



- Uwaga na nakładające się podsieci prywatne przy redystrybucji tras połączonych/statycznych do protokołów routingu dynamicznego!
- Trasy można odfiltrować nawet, jeśli redystrybucja jest włączona

Routing dynamiczny a inne zagadnienia

NAT

- **Filtrowanie sieci z RFC1918 przy redystrybucji tras połączonych/statycznych do OSPFa:**

```
router ospf
  redistribute kernel route-map NORFC1918

route-map NORFC1918 permit 10
  match ip address prefix-list 10

ip prefix-list 10 deny 10.0.0.0/8 le 32
ip prefix-list 10 deny 172.16.0.0/12 le 32
ip prefix-list 10 deny 192.168.0.0/16 le 32
ip prefix-list 10 permit any
```

Routing dynamiczny a inne zagadnienia

Filtrowanie ruchu

- **Protokoły routingu mają swoje wymagania co do przepuszczanego ruchu:**

RIPv1 – 520/udp

RIPv2 – multicast pod adres 224.0.0.9 na 520/udp

za pomocą wskazania wprost sąsiada można dodatkowo wysyłać pakiety unicast (przydatne przy tunelach IPsec)

OSPF – multicast, protokół IP numer 89

BGP – protokół TCP na/z portu 179

Routing dynamiczny a inne zagadnienia

Tunelowanie IP-w-IP i GRE

- Tunele GRE pozwalają przenieść multicasty oraz inne protokoły warstwy 3 – w szczególności IPX

wygodne i funkcjonalne połączenie dwóch sieci z możliwością zapewnienia działania protokołów RIPv2 i OSPF

quagga automatycznie konfiguruje interfejsy gre jako punkt-punkt, ale tylko te, które istnieją zanim zostanie uruchomiona

warto sprawdzić, czy interfejs jest widziany jako posiadający flagę **multicast** – różnie dla różnych kerneli i wersji pakietu

Routing dynamiczny a inne zagadnienia

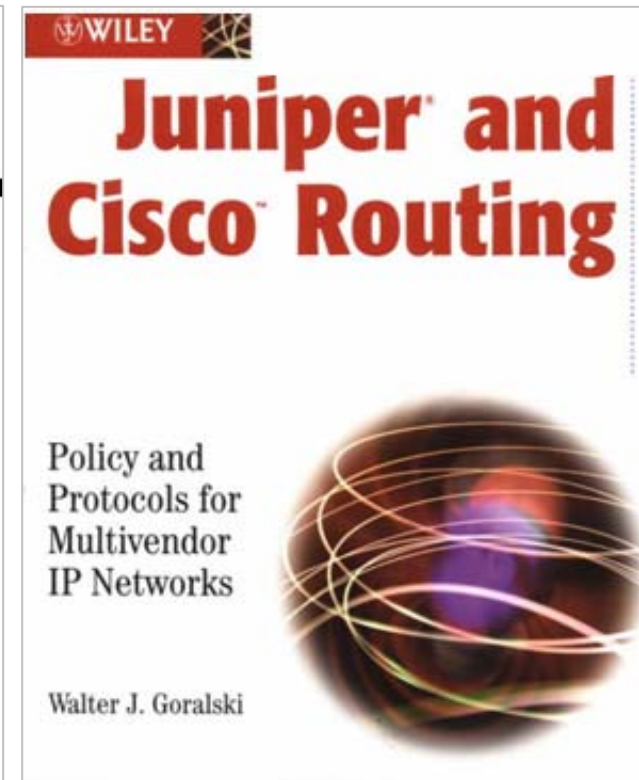
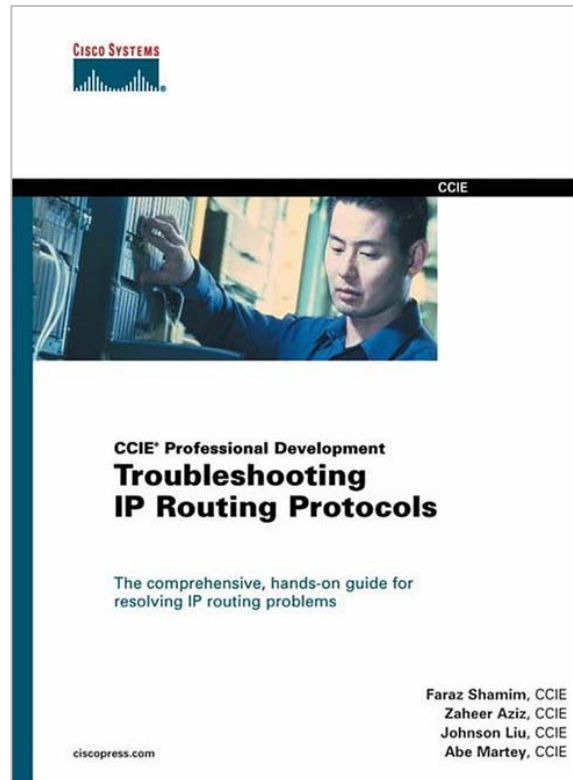
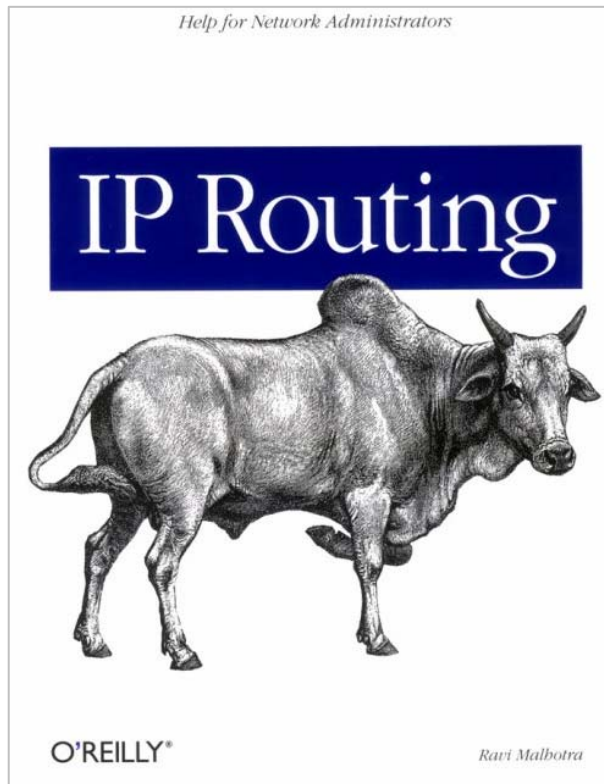
QoS

- Mechanizmy/polityka QoS powinna **w szczególności** dotyczyć protokołów routingu
- Priorytet dla pakietów **hello**, oraz zapewniających funkcjonowanie protokołów routingu powinien być pierwszym składnikiem polityki (przed gwarancjami dla VoIP itp.)
- Zastosowanie mechanizmów nawet, jeśli nie ma potrzeby ich stosowania daje ochronę w trakcie ataków (D)DoS oraz np. infekcji wirusami/trojanami

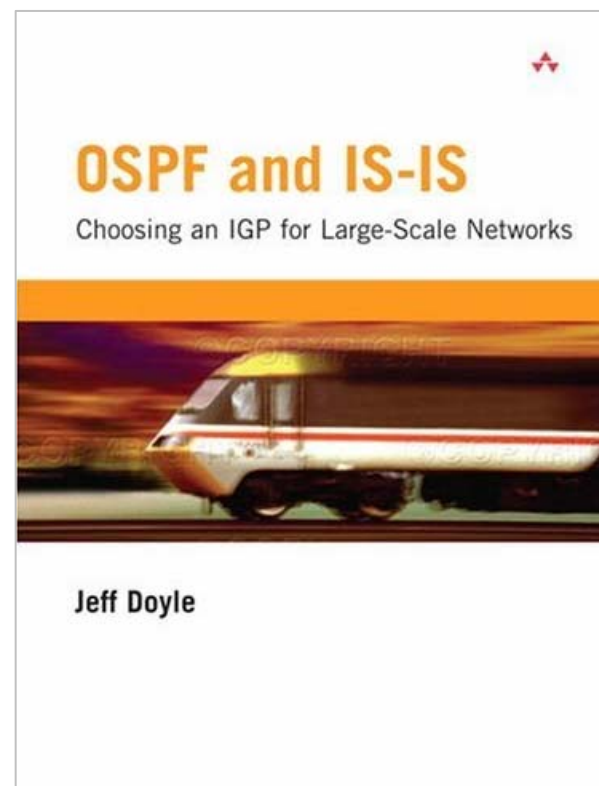
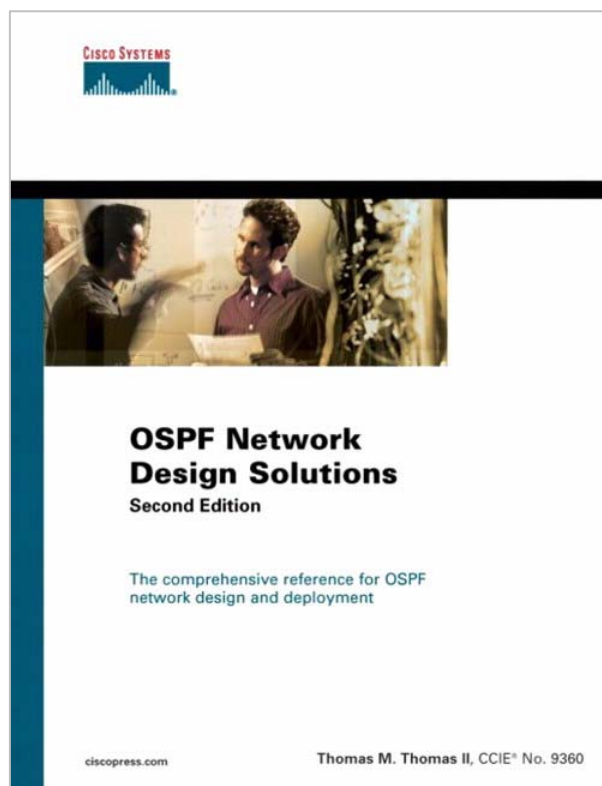
GDZIE WARTO RZUCIĆ OKIEM



Książki



Książki



Zasoby WWW

- **Pakiet Quagga:**

<http://www.quagga.net>

- **Pakiet XORP:**

<http://www.xorp.org>

- **Demon OpenOSPFd/OpenBGPd:**

<http://www.openbsd.org>

- **Einar – LiveCD wykorzystujące Xen i Quagę:**

<http://www.isk.kth.se/proj/einar/>

Q&A



ROUTING DYNAMICZNY

...a Linux

Łukasz Bromirski

lukasz@bromirski.net

