

# -1. Ostatnie zmiany

3., 2.3.2.2, 2.3.2.4

## 0. Konfiguracja routingu statycznego i dynamicznego

Wykonał: Patryk Kaniewski, 2 rok Informatyka ST PUSB Skierniewice.

## 1. Wprowadzenie

Routing jest to proces wybierania ścieżki w sieci lub pomiędzy różnymi sieciami. Przykładami routingu jest telefonia (dawniej operator fizycznie przełączający wtyczkę z naszym telefonem z innym gniazdem) jak i sieci komputerowe np. Internet. W sieciach komputerowych zazwyczaj jest to rozwiązane za pomocą routerów. Router posiada w sobie informacje o sąsiednich sieciach i na podstawie tych informacji i innych zasad (np. filtrowania, źródła) i przekazuje pakiety do następnego routera lub komputera.

### 1.1 Cel ćwiczenia

Konfiguracja tras statycznych i uruchomienie a następnie testowanie i monitorowanie zachowań protokołów routingu dynamicznego oraz zbadanie ich możliwości w różnych typowych i awaryjnych sytuacjach.

### 1.2 Wymagania wstępne

- Podstawy CLI (Linux Bash)
- Konfiguracja interfejsów sieciowych
- Konfiguracja usługi świadczącej routing dynamiczny (Linux: *quagga*)
- routing na podstawie stanu łącza - OSPFv2 (RFC 2328)
- routing na podstawie wektora odległości - RIPv2 (RFC 2458)
- Narzędzia diagnostyczne sieci (*tcpdump, traceroute, ping*)

### 1.3 Zakres ćwiczenia

W tym ćwiczeniu zajmiemy się docelowym routingiem w sieciach komputerowych używających modelu TCP/IP na podstawie MAC (Ethernet, WiFi itp.) Pod uwagę weźmiemy tylko protokoły bram wewnętrznych (ang. *Interior gateway protocol*) ze względu na zakres materiału w protokołach bram zewnętrznych (protokoły te obejmują zagadnienia routingu pomiędzy systemami dostawców usług internetowych, miast, krajów, kontynentów oraz zagadnienia takie jak cache, content delivery network w celach usprawniania działania sieci).

Ćwiczenie może być wykonane na maszynach wirtualnych (przykłady na podstawie GNU/Linux Debian 9 i *quagga* 1.2.4)

### 1.4 Zagadnienia

1. Jak działa routing w sieci komputerowej
2. Routing statyczny
3. Problemy routingu statycznego
4. Routing dynamiczny
  - 4.1. Wektor odległości (Przykład: RIPv2)
  - 4.2. Stan łącza (Przykład: OSPFv2)
5. Wady i zalety rozwiązań routingu dynamicznego
6. Dostosowanie rozwiązań do zapotrzebowań w różnych okolicznościach

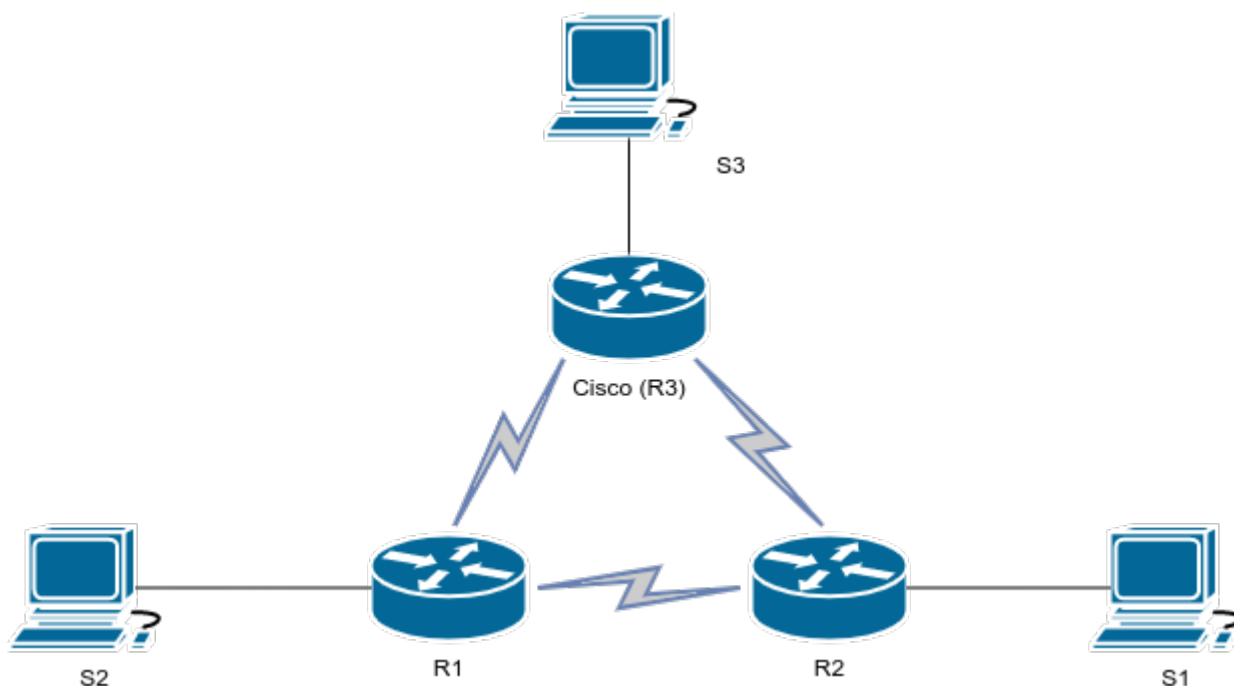
### 1.5 Dodatkowe definicje

System autonomiczny (ang. Autonomous system) – jest to zbiór sieci połączonych z sobą protokołami routingu bram wewnętrznych. Takie systemy natomiast łączą się z innymi za pomocą protokołów routingu bram zewnętrznych

VLSM (ang. Variable Length Subnet Mask) – jest to odejście od klasowej maski sieci w adresowaniu IP (24,16,8 dla klas C,B,A odpowiednio) i stosowanie dowolnej długości maski aby lepiej wykorzystać przestrzeń adresową (256 może być za mało adresów dla sieci firmowej a 16 tysięcy za dużo, tak samo z połączeniami point-to-point)

## 2. Przebieg ćwiczenia

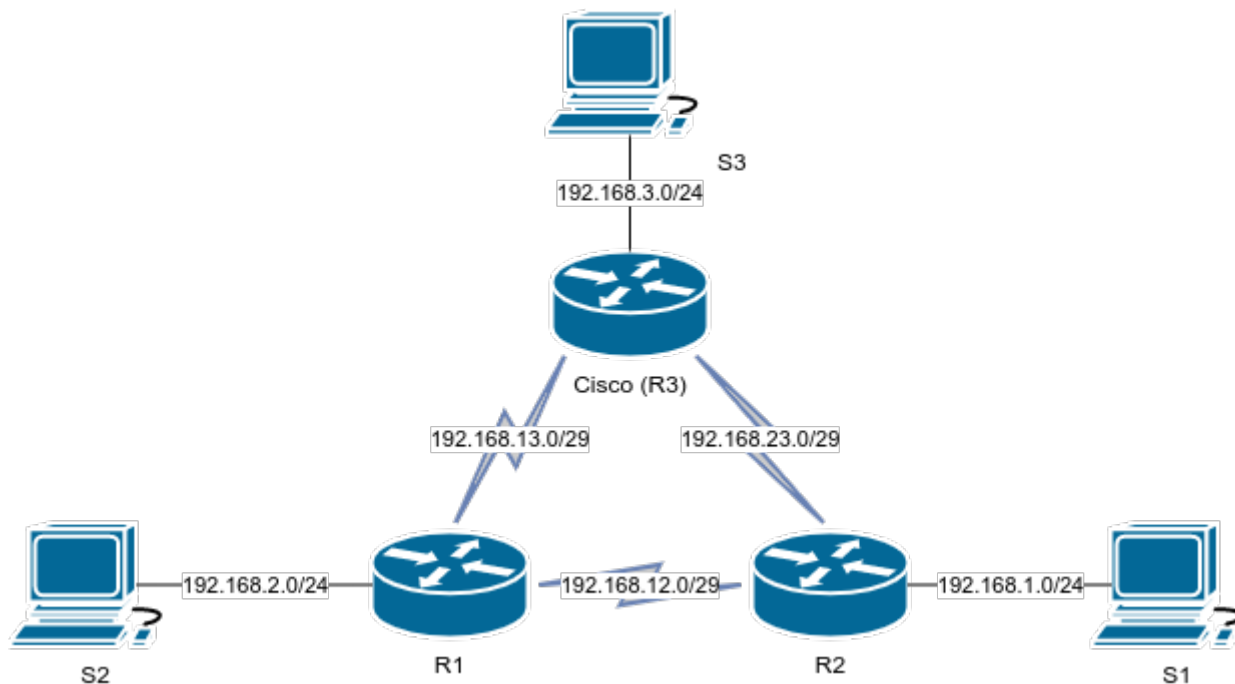
Fizyczna konfiguracja sieci:



Rys 2.1

## Lab1: Konfiguracja routingu statycznego i dynamicznego

Logiczna konfiguracja sieci:



Rys 2.2

Wyjaśnienie:

Routery łączymy w sieciach z długą maską sieciową ponieważ nie potrzebujemy wiele adresów do sieci point-to-point. Maszyny robocze zostawiamy w domyślnych podsieciach klasy C. Za pomocą tego możemy zweryfikować jak protokoły radzą sobie z maską sieci różnej długości (VLSM).

Konfiguracja interfejsów:

ręcznie

za pomocą skryptu: <https://github.com/p7tryk/administracja/blob/master/network.sh>

Instalacja programów diagnostycznych (jeśli potrzeba):

```
# apt install tcpdump traceroute ping
```

Włączenie routingu na R1,R2:

```
# Edycja sysctl.conf
```

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

## 2.1 Routing

Ogólny schemat działania routingu w TCP/IP:

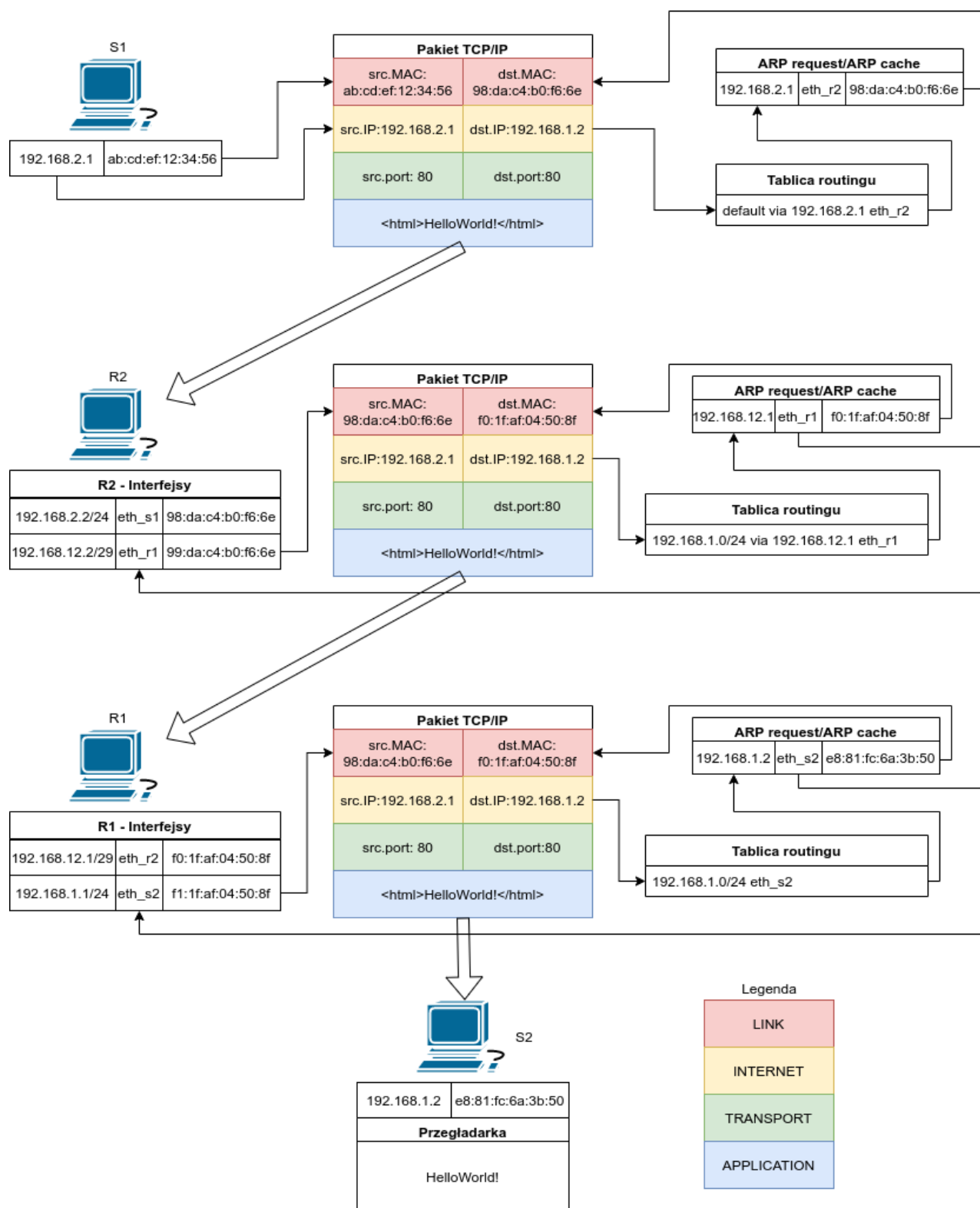
1. Router otrzymuje pakiet TCP/IP
2. Sprawdza adres MAC do którego jest zaadresowany.
3. Jeżeli pokrywa się z adresem urządzenia zdejmuję ramkę Ethernet

## **Lab1: Konfiguracja routingu statycznego i dynamicznego**

---

4. Sprawdza adres IP do którego jest zaadresowany
5. Jeżeli jest zaadresowany do innej sieci, porównuje adres tej sieci z tablicą routingu
6. Konsultowane jest ARP Cache lub wysyłany jest zapytanie ARP żeby znaleźć adres fizyczny związany z adresem IP
7. Po otrzymaniu adresu fizycznego zakładana jest ramka Ethernet z tym adresem i adresem routera
8. Nowo skonstruowany pakiet wysyłany jest przez interfejs podany przez tablice routingu.

## Lab1: Konfiguracja routingu statycznego i dynamicznego



Rys 2.3

Link, Internet, Transport, Application

### 2.2 Routing statyczny

Routing statyczny jest to najprostsze rozwiązanie problemu routingu. Zwykle polega na manualnym wprowadzeniu rekordów do tablicy routingu. Administrator sieci musi ręcznie konfigurować routing dla każdego routera. Zwykle używane w małych sieciach i prostych sieciach.

Specjalną drogą jest droga domyślna (default gateway) trafiają tam wszystkie pakiety których droga nie jest specyficznie wypisana w tablicy routingu

#### 2.2.1 Linux

Konfigurujemy routing statyczny ze stacji roboczej (S1,S2,S3) do przyłączonego do niej routera.

##### 2.2.1.1 ip route add

```
# ip route [-6] add $adres [opcje] via $adres [opcje] dev $interfejs  
<zdjecie>
```

##### 2.2.1.2 Default route

```
# ip route add default via $adres dev $interfejs
```

<zdjecie>

```
# edycja /etc/network/interfaces
```

<zdjecia>

#### 2.2.2 Weryfikacja

Działanie połączenia:

<zdjecie ping>

Sprawdzanie tablic routingu:

<zdjecie ip r>

### 2.3 Routing dynamiczny

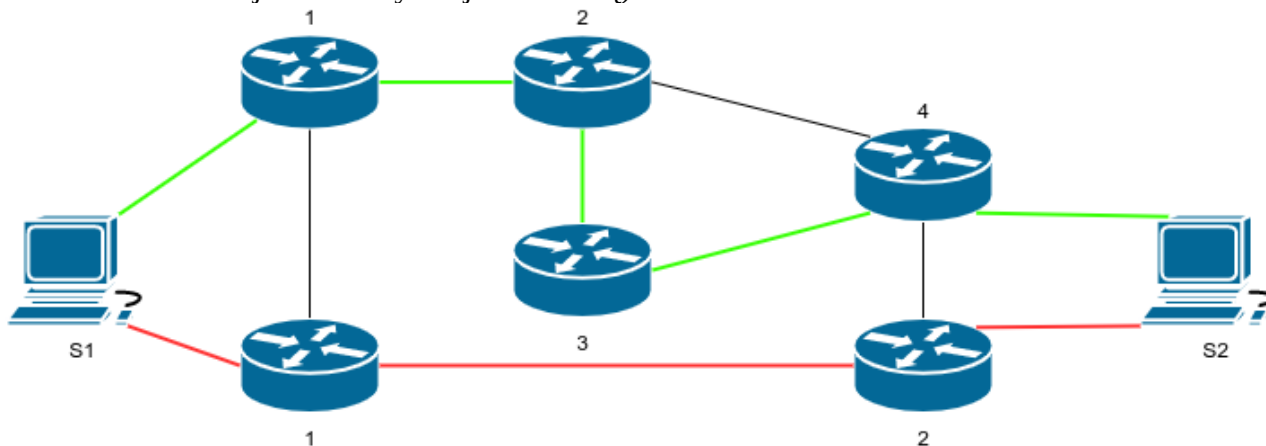
W miarę wzrostu złożoności sieci komputerowych narosła potrzeba bardziej inteligentnego i *dynamicznego* systemu który mogłby reagować na zmiany w ruchu sieciowym czy awarie na połączeniach.

Ogólna idea się nie zmienia (punkt 2.1) ale nasze tablice routingu są teraz generowane i aktualizowane automatycznie.

Dwoma głównymi metodami routingu dynamicznego w sieciach wewnętrznych jest routing oparty o *stan łącza* (*link-state routing*) i *wektor odległości* (*distance-vector routing*).

### 2.3.1 Distance-vector routing

Distance-vector routing polega na wyliczeniu liczby skoków (routerów) przez które pakiet przechodzi na danej drodze. Problemem w takiej sytuacji może być droga z małą przepustowością może być krótsza niż droga z dużą przepustowością (analogia świata rzeczywistego: jechanie przez centrum miasta vs. jechanie szybciej obwodnicą)



rys 2.4  
drogi pomiędzy S1 i S2  
droga zielona ma 4 skoki a czerwona 2 skoki

#### 2.3.1.1 RIPv2

RIPv2 jest protokołem opartym na algorytmie Bellman-Ford. Jest rozwinięciem protokołu RIP dodając nowe możliwości (szczególnie VLSM) ale zachowując prostotę protokołu.

- Protokół jest ograniczony do 15 skoków. Jest to zrobione specjalnie gdyż twórcy uważają że większe sieci nie powinny być konstruowane z tak prostym protokołem
- Protokół używa tylko jednej metryki – ilości skoków. Powoduje to problemy w sieciach z dużymi różnicami przepustowości pomiędzy routerami oraz nie pozwala na balansowanie ruchu sieciowego zdala od słabych połączeń
- Protokół nie rozróżnia pomiędzy sieciami a hostami (jeżeli potrzebne są specjalne drogi dla hostów)

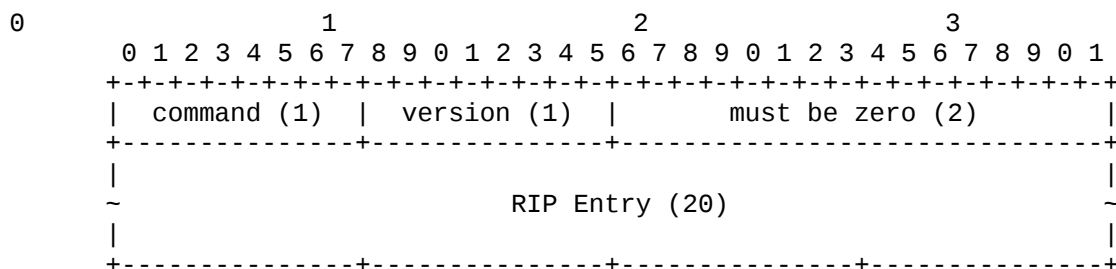
Router tworzy tabele z rekordami drogi do każdej możliwej lokalizacji i przechowuje jej długość i router który jest następnym skokiem. Co jakiś czas router wysyła aktualizacje do swoich sąsiadów. Sąsiad otrzymując taką aktualizację dodaje do informacji wysłanej liczbę skoków do sąsiada i porównuje ze swoją tabelą, jeżeli jakaś droga jest krótsza to dopisuje ją do swojej tablicy zamiast poprzedniej.

Problemem jest jeżeli topologia sieci się zmieni (awaria). RIP rozwiązuje to za pomocą regularnych aktualizacji domyślnie co 30 sekund (*\$update*). Jeżeli router nie odpowiada po

## Lab1: Konfiguracja routingu statycznego i dynamicznego

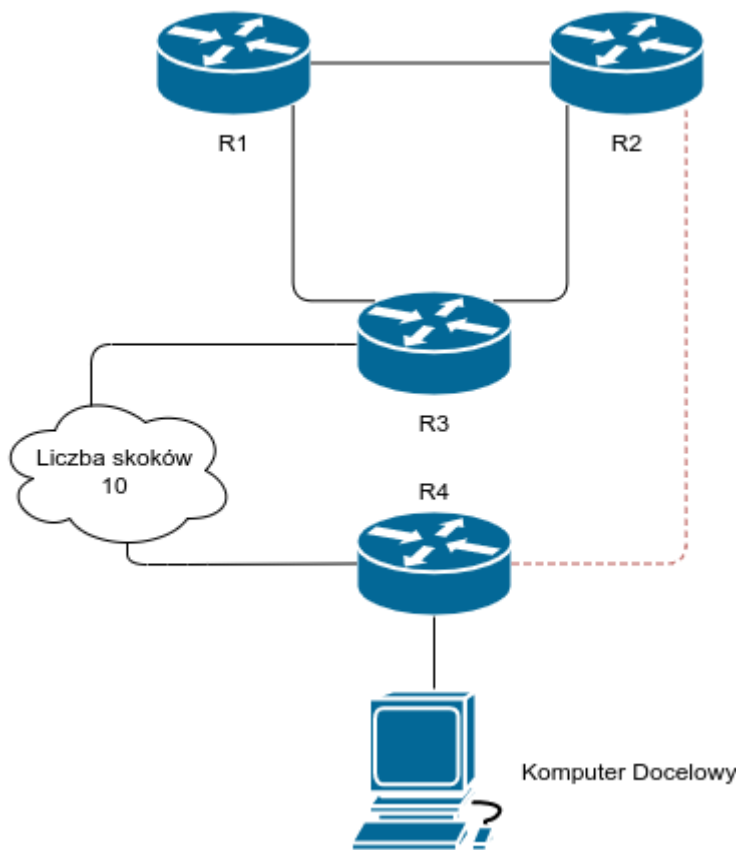
domyślnie 180 sekundach (*\$dead*) to ta droga jest zaznaczana jako nieważną a po następnych 180s (*\$garbagecollect*) usuwana z tabeli.

RIP używa pakietów UDP i zawiera od 1 do 25 rekordów RIP



*RIP packet RFC2453*

### 2.3.1.2 Problemy z distance-vector routing



*rys 2.5*

Weźmy ten przykład (rys 2.5) z dokumentu RFC. Czerwona droga została przerwana. R2 unieważnia drogę bo R4 nie odpowiada. Jednak R3 i R1 nadal wysyłają że mogą dostać się do sieci docelowej przez R2 (3 skoki). Nawet jeżeli dowiedzą się że ich droga przez R2 nie jest ważna to będą myśleć że mogą się dostać przez odpowiednio R1 i R3. Będą one liczyć samych siebie coraz wyżej aż alternatywna droga (przez R3) stanie się krótsza. Stanie się to zawsze (chyba że nowa



## Lab1: Konfiguracja routingu statycznego i dynamicznego

droga nie będzie możliwa – skoki >15) ale może zająć czas kiedy sieć będzie niedostępna. Problem ten nazywa się liczeniem do *nieskończoności* (naszą nieskończonością na szczęście jest maksymalna liczba skoków = 15)

### 2.3.1.3 Split Horizon

Jednym z problemów jest to że routery ogłaszają droge dostępne przez dany router do *tego routera* powoduje to że routery sobie ufają i może zająć dużo czasu zanim fałszywa droga (pętla) będzie gorsza niż droga alternatywna.

Prostym sposobem jest ominięcie w swoich ogłoszeniach dróg które usłyszeliśmy z hosta do którego wysyłamy.

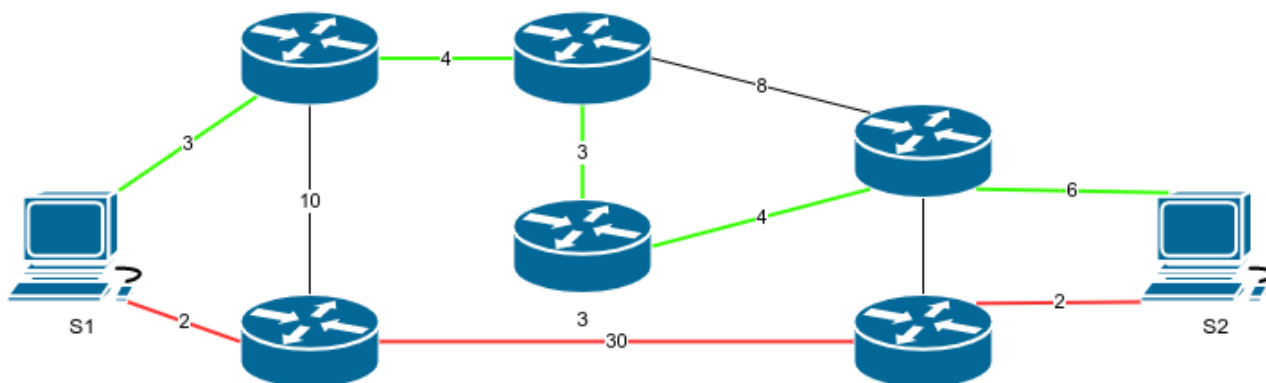
Lepszym rozwiązaniem jest zatrucie (ang. poisoning) takich dróg. Ustawiamy takie drogi na maksymalną ilość skoków (15). Powoduje to na natychmiastowe policzenie do *nieskończoności* i przerwanie pętli. Rozwiązuje to wszystkie pętle 2 routerów ale w systemach z większą liczbą routerów które mogą się połączyć w taką pętlę ten problem nadal może się zdarzyć.

### 2.3.1.4 Triggered Updates.

Aby rozwiązać problem większych pętli dodano możliwość wysyłania aktualizacji od razu po wykryciu utraty drogi. W najlepszym przypadku natychmiastowo rozprowadzi informacje i zapobiegne tworzeniu fałszywej drogi. Jeżeli jednak czasowa aktualizacja zdarzy się w podobnym czasie taka droga nadal może powstać. W takiej sytuacji rzadziej może zdarzyć się większa pętla więc ochrona przez split horizon (2.3.1.3) może zdecydowanie obniżyć szanse przestoju w sieci.

## 2.3.2 Link-state routing

Link-state routing polega na nadaniu danemu routerowi wagi zwykle oparte na szybkości połączenia i urządzenia oraz obciążenia. Rozwiązuje to problem obwodnicy ale zdecydowanie komplikuje działanie takiego systemu.



Rys 2.4

## Lab1: Konfiguracja routingu statycznego i dynamicznego

---

*drogi pomiędzy S1 i S2*  
*droga zielona ma koszt  $3+4+3+4+6=20$*   
*droga czerwona ma koszt  $2+30+2=34$*

### 2.3.2.1 OSPFv2

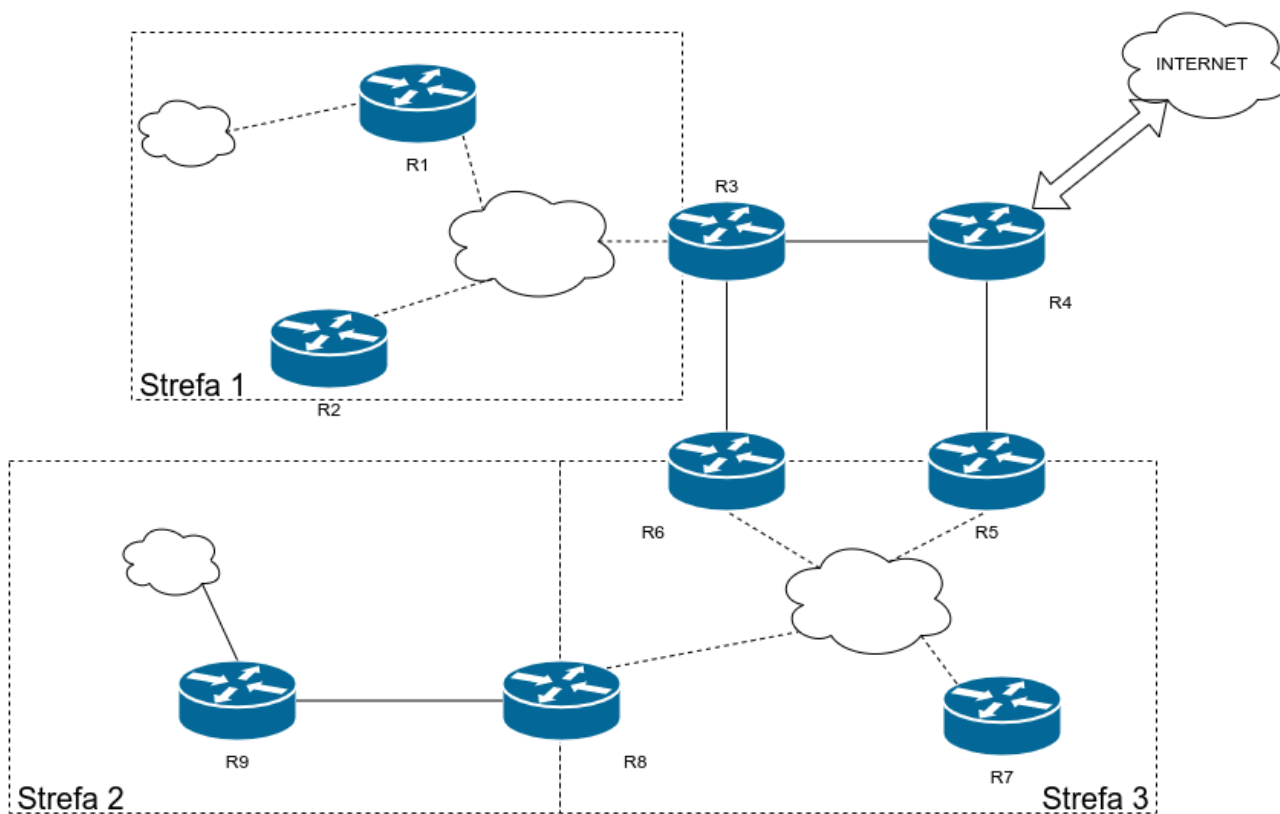
OSPFv2 jest implementacją link state routing dla bram wewnętrznych. Router OSPF ma swój unikalny numer 32bitowy (w systemie 1 bramy wewnętrznej) i przechowuje w sobie mapę topologii sieci wraz z wagami przypisanymi do każdego połączenia. Routery wymieniają pomiędzy siebie informacje o swoich bazach stanu. Za pomocą tej bazy danych (ang. link-state database) każdy router buduje drzewo nakrótszych dróg zaczynając od samego siebie (jest korzeniem tego drzewa). W tym drzewie liśćmi są informacje zewnętrzne dla systemu (takie otrzymane z protokołów bram zewnętrznych).

System ten może łączyć sieci w strefy (ang. area) by zredukować ilość informacji które są wysyłane pomiędzy routerami od siebie odległymi (równoznacznie: rozdzielenie systemu autonomicznego na części). Specjalną strefą jest strefa 0 (ang. backbone) która stanowi centrum systemu

Koszt drogi nie jest zdefiniowany jako prędkość a jedynie jako wartość którą można dostosować do wymagań systemu. Cisco domyślnie używa  $10^8$ /przepustowość, czyli 100Mb/s jest kosztem 1 a 10Mb/s jest kosztem 10.

Rodzaje routerów:

- wewnętrzne – połączone tylko z sieciami w jednej strefie
- granicy stref – ma interfejsy podłączone do strefy 0 i co najmniej jednej innej strefy.
- granicy systemu autonomicznego – posiada drogi zewnętrzne do systemu autonomicznego
- backbone – połączone z strefą 0



*R1,R2,R7,R9 – wewnętrzne*  
*R3,R5,R6,R8 – granicy stref*  
*R3,R5,R6,R4 – backbone*  
*R4 – granicy systemów autonomicznych*

*R3,R5,R6,R8 – granicy stref*

$R3, R5, R6, R4$  – backbone

#### R4 - granicy systemów autonomicznych

### 2.3.2.2 Link State Advertisement

Jest mechanizmem za pomocą którego router ogłasza sieci do niego podłączone.

### Wyróżnione typy LSA:

1. Router LSA – router ogłasza sieć podłączoną bezpośrednio niego
2. Network LSA - Designated router ogłasza listę routerów które są z nim związane (propaguje się tylko lokalnie)
3. Summary LSA (area boundary router) – jeżeli ogłoszenia przechodzą przez różne strefy są sumaryzowane zanim przekroczą te bariery
4. Summary LSA (autonomous area boundary router) – ogłasza drogę do routera z wyjściem z systemu autonomicznego
5. External LSA (autonomous area boundary router summary) – ogłasza drogi dostępne zewnętrznie przez granice systemu autonomicznego

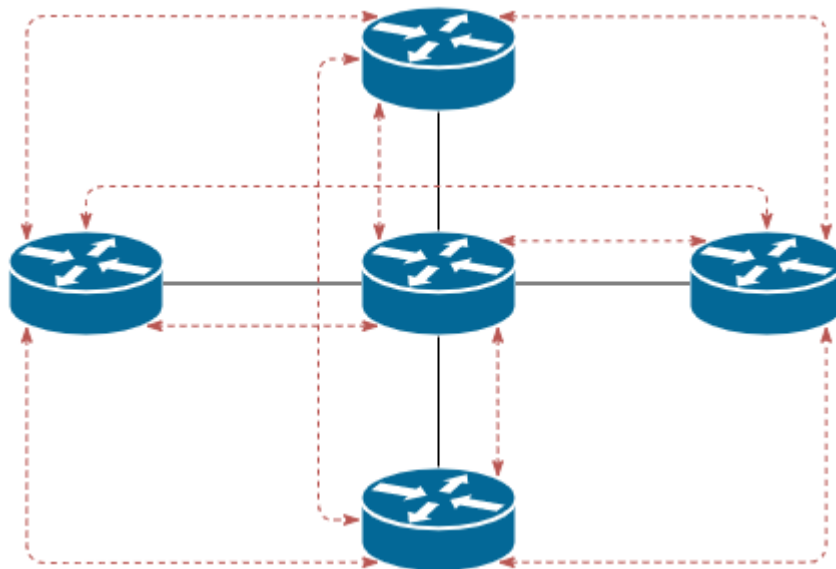
## Lab1: Konfiguracja routingu statycznego i dynamicznego

---

7. NSSA LSA – działa podobnie jak typ 5 w NSSA(2.3.2.4) po wyjściu ze strefy konwertowane na standardowy typ 5

### 2.3.2.3 Designated router

Designated router i backup designated router – mechanizm stworzony by ograniczyć liczbę wysyłanych LSA pomiędzy wszystkimi routerami w sieci.

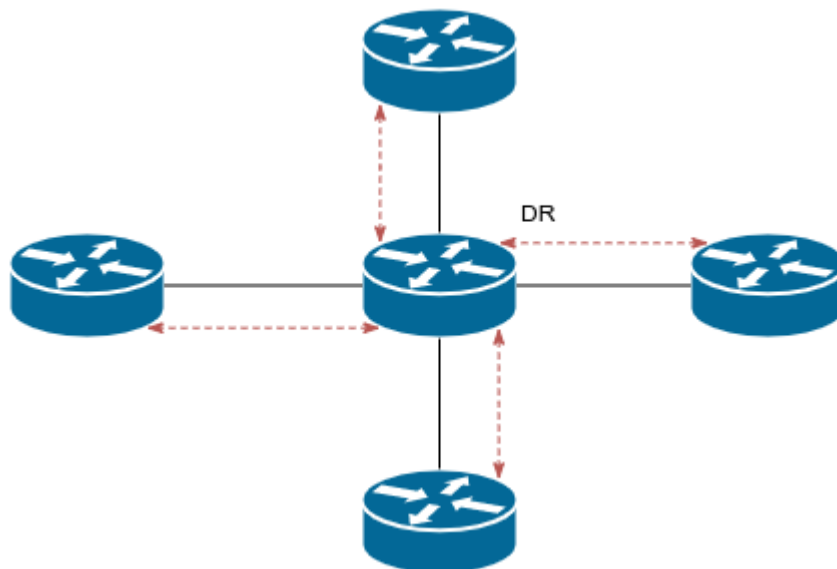


rys. 2.6

*routery wymieniające informacje pomiędzy sobą (czerwone linie)*

Jeden router jest wyznaczany jako designated router (według priorytetu) i LSA są wysyłane tylko pomiędzy routerami podrzędnym a DR.

Jeżeli DR przestanie odpowiadać to zapasowy DR zostanie DR.



Rys 2.7

*Routery komunikują się tylko z DR a nie między sobą (czerwone linie)*

### 2.3.2.4 Typy stref

W celu ograniczeniu wielkości tablic routingu strefy mogą być specjalnie wydzielone aby blokować specyficzne LSA zewnętrzne dla strefy.

1. Stubby area.

Zewnętrzne drogi (z poza systemu autonomicznego) nie będą tam rozprowadzane (LSA typ 5).

2. Totally Stubby area.

Stubby area (LSA5) + Informacje o drogach pomiędzy strefami nie będą tam rozprowadzane (LSA typ 3).

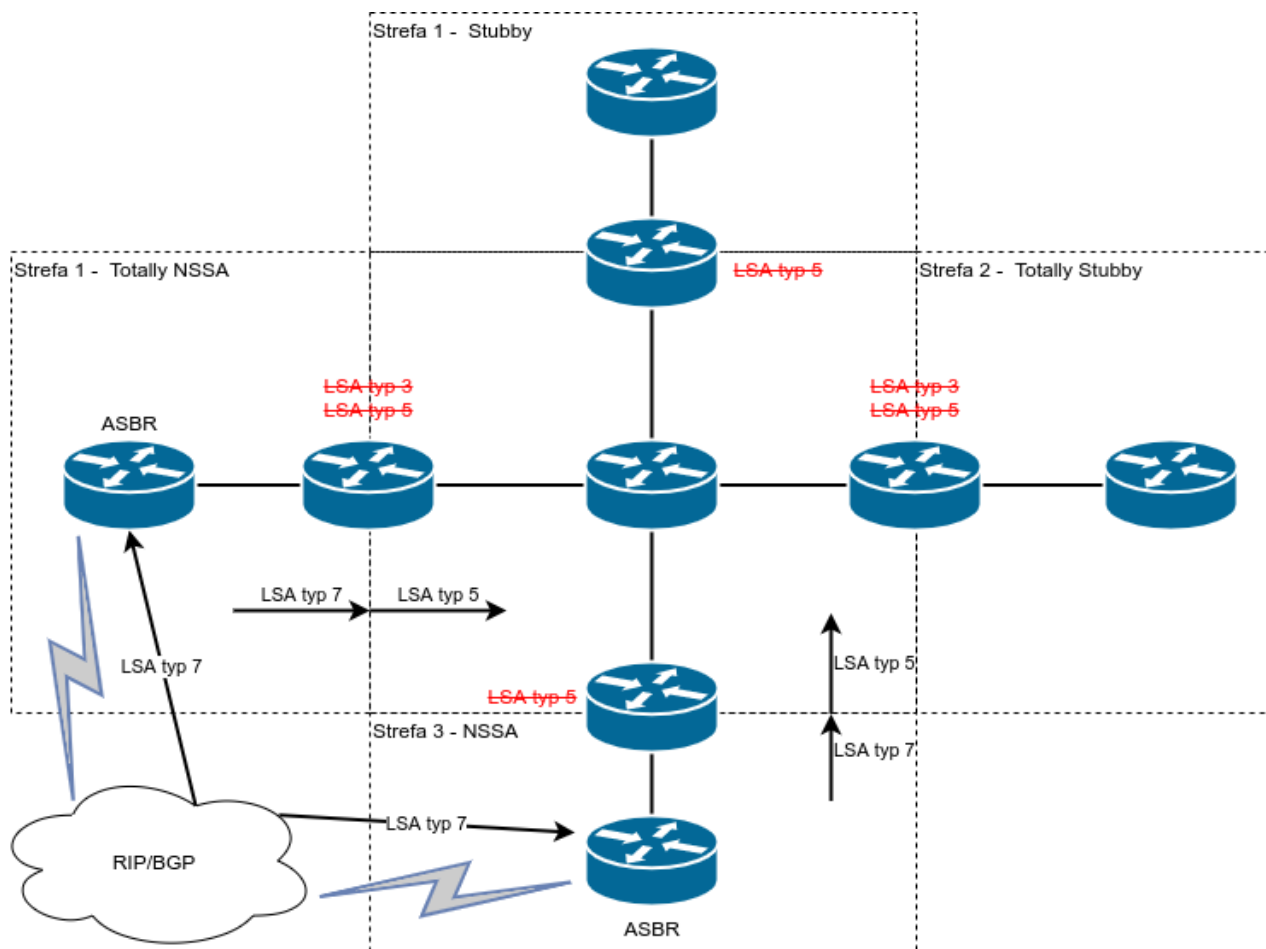
3. Not so Stubby area(NSSA).

Jak Stubby Area, ale może posiadać ASBR a jego drogi są rozprowadzane przez LSA typ 7 (konwertowane na LSA5 przy wyjściu ze strefy).

4. Totally NSSA.

Jak NSSA, ale może posiadać ASBR a jego drogi są rozprowadzane przez LSA typ 7 (konwertowane na LSA5 przy wyjściu ze strefy).

## Lab1: Konfiguracja routingu statycznego i dynamicznego



Rys X.X

Różne rodzaje stub area, widać że NSSA dostaje drogi zewnętrzne przez ASBR (typ7)

### 2.3.2.5 Komunikacja pomiędzy routerami OSPF

OSPF używa jednego 24 bajtowego nagłówka:

0										1										2										3																			
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1																		
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			
Version #										Type										Packet length																													
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			
										Router ID																																							
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			
										Area ID																																							
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			
										Checksum																				AuType																			
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			
										Authentication																																							
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			
										Authentication																																							
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+																			

OSPF Header RFC 2328 appendix A

## Lab1: Konfiguracja routingu statycznego i dynamicznego

---

Version – wersja protokołu (1/2)

Type – jeden z pięciu typów pakietów OSPF

5 typów:

1. Hello  
odpowiedzialny za nawiązywanie i utrzymywanie sąsiadów. Wysyłane regularnie (*hello-interval*)
2. Database Description  
Pakiet synchronizujący bazy sąsiednich routerów, posiadają listę LSA. Jeżeli sąsiedni router zobaczy wpis nowszy niż ma w swojej bazie wysyła LS Request
3. Link State Request  
Zapytanie wysyłane przez router kiedy sąsiedni DD ma LSA nowsze niż we własnej bazie.
4. Link State Update  
Każdy pakiet zawiera routing, i informacje o topologii części sieci
5. Link State Ack  
Pakiet używany do zatwierdzania innych pakietów OSPF. Wiele pakietów może być zatwierdzonych jednym pakietem.

### 2.3.3 Konfiguracja daemona routingu dynamicznego (quagga)

#### 2.3.3.1 Instalacja quagga

```
# apt install quagga quagga-doc
```

```
# edycja /etc/quagga/daemons
```

```
+ zebra=yes
```

```
+ ospfd=yes
```

```
+ ripd=yes
```

stworzenie pustych plików konfiguracyjnych (inaczej usługi nie zostaną uruchomione *dependencies not met*)

```
# touch /etc/quagga/{zebra.conf,ospfd.conf,ripd.conf}
```

#### 2.3.3.2 Konfiguracja podstawowa (OSPF)

uruchamiamy usługę:

```
# systemctl start ospfd.service //dla systemów z systemd jeżeli twój system używa innego systemu init skonsultuj się z jego konfiguracją
```

Łączymy się do servera ospfd za pomocą *telnet* lub *vttysh*:

numer portu możemy poznać za pomocą *nmap localhost | grep ospf*

## Lab1: Konfiguracja routingu statycznego i dynamicznego

---

```
$ enable //zeby przejsc do trybu uprawnionego
```

```
# configure terminal //zeby przejsc do trybu konfiguracji
```

```
# router ospf //zeby przejsc do konfiguracji OSPF
```

```
# network $adres_sieci/$subnet area $area //podajemy sieć którą będziemy ogłaszać
```

Po dodaniu naszych sieci do rozgłoszenia wychodzimy z konfiguracji i komendą *write* zapisujemy obecną konfigurację do pliku.

### 2.3.3.3 Konfiguracja zaawansowana (OSPF)

do punktu 3.

```
# cost $koszt //manualne ustawienie kosztu prześcia przez interfejs (domyslnie kalkulowane z trybu prędkości interfejsu (np. 100 BASE-T)
```

```
# dead-interval $sekundy //po jakim czasie nie odpowiadający router sąsiedni będzie uznany za martwy
```

```
# hello-interval $sekundy //specyfikuje co jak czas pakiet hello będzie wysyłany
```

```
# priority $numer //priorytet routera
```

```
# transmit-delay $sekundy //dodaje czas do pola czasu w pakiecie LSA
```

```
# retransmit-interval $sekundy //ustawienie częstotliwości retransmisji dopóki pakiet LSA nie zostanie potwierdzony
```

### 2.3.3.4 Konfiguracja podstawowa (RIP)

uruchamiamy usługę:

```
# systemctl start ripd.service
```

Łączymy się do servera ripd za pomocą *telnet* lub *vttysh*:

numer portu możemy poznać za pomocą *nmap localhost | grep ripd*

```
$ enable //zeby przejsc do trybu uprawnionego
```

```
# configure terminal //zeby przejsc do trybu konfiguracji
```

```
# router rip //zeby przejsc do konfiguracji OSPF
```

```
# rip ip send version [1,2,1 2] //zeby wybrać wersję RIP której używamy
```

```
# [no] network $adres_sieci/$subnet //podajemy sieć którą będziemy ogłaszać
```

```
# [no] network $nazwainterfejsu //podajemy sieć którą będziemy ogłaszać (przez interfejs)
```



Po dodaniu naszych sieci do rozgłoszenia wychodzimy z konfiguracji i komendą *write* zapisujemy obecną konfigurację do pliku.

### 2.3.3.4 Konfiguracja zaawansowana (RIP)

**do punktu 3.**

*# distance [1-255] [\$address/\$maska]* //ustawienie maksymalnego dystansu (liczby skoków) dla sieci

*# timers basic [\$update \$timeout \$garbagecollect]* //pokazuje/zmienia czasy

*//\$update* – czas wysyłania aktualizacji do *neighbour*(sąsiadów)

*//\$timeout* – czas do unieważnienia drogi

*//\$garbagecollect* – czas do usunięcia nieważnej drogi

### 2.3.4 Weryfikacja działania protokołu

<Ping, traceroute, ip route show, tcpdump (pakietów kontrolnych)? pomiędzy stacjami roboczymi, routing table routerów>

<zdjecia>

## 3. Testowanie „dynamiczności” routingu

### 3.1 Awaria interfejsu

Używamy tylko jednego demona (OSPF lub RIP)

1. Weryfikujemy że pakiety pomiędzy S1 i S2 są wysyłane przez R1 i R2 (traceroute)
2. Wywołujemy ping pomiędzy S1 i S2 i odwrotnie
3. Odłączamy interfejs pomiędzy R1 i R2 (*set link down* na jednym z routerów lub fizycznie odłączamy sieć)
4. Logujemy komunikacje pomiędzy routerami
5. Obserwujemy kiedy połączenie powróci i weryfikujemy że używa R3 (traceroute)
6. Podłączamy interfejs
7. Obserwujemy kiedy (i jeżeli) połączenie przełączy się na bezpośrednie R1-R2 (traceroute)

### 3.1.1 RIP

TBD

### 3.1.2 OSPF

TBD

## 3.2 Przeciążenie na sieci (OSPF)

Przeciążenie sieci (lub wolne połączenie) zasymulujemy dodając interfejsowi wysoki koszt

Ćwiczenie do OSPF:

1. Weryfikujemy że pakiety pomiędzy S1 i S2 są wysyłane przez R1 i R2 (*traceroute*)
2. Wywołujemy ping pomiędzy S1 i S2 i odwrotnie.
3. Nadajemy wysoki koszt interfejsowi S1 → S2
4. Logujemy komunikacje pomiędzy routerami
5. Sprawdzamy czy komunikacje w dwie strony przechodzi tą samą drogą (*traceroute*)
6. Jeżeli nie, to nadajemy wysoki koszt S2 → S1
7. Weryfikujemy że obie drogi przechodzą przez R3 (*traceroute*)

### 3.2.1 Wyniki

TBD

## 3.3 Sprawdzanie Overhead-u

W tym ćwiczeniu sprawdzimy wpływ ilości pakietów kontrolnych na działanie sieci porównując je z domyślnymi ustawieniami. Ćwiczenie te może być niereprezentowalne ze względu na dużą ilość zmiennych ciężkich do kontroli (CPU Boost, hypervisor scheduler itp.).

Maszyna fizyczna to thinkpad t440 z i7-4600u @2.1 GHz (2Cores/4Threads) z Ubuntu 18.04.3 LTS

Maszyny virtualne to Debian 9 z jednym logicznym rdzeniem i 1GB RAM

Skrypty: TBDBD (mogą jeszcze nie działać)

<https://github.com/p7tryk/administracja/blob/master/netcatclient.sh>

<https://github.com/p7tryk/administracja/blob/master/netcatserver.sh>

### 3.3.1 Kontrola

Testem szybkości będzie wysłanie wygenerowanego pliku z losowymi danymi 1GB (*dd if=/dev/random of=pliktestowy.bin bs=4M count=256*) za pomocą *netcat*

## Lab1: Konfiguracja routingu statycznego i dynamicznego

---

Monitorować również będziemy zużycie procesora na serwerach za pomocą *sar -u* (pakiet *sysstat*).

Poniższe kroki wykonać dla RIP i OSPF:

1. Zweryfikować połączenie pomiędzy S1 i S2 przez R2 i R1.
2. Otworzyć serwer *netcat* na S2.
3. Zarejestrować czas i rozpocząć rejestrowanie zużycia procesora.
4. Wysłanie pliku.
5. Zarejestrować czas i zatrzymać rejestrowanie zużycia procesora.

### 3.3.2 RIP

Zmienimy liczniki serwera RIP na 1 180 180 (*\$hello*, *\$dead*, *\$garbage-collect*) a następnie wykonamy kroki z 3.3.1.

### 3.3.3 OSPF

Zmienimy liczniki serwera OSPF na 1 40 5 (*\$hello-interval*, *\$dead*, *\$retransmit*) a następnie wykonamy kroki z 3.3.1.

### 3.3.4 Wyniki

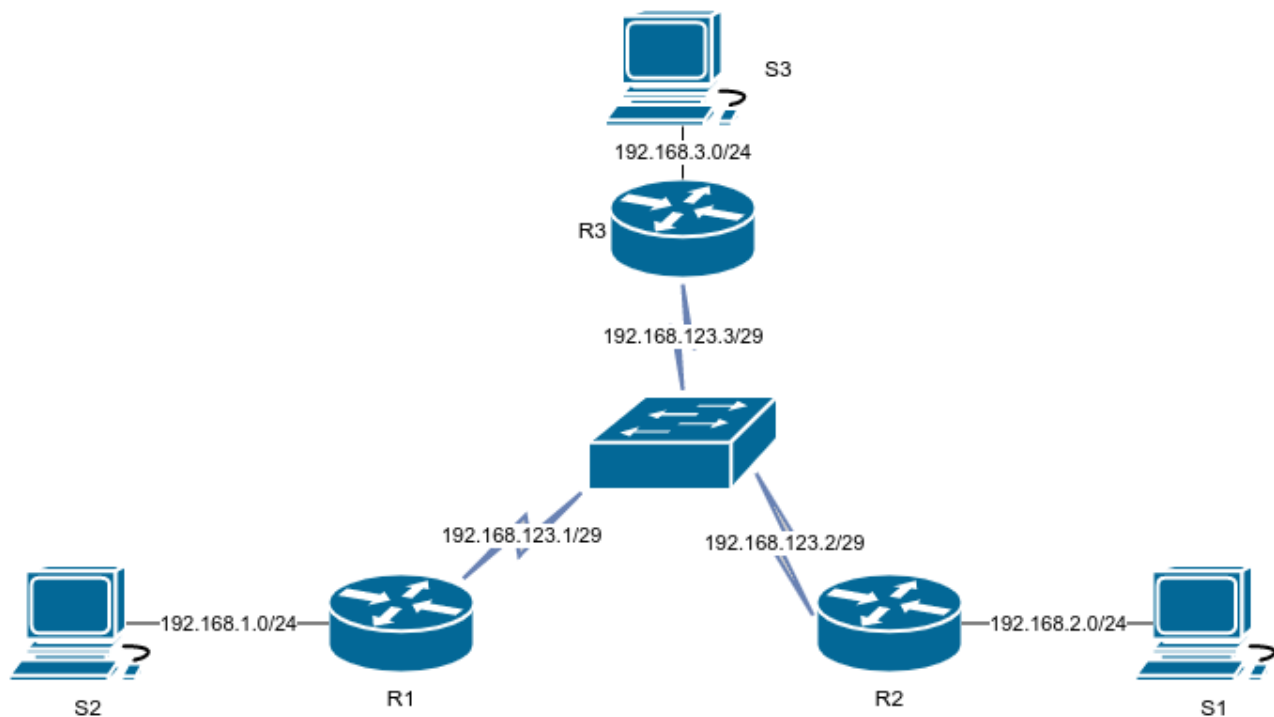
TBD

## 3.4 Designated router (OSPF)

Zmieniamy topologie sieci

## Lab1: Konfiguracja routingu statycznego i dynamicznego

---



Rys X.X

Ćwiczenie do OSPF:

1. Sprawdzamy który router jest DR (*show ip ospf neighbour*)
2. Weryfikujemy że routery komunikują się tylko z DR (*tcpdump*)
3. Podwyższamy priorytety routerów które **nie jest DR** do różnych wartości.
4. Sprawdzamy czy nowy DR będzie wynegocjowany.
5. Jeżeli nie, restartujemy usługi (lub serwery)
6. Sprawdzamy czy DR i backup DR zostały wynegocjowane wg naszych priorytetów.

### 3.4.1 Wyniki

TBD

## 4. Wnioski

TBD

### 4.1 Routing statyczny vs dynamiczny

TBD

T

## 4.2. RIP vs OSPF

TBD

## 4.3 Dopasowanie do wymagań i sytuacji

TBD

## 5. Bibliografia

TBD TBD

RFC 2453 RIP Version 2

RFC 2328 OSPF Version 2

Quaga documentation:

<https://www.nongnu.org/quagga/docs>

Cisco documentation:

[https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute\\_ospf/configuration/xe-16/iro-xe-16-book/iro-cfg.html#GUID-4AABEB56-2125-488B-B5A4-A5650F3159BB](https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute_ospf/configuration/xe-16/iro-xe-16-book/iro-cfg.html#GUID-4AABEB56-2125-488B-B5A4-A5650F3159BB)

<https://www.cisco.com/c/en/us/support/docs/ip/open-shortest-path-first-ospf/9237-9.html#q3a>

<https://www.ciscopress.com/articles/article.asp?p=26919>

<https://www.freeccnaworkbook.com/workbooks/ccna/configuring-basic-ospf>

TBD – muszę znaleźć ponownie