LABORATORIUM SIECI WIELO-USŁUGOWE

PROTOKOŁY RUTINGU W SIECIACH PAKIETOWYCH

POLITECHNIKA WARSZAWSKA

INSTYTUT TELEKOMUNIKACJI

Warszawa, 2013

1 Wstęp

Celem ćwiczenia jest zapoznanie z konfiguracją i działaniem podstawowych algorytmów wyboru drogi w sieciach pakietowych. Sprawdzane jest zachowanie mechanizmu doboru trasy w razie wystąpienia awarii łącza. Ćwiczenie zostanie wykonane z wykorzystaniem oprogramowania Quagga, umożliwiającego uruchomienie protokołów rutingu. Quagga została zainstalowana na systemie operacyjnym Tiny Core Linux, zainstalowanym w maszynach wirtualnych.

2 Mechanizm rutingu w sieciach pakietowych

Algorytmy wyboru drogi w sieciach pakietowych są realizowane w warstwie trzeciej modelu ISO OSI (warstwie sieciowej). Mechanizm doboru trasy w sieciach WAN i MAN jest określany jako **ruting**. **Protokół rutingu** jest metodą, zgodnie z którą ruter (układ działający w warstwie fizycznej, łącza danych i sieciowej) wymienia informacje z innymi ruterami i dokonuje wyboru trasy.

2.1 Zasada działania mechanizmu rutingu

Jeżeli w sieci stosowane są usługi **datagramowe** (np. w sieci Internet opartej o protokół IP), wówczas decyzje o wyborze drogi podejmowane są dla każdego pakietu, w momencie jego odebrania w węźle. Ruter analizuje wtedy adres miejsca przeznaczenia w nagłówku IP i kieruje pakiet na odpowiednie łącze wyjściowe, zgodnie z odpowiadającym temu adresowi wpisem w tzw. **tablicy rutingu**. Format nagłówka pakietu IP wersji 4. przedstawiono na rysunku 1., natomiast format nagłówka pakietu IP wersji 6. Przedstawiono na rysunku 2.

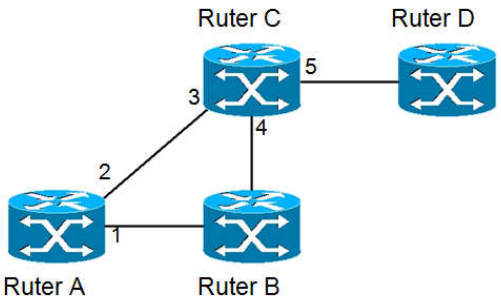
<table>
<tr><td>Wersja (4 bity)</td><td>Dł. nagłówka (4 bity)</td><td>TOS (8 bitów)</td><td colspan="2">Długość całkowita (16 bitów)</td></tr>
<tr><td colspan="3">Identyfikacja (16 bitów)</td><td>Znaczniki (3 bity)</td><td>Przesunięcie fragmentacji (13 bitów)</td></tr>
<tr><td colspan="2">Czas życia (8 bitów)</td><td>Protokół (8 bitów)</td><td colspan="2">Suma kontrolna nagłówka (16 bitów)</td></tr>
<tr><td colspan="5">Adres źródłowy (32 bity)</td></tr>
<tr><td colspan="5">Adres przeznaczenia (32 bity)</td></tr>
<tr><td colspan="5">Opcje (jeśli są)</td></tr>
<tr><td colspan="5">Dane</td></tr>
</table>

Rysunek 1. Format nagłówka pakietu IPv4

<table>
<tr><td>Wersja (4 bity)</td><td>Klasa ruchu (8 bitów)</td><td colspan="2">Etykieta przepływu (20 bitów)</td></tr>
<tr><td colspan="2">Długość danych (16 bitów)</td><td>Następny nagłówek (8 bitów)</td><td>Limit przeskoków (8 bitów)</td></tr>
<tr><td colspan="4">Adres źródłowy (128 bitów)</td></tr>
<tr><td colspan="4">Adres przeznaczenia (128 bitów)</td></tr>
</table>

Rysunek 2. Format nagłówka pakietu IPv6

Przykładowo załóżmy, że w sieci, którą przedstawia rysunek 3., ruter A odbiera pakiet skierowany do podsieci skojarzonej z ruterem D. Algorytm rutingu działający w ruterze A powinien skierować ten pakiet na port wyjściowy nr 2. Następny ruter na drodze tego pakietu, oznaczony jako C, także analizuje adres miejsca przeznaczenia i na tej podstawie podejmuje decyzję o skierowaniu tego pakietu na łącze wyjściowe skojarzone z portem nr. 5. W efekcie, mimo iż każdy ruter na drodze pakietu podejmuje decyzję niezależnie, na podstawie informacji zgromadzonych w lokalnej tablicy rutingu, pakiet powinien ostatecznie dotrzeć do miejsca przeznaczenia. Warunkiem prawidłowego działania tego mechanizmu jest utrzymanie spójności dróg skonfigurowanych w tablicach rutingu w poszczególnych węzłach sieci.



Rysunek 3. Przykładowa sieć IP

W sieciach, w których przekaz pakietów odbywa się na zasadzie **połączeń wirtualnych** (np. w sieci Frame Relay lub ATM), wybór drogi jest realizowany tylko w momencie zestawiania nowego połączenia wirtualnego. Wszystkie pakiety należące do danego połączenia wirtualnego będą przesyłane tą samą drogą. Rozwiązanie takie jest nazywane *sesją wyboru drogi*, ponieważ ustalona droga obowiązuje przez czas trwania sesji użytkownika.

Ogólnie, strategie wyboru drogi w sieci można zaliczyć do dwóch klas. **Ruting statyczny** zakłada, że tablice rutingu są skonfigurowane na stałe przez administratora sieci i nie zmieniają się w trakcie jej działania. W przypadku **rutingu dynamicznego**, drogi do poszczególnych miejsc przeznaczenia mogą zmieniać się w zależności od zmian topologii (np. powodowanych przez awarie węzłów lub łączy), lub też zmian aktualnych warunków ruchowych w sieci.

2.2 Protokoły rutingu

Realizacja rutingu dynamicznego wymaga zastosowania tzw. **protokołu rutingu**, który służy do wymiany informacji pomiędzy ruterami w sieci IP. Celem działania protokołu rutingu jest zagwarantowanie, że wszystkie rutery posiadają aktualny stan wiedzy o osiągalnych sieciach i warunkach panujących w sieci i mogą w sposób niezależny konstruować spójne tablice rutingu. Wymiana wiadomości rutingowych odbywa się okresowo (np. co 1 minutę), lub w wyniku zaobserwowania przez rutery pewnych zdarzeń powodujących konieczność zmiany dróg w sieci (np. awarii łącza).

Ze względu na rodzaj wymienianych wiadomości i sposób konstrukcji tablicy rutingu, można wyróżnić dwa typy protokołów rutingu dynamicznego:

- Protokoły „**wektora odległości**” (ang. distance-vector routing) zakładają, że ruter przesyła

do *wszystkich swoich sąsiadów* aktualną kopię tablicy rutingu. W ten sposób, sąsiad otrzymuje informację o adresach podsieci, które są osiągalne z danego rutera *bezpośrednio lub za pośrednictwem innych ruterów*, do których ruter ten zna drogę (wraz z kosztem, czyli **metryką** takiego połączenia). Ruter nie ma wiedzy o całej topologii sieci. Ścieżki są obliczane za pomocą algorytmu Bellmana-Forda.

- Protokoły „**stanu łącza**" (ang. link-state routing) zakładają, że każdy ruter rozsyła do wszystkich innych ruterów w sieci tylko informację o adresach podsieci i metrykach łączy bezpośrednio do niego podłączonych (komunikaty *link-state advertisement*). Dzięki temu każdy ruter w sieci może utrzymywać aktualną bazę danych informacji na temat topologii całej sieci, co pozwala mu na obliczenie, za pomocą algorytmu Dijkstry, najkrótszych dróg do poszczególnych miejsc przeznaczenia.

Sieć IP jest zbudowana z niezależnych domen, zwanych **systemami autonomicznymi AS** (ang. Autonomous Systems). Domena składa się z ruterów znajdujących się pod wspólną administracją. Każda z domen ma przydzielony numer AS, przydzielony przez organizację IANA (Internet Assigned Numbers Authority). W każdej z tych domen może być uruchomiony niezależnie dowolny (jeden lub więcej) protokół rutingu wewnątrzdomenowego. Do tego, by zapewnić łączność pomiędzy ruterami znajdującymi się w różnych domenach, potrzebny jest protokół rutingu zewnętrznego, pozwalający na wymianę informacji o ścieżkach pomiędzy systemami autonomicznymi.

Protokołami rutingu wewnątrzdomenowego są m.in. protokoły:

- **RIP** (Routing Information Protocol) - należy do klasy protokołów „**wektora odległości**". Stosowany przez niego algorytm rutingu oblicza koszt danej drogi na podstawie ilości skoków (liczby łączy), jakie musi pokonać pakiet na drodze od źródła do przeznaczenia. Zatem kary przypisane poszczególnym łączom są zawsze równe 1.
- **OSPF** (Open Shortest Path First) - należy do klasy protokołów „**stanu łącza**". Metryki poszczególnych dróg są obliczane na podstawie wartości kar przypisanych łączom w sposób administracyjny. *Jeżeli kara dla każdego łącza jest równa 1, wówczas drogi wyznaczane przez algorytmy OSPF oraz RIP są identyczne.*
- IS-IS (Intermediate System to Intermediate System) – także jest protokołem "stanu łącza". Algorytm wyboru drogi jest w tym przypadku w zasadzie równoważny algorytmowi zastosowanemu w protokole OSPF. Wagi poszczególnych dróg także są obliczane na podstawie kar administracyjnie przypisanych poszczególnym łączom. Jeśli kary przypisane łączom są takie same jak w przypadku protokołu OSPF, wyznaczone drogi są identyczne.
- IGRP (Interior Gateway Routing Protocol) – należy do klasy protokołów „wektora odległości". Całkowita metryka danej drogi jest obliczana na podstawie wartości metryk składowych: współczynnika przepływności, współczynnika wykorzystania łącza, współczynnika opóźnienia przekazu pakietów oraz współczynnika niezawodności.

Protokołem rutingu międzydomenowego wykorzystywanym obecnie w Internecie jest **BGP (Border Gateway Protocol)**. **Celem BGP jest wymiana między systemami autonomicznymi wiadomości o osiągalności prefiksów podsieci,** dzięki czemu możliwy jest ruting wolny od pętli pomiędzy systemami autonomicznymi. BGP jest protokołem „**wektora ścieżek**" (ang. path-vector).

Rutery BGP, by mogły wymieniać ze sobą wiadomości, wykorzystują protokół transportowy **TCP** (port 179). Każde dwa rutery, które mają zestawioną ze soba sesję TCP nazywane są sąsiadami. Rutery odpowiadające za wymianę wiadomości z ruterami należącymi do innych domen nazywane są **ruterami brzegowymi** (ang. border router). Wyróżniane są dwa rodzaje sesji BGP: eBGP (ang. exterior BGP), jeśli sąsiedzi wymieniający się wiadomościami BGP są zlokalizowani w dwóch różnych domenach, oraz iBGP (ang. interior BGP), jeśli rutery BGP znajdują się w ramach jednej domeny. Wiadomościami, które wymieniają między sobą rutery BGP są:

- OPEN – wiadomość wysyłana przez rutery po utworzeniu sesji TCP między nimi w celu otwarcia sesji BGP,
- UPDATE – za jej pomocą można rozgłaszać nowe lub aktualizowane ścieżki włącznie z ich atrybutami i wycofywać z użycia te ścieżki, które są nieaktualne,
- NOTIFICATION – wiadomość wysyłana w razie wykrycia błędu, połączenie BGP jest kończone niezwłocznie po wysłaniu tej wiadomości,
- KEEPALIVE – służy do sprawdzania czy sąsiad jest osiągalny.

Atrybutami ścieżek między domenami są m.in.:

- Origin – wskazuje czy informacja została wygenerowana przez wewnętrzny protokół rutingu czy zewnętrzny (BGP),
- AS_Path – lista AS-ów, przez które przechodzi dana ścieżka,
- Next_hop – adres następnego rutera brzegowego,
- Multi_Exit_Disc – do rozsyłania informacji o drogach do danego AS,
- Local_Pref – atrybut wykorzystywany do poinformowania ruterów z tej samej domeny o preferowanej ścieżce wyjścia z domeny.

Ścieżka jest wybierana w kolejności na podstawie algorytmu (preferowane ścieżki wg):

- najwyższa waga (WEIGHT, parametr Cisco),
- najwyższa wartości Local_Pref,
- najkrótsza ścieżka AS_path,
- najniższy typ Origin,
- najniższa wartość MED (dla dróg z tego samego AS),
- ścieżki z eBGP przed ścieżkami z iBGP,
- niższa metryka IGP (Interior Gateway Protocol, protokół wewnątrzdomenowy),
- itd.

2.3 Algorytmy wyboru drogi

Odpowiedni **algorytm rutingu** pozwala ruterowi wyznaczyć drogę do danego miejsca przeznaczenia, optymalizując przy tym ogólnie rozumiany koszt takiej drogi. Zwykle, algorytmy rutingu są oparte na algorytmach znajdowania najkrótszych ścieżek w grafie (np. algorytm *Dijkstry* lub *Bellmana-Forda*), przyjmując przy tym różne **metryki kosztu poszczególnych dróg** . Takimi metrykami mogą być:

- liczba łączy, przez które prowadzi dana droga,
- opóźnienie wnoszone przez daną drogę,
- przepustowość drogi, rozumiana jako najmniejsza przepływność łącza wchodzącego w jej skład,

- maksymalne obciążenie łączy wchodzących w skład danej drogi (zakłada się, że łącza w większym stopniu obciążone ruchem wnosą w efekcie większe opóźnienie),
- arbitralnie konfigurowany przez administratora koszt danej drogi.

Metryki dróg są obliczane przez algorytm rutingu na podstawie wartości kar przypisanych poszczególnym łączom, które wchodzą w skład danej drogi. Przykładowo, w sieci przedstawionej na rysunku 3., jeśli algorytm rutingu bierze pod uwagę liczbę łączy jako metrykę kosztu drogi, najkrótsza droga pomiędzy ruterami A i D to: A-C-D. W jej skład wchodzą łącza A-C i C-D, każde z karą równą 1. Całkowita metryka kosztu drogi A-C-D jest zatem równa 2. Dla porównania, metryka drugiej dopuszczalnej, lecz „droższej" drogi A-B-C-D jest równa 3.

Różne algorytmy rutingu mogą wybrać różne trasy jako najlepsze, zależnie od rodzaju użytych metryk i samego algorytmu wyboru drogi. Niezależnie od rodzaju algorytmu rutingu, można wymienić kilka cech, którymi powinny się charakteryzować wszystkie z nich. Należą do nich:

- poprawność obliczonych dróg,
- łatwość implementacji,
- odporność na uszkodzenia elementów sieci,
- stabilność,
- sprawiedliwość,
- optymalność.

Dwa pierwsze wymagania można uznać za oczywiste. Odporność na uszkodzenia sieci oznacza, że algorytm rutingu sprawnie realizuje swoje funkcje mimo uszkodzeń elementów sieci. Algorytm jest **stabilny**, jeśli mechanizm wyboru dróg dąży do równowagi. Przykładowo, jeśli drogi są wybierane na podstawie chwilowego obciążenia łączy, algorytm rutingu będzie starał się kierować ruch na łącza o małym obciążeniu, co w efekcie doprowadzi do zwiększenia ich obciążenia ruchowego. Takie działanie algorytmu może prowadzić do oscylacji i braku stabilności wybranych dróg w dłuższym okresie czasu. **Sprawiedliwość** określa, czy algorytm nie wykazuje tendencji do wyszukiwania „lepszych" dróg dla niektórych relacji w sieci, kosztem pogorszenia dróg znajdowanych dla pozostałych relacji.

Projektując optymalny algorytm rutingu, należy przede wszystkim podjąć decyzję o tym, która z metryk podlega optymalizacji. Często jako cele przyjmuje się: (1) minimalizację średniego czasu opóźnienia pakietu, lub (2) maksymalizację ogólnej przepustowości sieci. Przy czym należy pamiętać, że w każdym systemie masowej obsługi obciążenie bliskie nasycenia systemu ($\rho \rightarrow 1$) oznacza dużą wartość opóźnienia. Zatem cele te są sprzeczne. Kompromisowym rozwiązaniem, przyjmowanym dla wielu sieci, jest próba zminimalizowania liczby etapów pokonywanych przez pakiet w trakcie transmisji przez sieć, co często prowadzi do zmniejszenia opóźnienia pakietu, a także minimalizuje całkowitą ilość zasobów zajmowanych podczas przekazu pakietu przez sieć.

3 Przykładowe pytania na kolokwium

1. Do czego służą protokoły rutingu?
2. W której warstwie modelu OSI działają protokoły rutingu?
3. Czym różni się protokół rutingu typu „wektora odległości” od protokołu typu „stanu łącza”?
4. Jakie metryki poszczególnych dróg w sieci może brać pod uwagę algorytm rutingu?
5. Podaj najważniejsze cechy (typ protokołu, kary łączy) protokołów: RIP, OSPF, IS-IS.
6. Kiedy protokoły OSPF, IS-IS i RIP są równoważne?
7. Do czego służy protokół BGP?

4 Dodatkowa literatura

[1] A.S. Tanebaum, Computer Networks (3. edycja). Prentice Hall, 1997, Rozdział 5.2 (*Routing Algorithms,* str. 345), oraz 5.5.5 (*OSPF,* str. 424)

[2] W.Stallings, Data and Computer Communications (5. edycja), Prentice Hall, 1997, Rozdział 9.2 (*Routing,* str. 264), Rozdział 16.3 (*The Internet Protocol*, str. 543-546), 16.4 (*Routing protocols*, *BGP*, str. 549-566)

[3] J.Woźniak, K. Nowicki, Sieci LAN, MAN, WAN - protokoły telekomunikacyjne, Wydawnictwo Fundacji Postępu Telekomunikacyjnego, 1998, Rozdział 9.3.4 (*Protokoły wyboru trasy*, str. 464)

[4] RFC4721 A Border Gateway Protocol 4 (BGP-4)

[5] Dokumentacja Quagga, http://www.nongnu.org/quagga/docs/quagga.pdf

[6] Quagga, http://www.nongnu.org/quagga/

[7] Tiny Core Linux, http://tinycorelinux.net/

[8] Oracle VM Virtual Box, https://www.virtualbox.org/

5 Oprogramowanie – konfiguracja, skrócona instrukcja obsługi

Ćwiczenie zostanie wykonane z wykorzystaniem oprogramowania Quagga, umożliwiającego uruchomienie protokołów rutingu. Narzędzie to zostało zainstalowane w maszynach wirtualnych z zainstalowanym systemem operacyjnym Tiny Core Linux. Maszyny wirtualne są zarządzane przez menedżer maszyn wirtualnych Virtual Box.

5.1 Virtual Box

Na potrzeby każdej z maszyn wirtualnych w Virtual Box przeznaczono 128 MB pamięci RAM i obraz dysku o rozmiarze 100 MB.

Klonowanie maszyny wirtualnej:
W razie potrzeby wykonania kopii maszyny wirtualnej, należy wyłączyć maszynę, której kopię chcemy wykonać. Następnie z listy maszyn wirtualnych w VirtualBox należy wybrać maszynę wirtualną, kliknąć na niej prawym przyciskiem myszy i wybrać z listy *Clone.* Następnie trzeba nadać nazwę kopii i zaznaczyć *Reinitialize the MAC address of all network cards*, kliknąć *Dalej*, zaznaczyć opcję *Full Clone* i wykonać kopię klikając na przycisk *Clone*.

Ustawienie liczby dostępnych interfejsów sieciowych w zainstalowanym w maszynie wirtualnej systemie operacyjnym odbywa się w ustawieniach maszyny wirtualnej w sekcji *Sieć*. W zależności od potrzeb należy włączyć wybrane karty sieciowe, zaznaczając *Włącz kartę sieciową*. Kartę sieciową należy podłączyć do sieci wewnętrznej (wybrać z listy *Sieć wewnętrzna*), nadać nazwę sieci – wpisać nową nazwę lub wybrać istniejącą sieć z listy.

5.2 Tiny Core

Tiny Core jest minimalistycznym systemem operacyjnym Linux (plik z dystrybucją ma rozmiar 12 MB), którego funkcjonalność można rozszerzać poprzez dodatkowe moduły. System ten został zaprojektowany do uruchamiania swojej kopii w pamięci RAM, dlatego po jego wyłączeniu zmiany wprowadzone przez użytkownika nie zapisują się automatycznie.

Aby nie utracić danych, można zrobić kopię zapasową. Są dwa sposoby zrobienia kopii przed wyłączeniem systemu lub jego restartem:
1. wpisać w terminalu komendę: *filetool.sh -b*
i następnie wyłączyć (komenda: *sudo poweroff*) lub zrestartować system (komenda: *sudo reboot*).
2. kliknąć lewym lub prawym klawiszem myszy na pulpicie, wybrać z listy opcję *Exit*, a w pojawiającym się oknie wybrać z listy *Backup Options* - *Backup*

Lista zachowywanych folderów i plików znajduje się w pliku */opt/.filetool.lst*
(m.in. */opt*, */home* oraz */usr/local/etc/quagga* – folder zawierający pliki konfiguracyjne Quagga).
By ułatwić konfigurację plików Quagga, w systemie zainstalowano edytor tekstu *vi* oraz *mcedit,* a także przeglądarkę plików *mc* (Midnight Commander).

5.3 Quagga

Quagga jest oprogramowaniem umożliwiającym uruchomienie w systemie Linux protokołów rutingu, takich jak: RIPv1, RIPv2, RIPng, OSPFv2, OSPFv3, IS-IS, BGP-4 i BGP-4+. Narzędzie wspiera protokoły rutingu IPv4 oraz IPv6.

Pliki konfiguracyjne *zebra.conf*, *ospfd.conf* oraz *bgpd.conf* znajdują się w folderze */usr/local/etc/quagga*.
Pliki wykonywalne *zebra*, *ospfd*, *bgpd* znajdują się w folderze */usr/local/sbin*.
Zebra jest menedżerem daemonów i musi być uruchamiana jako pierwsza.
Pliki wykonywalne można uruchamiać ręcznie:
sudo /usr/local/sbin/<nazwa_pliku> &
Aby ułatwić włączenie plików wykonywalnych w folderze domowym */home/tc* pozostawiono skrypty *bgpd.sh*, *ospfd.sh*, *zebra.sh*, które można uruchomić za pomocą polecenia: *./<nazwa_skryptu>*
Aby wyłączyć daemon, należy zabić jego proces za pomocą polecenia *killall <nazwa_ daemona>* lub znaleźć numer procesu za pomocą (*ps –ef | grep <nazwa_daemona>*) i go zabić (*kill <numer_procesu>*).
W celu połączenia się do konsoli rutera należy wpisać komendę *vtysh*.

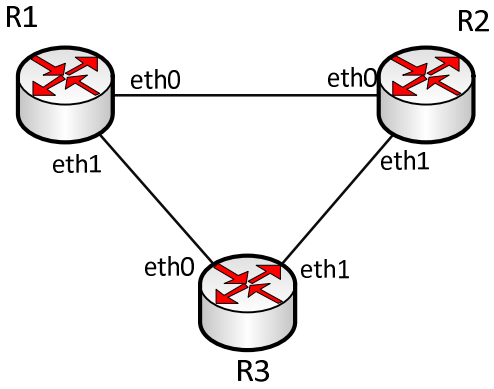
6 Zadania do realizacji w trakcie ćwiczenia

Celem ćwiczenia jest zapoznanie się ze sposobem konfiguracji oprogramowania do realizacji rutingu oraz zapoznanie się z działaniem protokołów rutingu wewnątrzdomenowego i międzydomenowego.

Przed przystąpieniem do ćwiczenia należy zaplanować adresację IP interfejsów ruterów oraz zapoznać się ze sposobem konfiguracji narzędzia Quagga.

6.1 Uruchomienie protokołu OSPF

Celem ćwieczenia jest uruchomienie w sieci protokołu rutingu OSPF i zapoznanie się z jego działaniem w sytuacji awarii łącza. Topologia sieci wykorzystana do testu protokołu OSPF została zaprezentowana na rysunku 4.



Rysunek 4. Topologia sieci do testu protokołu OSPF

Zadanie do wykonania:

1. Zaplanować adresację IPv4 interfejsów ruterów R1, R2, R3.
2. Uruchomić maszyny wirtualne R1, R2, R3 w programie Virtual Box.

W każdej z maszyn wirtualnych należy skonfigurować dwie karty sieciowe.

3. Skonfigurować interfejsy w każdym z ruterów poprzez plik konfiguracyjny *zebra.conf*.

Należy pamiętać, by w pliku *zebra.conf* została wpisana komenda umożliwiająca forwardowanie pakietów między interfejsami fizycznymi (*ip forwarding*).

Do czego służy komenda *link-detect* wykorzystywana przy konfiguracji interfejsów ruterów?

4. Skonfigurować w każdym z ruterów protokół rutingu za pomocą pliku konfiguracyjnego *ospfd.conf*. Ustawić rozgłaszanie podsieci dołączonych do ruterów (*network ... area 0*).
5. Uruchomić na każdym z ruterów oprogramowanie *zebra* oraz protokół OSPF – plik *ospfd*.
5. Zalogować się do każdego z ruterów z użyciem *vtysh*.
6. Sprawdzić konfigurację interfejsów (*show interface*).
7. Sprawdzić osiągalność adresów IP w sieci (komendy *ping, traceroute*). Wyjaśnić znaczenie uzyskanej informacji.
8. Sprawdzić i zapisać zawartość tablicy rutingu w każdym z ruterów (*show ip ...*). Jakie są ścieżki pomiędzy poszczególnymi ruterami?
9. Wyłączyć jeden z interfejsów w jednym z ruterów i sprawdzić co się dzieje w sytuacji awarii łącza. Czy każdy ruter jest osiągalny z pozostałych? Porównać tablice rutingu z sytuacją przed wystąpieniem awarii i po wystąpieniu awarii. Jakie są ścieżki pomiędzy poszczególnymi ruterami?
10. Jaką informację wyświetla polecenie: *show ip ospf database*?

Przydatne komendy:

a) Tiny Core:

route – sprawdzenie zawartości tablicy routingu

sudo ifconfig <eth> down – wyłączenie interfejsu *<eth>*

b) Quagga:

show running-config – komenda umożliwiająca sprawdzenie konfiguracji rutera

show interface description – komenda umożliwiająca sprawdzenie statusu interfejsów

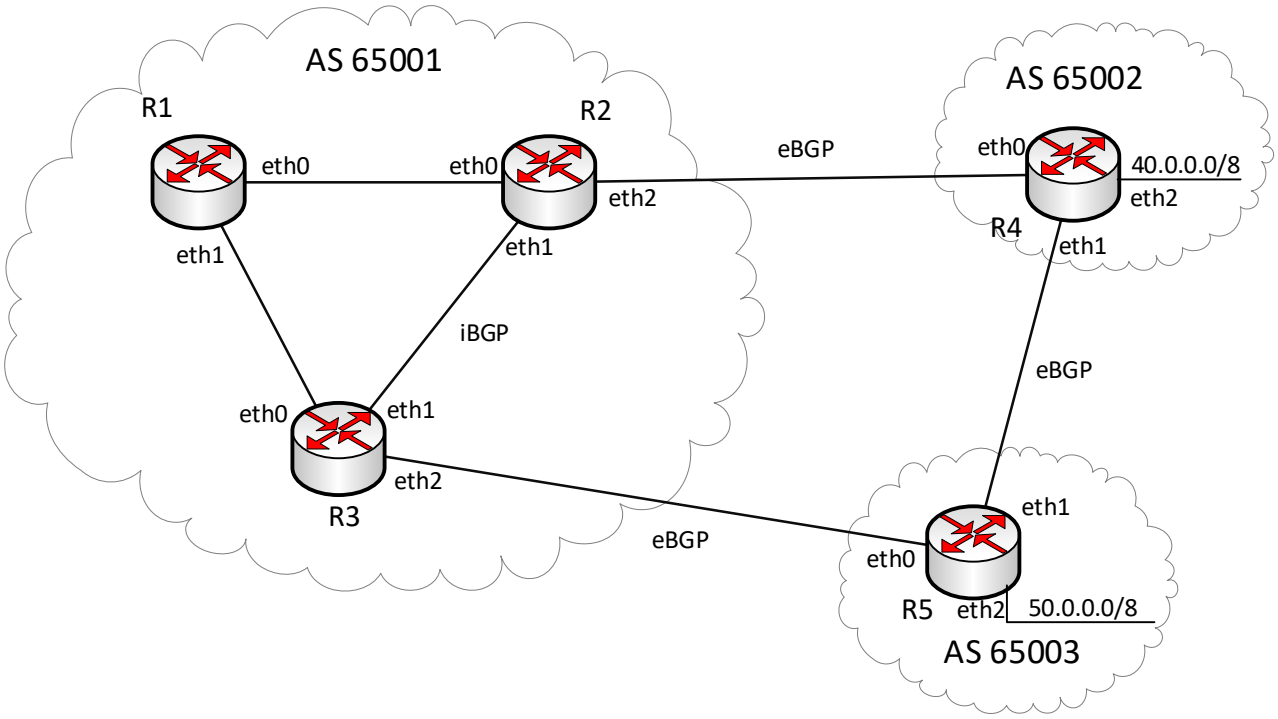
show ip forwarding – do sprawdzenia, czy włączony jest forwarding między interfejsami rutera

show ip ospf route – sprawdzenie tablicy rutingu OSPF

6.2 Zapoznanie się ze sposobem współpracy protokołu OSPF z protokołem BGP

Celem tego ćwiczenia jest zapoznanie się z konfiguracją protokołu międzydomenowego BGP tak, by możliwe było rozgłaszanie prefiksów podsieci między domenami. Zostanie również sprawdzone zachowanie protokołu w przypadku awarii łącza między dwiema domenami.

Topologia sieci wykorzystana w tym ćwiczeniu została przedstawiona na rysunku 5. System Autonomiczny AS 65001, składający się z 3 ruterów połączonych w topologii z ćwiczeń 6.1 i 6.2 został dołączony do dwóch innych domen – AS 65002 oraz AS 65003 (każda z tych dwóch sieci składa się z jednego rutera z uruchomionym protokołem BGP).



Rysunek 5. Topologia sieci do testu protokołu BGP

Zadania do wykonania:

1. Dodać poprzez oprogramowanie Virtual Box po jednej dodatkowej karcie sieciowej do ruterów R2 i R3. Uruchomić maszyny wirtualne, stanowiące rutery R4 i R5.
2. Przygotować adresację interfejsów ruterów R4, R5 i skonfigurować ją za pomocą pliku konfiguracyjnego *zebra.conf* w odpowiednich maszynach wirtualnych. Zaktualizować adresację interfejsów ruterów R1 i R2. Należy pamiętać, że zmiana konfiguracji wymaga restartu programu *zebra*.
3. W ruterach R2, R3, R4 i R5 należy skonfigurować za pomocą plików *bgpd.conf* protokół rutingu BGP.

a) Należy utworzyć sesje BGP pomiędzy R2 a R3, R2 a R4, R4 a R5, R3 a R5.
neighbor ... remote-as <AS_nr>

b) W ruterach R2, R3, R4, R5 statycznie określić rozgłaszane prefiksy podsieci (*network ...*).

4. W pliku konfiguacyjnym *ospfd.conf* na ruterze R2 należy ustawić rozgłaszanie podsieci adresów uzyskanych z BGP do sieci OSPF (*redistribute ...*).

5. Uruchomić w ruterach pliki wykonywalne:
a) R1 – *zebra.conf, ospfd.conf,*
b) R2, R3 – *zebra.conf, ospfd.conf, bgpd.conf,*
c) R4, R5 – *zebra.conf, bgpd.conf*.
6. Sprawdzić osiągalność adresów IP w sieci (komendy *ping, traceroute*). Wyjaśnić znaczenie uzyskanej informacji. (R4 – R2, R3, R5; R5 – R2, R3, R4; R1 – R4, R5)
7. Jakich informacji dostarcza polecenie *show ip bgp*?
8. Jakich informacji dostarcza polecenie *show ip bgp neighbors*?
9. Sprawdzić tablice rutingu w ruterach.
10. Zasymulować awarię łącza R2-R4 i sprawdzić co się wtedy stanie. Porównać tablice rutingu przed i po wystąpieniu awarii. Jakie są ścieżki pomiędzy poszczególnymi ruterami?

Sposób konfiguracji plików (należy uzupełnić wzór o potrzebne komendy):

zebra.conf

```
hostname router
password quagga
enable password quagga
log stdout
!
interface eth0
link-detect
!
interface eth1
link-detect
!
ip forwarding
```

ospfd.conf

```
password quagga

router ospf
```

bgpd.conf:

```
password quagga
router bgp <as_number>
bgp router-id <router_id>
```