Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Tvorba materiálů pro wiki projekt počítačových sítí

Veronika Hurtová

Bakalářská práce

2014

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2013/2014**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Veronika Hurtová**
Osobní číslo:	**I11075**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Tvorba materiálů pro wiki projekt počítačových sítí**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

Cílem práce je vytvořit materiály pro wiki projekt počítačových sítí na upce.cz. Autor práce připraví přehledný materiál, který bude integrován do projektu wiki počítačových sítí na upce.cz. Materiál bude zaměřen na výklad principů směrování, směrovacích protokolů, použitých algoritmů a základní konfigurace směrovačů. V další části práce se autor zaměří na nejpoužívanější technologie pro WAN sítě, které podrobně představí, popíše a představí nejpodstatnější části konfigurace vybraných technologií a přístupů.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

LAMMLE, Todd. CCNA: výukový průvodce přípravou na zkoušku 640-802. Vyd. 1. Brno: Computer Press, 2010, 928 s. ISBN 978-802-5123-591.
EMPSON, Scott. CCNA kompletní přehled příkazů: autorizovaný výukový průvodce. Vyd. 1. Brno: Computer Press, 2009, 336 s. ISBN 978-80-251-2286-0.
ODOM, Wendell, Rus HEALY a Naren MEHTA. Směrování a přepínání sítí: autorizovaný výukový průvodce. Vyd. 1. Brno: Computer Press, 2009, 879 s. ISBN 978-80-251-2520-5.

Vedoucí bakalářské práce: **Mgr. Josef Horálek**
Katedra softwarových technologií

Datum zadání bakalářské práce: **20. prosince 2013**
Termín odevzdání bakalářské práce: **9. května 2014**

Univerzita Pardubice Česká republika

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Lukáš Čegan, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2014

Prohlašuji:

Tuto práci jsem vypracovala samostatně. Veškeré literární prameny a informace, které jsem v práci využila, jsou uvedeny v seznamu použité literatury.

Byla jsem seznámena s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 17. 4.2014 Veronika Hurtová

Poděkování

Mé poděkování patří Mgr. Josefu Horálkovi, Ph.D. za odborné vedení, trpělivost a ochotu, kterou mi v průběhu zpracování bakalářské práce věnoval. Děkuji také Ing. Soně Neradové za spolupráci při získávání zdrojů k vypracování této bakalářské práce.

ANOTACE

Cílem práce je vytvořit materiály pro wiki projekt počítačových sítí na upce.cz. Autor práce připraví přehledný materiál, který bude integrován do projektu wiki počítačových sítí na upce.cz. Materiál bude zaměřen na výklad principů směrování, směrovacích protokolů, použitých algoritmů a základní konfigurace směrovačů. V další části práce se autor zaměří na nejpoužívanější technologie pro WAN sítě, které podrobně představí, popíše a představí nejpodstatnější části konfigurace vybraných technologií a přístupů.

KLÍČOVÁ SLOVA

CCNA2, sítě, směrování, protokol, konfigurace

TITLE

Creating learning materials for wiki project on wiki.upce.cz

ANNOTATION

The goal of this bachelor work is creating materials for wiki project on wiki.upce.cz. Author of this work will prepare transparent material, which will be integrated into the project wiki.upce.cz. The material will focus on the interpretation principles of routing, routing protocols, algorithms and the basic configuration on routers. In the next section, the author will focus on the most widely used technology for WAN networks, which will be introduce in details, describe and present the most important parts of the configuration of selected technologies and approaches.

KEYWORDS

CCNA2, network, routing, protocol, configuration

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Oblast infornačních technologií je jedna z nejrychleji se rozvíjejících a s tím úzce souvisí i problém aktuálnosti vzdělávacích materiálů, ze kterých studují budoucí specialisté v této oblasti. V posledních několika letech jsme mohli zaznamenat i rychlý vývoj v oblasti počítačových sítí a to zavedení protokolu IPv6.

Jelikož je nezbytné zachovat konzistenci a zároveň aktuálnost informací, z nichž studenti získávají své znalosti a dovedosti je nutné přistoupit k využívání moderních technologií. Tyto technologie umožňují jednoduchou editaci a interakci.

Zároveň, zde vzniká problém s rozsahem předkládaných informací, kdy je nutné mít na paměti, že je nutné zachovat diverzifikaci informací a specializací jednotlivých expertů z oblasti IT a počítačové sítě nevyjímaje. Proto byl zvolen přístup wiki, tedy otevřené encyklopedie, shrnující aktuální informace, jež odpovídají výše zmíněným požadavkům.

Cílem této práce, je připravit podrobný a přesný materiál, jež má sloužit jako základ znalostní databáze ve formě wiki nejen pro studenty FEI UPCE, ale pro širokou veřejnost, od které lze očekávat i řadu relevantních nápadů a připomínek. Práce má následujíí stukturu, jež vychází z celé řady konzultací nejen s vedoucím práce, ale i s dalšími specialisty na oblast vzdělávání a odborníky v oblasti datových sítí.

První část je zaměřena na obecný princip fungování směrovače, jeho vnitřní strukturu a slouží pro připomenutí informací získaných z předchozího studia. Dále přibližuje praktické informace, například lze uvést princip označování rozhraní na směrovači, či korektní rozpoznání jednotlivých rozhraní včetně jejich funkčnosti.

Další část je zaměřena na objasnění způsobu ukládání informací o zjištěných cestách na směrovači. Tato část je zaměřena zejména na směrovací tabulku, která představuje stěžejní datovou strukturu využívanou při směrování. Ke směrovací tabulce neodmyslitelně patří lookup proces. Důraz byl kladen zejména na použití paralel k běžnému životu, díky nimž si čtenář snadněji uvědomí i po nějakém čase letmý princip.

Poslední část je nejrozsáhlejší a zaměřuje se na směrovací protokoly, používané v počítačových sítích. Při zpracovávání této časti, byl kladen důraz zejména na aktuálnost

uváděných informací, a tudíž jsou probrány i směrovací protokoly používané nad IPv6. Část věnovaná směrovacím protokolům se opírá o požadavky kurzu CCNA, avšak jako nadstavbu uvádí i znalosti potřebné pro budoucí studenty navazujícího studia. Z těchto znalostí lze uvést principy algoritmů využívaných ve směrovacích protokolech.

1 VŠECHNO JE JEDNA VELKÁ SÍŤ

Osobní počítače, notebooky, mobilní zařízení jako třeba PDA, tablety či chytré telefony, to vše nás v současné době obklopuje na každém kroku. Zkusme se však zamyslet, k čemu by nám tato zařízení byla, kdybychom nedokázali pomocí nich vzájemně komunikovat nebo přenášet data. Ano, odpověď je zřejmá. Tato zařízení by ani neexistovala, jelikož by člověk neměl potřebu zdokonalovat výpočetní techniku. Tedy lze usoudit, že vznik moderních zařízení je závislý na vzájemném propojení. Toto propojení nazýváme síť. Ať už počítačová nebo mobilní, princip je vždy stejný a to navázat přes nejrůznější prostředky spojení se zařízením, které může být jen metr od nás nebo také na druhé straně polokoule.

Samotné počítače nebo notebooky by nám ovšem nestačily, tudíž musí existovat zařízení, která by dokázala řídit provoz mezi komunikujícími stanicemi. Tyto prvky sítě nazýváme aktivní a řadíme mezi ně směrovače, přepínače, rozbočovače a mnoho dalších. Celkově lze říci, že jsou to všechny prvky, které se starají o propojení, filtraci či zesílení přeposílaných dat.

2 BEZ SMĚROVAČŮ BY TO NEŠLO

2.1 VLASTNOSTI SMĚROVAČE

Prvním směrovačem byl Interface Message Processor, dále uváděný jako IMP. Nejednalo se o směrovač ve významu, ve kterém ho známe dnes, ovšem o zařízení, které mělo stejné funkce jako dnešní směrovače. IMP byl využíván v síti Arpanet. Nestaral se však o zajištění spolehlivosti či vytváření spojení, jak je tomu u jeho dnešních nástupců. O tuto funkci se i v dnešních zařízeních stará celá řada protokolů (např. RIP, RIPv2 či OSPF).

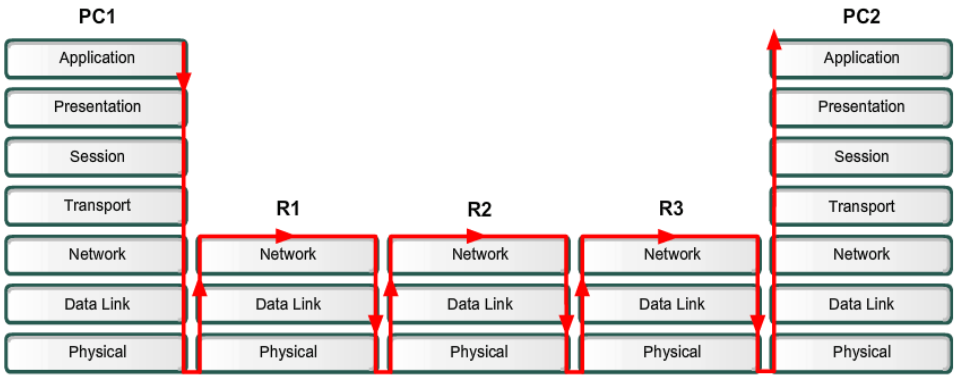
Na vývoji tohoto zařízení se podílela skupina autorů, mezi něž patří například Robert Kahn (teoretický informatik, který později stál u vývoje TCP/IP) či Severo Ornstein.

V prvních letech směrování (což je zhruba polovina 70. let až 80. léta) sloužily ke směrování v síti **minipočítače**. Tyto minipočítače umožňovaly provádění směrování.

Směrovač, ať již z pohledu dnešního nebo historického je **počítač**, i když tak na první pohled nemusí vůbec vypadat. Sice k němu nebývá nastálo připojen monitor ani klávesnice, nicméně obsahuje většinu prvků jako osobní počítač a speciální operační systém. Na rozdíl od osobního počítače obsahuje směrovací tabulku, která je využívána pro určování nejlepší cesty pro pakety. Směrovací tabulka, její struktura, význam a využití bude probrána později. Směrovač pracuje na prvních 3 vrstvách referenčního modelu ISO/OSI (respektive na prvních dvou vrstvách architektury TCP/IP).

Mezi hlavní dva úkoly směrovače řadíme rozhodování o tom, do které sítě posílaná data patří a jaká cesta se má zvolit, pokud chceme dodržet strategii řízení toku v síti. Dalším úkolem je propojování sítí, které mají rozdílné standarty a využívají rozdílné směrovací protokoly. Tedy ve zkratce se s nimi setkáme při propojování více sítí typu LAN nebo v případě, kdy chceme propojit sítě LAN a WAN.

[1][5]



Obrázek 1 Postup při zpracování paketu

2.2 ARCHITEKTURA SMĚROVAČE

V předchozí kapitole jsme na směrovač nahlédli z logického pohledu, ovšem stejně tak, jako každé jiné zařízení má i směrovač svou fyzickou strukturu. Směrovač je vcelku složité zařízení stejně tak jako osobní počítač. Obsahuje celou řadu fyzických komponent, které se starají o správnou funkci, výpočty i rozhodovací logiku. Ze všech těchto součástí je však nejsložitější proces **směrování**, čímž je nazývána samotná logika, díky níž může směrovač provádět veškeré funkcionality potřebné pro hladký a hlavně správný průběh směrování.

Pod specifickým case směrovače se skrývají stejné základní komponenty, které nalezneme i v klasickém počítači, tedy:

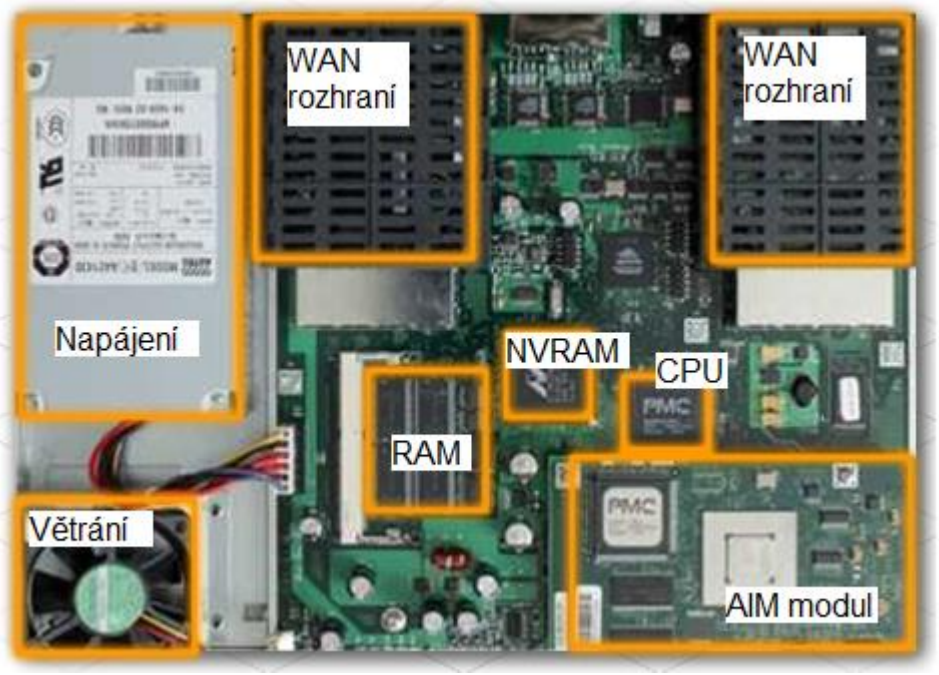
- napájení,
- fyzické I/O porty,
- základní deska,
- operační paměť,
- ROM,
- procesor,
- BIOS,
- operační systém.

Administrátor se při správném zacházení a v běžných provozních podmínkách nemusí nikdy setkat s většinou těchto komponent. Výjimku však tvoří potřeba rozšiřování systému, kdy pro jakékoliv rozšíření stávajícího zařízení musíme zasahovat přímo do fyzického uspořádání směrovače. Ovšem není to nikterak odlišné od běžného rozšiřování kapacit počítače.

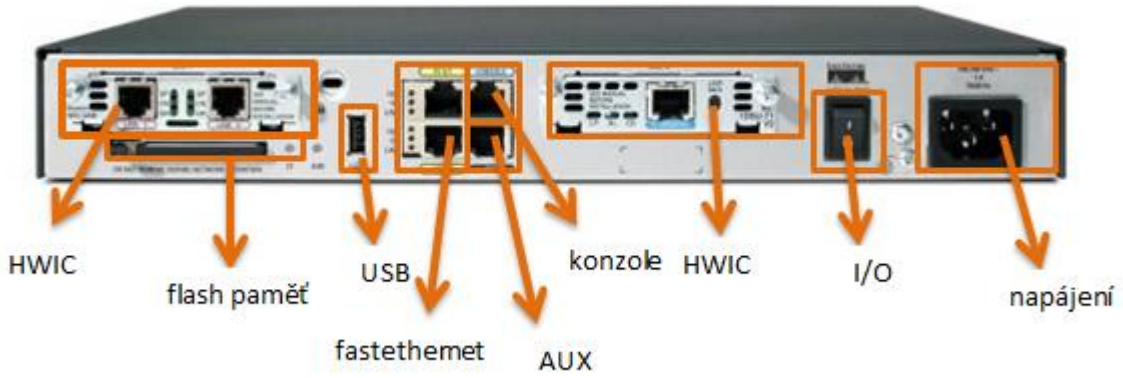
Jedinými komponentami, se kterými administrátor pracuje během své denní rutiny, jsou vstupně – výstupní porty a operační systém, který bude rozebrán později. Libovolný směrovač obsahuje minimálně dva I/O porty, které nazýváme rozhraními. Tato rozhraní slouží pro spojení přenosových prostředků konkrétní sítě se směrovačem a jsou indikována blikajícími LED diodami. Pomocí těchto diod je například možné rozpoznat aktuální stav rozhraní. Pro porozumění významu je nutné mít prostudovány příručky konkrétního typu směrovače.

2.3 VNITŘNÍ ARCHITEKTURA SMĚROVAČE

I když se jako běžný administrátor sítě většinou nebudete potýkat s nutností rozebírat směrovač, bylo by dobré mít alespoň představu, jak vnitřní uspořádání součástí směrovače vypadá.



Obrázek 2 Fyzické komponenty počítače



Obrázek 3 Viditelné komponenty pro administrátora

Na předchozích obrázcích jsme si ukázali fyzickou strukturu směrovače. Nyní si rozebereme jednotlivé komponenty:

- **CPU** lze s nadsázkou označit za „mozek“ směrovače, který se stará o vykonávání instrukcí operačního systému.
- **RAM** slouží k ukládání instrukcí a dat potřebných pro výpočty běžící v procesoru. Do RAM se také ukládá obraz běžícího operačního systému, který je do ní nakopírován ihned po startu. RAM dále obsahuje směrovací tabulku, konfigurační soubor obsahující počáteční konfiguraci směrovače, ARP cache (sloužící pro zjišťování MAC adresy) a dále slouží jako packet buffer (jsou zde ukládány pakety před jejich odesláním na konkrétní rozhraní nebo do doby, než začnou být zpracovávány). Pozor si musíme dávat na to, že tato paměť je při ztrátě napájení vymazána.
- **ROM**, která umožňuje trvalé uložení dat. Směrovače využívají tuto paměť k ukládání zaváděcích instrukcí pro operační systém. ROM dále obsahuje základní software sloužící pro diagnostiku chyb a odlehčenou verzi IOS.
- **FLASH** paměť sloužící k uložení obrazu IOS.
- **NVRAM** je využívána pro trvalé uložení konfiguračního souboru sloužícího pro zavedení konfigurace po spuštění systému.

2.4 PROPOJENÍ SMĚROVAČE

Stejně jako jakékoliv jiné zařízení, je možné i směrovač propojovat se zařízeními jinými. Těmito zařízeními mohou být jiné směrovače, či koncová zařízení. Bez propojení by nebylo možné směrovač, jenž neumožňuje bezdrátové připojení, využít.

2.4.1 Konvence při číslování rozhraní

V praxi rozlišujeme tři druhy rozhraní. Tato rozhraní jsou vždy označena sekvencí čísel oddělených lomítky.

Obecně lze zmíněnou sekvenci převést do tvaru:

slot/subslot/port

Tato posloupnost není vždy složena ze všech tří hodnot. Vždy záleží na použité platformě směrovače, na kterém se nacházíme. Konkrétně zápis například **se 0/hodnota/hodnota** je uváděn pro rozhraní umístěné na vložených kartách. Tyto karty jsou instalovány pomocí síťových slotů.

Zápis složený ve tvaru **se hodnota hodnota** je naopak používán pro rozhraní nativně vložené na základní desce. S nativně vloženými porty se setkáváme kupříkladu u směrovačů Cisco 2821. Na ostatních se využívají porty nainstalované, u kterých číslování vypadá následovně:

<rozhraní> 0/0/0, <rozhraní> 0/0/1

<rozhraní> 0/1/0, <rozhraní> 0/1/1

<rozhraní> 0/2/0, <rozhraní> 0/2/1

<rozhraní> 0/3/0, <rozhraní> 0/3/1

Zkrátka určujeme kde na směrovači je modul rozhraní zapojen.

Obrázek 4 Příklad zásuvného modulu

2.4.2 Porty pro správu směrovače

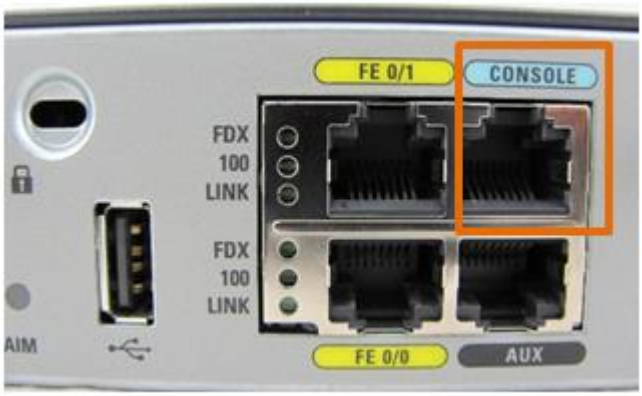
Zásadním rozdílem od klasických rozhraní je skutečnost, že tyto porty neslouží k přenášení paketů. Nejdůležitějším a zároveň nejvíce používaným je port pro připojení **konzole**, který slouží k propojení s počítačem, přes který je možné provádět konfiguraci směrovače. Ovšem

aby bylo možné tuto správu směrovače provádět, musí být na počítači (terminálu) spuštěn **emulační software**. Nejčastěji je přes tento port prováděna počáteční konfiguraci směrovače.

Dalším portem, který se řadí do této skupiny, je port **pomocný,** označovaný jako AUX. Tento port nemusí být implementován na každém směrovači. AUX slouží obdobně jako port konzolový, avšak na rozdíl od něj nám umožňuje i připojení modemu.

V našem výkladu budeme pro správu směrovače využívat port konzolový nikoli pomocný.

Konzolový port je značen u směrovačů firmy Cisco následujícím způsobem.



Obrázek 5 Konzolový port

2.4.3 Rozhraní

Je fyzický konektor umístěný na směrovači tak, aby k němu byl co nejjednodušší přístup. Slouží k připojování sítí různých typů a rozsahů. Každý směrovač obsahuje více typů rozhraní, aby bylo možné propojení LAN a WAN sítí, které využívají různé přenosové technologie. Každé rozhraní je navíc obohaceno o LED diody indikující aktuální stav rozhraní.

2.4.3.1 Typy rozhraní

Fastethernet je používán pro propojení koncového zařízení k směrovači. Není využíván pro konfiguraci směrovače. Nenalezneme jej samozřejmě pouze na směrovačích. Zatímco sériové rozhraní nalezneme výhradně na aktivních prvcích, fastethernet je obsažen i na zařízeních koncových, jelikož je jediným způsobem, jak tato zařízení zapojit do sítě. Směrovače běžně poskytují dvě rozhraní typu fastethernet. Pokud potřebujeme připojit do sítě více zařízení, je nutné využít přepínač. Již z hlediska budoucího rozšiřování sítě je nanejvýš vhodné používat přepínače i směrovače.

Obrázek 6 Rozhraní fastethernet

Gigabitethernet je obdobou předešlého avšak slouží pro připojení ethernetu využívající gigabytové rychlosti.

Sériové rozhraní, jež slouží pro vzájemné zapojení mezi směrovači. U sériových linek musíme zapojovat dva typy konektorů. Označujeme je za **DTE** a **DCE.** Jejich funkcí je řízení komunikace po lince. Při konfiguraci je nutné věnovat pozornost při zapojování, jelikož na rozhraní, kde bude připojen konektor označený DCE je nutné nastavit clock rate. Zpravidla jej nastavujeme na 64000, avšak může dosahovat celé množiny různých hodnot. Mějme například rozhraní se0/0/0 s IP adresou 192.168.20.1, na kterém má být nastaven clock rate . Celou konfiguraci provedeme následující sérií příkazů:

```
R # configuration terminal
R (config) # interface se0/0/0
R (config-if) # clock rate 64000
R (config-if)# ip address 192.168.20.1 255.255.255.0
R (config-if)# no shutdown
```

V praxi však můžeme, například ve firmě, narazit na kabel, který nebude popsán. V takovém případě bychom nebyli schopni ihned rozlišit, zda se jedná o DCE či DTE koncovku. V takovém případě si nejprve musíme zjistit podobu **spojovacího konektoru**. Nejjednodušším způsobem, jak vzájemně rozpoznat DTE a DCE, je převést si funkci na paralelu s běžným životem. V reálném světě většinou žena udává směr, tudíž i konektor, který je **samice**, udává tempo komunikace, tedy je označován jako **DCE.** Naopak **samec** tempu komunikace naslouchá, a tedy tento je tento konektor označen za **DTE**. Následující obrázek

znázorňuje podobu konektoru DTE a DCE, včetně spojových částí. Konektory samotné jsou identické.

Obrázek 7 Porovnání DTE a DCE konektorů

2.5 JAK VYBRAT VHODNÝ SMĚROVAČ

Ač není záměrem těchto skript být uživatelskou příručkou při výběru směrovače, považuji za velmi užitečné zmínit se alespoň o velmi obecných parametrech při jeho výběru.

V první řadě je vždy nutné zvážit, kde bude toto zařízení využíváno, tedy zda bude vytíženo pouze domácím použitím či budeme propojovat rozsáhlé sítě velkých korporací.

Zaměřme na běžného uživatele, který v domácnosti využívá notebooky, tablety či jiná koncová zařízení podporující připojení k internetu. Tento uživatel tedy potřebuje na sdílení jednoho internetového připojení, využít wi-fi směrovač[1]. Wifi směrovač je zařízení, měnící příchozí signál na bezdrátový přenos do všech výše vyjmenovaných periferií. Velmi často samotní poskytovatelé internetového připojení poskytují svým zákazníkům při zřízení připojení takzvané modem-routery, tedy zařízení, která zprostředkovávají zároveň i funkci modemu.

Mezi základní parametry, které je nutné při výběru zvažovat je rychlost portů. Dnešní dostupné směrovače mají nejmenší rychlost portů 100Mbit/s. Pro představu mějme na disku soubor o velikosti 100MB, tento soubor budeme stahovat pomocí tohoto portu zhruba 8s.

[1] V dnešních konsolidovaných zařízeních se spíše jedná o bridge.

U nenáročného uživatele je tato rychlost naprosto dostačující, ovšem pro náročnějšího, který kupříkladu bude chtít sledovat filmy ve FullHD kvalitě je lepší zvažovat porty o rychlostech 1Gb/s. Menší rychlost by u takovéhoto datového toku mohla způsobit kostičkování videa.

2.6 CISCO IOS

2.6.1 Obecně o IOS

Tento operační systém vyvinutý firmou CISCO je stejně tak jako operační systémy v osobních počítačích zodpovědný za správu veškerých hardwarových prostředků, přidělování paměti procesům a mimo jiné i za správu souborového systému a paměti. Cisco IOS je **multitaskingový operační systém**, tudíž umožňuje paralelní běh více procesů. Cisco IOS je specializovaný operační systém založený na Unixovém jádře. Důležitým úkolem IOS je také realizování směrovací logiky dle zvoleného typu směrování.

Uživatel zde nekomunikuje se zařízením pomocí grafického rozhraní, jak tomu známe z běžných operačních systémů. Ke komunikaci je využívána **příkazová řádka.**

2.6.2 Režimy práce s IOS

Aby bylo možné rozlišit práva uživatele v rámci systému, obsahuje IOS následující čtyři režimy práce:

- **Uživatelský mód**, jenž je implicitní po navázání spojení mezi terminálem a směrovačem. V tomto režimu je možné provádět pouze omezené množství příkazů, zejména pro ověření stavu směrovače a jeho konektivity. Tento režim je symbolizuje znak „>“.
- **Privilegovaný mód** je aktivován pomocí příkazu *enable* z módu uživatelského. Tento režim je již možné chránit heslem a zabezpečit tak systém před přístupem nepovolaných osob. Dovoluje nám zobrazit informace o systému, včetně jeho konfigurace, avšak není v něm možno měnit jeho nastavení. Tento režim je uvozován znakem „#“.

- **Konfigurační mód**, který umožňuje změny v nastavení operačního systému. Tento režim poznáme dle zápisu:

  ```
  R (config)#
  ```

- **Mód konfigurace rozhraní** je dostupný vždy až po zadání konkrétního označení rozhraní, či množiny rozhraní, na nichž se mají změny projevit.

  Pro jedno rozhraní, například fastethernet, se do tohoto režimu dostaneme následovně

  ```
  R (config)# interface fa0/0
  ```

  Pro množinu musíme zadat

  ```
  R (config)# interface range fa0/0 – 1
  ```

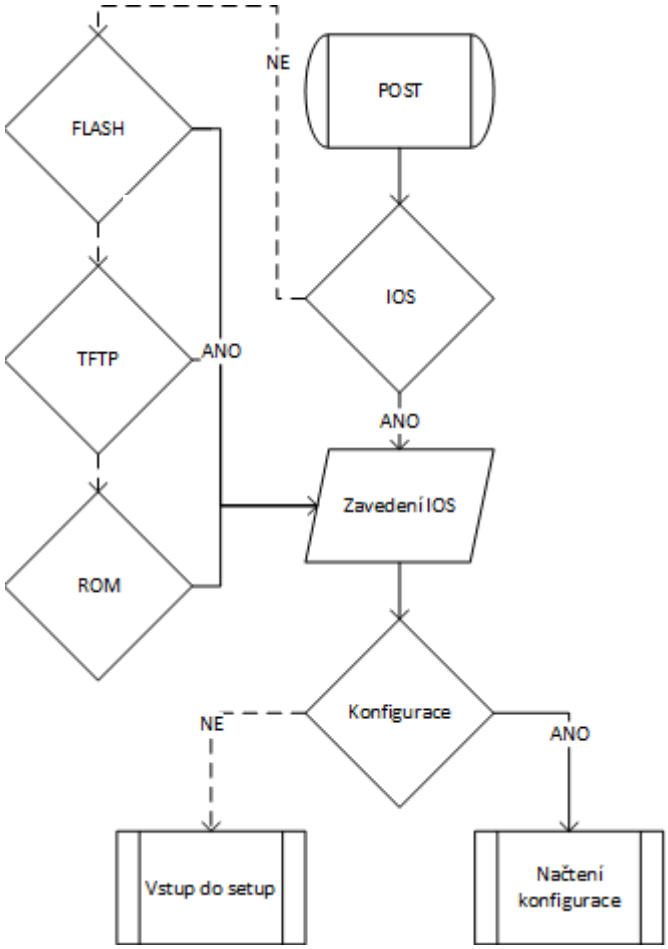
  V tento okamžik konfigurujeme současně rozhraní fastethernet0/0 a 0/1.

2.6.3 Postup zavádění IOS

Stejně tak, jako operační systém v osobním počítači, musí být i do směrovače nejprve zaveden jeho operační systém. Nejprve bychom měli naznačit, co vše musí proběhnout před zavedením IOS.

Každý směrovač nejprve provede POST test, při kterém dochází ke kontrole všech hardwarových komponent. Dalším nezbytným krokem je načtení zavaděče, což je krátký kód programu sloužící ke zkopírování operačního systému z flash paměti do paměti RAM s následným předáním řízení nad směrovačem. Nyní již máme zaveden operační systém, který však nemá žádnou konfiguraci, podle které by se mohl řídit. Nezná žádná hesla, ani nastavení rozhraní. Tedy je nutné nalézt (pokud existuje) počáteční konfiguraci. Ta je zpravidla uložena v NVRAM. Pokud není tato konfigurace nalezena, je spuštěn režim nastavování směrovače.

Celý postup znázorňuje následující schéma



Obrázek 8 Schéma načtení IOS

Pro práci s operačním systémem IOS, se používají standardizované příkazy, obdobné jako v operačních systémech s jádrem Linux, jejichž základní přehled je uveden v příloze A.

[1][3][5][10]

3 VŠECHNY CESTY (NE)VEDOU K CÍLI

3.1 CO JE PRO NÁS CESTA

Při delších či neznámých cestách využíváme ve svých automobilech GPS navigace. Důvodem je fakt, že nechceme ztrácet čas hledáním příslušných ulic či se ztratit ve spleti malých uliček ve velkých městech. Navigace nás navede přímo k cíli a my tak ušetříme čas i náklady.

Obrázek 9 Ilustrační obrázek silniční dopravy

Stejně tak tomu je i v počítačové síti. Každý náš požadavek na zobrazení webové stránky, či odeslání emailu přátelům si můžeme představit jako automobil, který má dojet co nejrychleji a nejúsporněji do cíle, čímž může být server webového portálu či jiný počítač.

V běžném životě využíváme na propojování měst silnice, mosty apod. V počítačové síti je to úplně stejné. Vzhledem k tomu, že počet počítačů i dalších koncových zařízení prudce stoupá, úměrně s lidskou potřebou být v kontaktu, není možné mít všechny počítače a ostatních zařízení propojeny kabelem do jedné obrovské sítě. Na propojování nám slouží směrovače a další aktivní prvky, které se starají o hladký průběh doručení našeho požadavku.

3.2 CO SE POSÍLÁ V SÍTI

3.2.1 Paket versus rámec

Na začátek si uvedeme základní rozdíl mezi pojmy paket a rámec, které v odborné literatuře i v běžném životě bývají nesprávně zaměňovány. **Paket** není nic jiného než blok dat konkrétního formátu, označovaný jako **PDU**. PDU vytvářen na síťové vrstvě modelu ISO/OSI

a sám o sobě nese informace o IP adrese příjemce a odesílatele, metadata a zapouzdřená data v podobě segmentu z transportní vrstvy.

Dříve než může být paket přenesen sítí, musí být zabalen do **rámce**. Rámce jsou vytvářeny na linkové vrstvě modelu ISO/OSI a velmi obecně lze říci, že nám umožňují přenos paketu sítí. Rámce vysílané mezi zařízeními jsou vždy dány maximální velikostí, která je důležitá pro to, aby si obě zařízení při komunikaci vzájemně rozuměla. Maximální velikost rámce označujeme jako **MTU**, na jejíž velikosti se komunikující zařízení nejprve domluví pomocí ICMP zpráv. Velikost je následně určena podle menší z oznamované velikosti. Z tohoto důvodu některé pakety síťové vrstvy musí být fragmentovány a umístěny do různých rámců. Podmínkou tohoto dělení do více rámců je schopnost správného rozpoznání protokoly sítě LAN. Kdyby tato vlastnost nebyla splněna, nebylo by možné pakety obnovit do jejich původní podoby. Je-li fragmentování zakázáno a přitom vyžadováno druhou stranou, skončí ICMP komunikace chybou a nelze provést přenos.

3.2.1 Formát rámce

V předchozích kapitolách jsme se dozvěděli, jak je rámec doručován v rámci počítačové sítě. Následujícím textem pochopíme, jak taková malá „dodávka“, přenášena sítí vypadá.

Dříve než si objasníme formát rámce, musíme říct, proč je důležité vytváření těchto malých dodávek, tedy respektive proč nemůžeme jednoduše vzít data, přidat jim cílovou adresu a odeslat je po síti. Důvod je zřejmý, počítač, který se snaží navázat komunikaci s jiným zařízením v síti, musí být schopen odeslat své datagramy skrze celou LAN síť. Odeslání nesmí být závislé na cílové adrese. Cílová adresa je důležitá pro ostatní zařízení v síti, která se starají o přeposílání tohoto datagramu. Jelikož samotné pakety, obsahují pouze adresu příjemce a odesílatele nebylo by možné v rámci sítě mezi zařízeními data přeposílat, protože by jednoduše nevěděla, jaké další zařízení má paket obdržet.

Celý ethernetový rámec může dosahovat velikosti od 72B až do 1532B, přičemž jeho přesná velikost je závislá na velikosti přenášených dat.

Ethernetový rámec si prošel poněkud delším vývojem. První formu tohoto rámce nazýváme podle původního Ethernetu, tedy **Ethernet II**. Jeho původ se datuje okolo roku 1980, kdy byl oficiálně představen a následně roku 1983 standardizován.

Tento typ rámce byl specifikován autory Ethernetu ve středisku PARC[2]. V současnosti je tento typ stále využíván a dost možná je i nejpoužívanějším.

Obrázek 10 Rámec Ethernet[3]

Následující přehled přibližuje význam jednotlivých položek.

- **Hlavička** je uvozující část, která oznamuje přijímacímu zařízení, že bude následovat rámec. Slouží k synchronizaci mezi komunikujícími zařízeními.
- **Cílová a zdrojová MAC adresa**, které jsou specifické pro každé ze zařízení v síti a jsou dány z části výrobcem zařízení a z části standardizační organizací. Platí, že adresa odesílatele musí být vždy unicastová, zatímco adresa příjemce může být unicastová, multicastová i broadcastová.
- **Typ** určuje protokoly aplikační vrstvy přijímacího zařízení, které jsou vyžadovány pro zpracování přijatého rámce.
- **Aplikační data** jsou samotný datový balíček přenášený po síti. Vždy je doplněn na minimální délku, pokud by ji nesplňoval nebo naopak fragmentován, pokud by překračoval délku maximální.
- **FCS** neboli frame check sequence je kontrolní součet sloužící na straně příjemce k detekci chyb vzniklých při přenosu. Způsob, jakým je FCS generován je závislý na použitém ethernetovém standardu.

[2] PARC (Palo Alto Research Center) je dceřiná společnost Xeroxu, podílející se na výzkumu a vývoji informačních technologií.

[3] Highteck. *Highteck* [online]. 2013 [cit. 2014-04-22]. Dostupné z: http://www.highteck.net/EN/Ethernet/Ethernet.html

Druhým typem je rámec dle standardu **IEEE 802.3**, jehož struktura je následující

Obrázek 11 Rámec 802.3[4]

Tento rámec se skládá ze dvou částí:

- hlavička MAC,
- hlavička LLC (IEEE 802.2).

První část slouží k určení, odkud rámec pochází a zároveň kam je adresován. Posledním polem, spadajícím do této části je délka. Významem tohoto pole je určit, pro jakou koncovou aplikaci je rámec určen a ve které má být zpracován. Proč se zavedlo používání délky záznamu jakožto identifikátor typu obsahu rámce? Důvod je jednoduchý, založený na skutečnosti, že rámec LLC je společný pro vícero přenosových technologií normovaných institutem IEEE. Jedná se o snahu umožnění snazší koexistence lišících se přenosových technologií. Ke každému záznamu přenášených dat je připojena LLC hlavička. Samozřejmě nechybí ani pole FCS, které slouží, obdobně jako u Ethernetu II, ke kontrole přenášených dat. Struktura LLC hlavičky je následující:

- cílové přístupové místo,
- zdrojové přístupové místo,
- kontrolní byte.

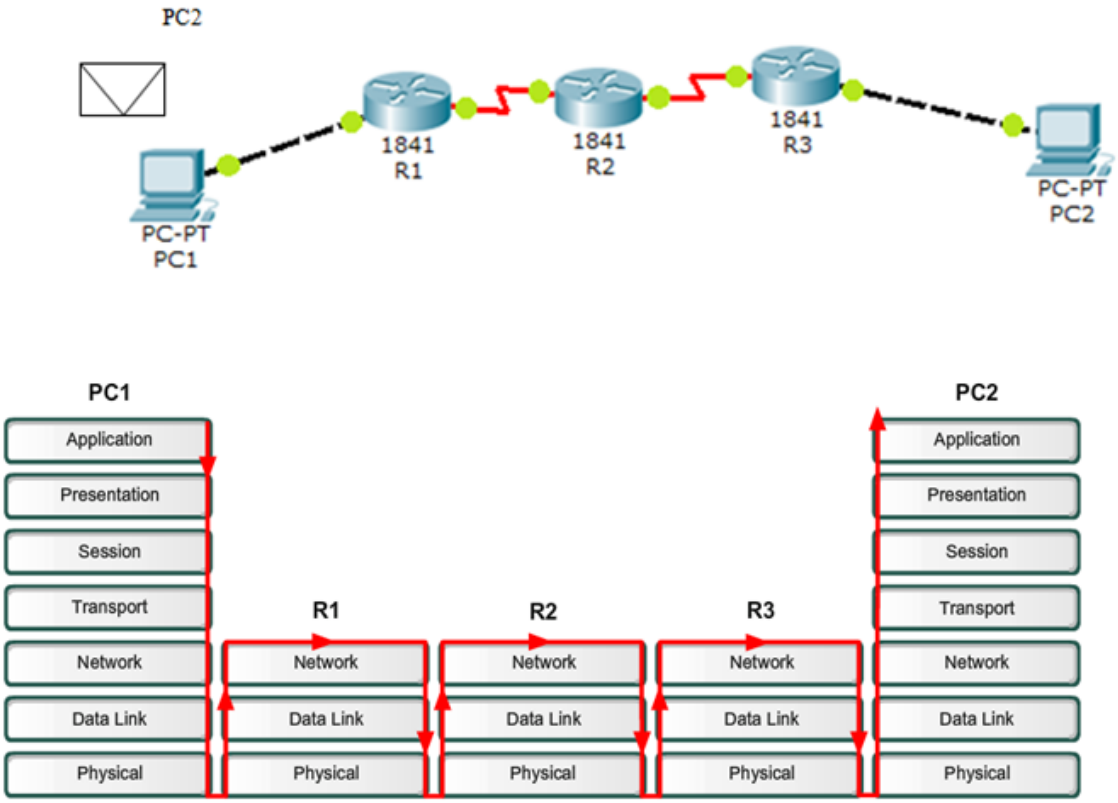
[4] Highteck. *Highteck* [online]. 2013 [cit. 2014-04-22]. Dostupné z: http://www.highteck.net/EN/Ethernet/Ethernet.html

Přístupové a zdrojové místo v síti jsou pojmy vážící se k referenčnímu modelu ISO/OSI. Oba určují koncovou vrstvu sítě referenčního modelu.

3.2.2 Jak směrovač pracuje s informacemi

Už víme, jaká je cesta paketu v počítačové síti. Obecně si tedy můžeme směrování definovat jako **proces**, který nám určuje použitou cestu na základě cílové adresy sítě umístěné v hlavičce každého paketu.

Celý tento proces probíhá na **třetí vrstvě modelu ISO/OSI** a jsou do něj zahrnuty veškeré směrovače nacházející se mezi zdrojovou a cílovou sítí. Těchto zahrnutých směrovačů může být velké množství, avšak pro samotný akt směrování jsou vybrány pouze některé. Množina směrovačů, jež je použita vychází z použitého směrovacího protokolu.



Obrázek 12 Postup při zpracování paketu

[1][2][3][5][12]

4 SMĚROVACÍ TABULKA A LOOKUP PROCES

4.1 CO JE A K ČEMU SMĚROVACÍ TABULKA SLOUŽÍ

Směrovací tabulka je datová struktura, nacházející se v operační paměti směrovače. Jejím hlavním úkolem je udržování zjednodušeného obrazu síťové topologie, na jejímž základu jsou odesílány pakety. Směrovací tabulka obsahuje vždy pouze nejlepší cestu pro každou z cílových sítí. Do paměti je nahrána zpravidla při startu, avšak aktualizuje se i za běhu směrovače.

Směrovací tabulku si můžeme představit jako třídící stůl na poště.

Obrázek 13 Ilustrační obrázek třídícího stolu používaného na poště

Každý posílá dopisy či balíky na jinou adresu, do jiného města nebo dokonce země. Aby bylo možné určit, kam balíky směřují, a tedy kudy povede jejich cesta, je nutné je projít a roztřídit je. Pro třídění je použit mechanismus, podle kterého jsou balíky filtrovány a následně naloženy do dodávky a odvezeny na následující sběrné místo k podrobnějšímu třídění. Takto jsou roztříděny až na úrovně jednotlivých poštovních úřadů, kde jsou již rozdělovány na jednotlivé adresáty.

Směrovací tabulka a lookup proces fungují na stejném principu. Dříve než však začneme se samotným zkoumáním směrovací tabulky, je důležité zavést, či připomenout několik pojmů.

- **Metrika** označuje soubor údajů o cestě v síti sloužící k výběru nejlepší cesty. Může být ovlivněna různými kritérii, které se liší v závislosti na použitém směrovacím protokolu.
- **Administrativní vzdálenost** označuje **spolehlivost** zdroje, ze kterého vybraný spoj pochází. Pokud do cílové sítě existuje vícero cest o stejné hodnotě metriky, avšak o různých administrativních vzdálenostech, je vybrána vždy ta s nejnižší hodnotou administrativní vzdálenosti, tudíž získána, z pohledu směrovače z nejspolehlivějšího zdroje. Tuto vzdálenost zkráceně označujeme AD a pro různé protokoly a způsoby směrování nabývá různých hodnot.

Hodnoty administrativní vzdálenosti shrnuje následující tabulka.

Tabulka 1 Přehled hodnot administrativních vzdáleností

Typ spojení	**Hodnota AD**
přímé spojení	0
statická cesta	1
EIGRP sumarizovaná cesta	5
externí BGP	20
interní EIGRP	90
IGRP	100
OSPF	110
RIP	120
externí EIGRP	170
interní BGP	200
Ostatní	255

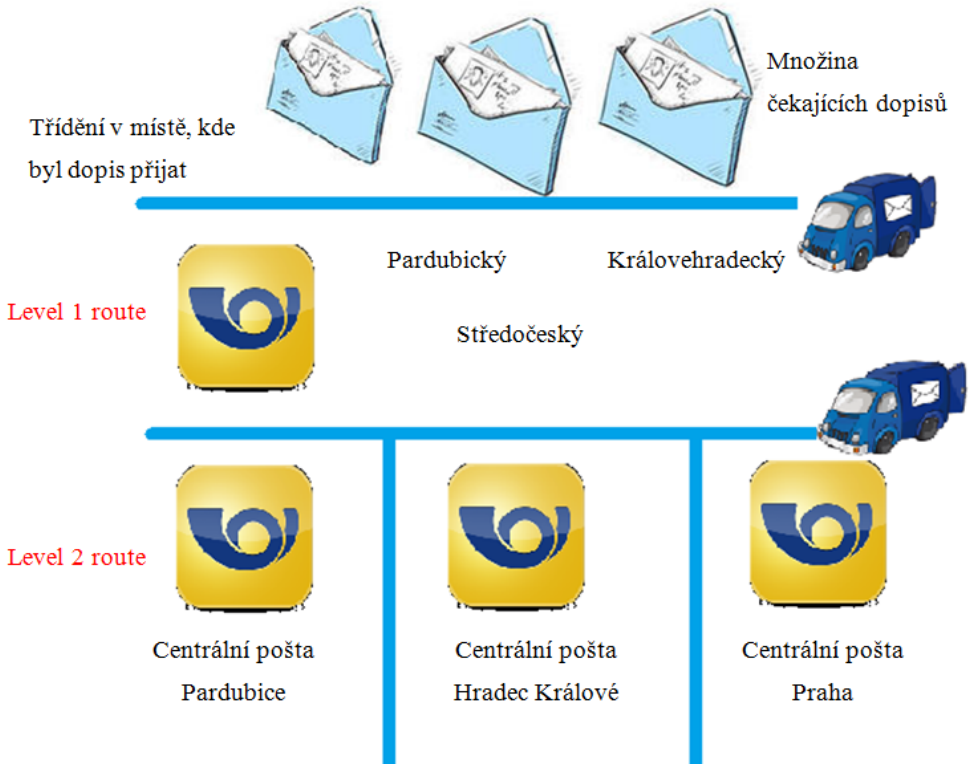
Jak tabulka číslo 1 udává, je nejspolehlivější cestou přímé spojení s cílovou sítí. Proto pokud má směrovač na výběr záznam, jenž je typu directly connected, s hodnotou AD rovnou 0 je automaticky preferován. Na druhou stranu spoje získané neznámým způsobem, označené jako ostatní, mají hodnotu nejvyšší, a tudíž se nebudou prakticky používat. Jediným okamžikem, kdy by byly použity je případ, kdy by neexistoval žádný jiný záznam.

4.2 TYPY CEST VE SMĚROVACÍ TABULCE

- **Level 1 route** představuje záznam ve směrovací tabulce shrnující vícero cest. Vraťme se k paralele s poštou, kde byly jednotlivé zásilky tříděny podle cílových adres. Není

však možné rozčlenit je ihned na základě města. Nejprve jsou rozčleněny na cílové kraje a teprve následně na města spadající do těchto krajů. Tyto kraje představují v počítačových sítích level 1 route.

A jelikož je známo, že obrázek řekne více než tisíc slov, ukažme si level 1 route pomocí obrázku.



Obrázek 14 Znázornění členění typu cest v rámci pošty

Level 1 route pro nás představuje roztřídění všech dopisů na jednotlivé kraje, do kterých jsou následně dopisy dopraveny. Zatímco level 2 route znamenají konkrétní záznamy v jednotlivých krajích.

- **Level 2 route** je také nazývána jako child route, tedy potomek. Takto označujeme cestu, která je podsítí cesty prvního stupně. Její maska je větší než třídní adresa. Level 2 route blíže specifikuje cestu.

- **Rodičovská cesta** je třídní cesta, která obsahuje potomky. Pokud její potomci neexistují, pak se nemůže jednat o rodičovskou cestu. Pro rodičovské cesty platí, že neobsahují odchozí rozhraní nebo IP adresu následujícího přeskoku. Bývá také označována jako cesta první úrovně (level 1 route).

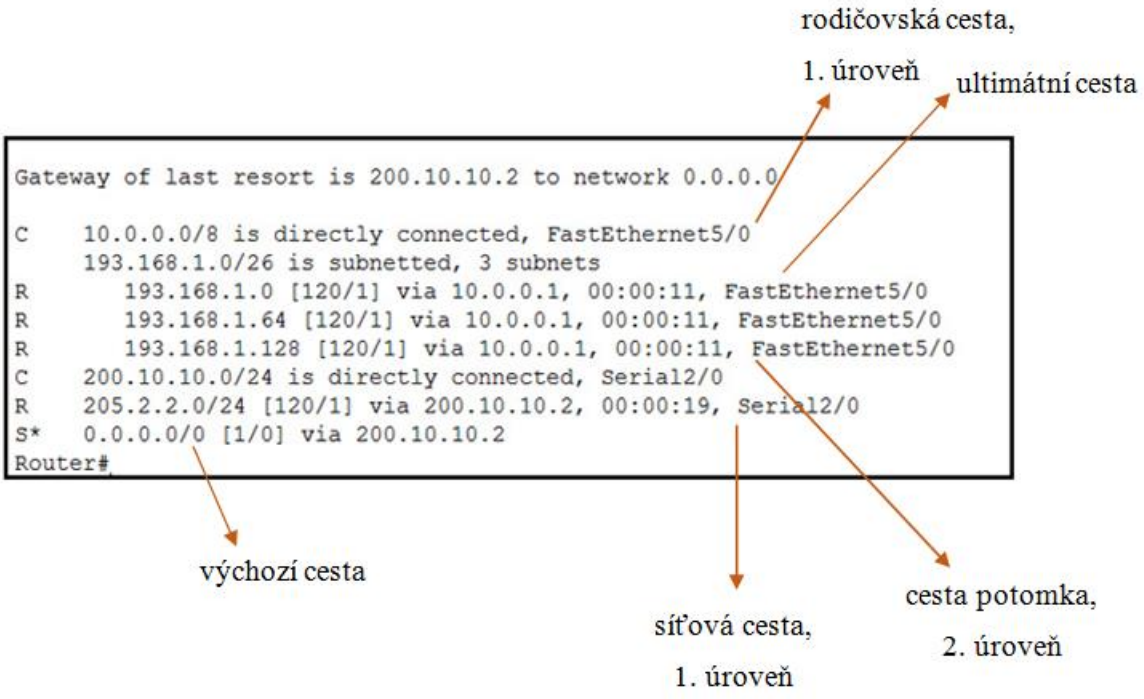
- **Cesta potomek** je podsíť k rodičovské cestě. Maska této adresy je vyšší než adresa cesty rodičovské. Označujeme ji často jako level 2 route.

- **Ultimátní cesta** je cesta obsahující odchozí rozhraní nebo adresu následujícího přeskoku.

- **Výchozí cesta** je „poslední útočiště“ pro paket. Je použita v případě, pokud ve směrovací tabulce není nalezen odpovídající záznam o cestě k cílové síti.

- **Síťová cesta** je cesta první úrovně navíc obsahující odchozí rozhraní.

Jak rozlišíme jednotlivé druhy cest, z výpisu směrovací tabulky znázorňuje následující obrázek



Obrázek 15 Členění typů cest ve směrovací tabulce

4.3 STRUKTURA SMĚROVACÍ TABULKY

Stejně tak jako všechny výpisy prováděné IOS, má i směrovací tabulka svou strukturu. Tato struktura obsahuje následující položky.

- **Legenda** je důležitá pro porozumění hodnotám, jež jsou vypsány v části obsahující záznamy o nejlepších cestách do cílových sítí. Z důvodu úspory místa není možné u každého záznamu vypisovat označení plným názvem, a proto jsou používány pouze zkratky. Bohužel ne vždy odpovídají tyto zkratky označení zdroje, ze kterého byla cesta získána. Následující tabulka shrnuje označení současně s jejich významem u zápisů, se kterými se budeme setkávat nejčastěji.

Tabulka 2 Významy kódů ve směrovací tabulce

kód	Význam
C	Síť získána přímo. Takto označené sítě směrovač rozpoznal sám na základě adresovaných rozhraní.
S	Adresa sítě získána staticky, tedy adresované přímo na směrovači.
R	Síť získaná dynamicky pomocí protokolu RIP, respektive RIPv2 nebo RIPng.
B	Adresa sítě získaná pomocí protokolu BGP.
D	Zde kód zcela neodpovídá. Takto označený záznam značí adresu sítě získanou dynamicky pomocí protokolu EIGRP.
O	Záznam získaný pomocí protokolu OSPF.
L	Linka lokální.

- Dalším polem je informace o **nastavení výchozí brány**. Pokud na směrovači používáme jedno rozhraní, není výchozí brána potřeba, jelikož pokud přijde paket, adresovaný do neznámé sítě, nemá směrovač jinou možnost než jej odeslat skrze toto rozhraní. Pokud však využíváme více rozhraní, nemohl by směrovač jednoznačně určit, které má paket zpracovat, a tudíž se zobrazí výpis

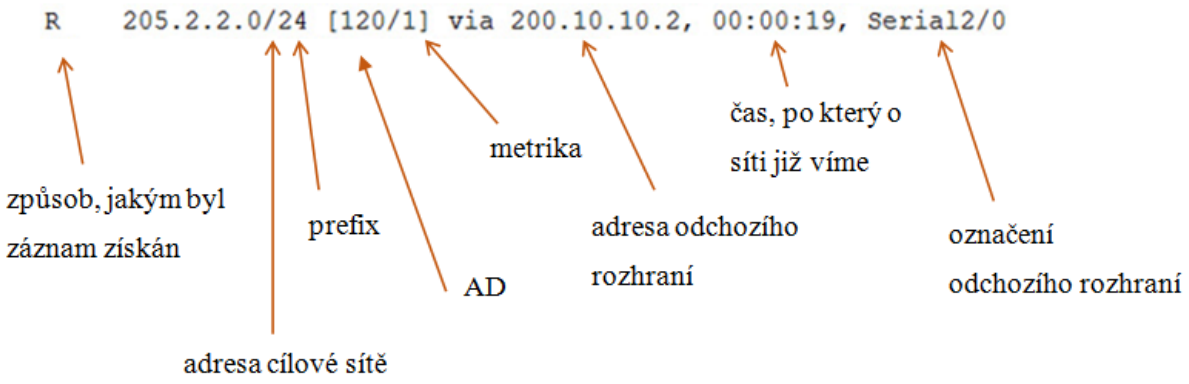
```
Gateway of last resort is not set
```

V tomto případě je nutné nastavit výchozí bránu. Pokud je výchozí brána nastavena, pak je tato informace reprezentována následně

```
Gateway of last resort is 10.10.2.4 to network 0.0.0.0
```

První informace určuje adresu rozhraní, jež má být zvoleno pro pakety určené pro neznámé sítě. Druhá informace symbolizuje libovolnou síť, tudíž kteroukoliv síť neznámou.

- Následuje přehledný **výpis směrovací tabulky**. Jak bylo uvedeno v předchozím textu, členíme záznamy, respektive adresy cílových sítí, na adresy první úrovně, neboli level 1 route a adresy úrovně druhé, které jsou označovány jako level 2 route. Aby bylo ve výpisu možné odlišit tyto dva druhy adres, jsou adresy druhé úrovně odsazeny a označeny kódem značícím způsob, jakým byla informace o cílové síti získána. Způsob, kterým je směrovací tabulka prohledávána označujeme jako **lookup proces** a bude mu věnována jedna z následujících částí této kapitoly.



Obrázek 16 Struktura záznamu směrovací tabulky

4.4 PRÁCE SE SMĚROVACÍ TABULKOU

Se směrovací tabulkou nelze provádět žádné úpravy, lze pouze provést její vypsání. Na libovolném Cisco zařízení ji vypíšeme následujícím příkazem

```
R # show ip route
```

Jak vypadá výpis, jež obdržíte, či význam zobrazených položek byl vysvětlen v předchozí části této kapitoly.

Bohužel jelikož slouží směrovací tabulka jako zdroj informací, není dostupných mnoho parametrů, jež by mohly modifikovat výpis. Ovšem máme možnost specifikovat vypisování informací pouze o konkrétních způsobech získání cest, či informace o konkrétní cestě k cílové síti. Pokud například potřebujeme vypsat informace o cestách získaných pomocí statického směrování, použijeme parametr static, zápis by vypadal následovně

```
R # show ip route static
```

Další možností je také vypsání informací o jedné konkrétní cílové síti, pro získání těchto informací použijeme následující příkaz

```
R # show ip route 192.168.20.0
```

Nesmíme opomenout ani směrovací tabulky sítí využívajících směrovaný protokol IPv6, v nichž nelze použít předchozí příkazy. V tomto případě využíváme příkazy obdobné, avšak modifikované tak, aby směrovač jasně věděl, kterou směrovací tabulku chceme vypsat. Veškeré parametry pak lze aplikovat i pro IPv6.

```
R # show ipv6 route
R # show ipv6 route static
```

4.5 LOOKUP PROCES

Lookup proces, česky proces vyhledávání, představuje systematické prohledávání směrovací tabulky. Název odpovídá anglickému look up, jež je používáno například pro označení vyhledávání ve slovníku. Toto označení až tak daleko od pravdy, jelikož proces prohledávání směrovací tabulky je obdobný.

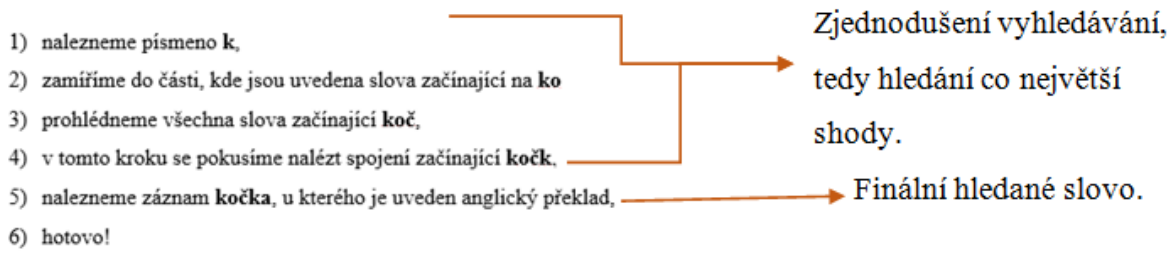
Obrázek 17 Ilustrační obrázek vyhledávání ve slovníku

Zamysleme se, jak vyhledáváme například slovo v anglickém slovníku. Vezměme si slovo kočka, u kterého chceme nalézt jeho anglický ekvivalent. K dispozici však nemáme inteligentní slovník, v němž bychom mohli jednoduše napsat slovo a ihned bychom měli jeho překlad. Vezměme pro náš příklad klasický slovník v papírové podobě, ve kterém musíme vyhledávat sami.

Algoritmus nalezení překladu slova kočka by mohl vypadat následovně:

1) nalezneme písmeno **k**,
2) zamíříme do části, kde jsou uvedena slova začínající na **ko**
3) prohlédneme všechna slova začínající **koč**,
4) v tomto kroku se pokusíme nalézt spojení začínající **kočk**,
5) nalezneme záznam obsahující slovo **kočka**, u kterého je uveden anglický překlad,
6) hotovo!

Shrňme si, ze kterých dvou částí se takové vyhledávání skládalo.



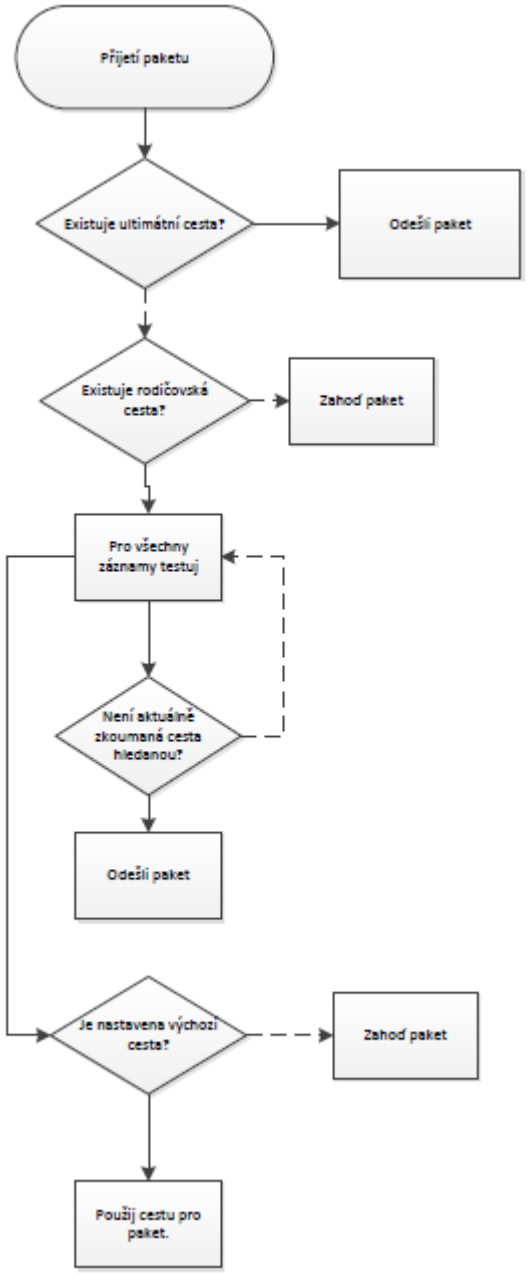
Obrázek 18 Postup při vyhledávání slova ve slovníku

A co se stane, pokud slovo, které jsme hledali, se ve slovníku chybí? Vezmeme si slovník jiný. Tedy si musíme hledané slovo stále pamatovat a dojít do knihovny, kde vezmeme další slovník a celý postup hledání opakujeme. Pokud ani tam slovíčko nebude, musíme pokračovat skrze všechny slovníky umístěné v knihovně. Ovšem abychom mohli začít prohledávat další slovník, musíme znát cestu ke knihovně.

Postup, který jsme zmínili výše je i ten, který využívá směrovač pro nalezení vhodné cesty pro paket. Nejprve se nalezne cesta souhrnná, do které by cílová adresa měla patřit, a teprve poté jsou prohledány jednotlivé podsítě tohoto záznamu.

[6][7][11]

Toto prohledávání znázorňuje následující vývojový diagram.



Obrázek 19 Postup prohledávání směrovací tabulky

5 UČÍME SMĚROVAČ, STATICKÉ SMĚROVÁNÍ

5.1 CO ZNAMENÁ „SMĚROVAT STATICKY“

Tento druh směrování je založen na využívání směrovací tabulky, tedy konkrétně statických záznamů obsažených v této tabulce. Při rozhodování jsou cesty vždy stejné, jelikož jsou

všechny cesty nastaveny pevně. Pokud nastane změna v topologii, nebo dojde k výpadku v síti, směrovač tuto skutečnost nemůže zjistit, jelikož si stále udržuje pouze pevně nastavené záznamy.

Statické záznamy jsou přidávány **manuálně administrátorem.** Jejich konfigurace je velice snadná. Obecně si lze tento proces představit tak, že sami určíme rozhraní, kterým bude paket odcházet v případě, že cílová adresa náleží do určité cílové sítě. Tento proces lze přirovnat k plánování cesty výletu na mapě. Určíme cíl, který chceme navštívit a následně vyhledáváme a stanovujeme města, přes která musíme k cíli projet. Nevýhodou je však nepředvídatelnost dopravních kolon, uzavírek nebo prací na silnici. Všechny tyto faktory nás při cestě pozdrží a tak příjezd do cíle může být zpožděn. Na druhou stranu můžeme pouze určit, do kterého místa se chceme dostat a jet nahodile, avšak volit cesty, které jsou nejrychlejší, například dálnice. V tomto případě nemůžeme určit, jak dalekou cestu pojedeme, ovšem dostaneme se do cíle poměrně rychle. První způsob, kdy vyhledáváme předem, označujeme ve světě počítačových sítí jako **statické směrování**, zatímco druhý uvedený jako **směrování dynamické.**

Statické směrování má ovšem několik zásadních nevýhod. Hlavní z nich je nezbytnost znalosti topologie administrátorem spravované sítě, jelikož právě administrátor musí rozhodovat o průběhu směrování. Stejně tak tomu je i při plánování cesty pomocí mapy, kdy naše povědomí o tom, jak město vypadá, zobrazuje mapa. Další neméně opominutelná nevýhoda nastává při konfiguraci. Statické směrování nelze provádět centrálně, a tudíž musí být prováděno na každém ze zařízení. Nejinak je tomu i při administrativních úpravách, či v případě poruchy na některém ze zařízení. Díky tomuto faktoru se stává statické směrování velmi časově náročné i v méně rozsáhlých počítačových sítích.

Z výše uvedeného vyplývá, že statické směrování je využíváno jen ve specifických případech, například lze uvést spojení, u kterého je nutnost 100% spolehlivosti přenosu. Jednoduše se nám nemůže u tohoto spoje stát, že by se nám paket ztratil. Nejčastějším případem, kdy se se statickým směrováním setkáme, je definování **výchozí cesty**.

5.2 DRUHY STATICKÝCH ZÁZNAMŮ

Pro statické záznamy rozlišujeme tyto druhy:

- standardní (klasická) cesta,

- výchozí cesta,
- sumarizovaná cesta,
- záložní cesta.

Při konfiguraci statického směrování můžeme využívat všechny uvedené typy cest, avšak každá je vhodná ke specifickým účelům. V následujících částech textu budou jednotlivé druhy podrobněji přiblíženy.

5.3 ZPŮSOB ZADÁVÁNÍ CEST

Dříve než si přiblížíme, v jakých situacích a jakým způsobem jednotlivé druhy cest aplikovat, je nutné shrnout, jakým způsobem se statické cesty směrovači zadávají. Pro zadání statické cesty lze použít následující tři způsoby

- pomocí odchozího rozhraní,
- adresou dalšího přeskoku,
- úplně, tedy odchozím rozhraním i dalším přeskokem.

```
R    (config)#    ip    route    <adresa_sítě>    <maska_sítě>
<adresa_přeskoku>

R    (config)#    ip    route    <adresa_sítě>    <maska_sítě>
<odchozí_rozhraní>

R    (config)#    ip    route    <adresa_sítě>    <maska_sítě>
<odchozí_rozhraní> <adresa_preskoku>
```

Jaký je mezi nimi rozdíl? Rozdíl je v možnosti budoucích změn sítě. Pokud je cesta dána prvním způsobem, tedy pomocí adresy dalšího přeskoku, potom v případě drobné změny topologie již není síť funkční, jelikož se odkazuje na adresu rozhraní směrovače, který se v síti již nemusí ani nacházet. Na druhou stranu druhý způsob je v tomto ohledu flexibilní. I v případě změny v topologii je pouze řečeno, jakým rozhraním ze směrovače má být paket odeslán. Tedy nejsme nikterak závislí na protějším směrovači. A nakonec třetí

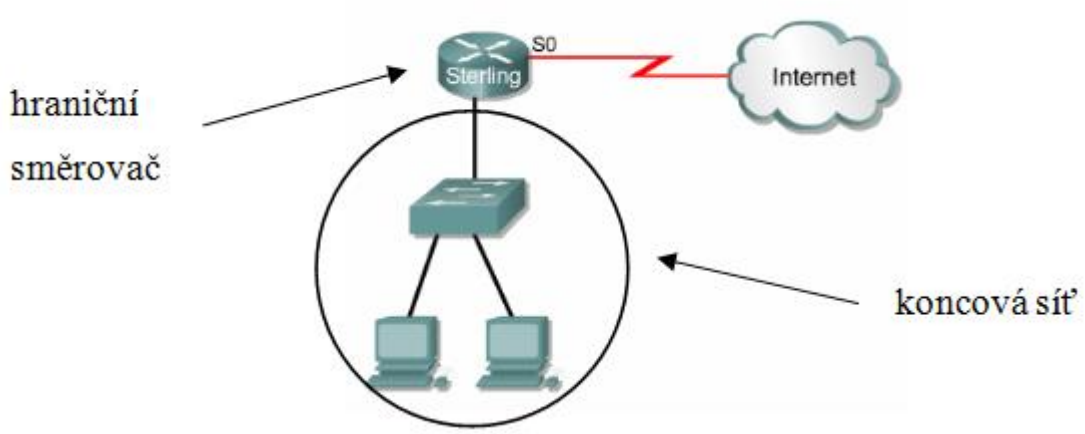
způsob, který využívá obou možností specifikace, se v praxi příliš nepoužívá, jelikož obdobně jako první způsob je velmi náchylný ke změnám v síti. Na druhou stranu je tento způsob vhodné použít tehdy, pokud komunikace musí probíhat pouze jedním specifickým rozhraním na konkrétní přeskok například z důvodu bezpečnosti.

5.4 STATICKÉ SMĚROVÁNÍ POMOCÍ VÝCHOZÍ CESTY

Tento druh směrování je vhodné využít tehdy, pokud je síť, či směrovač označen jako **hraniční**. Hraniční směrovač je propojen pouze s jedním dalším směrovačem. Za tímto hraničním směrovačem již následuje koncová síť, tedy například uživatelé v budově školy. Jelikož nemá směrovač na výběr jinou možnost, než odeslat paket na druhé rozhraní, než to ze kterého paket přijal, stačí využívat nastavení výchozí cesty. Tento druh směrování je výhodný svoji jednoduchost, jelikož nám postačí jeden jediný příkaz a to

```
R (config)# ip route 0.0.0.0 0.0.0.0 <odchozí_rozhraní|adresa_preskoku>
```

Kde první série nul označuje, že cesta je použita pro libovolnou adresu sítě a druhá čtveřice nul udává, že se jedná o sítě s libovolnou maskou. Následuje určení odchozího portu či dalšího přeskoku. Možnosti definování specifikace pro paket byly rozebrány v předchozí části této kapitoly.



Obrázek 20 Směrování pomocí výchozí cesty

Ovšem výchozí cesta, respektive výchozí směrování, není používána pouze pro koncové sítě. Dalším využitím je nastavení odchozího rozhraní pro případ, kdy ve směrovací tabulce nebude nalezen vyhovující záznam pro cílovou síť. Pokud je v takovém případě výchozí brána nastavena, je paket adresovaný do neznámé sítě odeslán touto cestou, ve druhém případě by byl paket zahozen.

Zda je na směrovači nastavena výchozí cesta zjistíme výpisem směrovací tabulky. Tedy provedeme ji následujícím příkazem

```
R # show ip route
```

Kde nastavení výchozí cesty interpretuje upozornění

```
Gateway of last resort is not set
```

respektive pokud výchozí cestu máme nastavenu

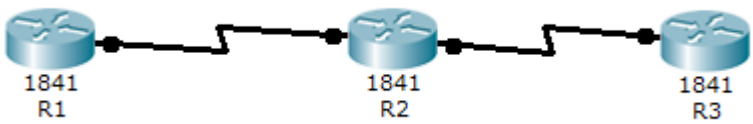
```
Gateway of last resort is 192.168.10.1 to network 0.0.0.0
```

Spojení last resort je zcela vypovídající, jelikož tento záznam je zkoumán jako poslední, teprve po úplném prohledání směrovací tabulky. Je vždy vhodné myslet při konfigurování sítě i na takové cílové sítě, jež nebudou v záznamech směrovací tabulky, a z tohoto důvodu nastavit výchozí cestu.

5.5 STATICKÉ SMĚROVÁNÍ A ZÁLOŽNÍ CESTA

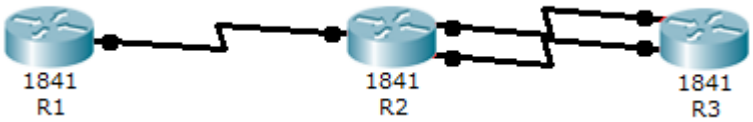
Každý má spojený pojem staticky konfigurované sítě s faktem, že v případě výpadku libovolné linky náležící cestě mezi zdrojovou a cílovou sítí, automaticky dojde k nedostupnosti cílového hosta. Avšak je možná, i když ne příliš často používaná, možnost nastavení záložní cesty pro případ výpadku spoje.

Mějme následující topologii



Obrázek 21 Topologie bez záložní cesty

Směrovač R2 obsahuje ve své směrovací tabulce záznam o výchozí cestě ke směrovači R1. Ovšem najednou dojde k výpadku linky mezi směrovači R1 a R2. V tuto chvíli a s konkrétním zapojením sítě nebude možné navázat spojení s ISP. V tomto případě není možné zabezpečit propojení pomocí jiné cesty. Jednoduše neexistuje spoj, přes který by tato komunikace mohla probíhat. Pro vytvoření zálohy je nutné mít mezi směrovači další spoj nebo směrovač, přes který by mohl být paket odeslán. V námi použité síti by bylo nutné přidat spoj následovně



Obrázek 22 Topologie obsahující záložní cestu

Nyní již je možné přidat záznam o záložní cestě do směrovací tabulky. Připomeňme si, že směrovač rozhoduje o použité cestě na základě důvěryhodnosti zdroje, z něhož informace o síti směrovač obdržel. Důvěryhodnost označujeme jako administrativní vzdálenost. Při zadávání záložní cesty využijeme malého triku s hodnotou.

Nastavíme dva spoje obsahující stejnou adresu koncové sítě, ovšem u druhé z nich připíšeme údaj o hodnotě administrativní vzdálenosti, kterou nastavíme ručně. Zápis na směrovači by mohl vypadat následovně

```
R (config)# ip route 0.0.0.0 0.0.0.0 serial0/0/0 2
```

Zvýrazněná část zobrazuje manuálně nastavenou hodnotu administrativní vzdálenosti. V našem příkladu byla použita hodnota 2, jelikož primární výchozí cesta je definována staticky a tudíž má hodnotu administrativní vzdálenosti rovnu 1.

A proč je použita hodnota pouze o jednotku vyšší? Důvod je takový, že pokud bychom záložní cestu označili například hodnotou 10 a na směrovači by byla konfigurována například cesta zjištěná jiným způsobem, pro směrovač důvěryhodnějším, byla by použita tato cesta místo námi požadované záložní cesty. Pokud zadáme AD pouze o jednotku vyšší, máme tak jistotu, že tuto cestu „nepředběhne“ žádná z definovaných cest.

Před tím, než začneme s konfigurací záložní cesty, je nutné stanovit, který spoj budeme používat jako **primární** (ten bude mít hodnotu AD nižší) a který jako **záložní**, jenž bude mít hodnotu AD o jeden bod vyšší než spoj primární.

Celý postup konfigurace na modelové topologii by na směrovači R3 byla provedena pomocí následujícího sledu příkazů.

```
R2> enable
R2# conf term
R2 (config)# ip route 0.0.0.0 0.0.0.0 se0/0/1
R2 (config)# ip route 0.0.0.0 0.0.0.0 se0/0/0 2
```

Zápis zvýrazněný žlutou barvou představuje specifikaci primární výchozí cesty. U tohoto zápisu nemusíme zadávat žádnou hodnotu AD, jelikož je přidělena aimplicitně. Definovali jsme statickou cestu, a tudíž bude hodnota AD rovna jedné. Nicméně nebylo by chybou uvést za zápis hodnotu jedna.

```
R2 (config)# ip route 0.0.0.0 0.0.0.0 se0/0/1 1
```

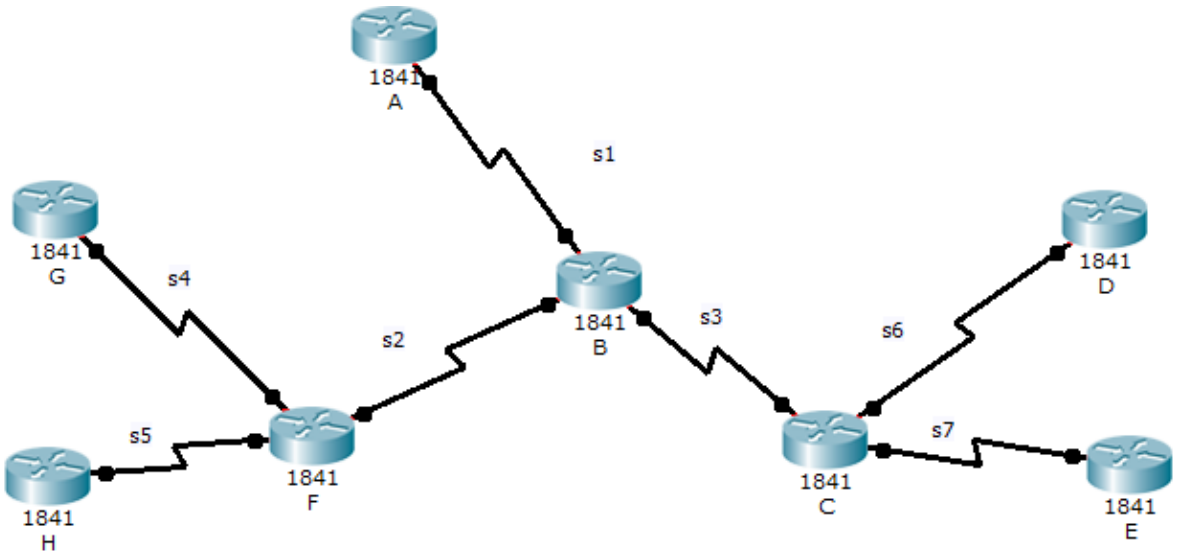
U druhého, zeleně zvýrazněného zápisu jsme definovali cestu záložní. Vidíme tedy, že pokud bude spoj skrze rozhraní se0/0/1 v provozu, bude používán pro odesílání paketů adresovaných do neznámých sítí, zatímco pokud dojde k výpadku, použije se rozhraní se0/0/0.

5.6 STATICKÉ SMĚROVÁNÍ POMOCÍ SUMARIZOVANÉ CESTY

Tento druh zápisu ve směrovací tabulce slouží zejména ke zjednodušení jejího obsahu. Směrovací tabulku se snažíme mít rozsahem co nejmenší, jelikož její velikost ovlivňuje nároky směrovače na hardwarové a systémové prostředky. Konkrétně je zvýšena potřeba procesoru a operační paměti. Kapacita procesoru je spotřebovávána zejména při prohledávání směrovací tabulky (lookup proces) a potřeba operační paměti vzrůstá současně s velikostí uložené směrovací tabulky.

Na obrázku 22 vidíme modelovou topologii sítě. Pro tuto síť bylo zvoleno využití statického směrování. Aby ovšem bylo možné komunikovat mezi jednotlivými uzly v síti, je nutné, aby každý ze směrovačů měl informace o celé topologii mimo sítí, jež jsou přímo připojené, jelikož o svých přímo púřipojených sítích ví směrovač automaticky.

Vezměme si směrovač náležící síti B. Směrovač B musí obsahovat informace o sítích A, D, E, G, H. Nezapomínejme také na sítě propojovací, tedy musel by vědět i o spojích S4, S5, S6 a S7. O spojích S1, S2 a S3 ví ihned, jelikož jsou tyto sítě přímo připojené. Představte si, že pro každou z těchto sítí musí existovat jeden záznam ve směrovací tabulce. Vznikla by nám tsk rozsáhlá směrovací tabulka. Navíc každý z těchto záznamů by musel být zadáván administrátorem, což je časově náročné a monotónní.



Obrázek 23 Topologie pro předvedení sumarizace

A teď se zamysleme, kam odchází pakety ze směrovače B, jsou-li adresovány do sítě D a E? Odchází skrze spoj S3 do směrovače C. Obdobný princip je i na stranou druhou, pro sítě G a H. Díky stejnému odchozímu rozhraní, respektive adrese dalšího přeskoku je možné sítě sjednotit a zapsat tak do směrovací tabulky na směrovači B pouze jeden záznam o sumarizované adrese pro sítě D a E včetně jejich spojů.

Z původních pěti záznamů rázem tři. Sumarizaci je možné provádět v sítích adresovaných pomocí IPv4 i IPv6. Způsobu, jakým sumarizovat cesty budou věnovány následující části této kapitoly.

5.6.1 Sumarizace v IPv4

Nejprve je nutné zmínit, že pro správné provedení sumarizace musí platit následující pravidla:

- Všechny sítě, jež chceme sumarizovat, musí mít stejné výchozí rozhraní, či adresu dalšího přeskoku.
- Adresy musí být ze stejného adresního rozsahu tak, aby bylo možné nalézt společné části adres.

Mějme následující adresy sítí, jež chceme sumarizovat

Tabulka 3 Adresy sítí pro sumarizaci

Název sítě	Adresa
A	172.14.0.2/16
B	172.20.0.0/16
C	172.16.0.0/16
D	172.22.0.0/16

5.6.1.1 Postup

Postup sumarizace lze shrnout do 3 kroků:

1. Všechny sítě přepíšeme do jejich **binární podoby** a zapíšeme do jednoho seznamu.

   172.14.0.2/16 = 10101100.00001110.00000000.00000000

   172.20.0.0/16 = 10101100.00010100.00000000.00000000

   172.16.0.0/16 = 10101100.00010000.00000000.00000000

   172.22.0.0/16 = 10101100.00010110.00000000.00000000

2. Projdeme adresy sítí bit po bitu zleva (od síťové části adresy) a hledáme nejvyšší možnou **shodu** v bitech

   172.14.0.2/16 = **10101100.000**01110.00000000.00000000

   172.20.0.0/16 = **10101100.000**10100.00000000.00000000

   172.16.0.0/16 = **10101100.000**10000.00000000.00000000

   172.22.0.0/16 = **10101100.000**10110.00000000.00000000

3. Přepíšeme shodující se bity a zbytek doplníme nulami. Prefix sumarizované adresy je dán počtem shodujících se bitů.

   10101100.00000000.00000000.00000000

   11 shodných bitů = prefix **/11**

Z původních 4 síťových adres bude zapsána jedna sumarizovaná, a to:

172.0.0.0/11

V příloze C, jsou připraveny příklady, pro procvičení této problematiky.

5.6.2 Sumarizace v IPv6

Postup sumarizace v IPv6 je téměř obdobný se sumarizací nad protokolem IPv4, avšak je zde přidáno několik kroků oproti IPv4. V následující části textu si letmo připomeneme alespoň základní informace o IPv6 adresaci.

5.6.2.1 Obecně o adresaci IPv6

Jistě si z předchozího studia vzpomenete, že nástupce protokolu IPv4 umožňuje na rozdíl od svého předchůdce mnohem větší, ba až nepředstavitelný rozsah adres. Konkrétně zde hovoříme o rozsahu 10^{38} adres, na rozdíl od IPv4 jež nám umožňuje zhruba 4 miliardy adres.

Toto rapidní rozšíření umožněného počtu adres však znamenalo celkovou změnu ve formátu adresy. Oproti zápisu složenému ze 4 oktetů oddělenými tečkami se zde setkáváme s různými tvary adres, čímž je zavedena notná dávka polymorfismu.

Uveďme alespoň pro zopakování, jak adresa v IPv6 může vypadat. Následující adresy představují tutéž adresu.

ff01:0000:0000:0000:0000:0000:0000:0101
ff01:0:0:0:0:0:0:101
ff01::101

Jelikož jednotlivé nulové položky IPv6 adresy mohou být vynechány a nahrazeny dvojtečkou, máme vždy více možností zápisu. Z důvodu částečného omezení polymorfismu byla organizací RFC zavedena kanonická forma zápisu. Každé zařízení musí být schopno pracovat se všemi formami zápisu IPv6 adresy, avšak aby došlo ke sjednocení výstupů, byla zavedena následující pravidla:

- všechny šestnáctkové hodnoty musí být zapsány malými písmeny,
- znak :: musí být využit efektivně tak, aby pokryl co největší možný rozsah nulových sousedních skupin,
- je nutností vynechávat počáteční nuly na začátcích jednotlivých skupin.

V IPv4 jsme jednotlivé části adresy nazývali jako oktety. Zde by toto označení nebylo nikterak pravdivé, jelikož adresa není 32 bitová, ale 128 bitová, tudíž nemůže být složena z oktetů. U IPv6 nemají jednotlivé skupiny ustálené označení, avšak v literatuře se můžeme setkat s označením **hextet.**

5.6.2.2 Postup sumarizace

Tabulka číslo 4 shrnuje adresy sítí, určených k sumarizaci.

Tabulka 4 Adresy sítí pro sumarizaci v IPv6

Síť	Adresa
A	2001:DB8:ACAD:1::/64
B	2001:DB8:ACAD:2::/64
C	2001:DB8:ACAD:3::/64
D	2001:DB8:ACAD:4::/64
E	2001:DB8:FEED:1::/64

Celý postup je možné rozdělit do 7 kroků.

1. Všechny adresy, jichž se týká sumarizace, zapíšeme do jednoho seznamu pod sebe. Díky přehlednému zápisu není problém identifikovat část, ve které došlo ke změně oproti ostatním adresám. Pro naše sítě tedy dostaneme následující seznam:

   2001:DB8:ACAD:1::/64

   2001:DB8:ACAD:2::/64

   2001:DB8:ACAD:3::/64

   2001:DB8:ACAD:4::/64

   Vidíme, že v našem případě došlo ke změně ve **4. hextetu**. V tomto místě se adresy jednotlivých sítí začaly lišit.

2. Hextet, ve kterém došlo ke změně, přepíšeme na rozvinutý tvar. Tedy dostaneme následující zápisy:

   2001:DB8:ACAD:**0001**::/64

2001:DB8:ACAD:**0002**::/64

2001:DB8:ACAD:**0003**::/64

2001:DB8:ACAD:**0004**::/64

3. Sekci, jež byla odlišná, převedeme do binární podoby. Můžeme převádět pomocí přímého převodu, či pomocí převodu do desítkové soustavy.

   2001:DB8:ACAD: 0000000000000001::/64

   2001:DB8:ACAD: 0000000000000010::/64

   2001:DB8:ACAD: 0000000000000011::/64

   2001:DB8:ACAD: 0000000000000100::/64

4. Stejně tak jako u sumarizace v IPv4 určíme nejdelší **shodu bitů**. Nesmíme však zapomínat, že zde reprezentuje jeden hextet 16 bitů, nikoliv 8 bitů.
   V našem případě se shodujeme v těchto počtech bitů:
   první hextet = 16 b
   druhý hextet = 16 b
   třetí hextet = 16 b
   čtvrtý hextet = 13 b (do výskytu prvního odlišného bitu)
   Prefix pro sumarizovanou síť bude celkem /61 (16+16+16+13).
5. V tomto kroku ponecháme shodný počet bitů beze změny a ostatní doplníme nulami.

   2001:DB8:ACAD: 0000000000000000::/64

   2001:DB8:ACAD: 0000000000000000::/64

   2001:DB8:ACAD: 0000000000000000::/64

   2001:DB8:ACAD: 0000000000000000::/64

   Zvýrazněné části adresy představují společné části určené prefixem.

6. Převedeme binární tvar rozdílného hextetu zpět na hexadecimální tvar.

2001:DB8:ACAD: 0000::

7. Zapíšeme výslednou sumarizovanou adresu včetně hodnoty prefixu.

2001:DB8:ACAD: 0000::**/61**

Příklady sumarizace nad IPv6 k procvičení jsou uvedeny v příloze číslo 3.

5.7 STATICKÁ KONFIGURACE JEDNODUCHÉ LAN

5.7.1 Co označujeme jednoduchou sítí

Jednoduchá LAN. Každý si pod tímto pojmem představí něco jiného. Studentovi IT oboru se vybaví síť do deseti koncových zařízení, sekretářce nejspíše místnost, ve které pracuje.

Pro další vysvětlení statického směrování použijeme síť, v níž se snaží komunikovat dva kamarádi, Pepa a Zbyněk.

Topologie a adresace využitá pro konfiguraci statického směrování, je uvedena v příloze 4.

5.7.2 Jak vybrat vhodný směr pro pakety

Uvědomme si, že statické směrování není nic jiného než „naprogramování“ jasně daných pokynů, dle kterých se má směrovač chovat v případě, že nastane určitá situace. Není zde tedy vyžadována a hlavně ani povolena žádná vlastní iniciativa směrovače.

V současné chvíli jsme na pozici „programátora“, který musí obdobně jako při programování nejprve zvážit, co vše je potřeba naprogramovat, ještě před tím, než se pustí do práce. My musíme zvážit, které cesty určíme do jednotlivých sítí. Ve většině sítí existuje totiž více cest, které mohou být pro přeposílání paketů využity. Bylo by dost neefektivní vybírat směr cesty zcela náhodně.

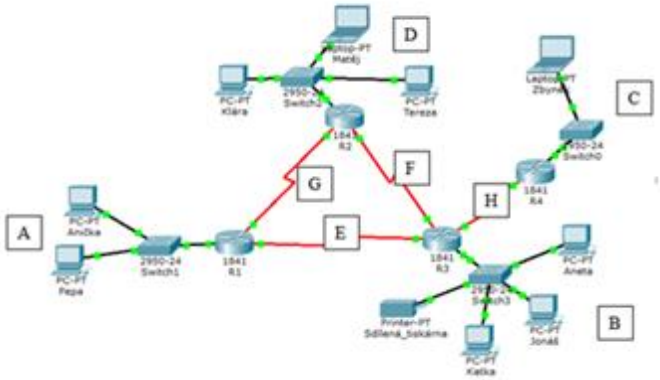
Při rozhodování o tom, kterou cestu zvolit, se můžeme řídit například:

- zatížením komunikace v dané části sítě,
- kvalitou použitých směrovačů na dané cestě,
- počtem mezilehlých směrovačů mezi zdrojem a cílem.

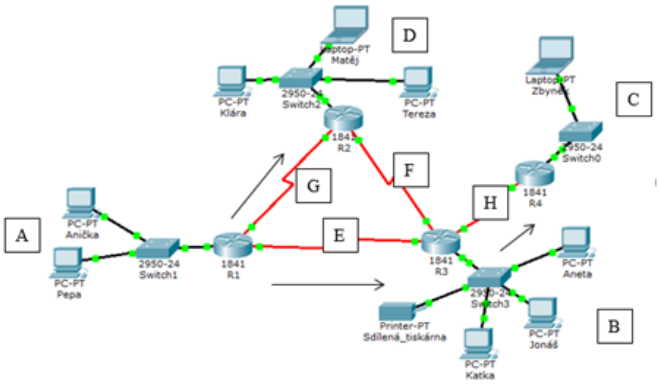
Kritérií existuje jistě mnohem více, zde jsme si uvedli pouze některá. Například poslední kritérium je důležité zejména z důvodu, že zpracování rámce na každém směrovači, ať již chceme nebo ne, trvá určitý časový interval. Tedy čím více těchto zpracovávání budeme

muset na cestě s rámcem provést, tím „déle“ (z hlediska uživatele není rozdíl patrný) bude doručení rámce trvat.

Následující obrázky ilustrují námi zvolené rozvržení cest do jednotlivých sítí včetně počátečního popisu sítě.

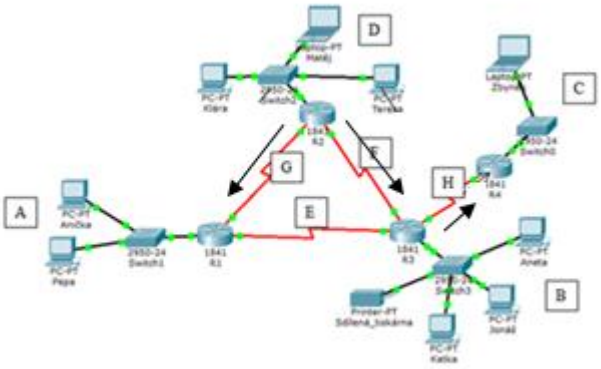


Obrázek 24 Počáteční rozložení sítí



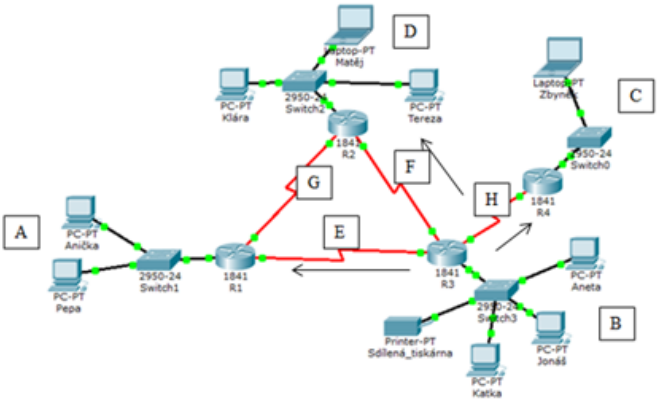
Směrovač R1	
Síť	Nutný průchod
B	E
D	G
C	E, H

Obrázek 25 Statické cesty ze směrovače R1



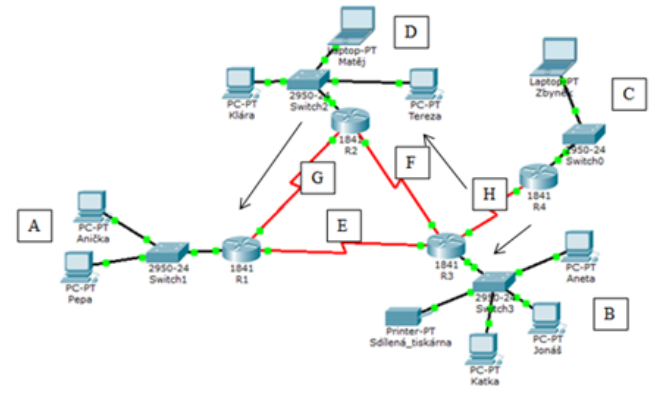
Směrovač R2	
Síť	Nutný průchod
A	G
B	F
C	F, H

Obrázek 26 Statické cesty ze směrovače R2



Směrovač R3	
Síť	Nutný průchod
A	E
D	F
C	H

Obrázek 28 Statické cesty ze směrovače R3



Směrovač R4	
Síť	Nutný průchod
A	H, F, G
B	H
D	H, F

Obrázek 27 Statické cesty ze směrovače R4

Vraťme se nyní k našemu úkolu, tedy umožnění komunikace mezi Zbyňkem a Pepou.

Z obrázku číslo 23 vidíme, že Zbyňkův notebook se nachází v síti C, zatímco Pepův počítač je umístěný v síti A. Budeme tedy potřebovat nejprve prozkoumat tabulky rozvržení cest pro

směrovače R1 a R4. Z nichž zjistíme, že pokud se budeme chtít dostat ze sítě C do sítě A, musíme rámec odesílat přes sítě H, F a G. Jistě si teď říkáte, že pokud by rámec byl odesílán pouze přes H a E byla by komunikace rychlejší. Ano, máte pravdu, nicméně toto je pouze cvičná topologie a tak předpokládejme, že síť E bývá velmi často zatížena, a proto veďme komunikaci „oklikou“. U směrovače R1 by byla komunikace ze sítě A do sítě C vedena průchodem přes sítě E a H. V tomto případě byla opravdu zvolena nejkratší cesta pro přeposílání paketu. Tuto analýzu je vždy nutné aplikovat ještě před samotným začátkem konfigurace směrování.

Postup, kterým nakážeme směrovači určitý typ chování je shrnut v příloze 5.

[1][10][11]

6 DYNAMICKÉ SMĚROVÁNÍ

Na rozdíl od směrování statického je více tolerantní ke změnám, vyžaduje méně administrativní režie a také je u konfigurace menší šance výskytu chyby ve směrovací tabulce, jelikož přidávání záznamů do směrovacích tabulek je prováděno automaticky pomocí směrovacích protokolů.

Největší výhodou dynamického směrování je reakce směrovače na výskyt chyby nebo výpadku. V případě takovéto situace směrovač okamžitě zareaguje a zvolí, tedy vypočte, novou trasu pro pakety.

Připomeňme si tři pojmy, a to:

- **Protokol** je označení pro sadu pravidel sloužících ke vzájemné synchronizaci komunikujících stran.
- **Autonomní systém** označuje ucelenou skupinu zařízení spadajících do jedné například firemní sítě. Autonomní systém označuje ucelenou skupinu zařízení pracujících pod společnou správou a udržujících jednu směrovací politiku. Příkladem mohou být internetoví poskytovatelé.
- **Směrovací doména** utváří ucelenou oblast, ve které dochází k výměně směrovacích informací. Tato oblast může být libovolně velká a platí pro ni, že veškerá mezilehlá

zařízení využívají stejný způsob určování trasy, tedy přesněji metriku pro výpočet nejlepší cesty.

Stejně tak jako směrování statické, členíme i směrování dynamické do několika skupin. Toto členění je dáno pohledem na směrovací protokoly. Nejčastěji se setkáme s následujícím členěním:

- podle výskytu,
- podle směrovací taktiky.

První skupina představuje členění **podle místa**, kde je směrovací protokol používán. Tato skupina obsahuje dvě podskupiny:

- **IGP** (Interior Gateway Protocol),
- **EGP** (Exterior Gateway Protocol).

Dalším kritériem, dle kterého lze členit směrovací protokoly je **podle složení metriky**, kterou protokoly používají k určení nejlepší cesty. Zde se setkáme s těmito dvěma skupinami

- protokoly typu vektoru vzdálenosti,
- protokoly typu stavu linky.

Více o jednotlivých skupinách popisují následující části této kapitoly.

6.1 INTERNÍ SMĚROVACÍ PROTOKOLY

Označovány také jako protokoly vnitřních bran. Jejich použití je zejména pro přeposílání směrovacích informací důležitých pro proces směrování **uvnitř** autonomního systému. To ovšem, jak jsme se již v předchozích částech textu dozvěděli, vůbec neznamená, že uvnitř jednoho autonomního systému je dovolen pouze jeden druh směrovacího protokolu. Nezapomeňme, že toto pravidlo“ je závazné pouze uvnitř směrovací domény.

6.2 EXTERNÍ SMĚROVACÍ PROTOKOLY

Jsou využívány pro výměnu informací mezi autonomními systémy. Ke své činnosti využívají protokoly typu vektoru cesty (path vector), které pracují na principu udržování informací o cestě, která je automaticky aktualizována. Celý postup probíhá tak, že v pravidelných intervalech jsou vysílány zprávy, obsahující informace o dostupnosti sítí v okolí aktuálně vysílajícího směrovače. O okolí směrovače je evidována metrika a následující přeskok.

6.3 PROTOKOLY VEKTORU VZDÁLENOSTI

První skupina interních směrovacích protokolů zahrnuje protokoly využívající **Bellman-Fordův algoritmus**, který bude popsán později v této kapitole. Tyto protokoly patří ke starším a pro určení nejlepší trasy je využíván pouze **počet přeskoků**. Nevýhodou však může být vyšší doba **konvergence sítě**. Konvergencí rozumíme okamžik, za který jsou všechna

zařízení ve směrovací doméně seznámena s aktuálním stavem sítě. K oznamování stavu sítě nedochází zcela automaticky, ale v pevně stanovených velmi krátkých časových intervalech. Nejspíše nikoho nepřekvapí, že o tyto časové úseky se stará **časovač**, který má odlišnou dobu generování intervalu pro jednotlivé protokoly.

Princip je tedy takový, že každý ze směrovačů obsahuje časovač, jež má být spuštěn řekněme každých 10 milisekund. Po uplynutí 10 milisekund směrovač využije svou směrovací tabulku a určí vzdálenosti ke všem ostatním směrovačům. Tuto souhrnnou informaci pak odešle na všechny okolní směrovače a zároveň informace dostává i tento směrovač.

Po dobu vzájemného přeposílání směrovacích tabulek nemají směrovače ucelenou představu o aktuálním rozvržení sítě. Díky tomuto faktu by paket nemusel spolehlivě dorazit do cíle, a proto v tento časový okamžik nelze odesílat pakety. Podle doby konvergence lze porovnávat jednotlivé protokoly mezi sebou.

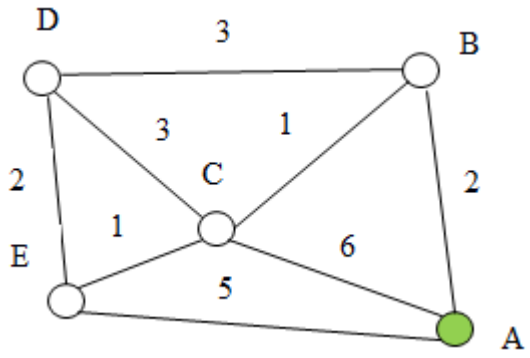
Interní směrovací protokoly lze podrobit dalšímu dělení, a to na protokoly skupiny RIP a IGRP (respektive EIGRP) rozdíly mezi nimi budou probrány dále v této kapitole.

6.3.1 Bellman – Ford algoritmus

Algoritmus Bellman-Ford se využívá v síťových grafech, k určení nejkratší cesty v grafu. Pokud nahlédneme na počítačovou síť, nejedná se o nic jiného než o síťový graf. Vrcholy v tomto grafu nám představují směrovače a hrany jsou reprezentovány sériovými spoji mezi těmito směrovači.

Podíváme-li se do literatury, nalezneme v souvislosti s tímto algoritmem pojem ohodnocení hrany. Ohodnocení hrany je důležité pro vlastní výpočet. Ohodnocením rozumíme hodnotu, udávající například spolehlivost nebo propustnost spoje. Není však výjimkou ani nejlehčí způsob odlišení hran, a to vzdálenost. Přímo tento způsob hodnocení hran využívají pro výpočty všechny směrovací protokoly typu vektoru vzdálenosti.

Mějme následující síťový graf

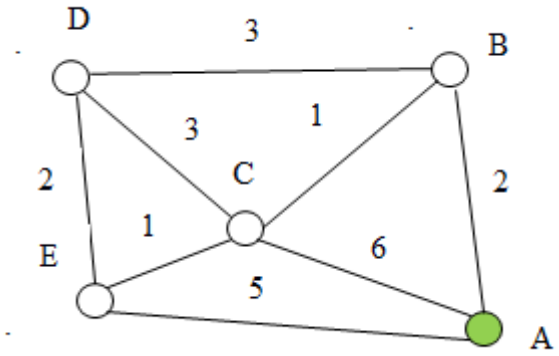


Obrázek 29 Bellman-Ford algoritmus

Na počátku jsou veškeré hrany grafu ohodnoceny „nekonečnou“ vzdáleností, která indikuje pouze stav, že tato hrana (respektive cesta) nebyla doposud analyzována. Nezapomeňme, že vždy je tento postup aplikován z jednoho **konkrétního vrcholu**. Tento počáteční vrchol je poté ohodnocen hodnotou 0, aby byl jasně odlišen od ostatních.

V našem příkladu by tedy vrcholy byly ohodnoceny následovně

vrchol	hodnota
A	0
B	∞
C	∞
D	∞
E	∞



Obrázek 30 Bellman-Ford algoritmus výchozí ohodnocení

Následují kroky, v nichž jsou vždy testovány sousední vrcholy, u nichž je hledána nejnižší možná hodnota ohodnocení.

Začněme ve vrcholu A, tento vrcjež obsahuje tři sousedící vrcholy, a to E, C a B. Tyto vrcholy jsou zatím ohodnoceny hodnotou nekonečno a tudíž jsou porovnávány následující podmínky:

Je 0 + 2 menší než nekonečno? Tedy je vrchol B ohodnocen hodnotou 2.

Je 0 + 6 menší než nekonečno? Tedy je vrchol C ohodnocen hodnotou 6.

Je 0 + 5 menší než nekonečno? Tedy je vrchol E ohodnocen hodnotou 5.

Nová tabulka ohodnocení vrcholů vypadá následovně.

Vrchol	hodnota
A	0
B	2
C	6
D	∞
E	5

Obrázek 31 Bellman-Ford algoritmus nové ohodnocení po kroku 1

Nyní jsou postupně prozkoumávány jednotliví sousedé těchto vrcholů, tedy z vrcholu B je prozkoumávána vzdálenost do vrcholů D, C a A. Bod A, jakožto bod výchozí není nutné ani zkoumat, jelikož hodnota jeho ohodnocení je nula, a tudíž žádné jiné ohodnocení nemůže být menší (Pozn.: záporné hodnoty nejsou povoleny). Zkoumejme tedy pouze cesty do vrcholu D a C. Při ohodnocování klademe tyto otázky

Je $2 + 3 < \infty$? Tedy je vrchol D ohodnocen hodnotou 5.

Je $2 + 1 < 6$? Tedy je vrchol C ohodnocen hodnotou 3.

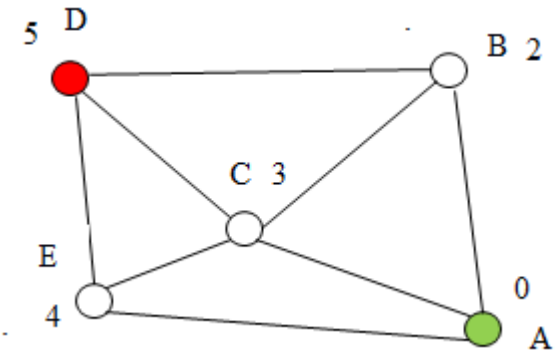
Stejným způsobem nalezneme ohodnocení sousedů vrcholů C a E jež jsou také sousedními vrcholy počátečního bodu A.

Po provedení tohoto ohodnocování dostaneme následující tabulku s ohodnoceními vrcholů.

vrchol	hodnota
A	0
B	2
C	3
D	5
E	4

Obrázek 32 Bellman-Ford algoritmus nové ohodnocení po kroku 2

Pokud již máme graf, u kterého nelze najít žádnou nižší hodnotu pro kterýkoliv vrchol, ukončili jsme ohodnocování. Nyní musíme zaznamenat nejkratší cestu. Vezměme například, že se chceme dostat do vrcholu D. Je před námi tedy takovýto graf s ohodnoceními.



Obrázek 33 Bellman-Ford algoritmus graf s novými ohodnoceními hran

Pokud chceme určit, kudy vede nejkratší cesta, musíme jít, na rozdíl od principu ohodnocování, od vrcholu koncového. Při tomto průchodu pak zkoumáme, zda

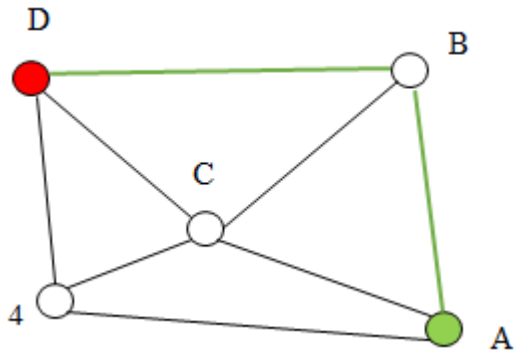
ohodnocení vrcholu – ohodnocení hrany

je rovno ohodnocení vrcholu sousedního. Pokud ano označíme spoj mezi nimi jako část nejkratší cesty.

Zpět k našemu příkladu, začínáme ve vrcholu D, jenž má ohodnocení 5. Od tohoto vrcholu vedou 3 hrany ohodnocené hodnotami 3, 3 a 2. Hledáme tedy sousední vrchol takový, jenž bude ohodnocen hodnotou 2 nebo 3. Tyto hodnoty jsme získali odečtením ohodnocení hrany

od ohodnocení vrcholu. Výsledek musí odpovídat následujícímu ohodnocení vrcholu, jenž cestě náleží. Sousední vrcholy vrcholu D mají ohodnocení 4, 3 a 2. Vybereme tedy spoj mezi vrcholy D a B a zařadíme jej do cesty. Dále pokračujeme z vrcholu B, ze kterého již vede cesta přímo do cílového vrcholu A.

Naše výsledná nejkratší cesta tudíž povede následovně a její délka bude pět jednotek.



Obrázek 34 Bellman-Ford algoritmus výsledná cesta

6.4 PROTOKOLY STAVU LINKY

Použití protokolů vektoru vzdálenosti je zejména v sítích menšího až středního rozsahu, zatímco pro rozsáhlejší sítě se využívají protokoly stavu linky. Jsou sice náročnější pro směrovač díky neustálému testování svých sousedů, avšak reagují tak mnohem rychleji na změny v síti a nadměrně ji nezatěžují svojí režií.

Prvním představitelem této skupiny protokolů byl IS-IS. Tento protokol slouží jako model založený na referenčním modelu ISO/OSI, zatímco OSPF či jiné jsou stavěny na architektuře IP.

6.4.1 Základní vlastnosti

Směrovací protokoly stavu linky jsou velmi náročné pro samotný směrovač. Tato náročnost je udána neustálým testováním všech sousedů směrovače, na němž je protokol stavu linky spuštěn. Každý ze směrovačů v síti má **kompletní přehled o celé topologii**. Díky této znalosti může být vypočítávána nejlepší cesta skrze síť. Nejlepší cestou zde nerozumíme pouze síť nejméně vzdálenou, či nejvíce propustnou. Nejlepší cesta je volena na základě kritérií, jež můžeme ovlivňovat, a může být volena na základě složených hodnot. Parametry

sítě, jež jsou použity pro výpočet ceny linky, určuje administrátor, avšak v implicitním stavu se u jednotlivých směrovacích protokolů liší.

Se znalostí celé topologie je možné dodržovat pravidla autonomního systému, či zajišťovat kvalitu služeb (QoS). Výpočet metrik probíhá pomocí **Dijkstrova algoritmu**, který bude zmíněn v následující části této kapitoly.

Oproti protokolům vektoru vzdálenosti přináší tento typ protokolů značné ulehčení komunikace v síti. Protokoly vektoru vzdálenosti zmíněné odesílají při zjištění změny v topologii celou svou směrovací tabulku všem zařízením, zatímco protokoly stavu linky odesílají směrovací tabulku pouze ke svým sousedům.

Další však neméně opomíjenou vlastností je autentifikace veškerých připojených zařízení, čímž je zajištěna vyšší bezpečnost sítě.

6.4.2 Dijkstrův algoritmus

Mezi základními vlastnostmi protokolů stavu linky bylo uvedeno, že výpočet nejlepších cest probíhá pomocí Dijkstrova algoritmu nejkratší cesty. Odpoutejme se nyní na okamžik od světa počítačových sítí a nahlédněme alespoň částečně do principu tohoto algoritmu, jenž byl vymyšlen roku 1959 nizozemským informatikem Edsgerem Dijkstri.

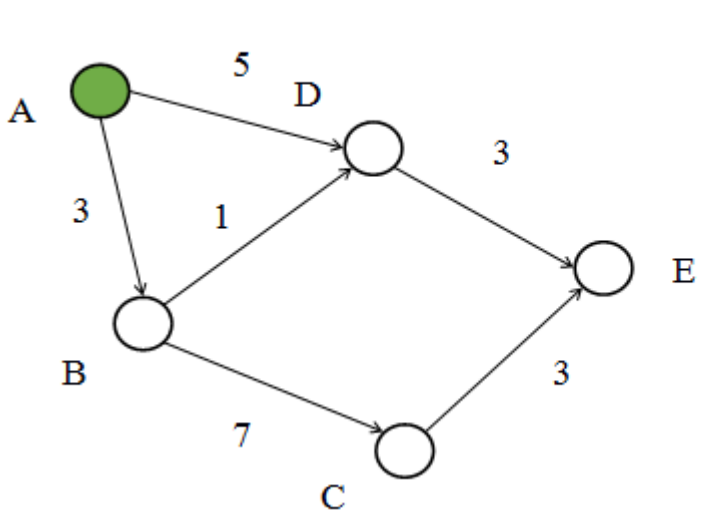
Z textu výše vyplývá, že se jedná o algoritmus nalezení nejkratší cesty v grafu. V našem případě je síť obyčejným síťovým grafem, kde ohodnocení jednotlivých hran představuje hodnota ceny linky. Pomocí těchto cen linek mohou být ohodnoceny jednotlivé vrcholy grafu. Graf je prohledáván do šířky. Graf není procházen podle počtu hran vedoucích od zdroje, avšak podle vzdálenosti ke zdroji.

Na začátku jsou všechny vrcholy mimo vrcholu počátečního ohodnoceny hodnotou nekonečno tak, aby byly jasně odlišeny od těch, které nebyly doposud zkoumány. Následně jsou všechny potomci aktuálního vrcholu testováni, zda se nenachází blíže ke zdroji, tudíž jestli jejich ohodnocení není nyní nižší, než bylo při předchozím testování. Pro každého z potomků je ověřována následující podmínka:

$$VZ_z + DH_{z,p} < VZ_p$$

kde $\mathbf{VZ_z}$ značí vzdálenost aktuálně testovaného vrcholu, $\mathbf{DH_{z,p}}$ představuje délku hrany mezi testovaným vrcholem a jeho potomkem a nakonec $\mathbf{VZ_p}$ značí vzdálenost potomka před prováděním testování.

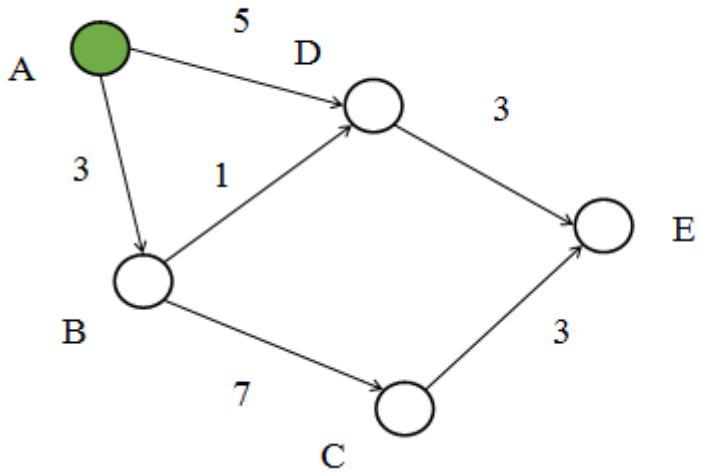
Pokud je výsledek této nerovnosti kladný, je nová vzdálenost kratší než původní nastavená a potomku je nastavena nová hodnota ohodnocení. K ukončení procházení grafu dochází v okamžiku prohledání všech vrcholů grafu.



Vrchol	Hodnota
A	0
B	∞
C	∞
D	∞
E	∞

Obrázek 35 Dijkstrův algorimus výchozí graf a ohodnocení

Následně jsou ohodnoceny potomci vrcholu A, jejich ohodnocením se stává ohodnocení hran, jelikož cokoliv bude nižší nežnekonečno a zároveň je počáteční bod ohodnocen hodnotou nula.



Vrchol	Hodnota
A	0
B	3
C	∞
D	5
E	∞

Obrázek 36 Dijkstrův algoritmus krok 1

V následujícím kroku postupujeme na vrchol, jehož ohodnocení, tedy vzdálenost od vrcholu počátečního je nejmenší. Pokud jich máme více, pak vybereme náhodně jeden z nich.

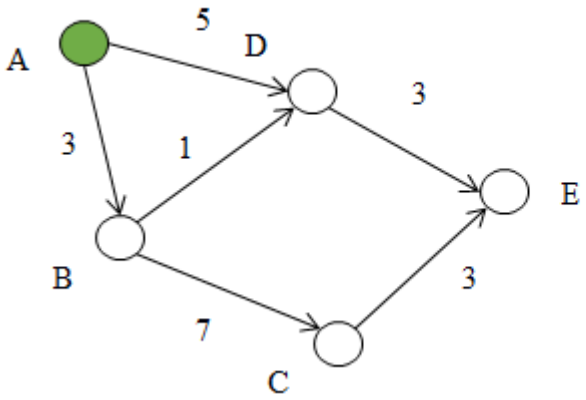
Zvolme tedy v našem příkladu vrchol B k dalšímu hodnocení a projděme všechny vrcholy, jež jsou jeho potomky. U každého z potomků se řídíme následující podmínkou, jež byla uvedena v úvodní části této kapitoly:

$$VZ_z + DH_{z,p} < VZ_p$$

3 + 7 < ∞ ? Ano, tedy ohodnotíme vrchol C hodnotou 10.

3 + 1 < 5 ? Ano, tedy nová hodnota ohodnocení vrcholu D bude 4, nikoliv původních 5.

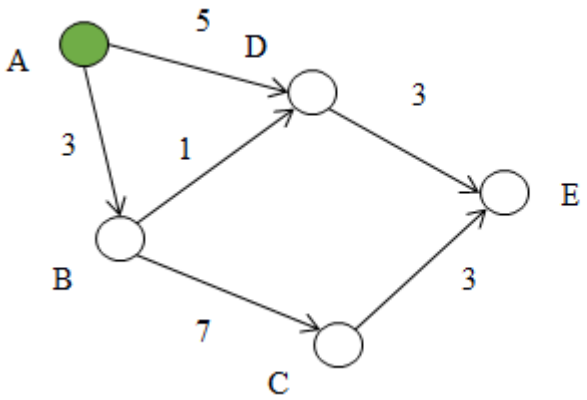
Jelikož máme graf orientovaný, pro vrchol A bychom ohodnocení odečítali, a tudíž by zůstala hodnota 0.



Vrchol	Hodnota
A	0
B	3
C	10
D	4
E	∞

Obrázek 37 Dijkstrův algoritmus krok 2

V dalším kroku budeme testovat potomky vrcholu D, jelikož je jeho ohodnocení nižší, než u vrcholu C. Testujeme vrchol E, který zatím nebyl ohodnocen, a tedy bude jeho ohodnocení pouze součtem ohodnocení vrcholu D spolu s ohodnocením hrany, vedoucí mezi vrcholy D a E.

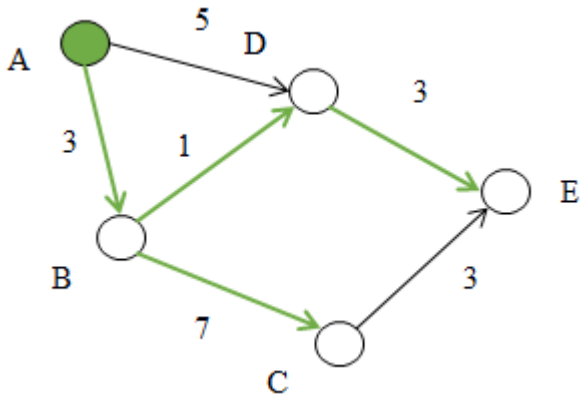


Vrchol	Hodnota
A	0
B	3
C	10
D	4
E	7

Obrázek 39 Dijkstrův algoritmus krok 3

Nakonec projdeme ještě poslední vrchol C, jeho potomkem je pouze vrchol E, který je ohodnocen hodnotou 7. Pokud bychom tento vrchol hodnotili z pohledu vrcholu C, byla by jeho nová hodnota rovna 10, tedy vyšší než stávající ohodnocení, a proto tento vrchol zůstane s ohodnocením 7.

Nyní máme kompletní představu o nejkratších cestách z vrcholu A do všech ostatních vrcholů grafu. Tyto cesty představuje následující obrázek a celý proes je označován jako výpočet **stromu nejkratších cest**.



Obrázek 41 Dijkstrův algoritmus výsledný strom

V tomto okamžiku máme jeden záznam o vzdálenostech vrcholů. Ovšem abychom měli kompletní představu o vzdálenostech mezi vrcholy v síti, museli bychom tento postup aplikovat pro každý z vrcholů.

[14][15][16]

6.5 PROČ ZVOLIT DYNAMICKÉ SMĚROVÁNÍ

V přechozí kapitole o statickém směrování jsme se dozvěděli, jak směrovač pracuje při přeposílání paketů a naučili jsme se zadávat statické cesty. Jak jsme však viděli, konfigurace statického směrování v naší topologii byla velmi pracná. Říkáte si „tak co, jednou si to nakonfiguruji a pak to bude pracovat“, ano, máte pravdu, v takovéto menší síti by se tento přístup dal aplikovat, pokud by nebylo plánováno budoucí rozšiřování sítě či topologické změny.

Nezapomeňte však, že nikdy nezabráníte chybám hardwaru. I nejvýkonnější zařízení se zaručenou vysokou spolehlivostí může například v důsledku výpadku elektrického proudu vypovědět službu.

V běžném životě jsou počítačové sítě tvořeny desítkami, stovkami i tisíci koncovými zařízeními. Při takovýchto rozměrech není již možné aplikovat první přístup, protože by konfigurace byla prováděna několik týdnů či měsíců. Navíc nezapomínejme, že síťový administrátor nemá přístup ke všem zařízením připojeným do sítě. Důvodů je několik. Koncová zařízení se mohou nacházet v různých částech státu či světa. Dalším případem je třeba neumožnění přístupu administrátora k počítači ve vládní organizaci, ve kterém se nachází citlivé údaje. Obdobou může být například i banka. Navíc v praxi, ve velkých sítích,

spolupracuje více administrátorů najednou a každý si spravuje svoji přidělenou autonomní oblast. V těchto případech je pravý okamžik na použití dynamických směrovacích protokolů.

7 NEPLÝTVÁME PROSTOREM, ANEB JAK ADRESOVAT VLSM A CIDR

7.1 DŮVOD PRO EFEKTIVNĚJŠÍ VYUŽÍVÁNÍ ADRESOVÉHO PROSTORU

Není tomu dlouho, co v Evropě došlo k vyčerpání IPv4 adres. K výskytu této události by zřejmě došlo mnohem později, kdybychom lépe využívali adresový prostor. Z počátku byly IP adresy využívány bez rozvahy nad budoucí možnou adresovou situací.

Adresy, jejichž přidělování bylo zcela náhodné, navíc značně rozšiřovaly velikost směrovacích tabulek, tedy i z pohledu směrování se jednalo o značně neefektivní způsob adresace. Navíc nezapomeňme, že směrovač je zařízení jako každé jiné, tudíž má omezenou operační paměť, která by rozsáhlou směrovací tabulku ani nemusela uchovat. Z těchto důvodů bylo nezbytné přejít na takový způsob adresování, u kterého bude možné přizpůsobovat si rozdělení přiděleného adresového prostoru.

7.2 CIDR

Prvním takovým adresním schématem byl CIDR, zkratka pro **C**lassless **I**nter-**D**omain **R**outing, který byl specifikován v RFC 4632 z roku 1993. S příchodem CIDR tudíž byly zrušeny třídy IP adres. Mezi základní 2 cíle, s nimiž CIDR vstoupil do síťového světa, patří

- zrušení limitu délky adresy sítě,
- možnost hierarchického přidělování adres.

Prvním rozumíme možnost využívání libovolných prefixů. Prefixy jsou zapisovány v obecném tvaru adresa**/prefix**, kde prefix označuje, kolik bitů je rezervováno pro adresu sítě a z kolika tedy můžeme adresovat hostitelská zařízení. Tento zápis označujeme jako **CIDR notace** a slouží k zaznamenání zjednodušené masky sítě. Již nemusíme psát

192.168.20.0 255.255.255.0

ale stačí nám zýpis, dle CIDR notace

192.168.20.0/24

Na tomto místě je vhodné poznamenat, že nelze získávat rozsah adres pouze na základě našeho přání. Pro přidělování adresového prostoru je celá řada přísných pravidel. Kdo se o přidělování adresových rozsahů stará? Pro tento účel existují **lokální** a **regionální registrátoři**. Mezi lokální registrátory (dále jen LIR) jsou zahrnovány zejména poskytovatelé připojení k Internetu, kteří se starají o správné rozdělení svého adresového prostoru, jenž jim byl přidělen od regionálního internetového registru, dále jen RIR. RIR stanovuje pravidla přidělování, jenž jsou uplatňovány u LIR.

V současné době existuje pět regionálních registrátorů. Tito registrátoři jsou rozčleněni dle jednotlivých kontinentů. S nadsázkou lze říci, že celý Internet je rozdělen mezi tyto registrátory. A jak si prostor rozdělily? Samozřejmě to nebylo pouze na nich, adresy s příslušným prefixem jim byly přiděleny organizací ICANN.

7.3 VLSM

Neboli **V**ariable **L**ength **S**ubnet **M**ask. Na rozdíl od CIDR, u kterého bylo nutností, aby všechny podsítě byly stejné velikosti, umožňuje VLSM přizpůsobení adresového prostoru přímo podle potřeb sítě. Lze si vytvořit adresové rozsahy přímo na míru, a tudíž tato metoda přispívá k nejefektivnějšímu využívání adresového prostoru.

7.3.1 Základní kroky

Postup adresace VLSM si lze rozdělit do 4 kroků.

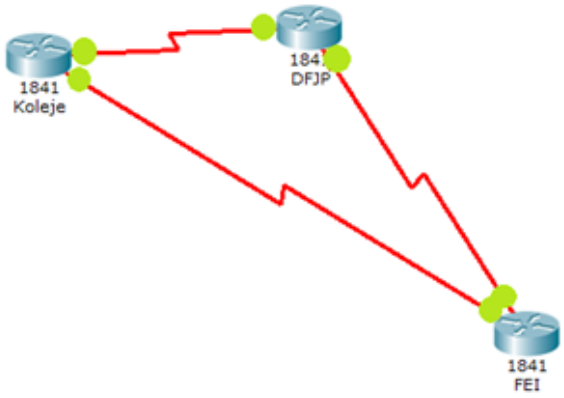
1. Určení počtu zařízení, jež budou připojeny do největší z adresovaných podsítí.
2. Výpočet masky podsítě pro největší podsíť a určení příslušných adres.
3. Opakování postupu je nutné opakovat postupně podle velikosti pro všechny sítě.
4. Adresace sériových spojů mezi zařízeními.

Není však nutné se striktně tohoto postupu držet, různé zdroje uvádí rozdílné způsoby, jak k adresaci VLSM přistupovat. Některé dokonce nezačínají od největších sítí, ale nejprve od sériových spojů. Ovšem asi nejlogičtějším a nejintuitivnějším postupem je začít od největší sítě, jež bude potřebovat nejvíce hostitelských adres.

7.3.2 Určování adres

Jak postupovat při samotném určování adres? Způsobů je vícero. Jedni používají pomocnou tabulku, jiní zase počítají z hlavy. V rámci našeho kurzu budeme využívat pomocnou tabulku, kterou naleznete jako přílohu K.

Pokud máme takovouto tabulku k dispozici, již nám nic nebrání začít adresovat. Dejme tomu, že jsme dostali od poskytovatele připojení adresu **172.168.20.0/24**, kterou můžeme využít pro adresování sítě. Pro náš příklad mějme následující síť s tímto rozložením hostitelských zařízení.



Označení	Počet hostů
FEI	230
Koleje	2300
DFJP	400

Obrázek 42 VLSM topologie a tabulka počtu hostů

Postup při adresování VLSM zmíněný výše radil, abychom s adresováním začali od největší podsítě. Tudíž začneme podsítí kolejí, ta dle tabulky musí být přístupná pro 2300 hostů. V tabulce pro adresování nalezneme takový záznam, který ve sloupci počet hostů obsahuje hodnotu, jež pokryje počet hostitelských adres. Vybíráme vždy ovšem tu nejnižší, abychom neplýtvali zbytečně adresami. Na druhou stranu nemůžeme pro největší podsíť využít prefix /21, jelikož ten dovoluje pouze 2048 adres, což pro naši podsíť nestačí. Pro naše potřeby odpovídá následující řádek tabulky:

Počet adres	Počet bitů	Prefix	Třída	Maska
4096	12	/20	16C	255.255.240.0

Je dobrým zvykem rozepsat si adresování sítě do takovéto podoby, nejlépe ve formě tabulky:

Tabulka 5 VLSM návrh přehledné formy zaznamenání

Adresa sítě	
Maska sítě	
Adresa 1. Hosta	
Adresa posledního hosta	
Broadcast	

Takovýto způsob záznamu je nejpřehlednější, jak pro naše potřeby, tak pro budoucí administrátory, kteří se v adresaci sítě lehčeji vyznají. Jednotlivá pole vyplníme následovně.

- **Adresa sítě** je vždy první adresa v rozsahu, v našem případě se jedná o adresu 172.168.20.0
- **Maska sítě** odpovídá sloupci maska sítě z tabulky.
- **Adresa 1. hosta** je první adresa, jež může být přidělena k použití v síti. Tato adresa je vždy o jednu adresu vyšší než adresa sítě.
- **Adresa posledního hosta** je poslední adresa, která může být přidělena. Doporučuji tuto položku vynechat a doplnit až po zjištění adresy broadcast. Adresu posledního hosta pak zjistíme jednoduše odečtením jedné hostitelské adresy.
- **Broadcast** je poslední adresa rozsahu. Nejjednodušším způsobem je využití součtu adresy sítě spolu s inverzní maskou podsítě. Inverzní maskou rozumíme její doplnění do podoby 255.255.255.255 a v literatuře bývá nazývána jako **wildcard mask**.

V našem případě by byla tato adresa spočtena následovně

172.168.20.0

+ 0.0.15.255

172.168.35.255

A kde bude začínat další síť? Na adrese 172.168.36.0, jelikož do adresy 172.168.35.255 nemůžeme v posledním oktetu přidat dalšího hosta. Tedy musíme zvýšit hodnotu třetího oktetu.

Správně vyplněná tabulka, pro podsíť obsahující nejvíce hostitelských zařízení, bude vypadat následovně:

Tabulka 6 VLSM pro koleje

Adresa sítě	172.168.20.0
Maska sítě	255.255.240.0
Adresa 1. Hosta	172.168.20.1
Adresa posledního hosta	172.168.35.254
Broadcast	172.168.35.255

Další největší sítí z výběru je podsíť pro DFJP. Její adresace bude vypadat následovně:

Tabulka 7 VLSM pro DFJP

Adresa sítě	172.168.36.0
Maska sítě	255.255.254.0
Adresa 1. Hosta	172.168.36.1
Adresa posledního hosta	172.168.37.254
Broadcast	172.168.37.255

Nejmenší sítí je podsíť pro FEI, jenž musí obsahovat 200 hostitelských adres:

Tabulka 8 VLSM pro FEI

Adresa sítě	172.168.38.0
Maska sítě	255.255.255.0
Adresa 1. Hosta	172.168.38.1
Adresa posledního hosta	172.168.38.254
Broadcast	172.168.38.255

Spoje mezi směrovači musí mít také naadresovány své podsítě. Jelikož adresa vypadá vždy stejně, uvedeme si pouze první spoj.

Tabulka 9 VLSM spoj

Adresa sítě	172.168.39.0
Maska sítě	255.255.255.252
Adresa 1. Hosta	172.168.39.1
Adresa posledního hosta	172.168.39.2
Broadcast	172.168.39.3

A kde bude začínat následující adresa sítě? Na adrese 172.168.39.4

POZOR! Nezapomínejte připočítávat vždy dvě adresy k celkovému počtu potřebných hostů. V některých případech se totiž může stát, že na první pohled by nám rozsah stačil, avšak po přičtení povinných 2 adres (adresa sítě a broadcastu) můžeme být nuceni využít nižší prefix.

[11][12]

8 RIP

8.1 OBECNĚ O PROTOKOLU

S tímto protokolem vektoru vzdálenosti se setkáme zejména u sítí malého až středního rozsahu. Rozhodneme se pro něj, pokud hledáme jednoduchý a navíc snadno aplikovatelný protokol. RIP využívá ke svému rozhodování metriku určenou vzdáleností. Způsob, jakým se vzdálenost určuje, byla předvedena v části nazvané Bellman-Fordův algoritmus.

Využitelnost mezi uzly v sítích do omezeného rozsahu je dána restrikcí možné maximální vzdálenosti uzlů. Pro protokol RIP je maximální přípustnou vzdáleností **15 přeskoků**. Pokud délka cesty přesáhne stanovenou limitní hodnotu je považována cílová síť za nedostupnou.

Velmi důležitou vlastností směrovacího protokolu je plno třídnost. Plnotřídní protokol nepodporuje adresaci pomocí VLSM.

Protokol RIP neumí rozpoznávat členění adresového prostoru, tudíž jsme omezeni pouze na rozsahy adres svých tříd. Mezi jednotlivými zařízeními odesílána vždy **souhrnná informace** týkající se aktuálně připojených sítí k vysílajícímu směrovači. Aby mohly být vysílány souhrnné informace, je nutná jednotnost a spojení vícero adres do jedné. Takto spojené cesty označujme za sumarizované. K této sumarizaci dochází automaticky na hranicích jednotlivých tříd. Nezapomeňme, že protokol RIP zná pouze plné adresy tříd, z toho vyplývá, že sítě, které pro nás představují díky svému prefixu zcela rozdílné oblasti, jsou spojeny do jedné. Směrovač tedy není na základě cílové adresy určit, o kterou podsíť se jedná, jelikož se celkově jeví jako jedna jediná větší síť.

V současné době, kdy se snažíme co nejvíce šetřit adresovým prostorem je proto výhodnější využívat protokoly, jež nám tuto vlastnost umožňují. Ač se nyní může zdát, že tento protokol nemá většího významu, nezapomeňme, že v době svého vzniku, tedy okolo roku 1988 byl přelomovým a velmi úspěšným. Jak bylo zmíněno v úvodu této kapitoly, i dnes si najde své využití.

8.2 Časovače protokolů RIP

Časovače slouží k řízení výkonnosti sítě. Mezi výkonnost sítě řadíme zejména co nejefektivnější způsob šíření aktualizací sítí či zdržení vyslání aktualizace v případě neplatné koncové sítě. Mezi základní časovače řadíme:

- **Update timer** sloužící k periodickému odesílání aktualizací. Implicitní nastavení je 30 sekund. Ovšem nelze z důvodu možného zahlcení linek, každých 30 sekund najednou umožnit všem směrovačům odeslat své aktualizace. Z tohoto důvodu je vždy přičten náhodný, avšak velmi krátký, časový interval a díky tomu nedochází k zahlcování sítě.

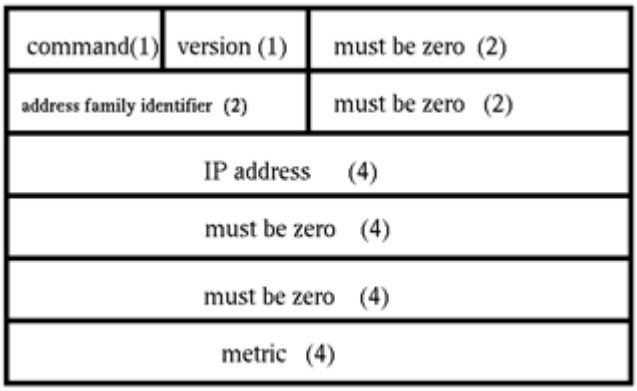
- **Timeout timer** po jehož vypršení tohoto časovače je cesta označena za neplatnou.

- **Flush timer**, který po svém po vyčerpání spustí vymazání všech neplatných záznamů. Záznam považujeme za neplatný, je-li metrika cesty rovna 16.

S časovači se setkáme v mírných modifikacích i u ostatních směrovacích protokolů. Dalšími nástroji směrovacích protokolů, sloužícími k prevenci vzniku smyček a likvidaci neplatných cest, jsou split-horizon, reverse poisoning nebo holddown časovač.

8.3 Hlavička RIPv1

Nyní víme, jak protokol RIP v síti nakonfigurovat i jak nastavení zkontrolovat, avšak stále nevíme, jak vypadá formát zprávy, která je přeposílaná po síti. Formát je následující:



Obrázek 43 Hlavička protokolu RIPv1

Celá hlavička protokolu RIPv1 se skládá z 8 polí. První pole **command** slouží k indikaci, zda je vysílaný paket takzvaně požadavkový či obsahuje odpověď. Nikoho zcela jistě nepřekvapí,

že požadavkový paket vyžaduje od jiného směrovače určité informace. V tomto případě požadavkový paket vyžaduje ověření, zda byla odeslána kompletní směrovací tabulka. Naopak paket odpovědový může být přijat i nevyžádaně a zpravidla se jedná o update. Odpověďový paket v sobě obsahuje jednotlivé záznamy směrovací tabulky. Pokud se nevejde obsah směrovací tabulky do jednoho RIP paketu, je jednoduše celý obsah rozdělen do několika po sobě jdoucích paketů.

Dalším polem, které je obsaženo v RIP formátu zprávy je **version number**, což není nic jiného, než označení o kterou verzi protokolu RIP se jedná. Možná vás při prohlížení struktury zarazilo, že se zde vyskytují čtyři pole s označením **zero**. Nejsou zde však zcela bezdůvodně. Každá z verzí protokolu RIP má lišící se strukturu zprávy, a tak je nutné, aby byla zajištěna vzájemná kompatibilita. Položka **AFI**, neboli *adress-family identifier*, slouží k identifikaci použité adresní rodiny. Následuje další **zero** pole. Pole **adress** specifikuje IP adresu jednotlivých záznamů. Přeskočme další dvě části obsahující zero pole a posledním polem v hlavičce je metric. **Metric** slouží ke kontrole, kolik bylo v daném okamžiku překročeno směrovačů na cestě do cílové sítě.

Nyní si nejspíše kladete otázku, zda pro každý záznam ze směrovací tabulky musí být vyslán samostatný RIP paket. Pokud by tomu bylo takto, nebylo by zřejmě využití protokolu RIP efektivní, a proto v jedné zprávě protokolu RIP je možno uvést až **25 různých záznamů** směrovací tabulky.

8.4 KONFIGURACE RIP

Mezi výhody protokolu RIP je jeho velmi snadná konfigurace. Celé nastavení provedeme sérií příkazů, provedenou v globálním konfiguračním režimu.

```
R (config)# router rip
```

Výše uvedeným příkazem směrovači řekneme, že chceme pro směrování zapnout proces protokolu RIP. Následné přidávání sítí do směrovací tabulky směrovače provádíme tímto příkazem:

```
R (config-router)# network <adresa přímo připojené sítě>
```

Tento příkaz provedeme pro všechny přímo připojené sítě ke směrovači. Takto naplněná tabulka je poté distribuována všem zařízením v síti. Je velmi vhodné nastavovat také výchozí bránu na hraničních směrovačích.

```
R (config)# router rip
R (config-router)# default information originate
```

Avšak dávejte si při konfiguraci pozor. Tímto příkazem se **nenastaví výchozí brána**! Pouze jím zajistíme, že nastavená výchozí brána bude sdílena mezi zařízeními. Pro zajímavost uveďme, že existuje i možnost publikovat vždy výchozí bránu, ať je již nastavena nebo ne. Tuto možnost aktivujeme pomocí parametru *always* přidaného k předchozímu příkazu.

Dalším, avšak ne velmi častým příkazem je

```
R (config-router)# passive-interface <označení rozhraní>
```

Tento příkaz zajistí, že rozhraní nebude vysílat aktualizace, ale pouze je přijímat.

Ukázkové řešení využití protokolu RIP na ukázkové topologii, včetně možnosti kontroly konfigurace je v příloze 8.

[3][4][6]

9 RIPv2

9.1 RIPv2 OBECNĚ

V předchozí kapitole jsme se dočetli o protokolu, který ve své době slavil velký úspěch, avšak nyní je pro své limity zastaralý. Protokol RIP nezůstal na cestě za vývojem osamocen. I v dnešní době existují uživatelé využívající tento starší protokol pro svou jednoduchost, rozhodlo sdružení IETF nominovat výbor, který měl jeden jasný cíl, a to prozkoumat další aktualizace a také rozšířit původní protokol o podporu adresování s proměnnou délkou masky.

Roku 1994 byl specifikován protokol RIPv2. Již samotné označení „verze 2“ evokuje, že se jedná o pokročilejší verzi předešlého protokolu RIP a jak je tomu u nástupců, přináší i RIPv2 vylepšení svého předchůdce. Samozřejmě, že stále se jedná o protokol typu vektoru vzdálenosti, to nemohlo být změněno, jelikož by se již nejednalo o rozšíření původní verze, ale o zcela nový protokol. Nicméně bohužel nepřinesl RIPv2 zrušení maximálního počtu přeskoků, a tudíž jsme stále omezeni 15 přeskoky.

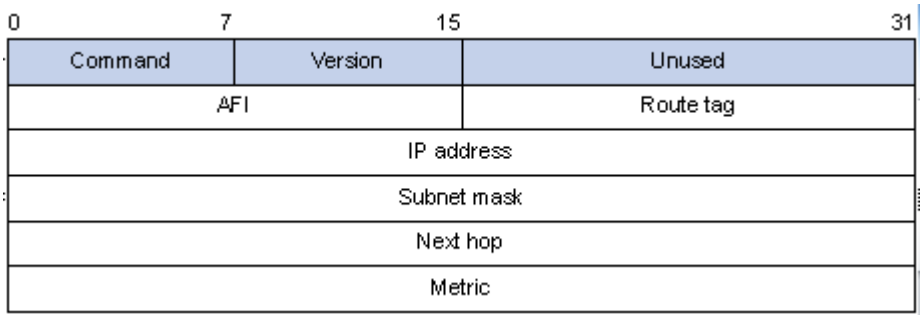
9.2 SPOLEČNÉ A ROZDÍLNÉ VLASTNOSTI RIP A RIPV2

V následující tabulce jsou shrnuty společné vlastnosti protokolů RIP a RIPv2. Tabulka zároveň uvádí i nové vlastnosti protokolu RIPv2.

Tabulka 10 RIPv1 a RIPv2 přehled vlastností

Společné vlastnosti	**Nové vlastnosti v RIPv2**
vektor vzdálenosti	podpora VLSM
maximálně 15 přeskoků	authentifikace update
administrativní vzdálenost 120	multicastové aktualizace
Časovače	vypnutí sumarizace

9.3 HLAVIČKA PROTOKOLU RIPV2



Obrázek 44 RIPv2 formát hlavičky

U hlavičky RIPv2 si lze všimnout, že není nikterak rozdílná od RIPv1.

Základní pole známe z předchozí verze protokolu RIP. Rozšiřme si nyní však naše znalosti o nově přidaná pole. První nové pole je **route tag**, jehož hlavním úkolem je zajištění rozlišování, mezi interně a externě zjištěnými cestami. Hlavním rozdílem od zprávy RIPv1 je existence pole **subnet mask**, které slouží pro uvedení masky podsítě. V protokolu RIPv1 nebylo toto pole potřeba, jelikož docházelo k automatické sumarizaci cest do jejich plno třídních podob. Posledním rozdílným polem oproti předchozí verzi je pole **next hop**, které jak již označení napovídá, obsahuje IP adresu dalšího přeskoku, na který mají být pakety přeposlány z aktuálního směrovače.

9.4 KONFIGURACE RIPV2

Jediným rozdílem oproti protokolu RIP je nutnost specifikace verze protokolu RIP, jenž má být na směrovači spuštěna. Pokud tuto specifikaci neprovedeme, směrovač implicitně zapne proces protokolu RIPv1.

Specifikaci verze používaného protokolu provedeme následovně.

```
R# conf term
R (config) # router rip
R (config-router)# version 2
```

Podrobná konfigurace na námi používané ukázkové topologii, je přiložena jako příloha F.

[1][10][11][13]

10 RIPNG

10.1 DŮVOD VZNIKU

Dříve, respektive v době, kdy byly protokoly RIP i RIPv2 navrhovány, a stávaly se standarty, bylo zcela běžné využívat směrovaný protokol IPv4. Ovšem s rozmachem počítačových sítí vzrůstala potřeba rozšiřování adresového prostoru. Možná jste jako studenti informatiky zaznamenali v médiích, že v září roku 2012 došlo k vyčerpání IPv4 adres. Samozřejmě to neznamenalo konec sítí, jelikož byl již prováděn postupný přechod na IPv6. Ovšem s přechodem na nové adresování musely nastat určité úpravy a vznik nových směrovacích protokolů. Stejný případ nastal i s protokolem RIP respektive RIPv2. Jakožto protokoly koncipované zejména na využití s protokolem IPv4 nebyly schopné vyrovnat se nově rozšiřujícímu způsobu adresování. Proto byl od roku 1997 vyvíjen nový protokol RIPng.

Z výše uvedeného vyplývá, že protokol RIPng je koncipován pro protokol IPv6. Jinak se chová zcela obdobně jako jeho IPv4 verze, tedy například lze vyjmenovat vysílání updatů každých 30 sekund či délka časovače pro vypršení limitu nastavená na 180 sekund.

10.2 KONFIGURACE RIPNG

V oblasti konfigurace přináší výhody oproti svým dvěma předchůdcům. Již nemusíme používat příkaz „network x.x.x.x“ pro stanovení všech přímo připojených sítí. Nejprve je nutné zahájit na směrovači proces RIPng. Tuto aktivaci provedeme příkazem:

```
R (config)# ipv6 router rip <název>
```

Poté nám již pouze stačí na všech rozhraních, jejichž přímo připojené sítě mají spadat do redistribuce cest, v konfiguračním režimu rozhraní zadat příkaz:

```
R (config-if)#ipv6 rip <název> enable
```

Kompletní řešení na ukázkové topologii včetně příkazů pro konfiguraci je uvedeno v příloze E.

[10][11][13][14]

11 PROTOKOL EIGRP

11.1 OBECNĚ O EIGRP

11.1.1 Předchůdce EIGRP

Předchůdcem tohoto protokolu typu vektoru vzdálenosti je protokol IGRP, tedy anglickým názvem **I**nterior **G**ateway **R**outing **P**rotocol. K rozhodnutí o vyvinutí nového protokolu v roce 1980 vedlo zejména omezení protokolu RIP na 15 přeskoků, či výpočet metriky založený pouze na vzdálenosti. Je důležité mít na paměti, že protokol IGRP respektive i EIGRP jsou **proprietární.** Neznamená to však nic jiného než fakt, že protokoly jsou patentován a tudíž mohou být využívány pouze na zařízeních společnosti CISCO.

11.1.2 Vlastnosti EIGRP

Tento protokol, plným názvem **E**nhanced **I**nterior **G**ateway **R**outing **P**rotocol, je inovací, nikoliv však nástupcem původního protokolu IGRP. Možná si říkáte, proč nebyl tedy pojmenován jako IGRPv2. Důvod je takový, že protokol EIGRP je po celou dobu zamýšlen jako úplně nový směrovací protokol, který přináší odstranění nedostatků protokolu IGRP. Pokud by společnost CISCO přistoupila k cestě změny stávajícího protokolu IGRP místo vytváření protokolu nového, mohly by nové modifikace znamenat narušení provozu s nutným přechodem na novou verzi. Takto si zákazníci mohou sami vybrat, zda zůstanou u původního protokolu IGRP, či převedou síť na EIGRP.

Některé vlastnosti zůstaly v obou protokolech stejné. Oproti svému předchůdci však EIGRP přinesl nové funkce a zejména podporu beztřídní adresace, tedy možnost využívání VLSM a CIDR. Nové technologie, které tento protokol přinesl, byly zavedeny zejména za účelem zkrácení doby konvergence sítě a zlepšení její stability. Hlavní technologií, která tomuto vylepšení slouží je algoritmus DUAL. Tomuto algoritmu bude věnována jedna z následujících částí této kapitoly.

I přes změny, které EIGRP přinesl, byla zachována vzájemná kompatibilitas protokolem IGRP. V jedné síti tak dokáží oba protokoly vzájemně vyměňovat své směrovací informace. Tato výměna je umožněna díky doplňování délky metriky, kterou mají oba protokoly v rozdílném formátu. Dále protokol EIGRP rozlišuje cesty na interní a externí, což protokol IGRP neumožňuje, nicméně pokud přijde směrovači se spuštěným procesem IGRP

aktualizace od EIGRP, zaznamená si směrovač všechny cesty jako klasické a nerozlišuje, zda byly interní nebo externí. Aby byla však možná tato kompatibilita obou protokolů, je nutné, aby spadaly pod jeden autonomní systém.

Díky podobným, ba i stejným vlastnostem obou protokolů, nebyl přechod mezi těmito protokoly nikterak náročný a nevyžadoval odstavení provozu. Nejprve byl protokol EIGRP využit například na strategické úseky, jako například páteřní vedení a následně byl rozšiřován do jednotlivých oblastí sítě. Změna výpočtu metriky je minimální a navíc spolu i nadále dokázaly komunikovat jak směrovače využívající IGRP, tak i ty, které již přešly na novější verzi, tedy na protokol EIGRP.

Protokol EIGRP rozlišuje dva druhy cest uvedených ve směrovací tabulce. Toto členění vzniklo v závislosti na zpětné kompatibilitě s protokolem IGRP. EIGRP dokáže informace získané ze směrovačů používajících protokol IGRP analyzovat a použít. Aby odlišil cesty zjištěné od směrovačů se starší verzí, označuje tyto cesty jako externí, zatímco cesty zjištěné od směrovače používajícího taktéž protokol EIGRP označuje jako interní.

11.1.3 Společné a rozdílné vlastností protokolů IGRP a EIGRP

Tabulka 11 Společné vlastnosti IGRP a EIGRP

Společné	Přínos EIGRP
protokoly vektoru vzdálenosti	algoritmus DUAL
Proprietárnost	delší pole pro výpočet metriky
využití složené metriky	menší spotřeba šířky pásma
stejné váhy položek při výpočtu metriky	podpora beztřídní adresace sítě
	protokol RTP

11.2 METRIKA V EIGRP

Oproti skupině protokolů RIP využívají protokol EIGRP i jeho předchůdce, protokol IGRP, **kompozitní metriku**. Kompozitní v tomto významu znamená, že se neřídí při posuzování nejlepší cesty pouze jedním údajem, avšak zkoumá několik aspektů současně. Metrice, jež se skládá z několika částí, říkáme **složená metrika**. Mezi faktory, které jsou zohledňovány při určování cest, patří:

- **Počet přeskoků,** jenž stejně tak, jak tomu bylo u předchozích protokolů, umožňuje protokolu EIGRP možnost měření délky cesty pomocí počtu přeskoků. Je nutné však podotknout, že EIGRP nepřevzal od protokolů skupiny RIP omezení na maximální vzdálenost 15 přeskoků. U EIGRP je implicitní hodnota nejdelší možné cesty nastavena na 100, avšak může být administrátorem změněna až na 224 přeskoků do cílové sítě.

- **Šířka pásma**, která vyjadřuje rychlost přenosu dat na jednotlivých rozhraních. Není žádným překvapením, že tuto rychlost udává použitý přenosový prostředek. Implicitní hodnota je nastavena na 1,544 Mb/s. což odpovídá připojení pomocí linky typu T1. Nejedná se však o nejnižší možnou hodnotu. Rychlosti jednotlivých linek spadají do rozmezí 1200 b/s až 10 Gb/s. Mějme na paměti, že pokud je implicitní hodnota nastavena na 1,544 Mb/s, avšak na daném portu bude připojena linka o rychlosti například 1200 b/s, nebude se port nikdy snažit využít implicitní možnou šířku pásma. Vždy se řídí aktuálně připojenou linkou. Pokud však necháme výchozích 1,544 Mb/s, nemůže být šířka pásma zapojena do výpočtu metriky, respektive by na její výpočet neměla žádný význam, jelikož by byla všude stejná. Z tohoto důvodu je v případě potřeby využívání šířky pásma jako jedné z parametrů pro výpočet metriky, nutné nastavit reálné rychlosti připojených linek na jednotlivých rozhraních. Nejnižší z hodnot nastavených pro jednotlivá rozhraní je volena jako limitní velikost.

- **Spolehlivost** je parametr vyjadřující pravděpodobnost správného doručení paketu. Tedy ze základů pravděpodobnosti dochází k tomuto výpočtu

Spolehlivost linky = počet paketů doručených bez chyb / počet všech odeslaných paketů

Vypočtená metrika je následně převedena do rozmezí 0 – 255, kde 0 značí naprosto nespolehlivou linku a naopak linka se spolehlivostí 255 nabývá stoprocentní spolehlivostí.

- **Zpoždění**, které je určováno automaticky protokolem EIGRP, jenž obsahuje informace o průměrných dobách zpoždění pro dané typy linek. Není tedy bohužel implementován žádný pokročilý algoritmus, který by dokázal určit aktuální míru zpoždění v síti. A co vlastně pojem zpoždění v síti znamená? Vezměte si kupříkladu cestu z Prahy do Pardubic. Na cestě je mnoho silnic či dálnic. I když předem přesně víte, po které cestě máte jet, nebudete v cíli hned. Na každém úseku cesty pojedete po určitý čas. Stejný způsob dopravy má i paket a tedy zpožděním udáváme jak dlouho paket „pojede“ po všech linkách, než dorazí do cíle. Dovedete si jistě představit, že čím delší cesta, tím bude zpoždění vyšší, stejně jako tomu je při cestách automobilem.

- **Zatížení** je položka metriky snažící se o přiblížení provozu v síti realitě. Tedy zatímco zpoždění bylo automaticky ohodnoceno průměrnou hodnotou pro daný typ linky a nebylo možné jej upravovat, zatížení představuje aktuální míru síťového provozu na dané cestě. Díky tomu se protokol EIGRP může rozhodnout, že nepošle paket skrze velmi zatížené místo, jelikož platí, že čím více zatížený úsek, tím delší zpoždění. I v reálném světě existují moderní navigace, které pomocí přijímání signálů dokáží zobrazovat kolony a řidič se jim tak může vyhnout. Zatížení si lze představit jako takovouto navigaci. Hodnota této metriky může být měněna, avšak je to velmi nebezpečné, zejména při provozu sítě. Odklánění paketů se může totiž projevit, až při vyšším zatížení linky a tudíž si nelze ihned ověřit dopady našich úprav. I z tohoto důvodu není ve výchozím nastavení protokolu EIGRP zahrnováno zatížení do výpočtu metriky.

- **MTU**, celým názvem maximal transfer unit, nevstupuje do samotného výpočtu metriky, avšak je mezi uzly informace o této hodnotě vyměňována. Udává největší možnou velikost datagramu, kterou dokáže směrovač, na němž běží protokol EIGRP přijmout a zpracovat. Tento údaj je důležitý pro vzájemnou komunikaci mezi uzly,

protože samotné doručení paketu je bezvýznamné, pokud ho protější směrovač neumí zpracovat. Pokud jsou data, jenž mají být přenesena větší než je MTU, jsou jednoduše rozdělena do několika datagramů a odeslána. Obecně platí, čím větší je dovolená velikost MTU, tím delší je doba zpracování na jednotlivých směrovačích.

Nyní si nejspíše říkáte, zda všechny mají stejnou váhu, i když ve skutečnosti jsou pro nás některé parametry důležitější. V běžném použití si administrátor může upravovat hodnoty důležitosti jednotlivých položek, podle požadavků na síť. Tudíž může být síť zaměřena na přenos s co nejnižším zpožděním, či naopak na vysokou šířku pásma. V tomto ohledu je EIGRP pro administrátora flexibilní, avšak se doporučuje, aby změny byly prováděny velmi opatrně, jelikož chybně nastavená hodnota priority může znamenat například rapidní zpomalení celé sítě.

Implicitně je tento protokol koncipován tak, aby nejlepší cesty určoval na základě šířky pásma a zpoždění.

Celý vzorec pro výpočet složené metriky vypadá následovně:

*Metrika = [K1 * šířka pásma + (K2 * šířka pásma) / (256 - zatížení) + K3*zpoždění] * [K5/(spolehlivost + K4)]*

kde jednotlivé konstanty K1 až K5 znamenají váhu při výpočtu metriky pro dílčí položky. Jelikož je implicitně nastaven výpočet metriky pomocí šířky pásma a zpoždění, je jasné, že konstanty K1 a K3 budou rovny 1, zatímco ostatní rovny 0.

Nedojde-li ke změně váhových konstant pro jednotlivé položky, lze vzorec pro výpočet zjednodušit takto:

Metrika = zpoždění + šířka pásma

11.3 SMĚROVÁNÍ PO VÍCE CESTÁCH A VYROVNÁVÁNÍ ZÁTĚŽE

Na rozdíl od protokolů skupiny RIP podporuje EIGRP novou funkci nazývanou **vícecestné směrování**. Směrování po více cestách neznamená však nic jiného, než možnost využití jiné cesty v případě výpadku cesty aktuální, po níž paket směřuje. Zmíněný protokol RIP si dokázal do každé cílové sítě udržovat pouze jeden záznam o cestě. Pokud tedy došlo k výpadku tohoto jediného spoje, nebylo možné do koncové sítě doručit paket. Oproti tomu

EIGRP dokáže evidovat až 6 různých cest do koncové sítě. Evidence více cest neslouží pouze k udržení spojení v případě výpadku jedné linky, ale zároveň mohou být tyto cesty využity k vyrovnávání zátěže.

Vyrovnávání zátěže dovoluje využívat pro doručení paketu více cest najednou. Tímto vyvažováním tak opět dochází ke zrychlování komunikace v síti. Vyrovnávání zátěže dělíme v souvislosti s náklady jednotlivých cest následovně:

- vyrovnávání zátěže mezi cestami o stejných nákladech,
- vyrovnávání zátěže mezi cestami o rozdílných nákladech.

První uvedený způsob nijak neodlišuje jednotlivé cesty, a tudíž je provoz rozdělen mezi všechny rovnoměrně. Ovšem i přesto nedochází ke zcela nahodilému rozposílání po jednotlivých cestách, Směrovač může rozkládat zátěž **podle paketů**. Při tomto způsobu však jednotlivé části dat, jež jsou rozděleny do více segmentů, mohou přijít v nesprávném pořadí, jelikož byly odesílány přes různá rozhraní.

Druhým způsobem je odesílání **podle cílů**. Odesílání podle cílů neznamená nic jiného než zajištění odeslání všech souvisejících paketů skrze jeden port. Díky omezení dělení jednoho toku dat do více cest je zaručeno, že do cílového zařízení dorazí pakety ve správném pořadí.

Druhý způsob je svým principem intuitivnější. Směrovač má zde možnost určit takzvanou **primární cestu**. Tato cesta má zpravidla nejnižší náklady, a pokud je v provozu, je využívána pro odesílání paketů. Ostatní **cesty** jsou označeny za **alternativní** a jsou využity pouze v případě výpadku cesty primární.

11.4 KOMPONENTY EIGRP

Na rozdíl od ostatních směrovacích protokolů se neeviduje u EIGRP pouze směrovací tabulka. Ke svým výpočtům, tedy ke správné funkci algoritmu DUAL, potřebuje tento protokol následující datové struktury ukládané v operační paměti směrovače. Dříve než se však seznámíme s těmito datovými strukturami, je nutné uvést několik základních pojmů, se kterými se v rámci této kapitoly budeme setkávat.

- **Následník** (successor) je cesta s nejnižší hodnotou metriky vedoucí k určitému cíli. Platí, že pro jeden cíl můžeme mít vícero následníků.

- **Přípustný následník** (feasible successor) označován také jako cesta záložní. Pro tuto cestu platí, že má nižší hodnotu oznamované vzdálenosti než přípustná vzdálenost aktuální cesty.
- **Udávaná vzdálenost** (reported distance) může být také označována jako reklamovaná vzdálenost (advertised distance). Tato hodnota označuje nejnižší hodnotu metriky, jež je nasčítána po cestě skrze následníky sousedního směrovače k cíli. Tato vzdálenost je oznamována sousedním směrovačem.
- **Přípustná vzdálenost** (feasible distance) označuje vzdálenost, která je součtem udávané vzdálenosti od souseda spolu s cenou, kterou my musíme vynaložit na cestu k tomuto sousedovi.
- **Přípustná podmínka** (feasible condition) označuje podmínku, která je testována za účelem eliminace cest obsahujících smyčky. Tato podmínka zní **RD<FD**, tedy porovnáváme, zda je udávaná vzdálenost souseda menší než vzdálenost od souseda spolu se vzdáleností tohoto souseda od aktuálního směrovače. Stručněji lze říci, že soused musí ležet na cestě blíže k cíli než aktuální směrovač.

Mezi datové struktury, které jsou ukládány pro efektivní funkčnost protokolu EIGRP, řadíme tyto tabulky:

- **Směrovací tabulka** (routing table) stejně tak jako každý jiný směrovací protokol, musí i EIGRP mít prostor pro evidování nejkratších cest do cílových sítí. Tato tabulka je naprosto obdobná jako u jiných směrovacích protokolů. Jelikož jsou v ní obsaženy pouze ty záznamy, jež jsou ohodnoceny nejnižší hodnotou metriky, obsahuje tedy vždy následníka k záznamu z tabulky topologie, jemuž algoritmus DUAL vypočítal nejnižší náklady. Ke každému cíli může evidovat až šest různých cest. Pokud nastane v topologii jakákoliv změna ovlivňující směrovací tabulku směrovače, ihned o tom

informuje ostatní směrovače v síti. Další důležitou poznámkou je evidence směrovací tabulky pro každý z použitých směrovaných protokolů. Jako příklad lze uvést, že v síti nám poběží současně AppleTalk a protokol IP, potom pro každý z těchto protokolů bude směrovač evidovat vlastní směrovací tabulku.

- **Tabulka topologie** (topology table) obsahuje informace o veškerých cestách do cílových sítí. Mezi těmito informacemi je evidována hodnota metriky. Záznamy do této tabulky jsou získány pomocí informací od sousedních směrovačů, které jsou uvedeny v tabulce sousedů. Na základě této tabulky je určována primární a záložní cesta, tedy následník a přípustný následník. Jednotlivé záznamy obsahují vždy informace o udávané vzdálenosti a zároveň i informace o přípustné vzdálenosti. Další hodnotou, jež je evidována o každé z cest je její stav, tento stav může být aktivní a pasivní. Zcela jistě by každého napadlo, že aktivní stav bude udávat, zda se daná cesta může použít pro směrování, zatímco u cest pasivních by tomu bylo naopak. Nicméně zde stav aktivní a pasivní neznamená možnost použití. V tomto případě označení stavu za aktivní udává, že je daná cesta aktuálně přepočítávána a díky tomu není aktuálně k dispozici a nelze jí využít pro odesílání paketu. Na druhou stranu, pokud je záznam označen za pasivní, neprobíhá s danou cestou žádný výpočet a je tudíž považována za stabilní. Takováto cesta může být použita pro posílání paketu.
- **Tabulka sousedů** (neighbor table) obsahuje informace o všech přímo připojených směrovačích. Je důležité mít na paměti, že těchto tabulek může existovat vícero, jelikož jsou závislé na použitém směrovaném, nikoli směrovacím, protokolu. O každém sousedním směrovači jsou evidovány informace o rozhraní a adrese. Jelikož chceme evidovat pouze ty sousední směrovače, jež jsou aktivní, a tudíž mohou být použity v procesu směrování, jsou rozhraní všech sousedních směrovačů testovány pomocí hello paketů. Co hello paket znamená, bude zmíněno v následující části textu.

Tato tabulka je nejdůležitější z datových struktur protokolu EIGRP, jelikož sleduje veškeré vztahy mezi směrovači. Tabulka sousedů je důležitá pro veškeré výpočty spojené se směrovacími aktualizacemi. Další informací, která je evidována, je číslo posledního přijatého paketu. Toto číslo je mimo jiné použito také pro spolehlivý přenos, jelikož díky němu lze provádět zpětnou vazbu.

Ve spojení s tabulkou sousedů jsme se zmínili o existenci hello paketů. Protokol EIGRP nevyužívá ke své činnosti pouze tento typ paketů. Pakety, které jsou použity v tomto protokolu, shrnuje následující přehled. Je nutné mít však na paměti, že EIGRP je proprietárním protokolem firmy CISCO a tudíž je patentovaný. Nelze tedy dohledat přesnou strukturu paketů, avšak dokážeme o těchto paketech určit jejich funkčnost a také jejich použití.

- **Hello paket** bývá označován také jako kontaktní paket. Jak již název napovídá, slouží ke kontaktování sousedních směrovačů za účelem ověřování stálého spojení. Pokud dojde v síti k výpadku, dochází tak ke změně topologie. Pomocí hello paketů tak směrovač dokáže rozpoznat, zda některý z jeho sousedních směrovačů není v provozu. Kontaktní pakety jsou však také využívány pro rozpoznávání nových sousedů, či naopak zjišťování, zda neaktivní soused není opět v provozu. Asi se nyní ptáte, jak směrovač pozná okamžik, kdy odeslat tento kontaktní paket. Hello paket je odesílán v pevně daném intervalu označovaném jako **hello interval**. Ovšem jelikož je jakékoliv odesílání paketů v rámci sítě zátěží pro provoz v síti, je nutné, aby odesílání probíhalo v závislosti na dostupné šířce pásma u spoje. U linek s malou šířkou pásma, jmenujme například okruhy typu frame relay nebo X.25, dochází k odesílání hello paketů každých 60 sekund, zatímco u linek poskytujících vyšší šířku pásma, kupříkladu linky typu T1, může k tomuto kontaktování docházet i každých 5 sekund. Každý směrovač si do tabulky sousedů eviduje mimo jiné i čas, ve kterém od svého sousedního uzlu

obdržel naposledy odpověď. Tato odpověď není zjišťována pouze pomocí hello paketů, ale jakoukoliv formou komunikace, kdy protější uzel odpoví. Na druhou stranu pokud neobdrží směrovač odpověď od svého sousedního uzlu, neodstraní jej hned z tabulky sousedů. Stejně tak jako u jiných směrovacích protokolů, spustí se časovač udržení cesty, který udržuje záznam do vypršení časového intervalu. Délka tohoto intervalu je zpravidla taková, aby mohlo dojít ke třem pokusům o navázání spojení s neaktivním směrovačem, tedy 180 či 15 sekund. V tomto intervalu jsou prováděny opětovné pokusy o navázání spojení s uzlem a teprve po vypršení časového limitu dojde k odstranění záznamu a zároveň k oznámení změny stavovému algoritmu DUAL, který okamžitě zahájí proces konvergence v síti.

- **Aktualizační pakety** mají dva významy. Prvním důvodem k odesílání aktualizačních paketů je potřeba oznámení podoby topologie nově přidanému směrovači. Když je do sítě přidán nový směrovač, nemá vůbec žádné informace o podobě topologie. V tento okamžik začnou okolní směrovače vysílat tomuto nově přidanému směrovači své zjištěné informace o topologii sítě. Přidaný směrovač tyto údaje analyzuje a použije je k sestavení vlastní představy o podobě topologie. S tímto odesíláním se nesetkáváme až tak často jako s typem druhým. Druhým významem pro odesílání aktualizačních paketů jsou veškeré změny v síťové topologii. Změnou v topologii rozumíme výpadek některého směrovače či změnu v ceně některé z cest. Na rozdíl od první skupiny aktualizačních paketů, jež jsou odesílány pouze jednomu konkrétnímu směrovači v síti, jsou pakety druhé skupiny odesílány všem dostupným sousedním zařízením. Dalším z vylepšení protokolu EIGRP je forma odesílání aktualizací směrovací tabulky. Zatímco předešlé směrovací protokoly odesílaly vždy úplný obsah své směrovací tabulky, směrovač využívající protokol EIGRP odesílá pouze záznamy ze směrovací tabulky, u nichž došlo od posledního odeslání ke změně. Díky této formě

odesílání aktualizačních informací nemohou být ovšem odesílány aktualizace v pravidelných intervalech. Tyto aktualizace reagují okamžitě na změny v síťové topologii a jsou vysílány bezprostředně po zjištění těchto změn.

- **Dotazovací pakety** jsou využívány v případě, kdy směrovač potřebuje zjistit informace od okolních směrovačů. Nejedná se přitom o aktualizační informace, ale o konkrétní dotazy. Jako příklad, při kterém dojde k využití dotazovacích paketů, můžeme uvést situaci, kdy se směrovač, označme jej například A, snaží vyhledat alternativní cestu do cílové sítě. Poznamenejme, že v této situaci směrovač A nemá žádné vlastní informace o cestách do cílové sítě. V takovém případě odešle směrovač A dotazovací paket obsahující žádost o uvedení všech alternativních cest do konkrétní sítě, o nichž příjemce paketu ví.
- **Pakety odpovědí** jsou reakcí na pakety dotazovací. Směrovače v nich odesílají informace, které byly vyžádány pomocí dotazovacích paketů. Pokud se vrátíme k předchozímu příkladu s vyhledáváním alternativních cest ze směrovače A, byly by obsahem odpovědi od jednotlivých příjemců dotazovacího paketu výpis všech následníků. V současnou chvíli je tedy zcela zřejmé, že dotazovací pakety a pakety odpovědí musí koexistovat, jelikož pokud by jeden z nich nebylo možné odesílat, nemohlo by docházet k této komunikaci.
- **Potvrzovací pakety** jsou používány pro potvrzování jakéhokoliv přijatého paketu. Důvodem pro toto potvrzování je využití přenosového protokolu RTP. Více o tomto protokolu bude probráno v následující části textu. Toto potvrzení slouží směrovači, jenž paket odeslal jako potvrzení, že tento paket dorazil do cíle. Ve své podstatě se potvrzovací paket nijak neliší od kontaktního paketu hello. Jediný rozdíl je v datové části paketu. Na rozdíl od kontaktního paketu neobsahuje paket potvrzovací žádná přenášená data. Tedy datová část je u potvrzovacího paketu prázdná. Dalším rozdílem

oproti kontaktnímu paketu jsou příjemci. Kontaktní paket je odesílán všesměrově, na rozdíl od potvrzovacího paketu, který je odesílán pouze jednosměrně na směrovač, ze kterého byl původní paket přijat.

11.5 PROTOKOL RTP

Zkratka RTP představuje spojení **R**eliable **T**ransport **P**rotocol. Tento spolehlivý přenosový protokol je dalším z vylepšení, které protokol EIGRP přinesl. Protokol RTP byl do EIGRP zaveden, aby bylo umožněn spolehlivý přenos jednotlivých typů paketů. Protokol RTP je protokolem transportní vrstvy a není otevřeným standardem. Je jistou obdobou protokolů TCP a UDP používaných pro směrovaný protokol IP. Ačkoliv RTP využívá ke své funkci vlastnosti protokolů TCP a UDP, nevyužili konstruktéři pouze jednoho z již existujících protokolů, ale pustili se do tvorby protokolu nového, tak aby nebylo nutné využívání protokolu IP.

Díky tvorbě nového protokolu je tak možné využívat transportní protokol RTP nezávislé na používaném směrovaném protokolu. Ovšem ze svých předloh si RTP vzal zejména schopnost spolehlivého i nespolehlivého přenosu. Dokonce je schopen využívat spolehlivý i nespolehlivý přenos současně.

Vždy je nutné určit, pro který typ paketu je vhodný spolehlivý a pro který nespolehlivý typ přenosu. Například pro kontaktní pakety hello by bylo vcelku zbytečným zatížením využívání spolehlivého přenosu a tudíž je pro jejich posílání použit nespolehlivý přenos.

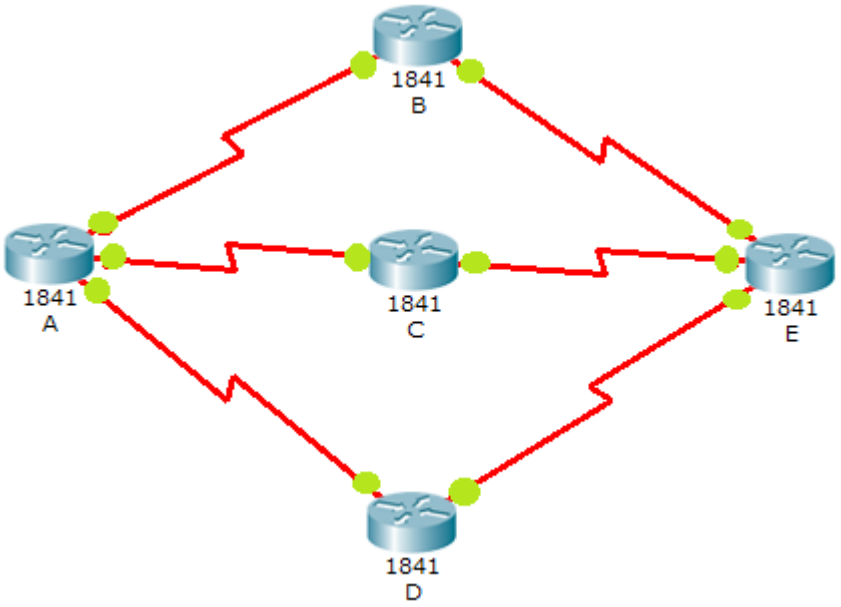
11.6 ALGORITMUS DUAL

Mimo spolehlivého transportního protokolu přinesl směrovací protokol EIGRP i velmi výkonný algoritmus DUAL, plným názvem Difussing Update Algorithm, který představuje

stroj pro výpočet směrovacích cest. Zcela jistě nikoho nepřekvapí, že tento algoritmus má implementovány veškeré funkce a metody, jež vedou k výpočtu i porovnávání jednotlivých cest v síti využívající protokol EIGRP. Mimo výpočtů konkrétních cest jsou také zkoumáni přípustní následníci směrovače, od kterého byla cesta zjištěna.

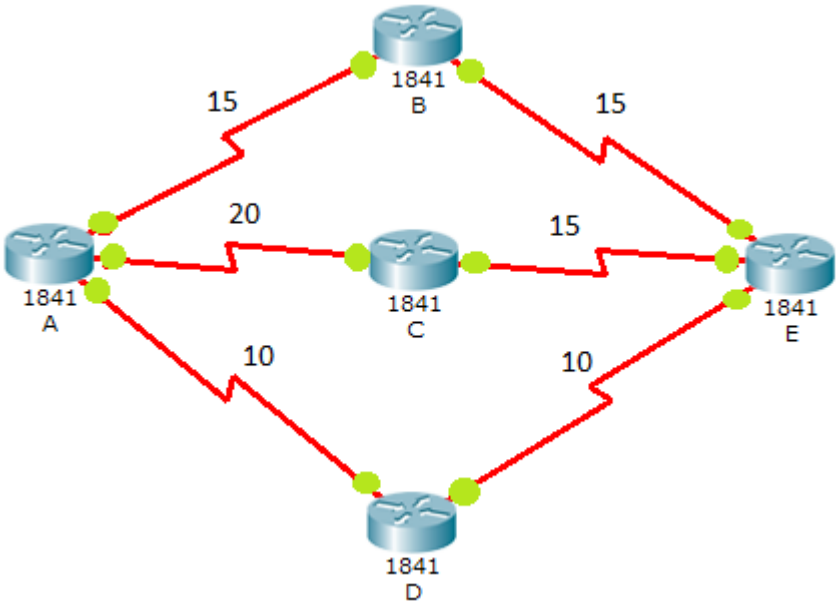
Základním požadavkem na cestu zjištěnou od sousedního směrovače je absence směrovacích smyček. Dalším požadavkem je poté co nejnižší hodnota metriky. Pokud jsou tyto dva požadavky úspěšně splněny, je zjištěná cesta zapsána do směrovací tabulky a dále používána pro směrování paketů.

Samotný průběh vyhodnocování cest demonstrujme na následujícím příkladu. Mějme zjednodušenou topologii, ve které je spuštěn algoritmus DUAL na směrovači A, přičemž je hledána nejlepší cesta ke směrovači E.



Obrázek 45 DUAL výchozí topologie

Každá z hran mezi směrovači je ohodnocena cenou linky.



Obrázek 46 DUAL ohodnocení

Při průběhu konvergence směrovače B, C a D oznamují svou vzdálenost, se kterou mohou být dosaženy ze směrovače A. Tedy doslova směrovač B vyšle následující zprávu:

„Já jsem od tebe vzdálen 15 jednotek.“

Směrovač C odesílá směrovači A informaci:

„Já jsem od tebe vzdálen 20 jednotek.“

A nakonec vysílá informaci o své vzdálenosti i směrovač D.

„Ke mně se dostaneš se vzdáleností 10 jednotek.“

Poté, co směrovač obdrží tyto informace, vyhodnocuje také přípustnou vzdálenost, neboli feasible distance. Přehled vzdáleností z vrcholu A do vrcholu E zobrazuje následující tabulka.

Tabulka 12 DUAL přehled vzdáleností z vrcholu A vzhledem k vrcholu E

Odkud	Průchod	Hodnota
A	B	30
	C	35
	D	20

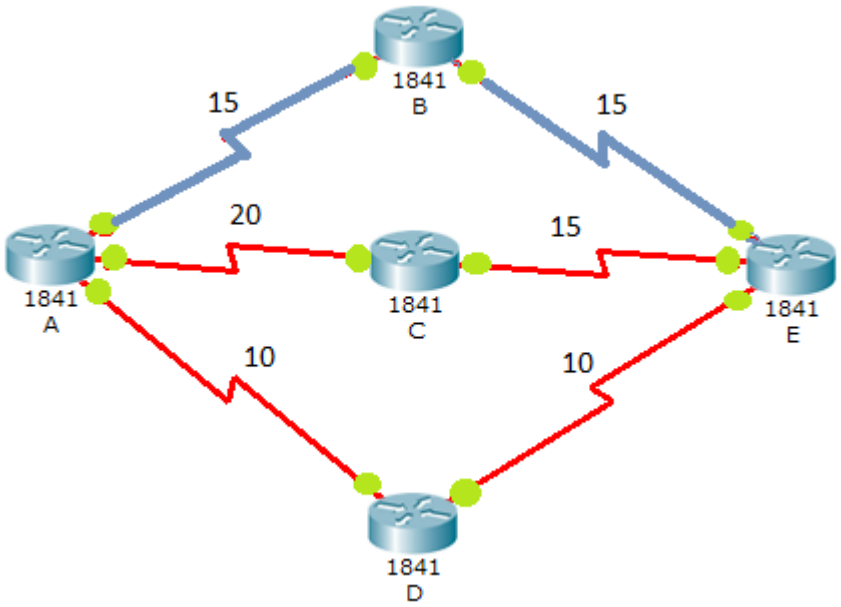
Na základě těchto informací je vybrán následník směrovače A pro pakety určené směrovači E. Tímto následníkem se stává ten směrovač, jehož hodnota k cíli spolu s ohodnocením spoje je co nejmenší. V našem modelovém příkladu byl tudíž jako následník zvolen směrovač D.

V tuto chvíli již máme určeného následníka pro cestu ze směrovače A do směrovače E, ale stále nemáme informaci o přípustném následníkovi. Tento přípustný následník je zkoumán přípustnou podmínkou, která jak již víme, zní **RD < FD**, v našem případě by tedy byly testovány tyto podmínky:

- Směrovač B: „Je 15 menší než 30?“ Ano, tedy tato cesta neobsahuje smyčky.
- Směrovač C: „Je 20 menší než 35?“ Ano, tedy i tato cesta neobsahuje smyčky.
- Směrovač D: „Je 10 menší než 20?“ Ano, tedy i poslední cesta neobsahuje smyčky.

Jelikož směrovač D byl zvolen jako následník, stane se přípustným následníkem směrovače A směrovač B, jelikož hodnota jeho přípustné vzdálenosti je další nejmenší, pomineme-li hodnotu odpovídající následníkovi.

Jako cesta mezi směrovačem A a směrovačem E by tudíž byla zapsána následující cesta zobrazena na obrázku.



Obrázek 47 DUAL výsledná zvolená cesta

Celkové ohodnocení této cesty je 30 jednotek. Tedy tento záznam bude zapsán do směrovací tabulky.

11.7 METODY ZAJIŠTĚNÍ STABILITY A KONVERGENCE

Tvůrci protokolu EIGRP se samozřejmě snažili také o co nejlepší **stabilitu při konvergenci** sítě, tu zajišťují základní čtyři funkce:

- odstavení cest,
- rozdělení horizontu,
- pozměnění zpětné aktualizace,
- blesková aktualizace.

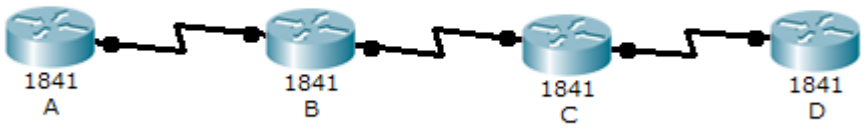
Všechny tyto funkce protokolu EIGRP jsou implementovány s cílem, co nejvíce zkrátit dobu konvergence a zároveň zvýšit stabilitu sítě.

- **Odstavení cest** představuje časové období, které je měřeno od přijetí první informace o nedostupnosti cílové sítě. Kdyby v témže okamžiku byl vyslán zrovna update,

obsahoval by i neplatnou cestu, která by byla takto šířena dále v síti. Aby se předcházelo šíření této mylné informace, je zapnut **časovač**, který zdrží odeslání updatu směrovací tabulky po takovou dobu, aby mohlo dojít k rozšíření nové vlny aktualizací a neplatná cesta by byla tudíž vyřazena. Z předchozího vyplývá, že perioda zdržení odeslání aktualizace musí mít hodnotu vyšší než je čas na konvergenci celé sítě. Stejný mechanismus je využíván i u protokolu RIP.

- **Rozdělení horizontu** je další metoda, jež byla využívána i v protokolu RIP. Celý mechanismus rozdělení horizontu je založen na myšlence, že informace o dostupných sítích zjištěné z jednoho směrovače nemá smysl odesílat v aktualizaci zpět na toto zařízení. V aktualizaci jsou odeslány pouze ty informace, které nebyly obsaženy v původní zprávě. Díky metodě rozdělení horizontu dochází k předcházení menších **smyček**. Předveďme si tuto metodu na příkladu.

  Mějme 4 směrovače, jejichž rozložení zobrazuje obrázek:



Obrázek 48 EIGRP rozdělení horizontu

  Zamysleme se. Je nutné informovat směrovač B o existenci sítí, které byly uvedeny v aktualizaci přijaté od tohoto směrovače? Budeme tedy nutně informovat směrovač B z pohledu směrovače C o existenci jisté sítě A, i přestože síť A byla oznámena v aktualizaci přijaté od směrovače B? Ne, docházelo by zcela zbytečně k šíření duplicitních informací. Split-horizon je založen na přeposílání pouze takových sítí, o kterých nás daný směrovač sám neinformoval. Princip je takový, že přijatá směrová tabulka je porovnána se směrovou tabulkou příjímacího směrovače. V tomto okamžiku dojde k vyhodnocení obsahu obou tabulek a v aktualizaci, jsou odeslány pouze ty, jenž v přijaté aktualizaci chyběly.

- **Pozměnění zpětné aktualizace** slouží k označení neplatných cest. Princip je postaven na předpokladu, že k nárůstu směrovací metriky dochází při výskytu smyček. Jakmile

obdrží směrovač aktualizaci, porovná ji se svoji směrovací tabulkou, a pokud nalezne stejný záznam, avšak s vyšší hodnotou metriky, usoudí, že cesta je neplatná. Samozřejmě, že jeho předpoklad o neplatnosti cesty nemusí být vždy pravdivý. Neodstraní tedy cestu ze směrovací tabulky ihned, ale označí ji za neplatnou a informuje ostatní směrovače. Teprve na základě příchozích informací z ostatních směrovačů je rozhodnuto, zda se doopravdy jedná o cestu neplatnou. Metoda pozměnění zpětných aktualizací je prevencí proti vzniku smyček v rozsáhlých sítích, kde je rozdělení horizontu již nepoužitelné.

- **Blesková aktualizace** je metoda, sloužící k výraznému zrychlení doby konvergence sítě. Směrovač nemusí čekat na vypršení aktualizačního časovače a může aktualizaci směrovací tabulky zahájit okamžitě.

11.8 Časovače protokolu EIGRP

Mimo mechanismů konvergence využívá protokol EIGRP ke své činnosti časovače. Tyto dvě skupiny však nejsou vzájemně nikterak odděleny, jelikož jak jsme mohli vidět například při odstavování cest, kdy tato metoda konvergence využívala ke své činnosti časovač.

V protokolu EIGRP jsou využívány tyto časovače:

- aktualizační,
- časovač pro odstranění cesty,
- časovač vyprázdnění cesty,
- časovač neplatnosti cesty.

Není žádným překvapením, že tyto časovače fungují obdobně jako u jiných směrovacích protokolů.

- **Aktualizační směrovač** je obsažen v každém směrovači a odpočítává dobu do dalšího odeslání aktualizace směrovací tabulky. Implicitní hodnotou je 90 sekund. Délka tohoto intervalu může být měněna administrátorem.
- **Časovač odstranění cesty,** se kterým jsme se již setkali při metodě odstavení cest. Označuje interval, po jehož dobu je zastaveno odesílání aktualizací směrovací tabulky do okolních směrovačů.

- **Časovač vyprázdnění cesty**, který se stará o vymazání cest ze směrovací tabulky, u kterých k následnému obnovení spojení nedošlo. Ve výchozím nastavení je doba udržení záznamu ve směrovací tabulce 630 sekund, tedy je celkem 7 provedení aktualizací. v níž by se mohla objevit cesta znovu platná. K vymazání cesty označené za neplatnou nedojde ihned. Je to z důvodu případného obnovení spojení s cílovou sítí.
- **Časovač neplatnosti cesty** udává směrovači, za jak dlouhý časový úsek má cestu označit za neplatnou, pokud neobdržel aktualizační zprávu. Pokud z daného směru (respektive cesty) směrovač neobdrží aktualizační zprávu, je jasné, že na této cestě došlo k chybě nebo je koncová síť nedostupná. Hodnota pro tento časovač je ve výchozím stavu rovna trojímu provedení aktualizace, tedy 270 sekundám. Jestliže ani po tuto dobu neobdrží směrovač aktualizaci, označí cestu za neplatnou.

11.9 KONFIGURACE EIGRP

Konfigurace protokolu EIGRP není nijak složitá a její ukázková konfigurace na naší cvičné topologii je uvedena v příloze H.

[10][11]

12 OSPF

12.1 OBECNĚ O PROTOKOLU

Společně s rozvojem internetu, tedy koncem osmdesátých let, začalo být směrování pomocí vektoru vzdálenosti značně nevýhodné. Z tohoto důvodu se začal vymýšlet nový způsob směrování, založený na sledování stavu linky. A co je to linka? Linkou rozumíme spojení mezi dvěma směrovači. Toto spojení má své vlastnosti, dle kterých lze linky rozlišovat a určovat jejich aktuální stav.

Základním představitelem protokolu řídícího se sledováním stavu linky, je protokol OSPF. Tento protokol byl navrhnut sdružením IETF, které jmenovalo zvláštní skupinu odborníků z oblasti počítačových sítí, zaměřenou na vývoj nového směrovacího protokolu. Předlohou pro vznik byla celá řada proprietárních protokolů typu nejkratší cesty založených na Dijkstrově algoritmu. O principu Dijkstrova algoritmu jste se mohli dočíst v kapitole věnované směrovacím protokolům typu stav linky. Tyto proprietární protokoly byly využívány různými výrobci, jelikož byly v této době velmi prosazovány. Ovšem od doby vzniku protokolu OSPF došlo k mnoha úpravám a vývoji tohoto protokolu. Společně s vývojem Internetu bylo a bude potřeba udržovat krok, a tudíž lze očekávat, že i v budoucnu bude muset tento protokol reagovat na pokrok s technickým vývojem počítačových sítí. Je nutné poznamenat, že v literatuře i v jiných odborných textech jsou rozlišovány dva druhy protokolu OSPF a to

- OSPFv2 určený pro využití v sítích využívajících směrovaný protokol IPv4.
- OSPFv3 určený pro využití v sítích využívajících směrovaný protokol IPv6.

Pozn.: V našem textu budeme používat pouze označení OSPF, jenž bude zastupovat OSPFv2.

Zajímavostí ve vývoji protokolu OSPF je úplně první pokusný protokol OSPFv1. Tento protokol sloužil jako koncept a ve skutečnosti nebyl nikdy vydán. Proto se všude hovoří až o protokolu OSPFv2. OSPFv1 měl dvě odlišné implementace. První implementací, jež byla napsána, byla verze pro stanice využívající operační systém typu UNIX, zatímco druhou implementací rozumíme rozšířený UNIXový proces známý jako GATED. Po tomto prvním

experimentálním protokolu OSPFv1 došlo k notným úpravám a roku 1991 byl zveřejněn OSPFv2 v RFC 1247.

Sdružení ISO také současně pracovalo na protokolu typu stavu linky, který je známý jako IS-IS. Ovšem IETF zvolilo protokol OSPFv2 jako doporučovaný interní směrovací protokol. Toto doporučení odráží administrativní vzdálenost, jenž je přiřazena protokolu OSPF.

Jak můžeme vidět v následující tabulce, je tento protokol při výběru zvýhodňován před protokolem IS-IS a mnoha dalšími.

Tabulka 13 Ukázka postavení hodnoty AD protokolu OSPF v celkovém přehledu hodnot AD

Typ spojení	Hodnota AD
přímé spojení	0
statická cesta	1
EIGRP sumarizovaná cesta	5
externí BGP	20
interní EIGRP	90
IGRP	100
OSPF	110
IS-IS	115
RIP	120
externí EIGRP	170
interní BGP	200
Ostatní	255

Netrvalo dlouho a roku 1999 stejný tým vývojářů, jako v případě protokolu OSPFv2, zveřejnil verzi OSPFv3, jenž slouží pro práci v sítích se směrovaným protokolem IPv6.

Důležitou vlastností, jenž chybí protokolu OSPF je možnost využívání více směrovaných protokolů. OSPF byl navržen výhradně pro využití se směrovaným protokolem IP, tudíž ho nelze implementovat v autonomních systémech využívajících kupříkladu IPX nebo AppleTalk.

Hlavní funkce a vylepšení oproti protokolům typu vektoru vzdálenosti, přinesl OSPF v podobě následujících vlastností:

- beztřídnost,
- rychlá konvergence,
- vyšší bezpečnost,
- snadná rozšiřitelnost,
- efektivita.

12.2 DATOVÉ STRUKTURY V OSPF

Stejně tak jako předchozí směrovací protokoly, potřebuje i OSPF ke své činnosti evidovat různé typy informací. Tyto informace jsou evidovány v různých datových strukturách. S některými jsme se mohli setkat v kapitole věnované EIGRP.

Databáze informací, které protokol OSPF ke své činnosti potřebuje evidovat, členíme na

- databáze sousednosti,
- databáze stavu linky,
- databáze doručení.

Všechny tyto databáze jsou evidovány v paměti RAM, jelikož dochází k jejich změnám pokaždé, když nastane nějaká změna v topologii sítě.

- **Tabulka sousedů** (neighbor table) obsahuje první zmíněný typ databáze, tedy databázi sousednosti. Tabulka sousedů obsahuje informace o všech sousedních směrovačích, k nimž je daný směrovač připojen. Nikoho tudíž nepřekvapí, že tato tabulka je vždy rozdílná pro každý směrovač, jelikož každý ze směrovačů v síti je propojen s různými sousedy. Tabulku sousedů lze na směrovači vypsat pomocí příkazu

  ```
  R# show ip ospf neighbor
  ```

- **Tabulka topologie** (topology table) na rozdíl od tabulky sousedů, obsahuje tato datová struktura informace o kompletní topologii sítě, respektive o všech ostatních směrovačích v rámci oblasti. Jelikož eviduje informace o celé topologii sítě, nikoho nepřekvapí, že bude totožná pro všechny směrovače v rámci jedné oblasti. Pro vypsání této tabulky použijeme příkaz

  ```
  R# show ip ospf database
  ```

- **Směrovací tabulka** (routing table) je neopominutelná součást každého směrovacího protokolu. Obsahuje záznamy, které byly zvoleny v rámci tabulky topologie jako nejvýhodnější cesty do cílových sítí. Tyto záznamy byly vybrány pomocí algoritmu nejkratších cest (SPF), využívající Dijkstrův algoritmus. Směrovací tabulku vypíšeme pomocí příkazu

  ```
  R# show ip route
  ```

12.3 OBLASTI V OSPF

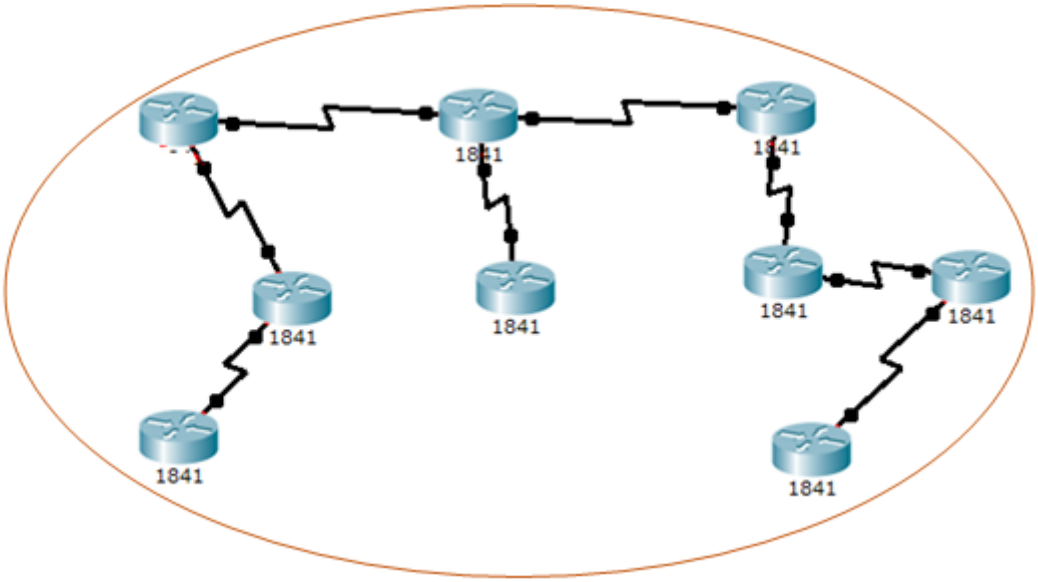
Vývojáři protokolu OSPF se snažili o základní dva cíle, na něž kladli při vývoji velký důraz:

- co nejnižší doba konvergence sítě,

- umožnění vyšší a lepší škálovatelnosti sítě.

Aby však bylo možné splnění těchto dvou cílů, bylo nutné rozdělit celou síť do jednotlivých menších částí. Tyto části označujeme jako oblasti (area).

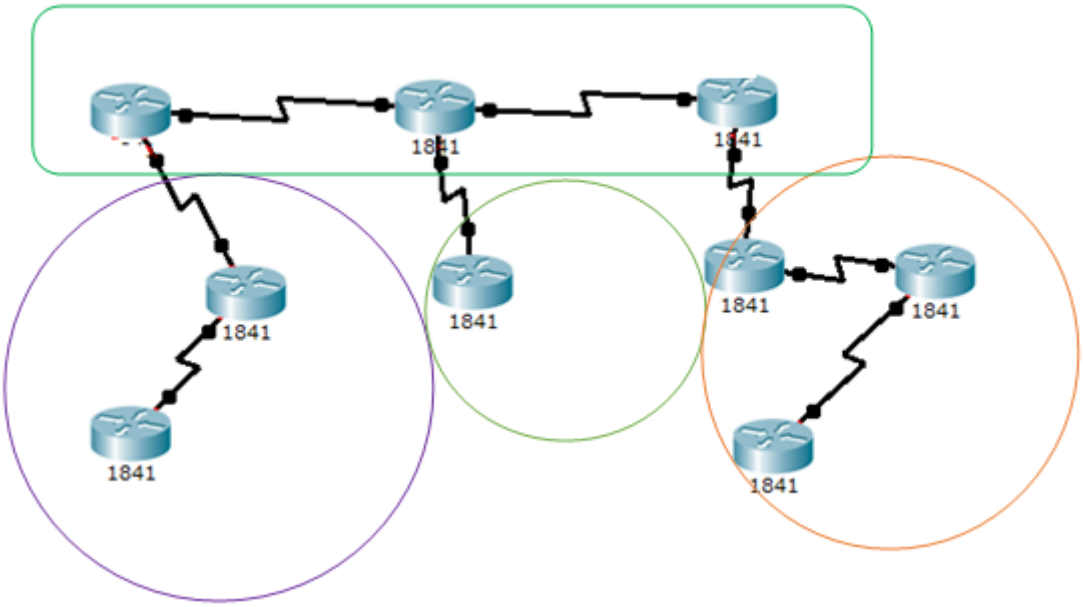
Síť před rozdělením může být libovolně rozsáhlá a tak vznikne vícero oblastí. Nikoho nepřekvapí, že tyto oblasti jsou odlišeny číslem oblasti, respektive rozhraní, jež mají spadat do konkrétní oblasti, jsou ohodnoceny stejným číslem oblasti. Není však povinností členit síť do oblastí, můžeme mít pouze jednu jedinou oblast. Tento způsob směrování nazýváme směrováním v rámci jedné oblasti (single area routing). Pokud využíváme ke směrování pouze jednu oblast, označujeme ji jako oblast 0, neboli páteřní oblast.



Obrázek 49 OSPF single area

Zároveň však není možné mít oblastí nekonečně mnoho, již z toho důvodu, že v síti by později mohl vzniknout zmatek. Členění sítě je tedy vhodné provádět na základě určitých geografických podmínek, či podle funkčnosti dané části sítě. Vezměme si například modelovou topologii, v níž chceme vhodně rozdělit celou topologii do jednotlivých oblastí.

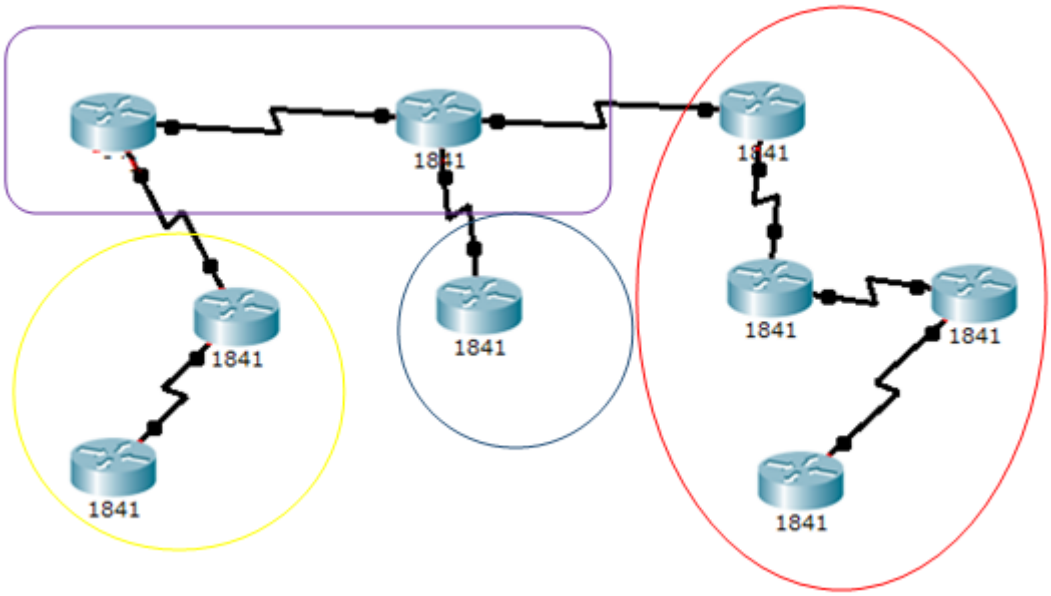
Způsob, jakým by mohla vypadat tato síť po rozdělení do oblastí, zobrazuje následující obrázek.



Obrázek 50 OSPF rozvržení oblastí 1

Jak můžeme na obrázku vidět, celá síť je rázem rozdělena na tři samostatné oblasti propojené jednou oblastí páteřní. Platí, že pokud chceme síť rozdělit do více oblastí, musí existovat vždy síť páteřní, jenž bude propojovat jednotlivé oblasti dohromady.

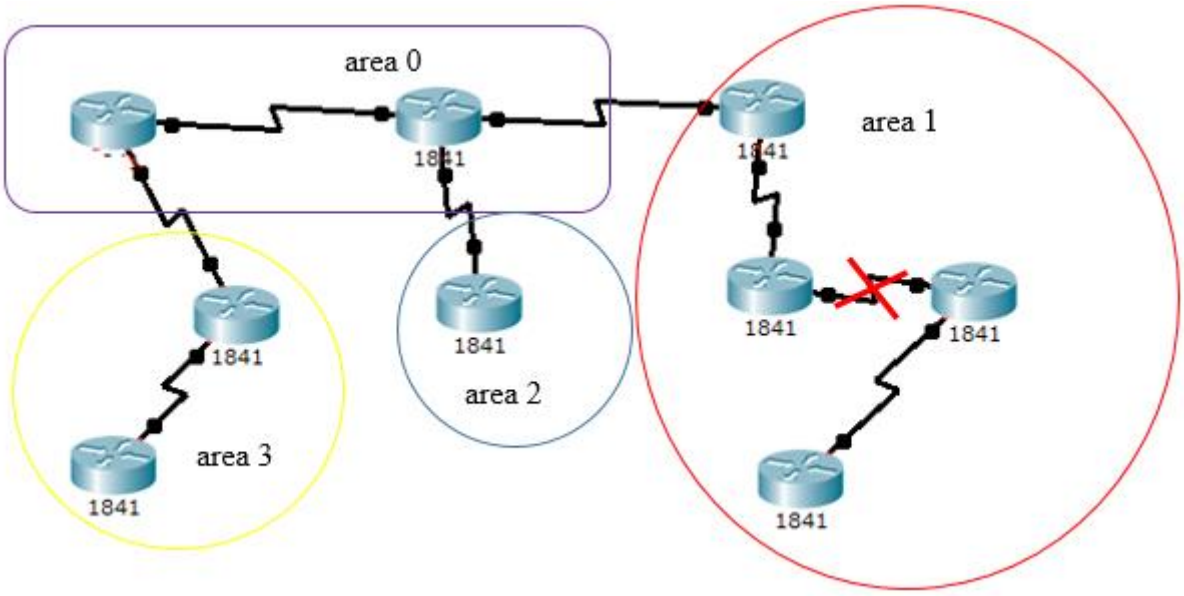
Síť může být, jak již bylo řečeno, rozdělena více způsoby, a tudíž lze modelovou topologii rozdělit i následovně.



Obrázek 51 OSPF rozvržení oblastí 2

Z předchozího obrázku je zřejmé, že rozdělení oblastí závisí zcela na nás a na našich potřebách v rámci sítě, jelikož pouze my jako administrátoři, můžeme určit, které prvky spolu určitým způsobem souvisí.

Každá z těchto oblastí je propojena k páteřní síti, ovšem propagace probíhá pouze mezi směrovači v rámci oblasti. Tato restrikce velikosti sítě určené pro propagování cest snižuje nároky na šířku pásma, jelikož aktualizační informace zatěžují provoz v síti. Pokud by tedy například došlo k následující situaci



Obrázek 52 OSPF havárie linky

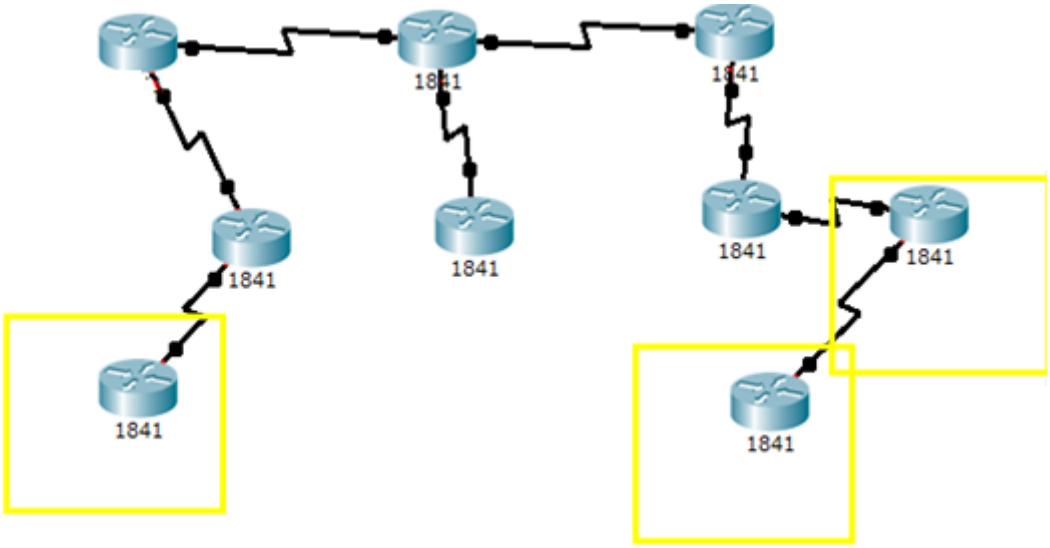
budou pakety LSU odeslány pouze v rámci oblasti ohraničené červenou čarou. Ostatních oblastí se tato změna nikterak nedotkne.

Pozn.: V rámci této kapitoly se zaměříme na směrování OSPF pouze v rámci jedné oblasti. Směrování OSPF s vícero oblastmi bude věnována samostatná kapitola.

V souvislosti s oblastmi využívanými v protokolu OSPF je důležité zmínit, že dochází ke členění směrovačů, podle jejich výskytu:

- **Interní směrovače** jsou takové, které se nacházejí uvnitř jedné oblasti. Všechny porty tohoto směrovače spadají do jedné oblasti. Každý z interních směrovačů si musí vyměňovat oznámení LSA s každým směrovačem náležícím stejné oblasti, tedy ostatními interními směrovači a zároveň se směrovačem hraničním. LSA jsou šířeny

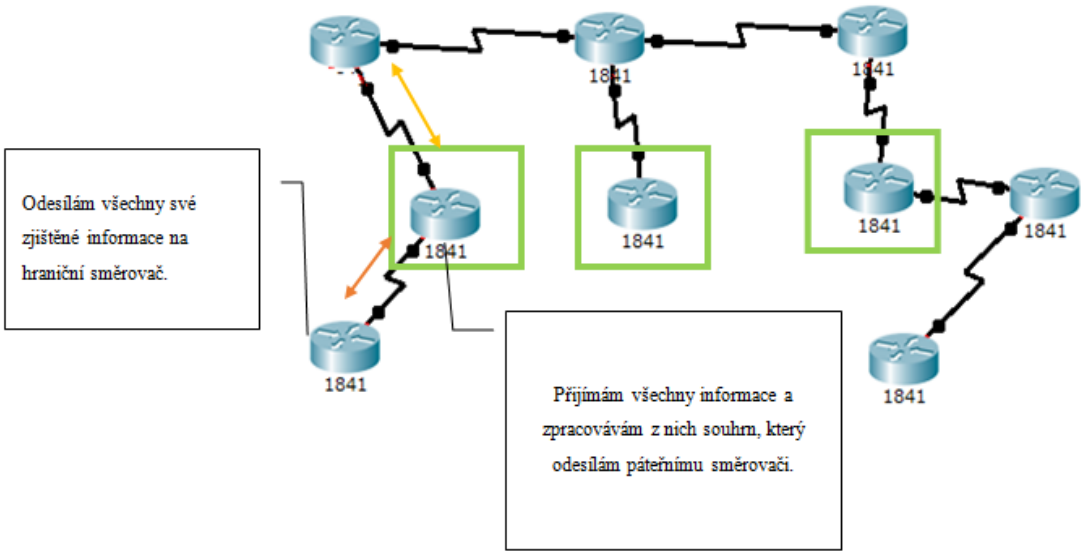
mezi interními směrovači.



Obrázek 53 Zobrazení interních směrovačů

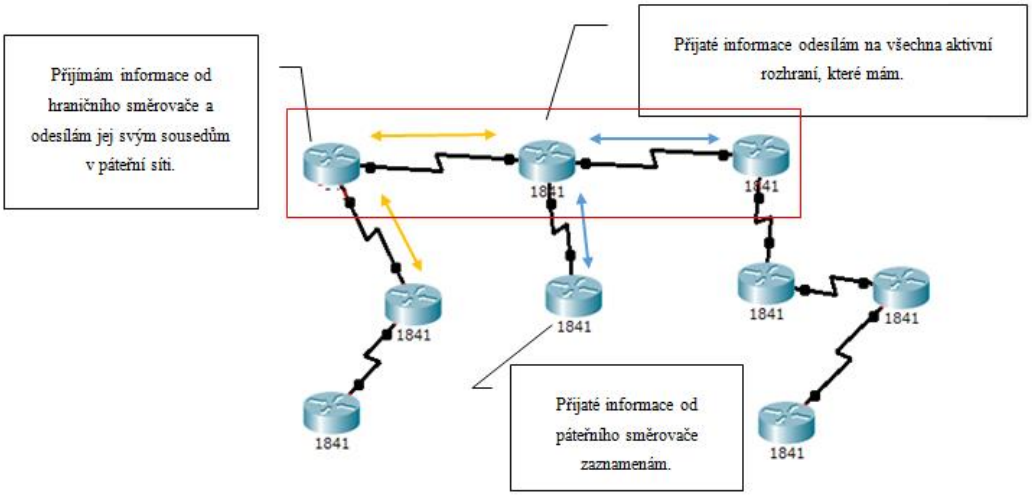
- **Hraniční směrovače** leží na okraji oblasti. Tyto směrovače obsahují porty náležící vícero oblastem. Odpovídají za aktuální stav databází topologie jednotlivých oblastí. Hraniční směrovač přijímá informace od interních směrovačů náležících jeho přímo připojené oblasti, tyto informace zpracuje a odešle do páteřní oblasti, díky které dojde k rozšíření souhrnné informace do ostatních oblastí. Díky publikování souhrnných informací a nikoliv široké škály cílových adres, jenž jsou obsaženy v konkrétní oblasti, nedochází k tak výraznému zatížení šířky pásma. Navíc okolní oblasti potřebují pouze znát, jaká je sumarizovaná adresa.

Odeslání paketu do správné cílové sítě je již prováděno v rámci oblasti.



Obrázek 54 Činnost hraničních směrovačů

- **Páteřní směrovače** jsou odpovědné za udržování informací o páteřní lince a zároveň je jejich úkolem šíření souhrnných informací, získaných od jednotlivých hraničních směrovačů, mezi oblastmi. Pokud páteřní směrovač obdrží paket s informacemi, informuje všechny své přímo připojené sousedy, tedy všechny hraniční směrovače, jež jsou připojeny k páteřní síti, a zároveň informuje všechny sousední páteřní směrovače.



Obrázek 55 Činnost páteřních směrovačů

12.4 TYPY ZPRÁV

V rámci směrovacího protokolu OSPF rozlišujeme pět druhů paketů sloužících pro správu tohoto protokolu. Řada zpráv v rámci komunikace funguje obdobně jako u protokolu EIGRP.

Rozlišujeme:

- **Hello pakety** jsou označovány jako pakety typu 1. Obdobně tak jako u protokolu EIGRP slouží tento typ paketů k udržování stávajících spojení a zároveň k navazování spojení nových mezi sousedními směrovači. Vztah mezi dvěma směrovači označujeme jako sousednost. Sousednost je základním kamenem výměny veškerých směrovacích informací v rámci protokolu OSPF. Pro navázání sousednosti se musí dva směrovače shodnout na určitých parametrech. Aby tyto parametry byly zřejmé oběma směrovačům, mezi nimiž má být sousednost navázána, musí být mezi těmito směrovači sděleny. K tomuto účelu jsou využívány hello pakety. Tyto pakety jsou odesílány v OSPFv2 na multicastovou adresu 204.0.0.5 a u OSPFv3 na adresu FF02::5. Interval, po kterém jsou hello pakety odesílány může být 10 sekund u sítí typu point-to-point, nebo 30 sekund u sítí typu frame relay. Druhým intervalem, který je evidován je takzvaný **dead interval**, po který směrovač čeká na přijetí hello paketu od souseda, než jej označí za nedostupný. Pokud dojde k vypršení tohoto intervalu, vymaže směrovač nedostupného souseda ze své databáze stavu linek a informuje pomocí LSU všechny ostatní směrovače v síti. Dead interval je stanoven na čtyřnásobek doby intervalu hello paketu.
- **Database description packet** (DBD) který obsahuje informace s popisem databáze. Odesílání těchto paketů je zahájeno v případě zahájení navazování sousednosti mezi dvěma směrovači. Pozor, tento paket nepřenáší žádnou databázi, ale pouze informace

popisující databázi stavu linek. Při procesu sousednosti je zvolen jeden ze směrovačů jako nadřízený, jenž odešle celý obsah své směrovací tabulky. Oproti tomu směrovač podřízený musí potvrzovat příjem paketů s popisem databáze stavu linek. Volba nadřízeného a podřízeného směrovače není nikdy závazná pro celou dobu výměny popisů databáze. Pokaždé, když probíhá tato výměna, může mít roli nadřízeného i podřízeného směrovače jiný ze směrovačů. Ovšem podmínkou je, že v síti nesmí být pouze všechny směrovače nadřízené nebo naopak všechny podřízené. Platí, že databáze stavu linek musí být na všech směrovačích shodná, aby bylo možné vytvořit SPF strom pro umožnění výpočtů nejvýhodnějších cest.

- **Link-state request packet** (LSR) označován jako paket typu 3. Z názvu vyplývá, že slouží k odeslání určitého požadavku. Směrovač odesílající tento paket požaduje od adresovaného směrovače určitou část databáze stavu linek. Vyžádání této informace je možné ovšem pouze u směrovače sousedního. A kdy napadne směrovač vyslat takovýto požadavek? Díky němu je totiž směrovač schopen porovnat, zda sousední směrovač neobsahuje aktuálnější informace nebo naopak neobsahuje informace, jenž odesílajícímu směrovači chybí. Ovšem aby dostal odesílající směrovač informace, které požaduje, musí jasně v této žádosti specifikovat, o které záznamy má zájem.
- **Link-state update packet** (LSU) paket typu 4. Slouží pro odesílání odpovědí na LSR a zároveň pro odesílání nových informací. Dalším důvodem k odesílání LSU jsou změny v topologii. LSU může obsahovat až 11 druhů LSA zpráv.
- **Link-state acknowledgement packety** (LSAck) jsou pakety typu 5. Jejich úkolem je potvrzení přijetí LSU. Jelikož slouží pouze k potvrzování, je datová část těchto paketů prázdná.

Všechny výše uvedené pakety jsou použity pro výměnu směrovacích informací a zároveň také pro objevování nových sousedních směrovačů, či reakce na změny v síťové topologii.

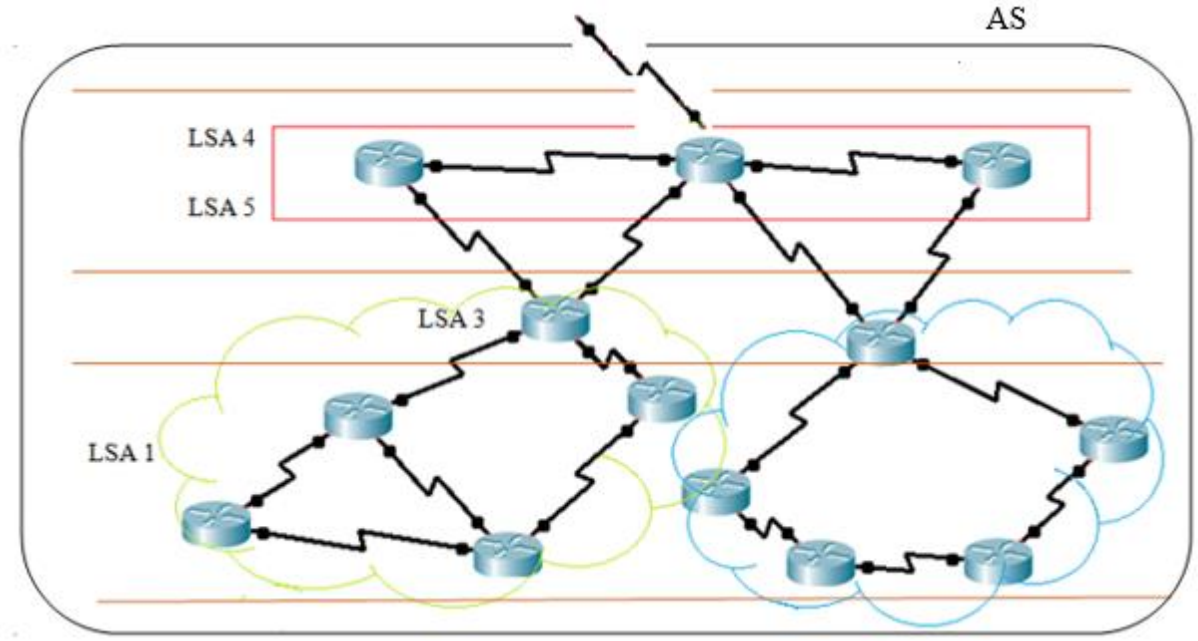
Zmínili jsme pojem **LSA**. Tato tři písmena jsou zkratkou pro link state advertisement. Tento paket je základním kamenem komunikace v OSPF a je důležité si uvědomit, že není vždy odesílán v podobě „záplavy“ celou sítí. Může být odeslán pouze na některá rozhraní. Tento rozdíl v odesílání způsobily čtyři druhy směrovačů v rámci sítě. Nikoho tak nepřekvapí, že například páteřní směrovač bude komunikovat jiným způsobem než směrovač interní. Pomocí paketů LSA je směrovač schopen komunikovat s ostatními směrovači v rámci jedné oblasti. Každý LSA je odlišen svým číslem ID, dle kterého je možné později komunikaci srovnat. Rozlišujeme 11 druhů LSA paketů, tedy „zpráv“, které mohou být obsahem jiné komunikace.

- **LSA typu 1**, označované také jako LSA směrovače. V tomto typu LSA směrovač propaguje v rámci své oblasti informace o svých spojeních s ostatními směrovači včetně informace o metrice k těmto spojům přiřazené. Neobsahuje tedy nic jiného než informace o samotném odesílajícím směrovači, jeho rozhraních a směrovačích, s nimiž má navázán vztah sousedství. Tyto pakety jsou ohodnoceny LSA ID o hodnotě ID směrovače.
- **LSA typu 2**, jenž jsou označovány jako LSA pro síť. Označený směrovač (designated router), dále jen DR, odesílá na broadcastovou adresu seznam všech směrovačů, které jsou propojeny dohromady v jeden segment. Jak tedy z předchozího vyplývá, jsou LSA typu 2 odesílány pouze v rámci oblastí, ve kterých je zvolen DR. Jejich rozesílání je prováděno pouze v rámci této oblasti. Jelikož tento typ obsahuje vždy celkový seznam směrovačů, redukuje tak velikost topologické tabulky a tudíž ulehčuje výpočet pomocí SPF pro tuto oblast. Ohodnocení těchto paketů LSA je hodnota IP adresy rozhraní na DR. Spolu s LSA typu 1 obsahují mnoho informací o topologii sítě, které jsou šířeny v rámci konkrétní oblasti.

- **LSA typu 3**, které jsou sumarizované LSA. Jelikož mezi jednotlivými oblastmi jsou vysílané pouze souhrnné informace, je tento typ LSA paketu použit právě na hraničním směrovači. Hraniční směrovač vezme veškeré informace, které se dozvěděl od připojené oblasti, tyto informace zpracuje v souhrn, který odešle dalším oblastem, ke kterým je připojen. Informace od připojené oblastí získává hraniční směrovač pomocí LSA typu 1 a 2.
- **LSA typu 4**, které jsou obdobou typu předešlého, avšak souhrnné informace odesílá hraniční směrovač autonomního systému. Tyto informace jsou důležité pro využití LSA typu 5. Jsou šířeny do všech ostatních oblastí. Protože jsou LSA pakety typu 5 odesílány do celé sítě, nebyly by dostupné informace o dalších přeskocích. Tento typ LSA paketu slouží k umožnění výpočtu metriky do připojené oblasti u ostatních směrovačů. LSA ID bude v tomto případě ID směrovače, který plní funkci hraničního směrovače autonomního systému.
- **LSA typu 5,** jež obsahují informace o všech importovaných cestách do směrovače, získaných pomocí protokolu OSPF. Jsou rozesílány do všech oblastí, avšak sami o sobě nemají dostatek informací k výpočtu externích cest. Proto musí být odesílány spolu s LSA pakety typu 4. Společně s LSA typu 4 tak slouží ke kalkulaci nejvýhodnější cesty a metriky k externím cestám.
- **LSA typu 6** jsou používány pro multicastové odesílání informací. Je důležité poznamenat, že odesílání těchto informací je podporováno pouze na některých zařízeních. Většího využití se očekává od verze OSPFv3. Zatím není tento typ používán a plánuje se spíše do budoucna.
- **LSA typu 7**, jenž jsou využívány u NNSA (not-so-stuby-area), ve kterých není dovoleno přijímání LSA typu 5, tedy takovéto oblasti by neměly žádné informace o externích cestách. Z tohoto důvodu jsou odesílány pakety LSA typu 7.

- **LSA typu 8**, což je paket odesílající informace pro externí směrovací protokol Border Gateway Protocol.
- **LSA typu 9, 10, 11**, které jsou ponechány pro budoucí využití.

Kde jsou odesílány základní LSA pakety, znázorňuje následující obrázek.



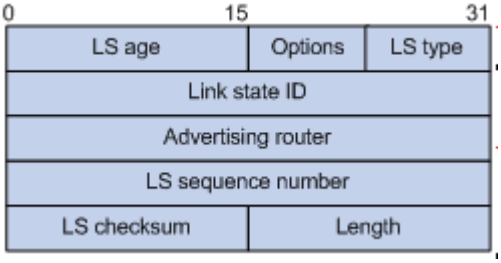
Obrázek 56 OSPF oblasti odesílání jednotlivých typů LSA

12.5 HLAVIČKA LSA

Nyní již víme, jaké druhy LSA zpráv rozlišujeme, ovšem nevíme, jakou musí každá LSA zpráva obsahovat hlavičku, aby v síti mohlo dojít k jasnému rozpoznání jednotlivých typů. Také již víme, že LSA zprávy slouží jako oznámení o stavu linek, a tudíž jsou pro protokol, jenž je založený právě na stavu linek nejdůležitější součástí.

Všechny LSA pakety obsahují stejnou hlavičku, liší se pouze svým typem, který je uvnitř hlavičky jasně specifikován. Hlavička je u všech typů o velikosti 20 oktetů, tedy 160 bitů a je připojována na konec hlavičky standartní pro protokol OSPF, jenž je dlouhá 192 bitů, tedy 24 oktetů.

Jak hlavička paketu LSA vypadá, znázorňuje následující obrázek.



Obrázek 57 Hlavička LSA

Význam jednotlivých položek:

- **Stáří stavu linky (LS age)** udává stáří konkrétního záznamu. Toto stáří je udáváno v sekundách a vyjadřuje počet sekund, které uplynuly od vytvoření LSA. Pomocí tohoto pole tedy lze porovnávat, jak aktuální je přijatý záznam.
- **Volby (Options)** označení služeb, které mohou být podporovány v OSPF.
- **Typ stavu linky (LS type)** označuje typ záznamu, které konkrétní LSA obsahuje. Každý typ má přitom jiný formát, a tudíž je nezbytně nutné vědět, jaký formát dat nalezneme za hlavičkou tohoto paketu. Může zde být tedy obsaženo celkem 11 různých příznaků. Jaké typy oznámení LSA rozlišujeme, jsme probrali v předchozí části textu.
- **Identifikátor stavu linky (Link state ID)** obsahuje popis oblasti, pro kterou se dané LSA oznámení vztahuje a kterého se týká. Obsahuje vždy příslušné ID LSA. Jaké mají jednotlivé typy LSA hodnoty ID, bylo probráno také v předchozí části textu.
- **Oznamující směrovač (Advertising router)** je směrovač, který tuto zprávu LSA vytvořil a odeslal. Toto pole tedy obsahuje ID daného směrovače.
- **Pořadové číslo stavu linky (LS sequence number)** obsahuje pořadové číslo LSA zprávy. S každou vygenerovanou LSA zprávou je totiž inkrementován čítač. Směrovač, který LSA paket příjme je tudíž schopný snadno rozlišit, který záznam je

novější. Prvním způsobem je rozlišení podle doby stáří stavu linky (pole 1) a druhým je právě toto pořadové číslo, jelikož záznam, jenž je aktuálnější, byl vygenerován později, a tudíž i jeho pořadové číslo bude vyšší. Pokud bychom k určování aktuálnosti záznamu používali první způsob, mohlo by se lehce stát, že novější informace by například z důvodu vyššího síťového provozu dorazila později, a tudíž by měla vyšší hodnotu stáří záznamu, i přestože by se jednalo o záznam aktuálnější. Aby se předcházelo těmto nesrovnalostem, využívá se pro kontrolu aktuálnosti druhý způsob, tedy podle pořadového čísla, jelikož to zůstává vždy neměnné.

- **Kontrolní součet stavu linky (LS checksum)**, které stejně tak jako jiné kontrolní součty, slouží i toto pole pro detekci případných chyb nebo poškození oznámení, při přenosu do cíle.
- **Délka stavu linky (Length)** udávající celkovou velikost oznámení LSA.

12.6 VOLBA DR A BDR

DR neboli designated router je česky označován jako **pověřený směrovač.** BDR je backup designated router a česky jej označujeme jako **záložní pověřený směrovač**. A co je pověřený směrovač? Pověřený směrovač je vybraný směrovač, který je zodpovědný za provedení aktualizací na všech ostatních směrovačích. Tyto ostatní směrovače označujeme jako DRother.

Jistě nikoho nepřekvapí, že úkolem záložního pověřeného směrovače je monitorování činnosti směrovače pověřeného, a v případě výpadku tohoto sledovaného směrovače převzetí jeho funkce. Pověřený i záložní pověřený směrovač jsou voleny na základě **priority rozhraní**.

Sami můžeme ovlivnit, jak bude volba DR a BDR probíhat, přiřazením této priority. Jednoduše u směrovače, jež si přejeme, aby se stal DR, nastavíme hodnotu priority vyšší a u toho, jenž se má stát BDR nastavíme druhou nejvyšší. Implicitně jsou hodnoty priorit rovny jedné a tudíž je DR a BDR volen na základě čísla ID směrovače. Zatímco implicitní hodnota priority je rovna jedné, je nejvyšší možná hodnota rovna 255.

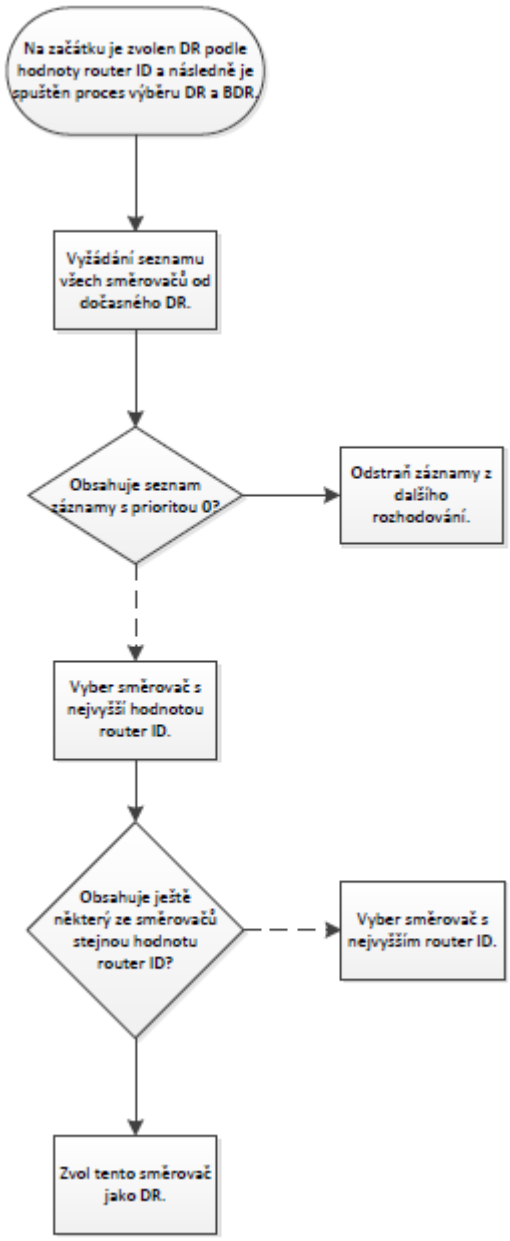
Změnu priority směrovače provedeme následovně.

```
R > enable

R #conf term

R (config)# int se0/0/0

R (config-if)#ip ospf priority 255
```

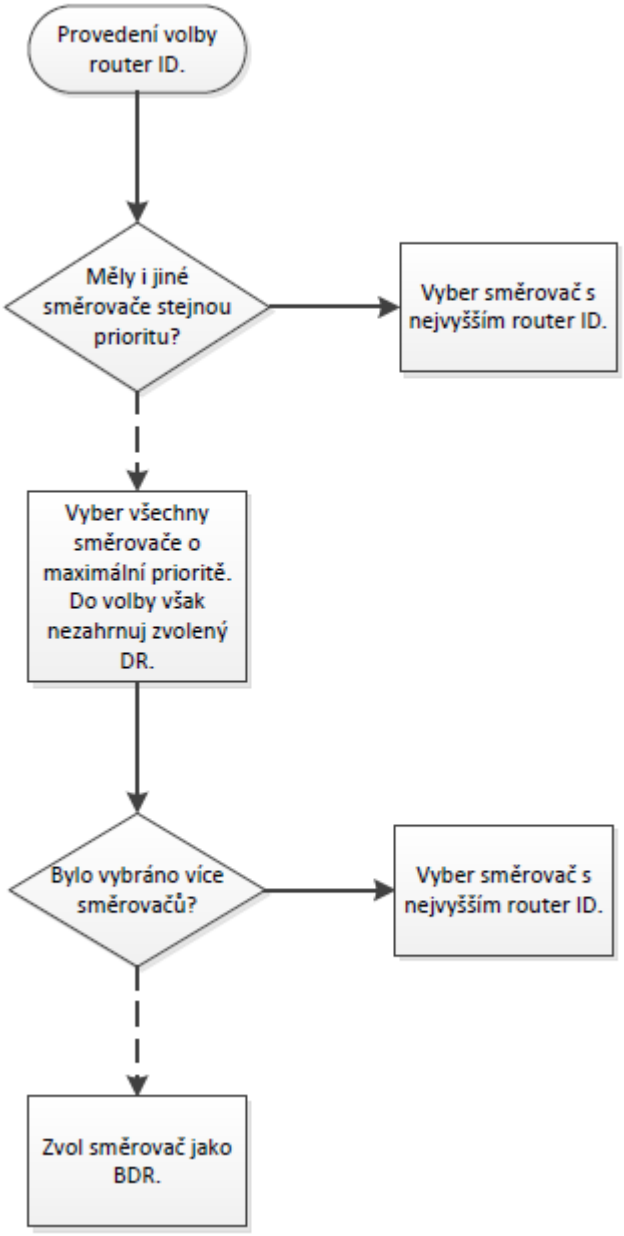
Pokud si chceme ověřit, na základě jakých hodnot dochází k určování DR a BDR, můžeme si snadno vypsat informace o hodnotě priority na konkrétním rozhraní. Tento výpis provedeme následovně.

```
R # show ip ospf interface se0/0/0
```

Celý princip volby DR a BDR znázorňují následující diagramy.



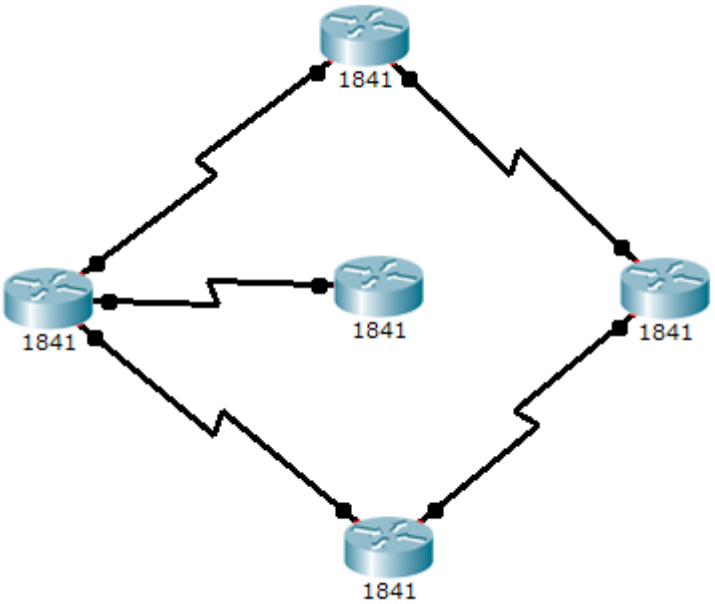
Obrázek 58 OSPF volba DR směrovače



Obrázek 59 OSPF volba BDR

12.7 PRINCIP OSPF V KOSTCE

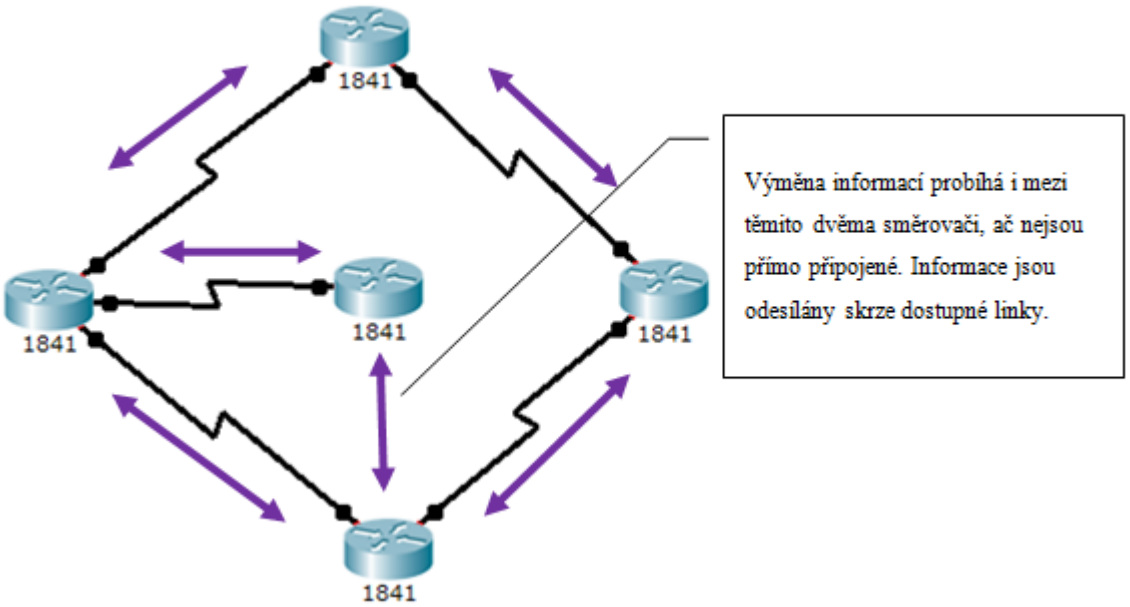
Mějme pro následující výklad modelovou, kterou zobrazuje následující obrázek.



Obrázek 60 OSPF v kostce výchozí topologie

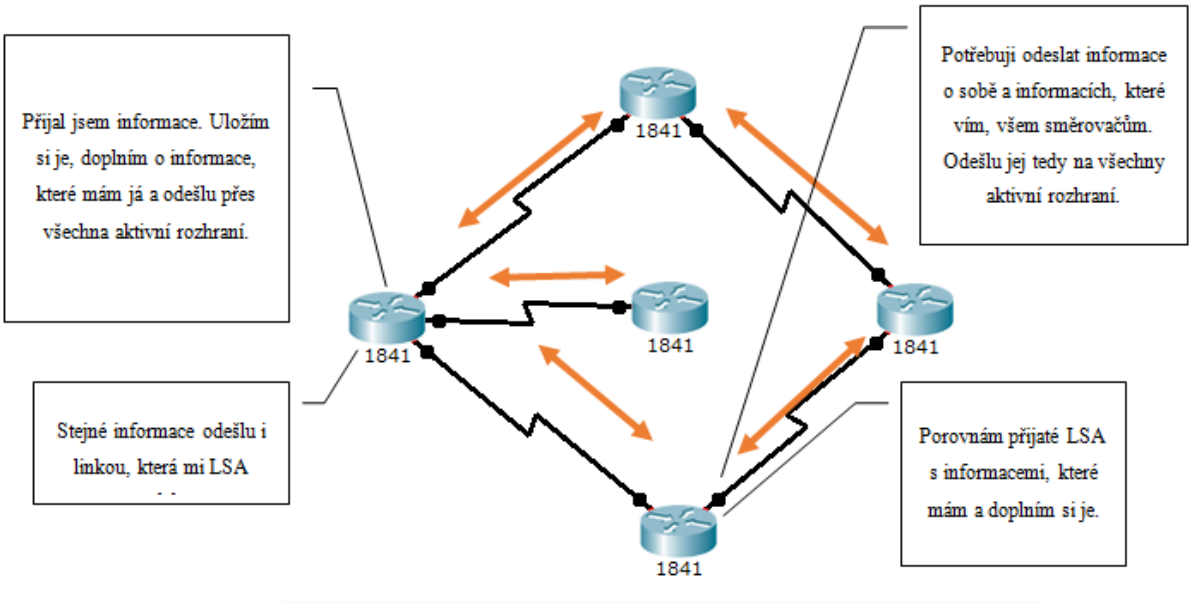
Ihned poté, co jsou jednotlivé směrovače vzájemně propojeny, začínají vysílat hello pakety. Tyto hello pakety informují ostatní směrovače o vzájemné existenci. Odesílání hello paketů lze přirovnat k pozdravu, všechny směrovače se vzájemně představí. Jak již bylo řečeno v části o interních směrovačích, není nezbytně nutné, aby spolu byly dva směrovače přímo propojené na to, aby mohly vzájemně vyměňovat hello pakety, či následně LSA pakety.

V naší topologii by došlo k následující výměně hello paketů.



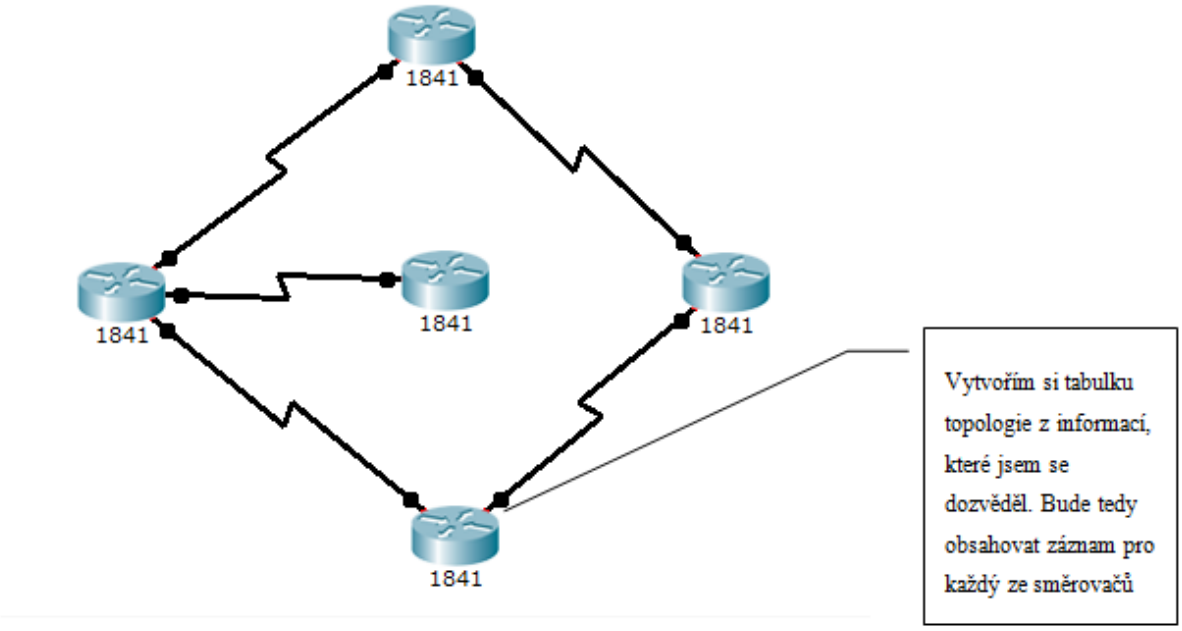
Obrázek 61 OSPF a záplava hello paketů v rámci sítě

Dalším krokem, který musí následovat je výměna LSA informací. LSA informace, jak již víme, obsahují cenu každé přímo připojené linky. Výměna LSA probíhá v tomto případě formou záplavy. **Záplava** je stav, při kterém směrovač, jenž příjme informace od svého souseda, získané údaje LSA uloží a zároveň ihned odešle všem dalším sousedním směrovačům. Díky tomuto postupu tak dojde během krátké doby k informování všech směrovačů v síti. Výměna LSA proběhne mezi všemi směrovači.[13]



Obrázek 62 OSPF výměna LSA paketů v rámci sítě

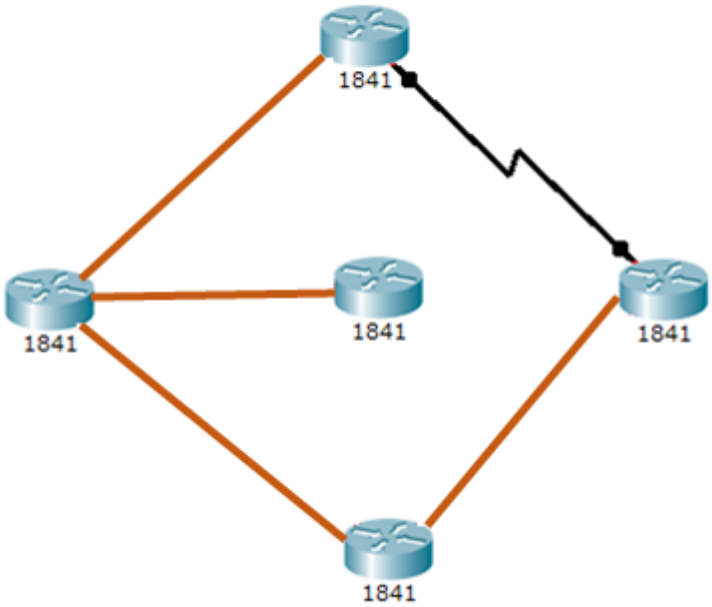
V okamžiku, kdy mají všechny směrovače potřebné informace, začnou si na základě informací přijatých pomocí LSA vytvářet tabulku topologie. Jelikož má každý ze směrovačů informace o celé topologii, obsahuje tato tabulka vždy stejné informace na každém ze směrovačů.



Obrázek 63 OSPF vytvoření tabulky topologie

Poté, co je vyplněna tabulka topologie, je bezprostředně spuštěn algoritmus SPF. Tento algoritmus určí, na základě informací uložených v tabulce topologie **strom nejkratších cest**. Strom nejkratších cest představuje informace o tom, jak se z konkrétního směrovače, dostaneme, co nejkratší cestou, do všech ostatních uzlů. Strom nejkratších cest pro levý směrovač by tak mohl vypadat následovně.[8][11][13]

Pozn.: Neopíráme se zde o žádné výpočty, příklad je pouze ilustrační. Přesný popis algoritmu se můžete dočíst v kapitole Protokoly stavu linky.



Obrázek 64 OSPF výsledný strom cest mezi směrovači

Jakmile na směrovači doběhne algoritmus SPF a směrovač tudíž zná nejkratší cesty ke všem ostatním směrovačům, zapíše si tyto informace do své směrovací tabulky.

12.8 Postup při navazování stavu přilehlosti

Výměna hello paketů je první krokem k navázání přilehlosti (adjencies). **Přilehlost** je stav, ve kterém si mohou dva směrovače vzájemně vyměňovat informace o stavu linek, respektive LSA pakety. Před navázáním plného sousedství mezi směrovači prochází směrovač následujícími stavy.

- **Init state** stav, kdy směrovač odešle hello paket a čeká na obdržení odezvy.
- **Two-way komunikace** představuje stav, kdy byl odeslaný hello paket potvrzen sousedním směrovačem. V tento moment se vybere pověřený směrovač z těchto dvou komunikujících směrovačů.
- **Exstart** směrovače si začnou formovat, který z nich bude podřízený a který nadřízený. Tento stav je nutný k tomu, aby byla možná korektní výměna informací.
- **Exchange** mezi nadřízeným a podřízeným směrovačem začne pobíhat výměna paketů s popisem databáze (DBD).
- **Loading**, v tento okamžik jsou odesílány LSR (link-state request) za účelem získání informací, které zatím nebyly předány. Informace nemusela dorazit korektně, a tudíž si ji směrovač znovu vyžádá.
- **Full state** v tomto stavu jsou dva směrovače plně přilehlé, tedy navázali sousedství. Tento stav signalizuje plnou synchronizaci link-state databází obou směrovačů.

Informaci o tom, v jakém z výše uvedených stavů, jsou dva sousední směrovače lze zjistit výpisem tabulky sousednosti. Tento výpis provedeme příkazem:

```
R# show ip ospf neighbor
```

12.9 METRIKA OSPF

Ať již protokol vektoru vzdálenosti nebo protokol stavu linky, každý potřebuje ke své činnosti parametr, díky kterému je možné jednotlivé cesty porovnávat a určovat tak nejvýhodnější cestu pro paket. Tento parametr, podle něhož lze cesty členit označujeme jako cenu (cost). Cena je libovolné číslo v rozmezí 1 až 65535. Tedy libovolné není tak úplně správný výraz. Při jeho přiřazování hraje roli právě stav linky. Platí, že čím je cena linky nižší, tím lepší metriku cesta má. Pokud neupravíme hodnotu metriky sami, je automaticky přiřazována následujícím vztahem.

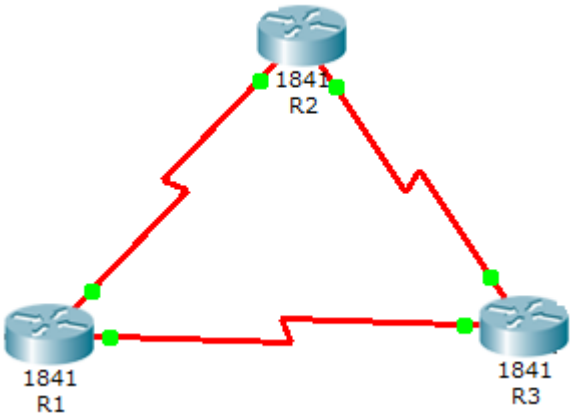
cena = 100000000 / šířka pásma [bps]

Jako příklad uveďme linku o šířce pásma 256 kbps. Takováto linka bude po dosazení ohodnocena následovně.

cena = 100000000/256000

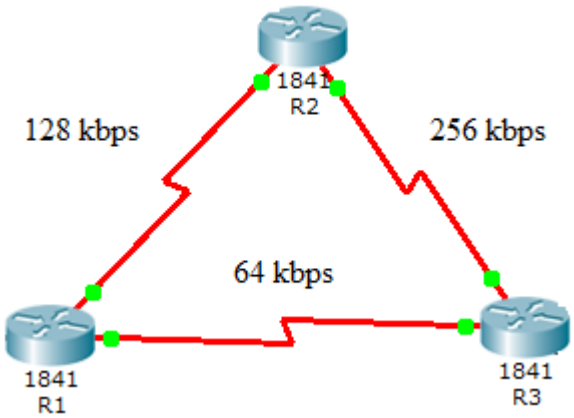
cena = **390**

Mějme následující topologii.



Obrázek 64 OSPF metrika modelová topologie

Za každým z těchto směrovačů leží síť příslušného směrovače. Vezměme pro představu výpočtu metriky následující typy linek.



Obrázek 65 OSPF ohodnocení spojů

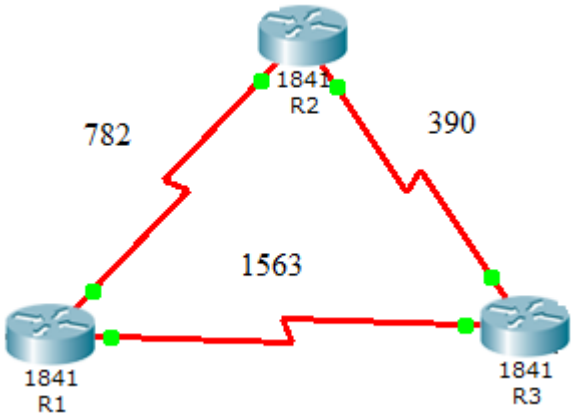
Pokud nebudeme nikterak ručně upravovat ceny jednotlivých spojů, dojde k následujícímu výpočtu cest automaticky.

cena linky R2 – R3 = 100000000 / 256000 = 390

cena linky R1 – R2 = 100000000 / 128000 = 782

cena linky R1 – R3 = 100000000 / 64000 = 1563

Z pohledu OSPF bude ohodnocená topologie vypadat následovně.



Obrázek 66 OSPF výsledná cena linek

Jako příklad nyní vezměme cestu ze směrovače R1 do sítě ležící za směrovačem R3. Při rozhodování, která cesta bude použita, porovnává OSPF následující dvě varianty:

- Pokud povede cesta přímo do směrovače R3, potom bude její metrika rovna 1563.
- Pokud povede cesta skrze směrovač R2, bude metrika rovna 782 + 390 = 1172.

Tedy překvapivě bude pro paket směrovaný do sítě R3 zvolena cesta skrze směrovač R2, i přestože bychom na první pohled předpokládali, že bude zvolena cesta vedoucí přímo do R3. Hodnotu ceny, respektive metriku z ní vypočítanou můžeme měnit a tím ovlivňovat cesty v síti. Prvním způsobem, kterým je možné změnit hodnotu je změna přiřazené šířky pásma na konkrétním rozhraní. Tuto změnu provedeme následovně.

```
R (config)# interface se0/0/0
R (config-if)# bandwidth 64
```

Pokud jsme takovýmto způsobem změnili šířku pásma, je automaticky proveden přepočet hodnot metriky. Druhým způsobem je nastavení hodnoty ceny linky.

```
R (config-if) # ip ospf cost 628
```

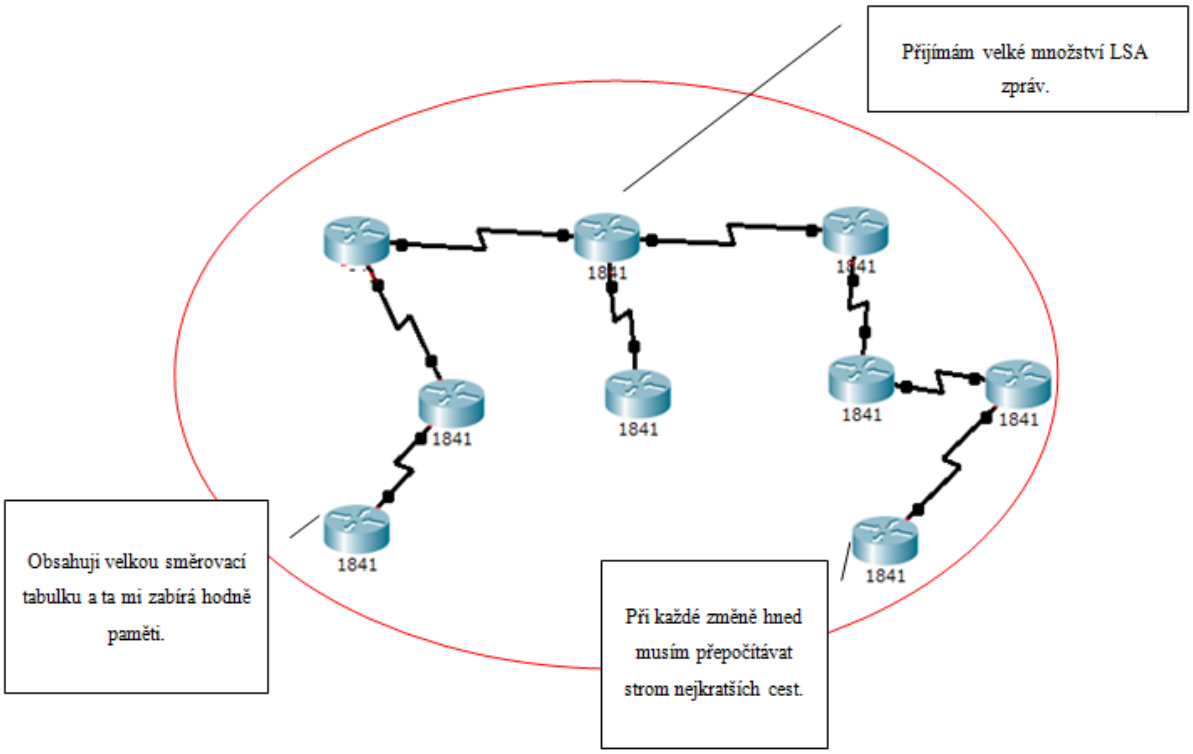
V tomto případě nebude docházet ke spuštění přepočtu ceny, ovšem dojde k vyhodnocení stromu nejkratších cest.

12.10 KONFIGURACE OSPF V RÁMCI JEDNÉ OBLASTI

Praktické využití výše zmíněných vlastností protokolu OSPF na ukázkové konfiguraci je uvedeno v příloze I.

12.11 SMĚROVÁNÍ V RÁMCI VÍCE OBLASTÍ

Ačkoliv je OSPF protokol jako takový velice rychlý a spolehlivý, se zvětšujícím se počtem zařízení začíná i tento protokol, pokud používáme směrování pouze v rámci jedné oblasti nula, kterou označujeme jako oblast páteřní, mít značné nevýhody. Hlavním problémem u zvětšující se sítě je velký obsah směrovacích tabulek, které zabírají hodně místa v paměti. Dalším problémem je také odesílání link-state advertisement (LSA), které zahlcují směrovače. Dále je také potřeba s jakoukoliv změnou v rámci síťové topologie spustit algoritmus SPF. Tyto výpočty potřebují ke své činnosti procesor, a tudíž dochází ke značnému zpomalování.



Obrázek 67 OSPF problémy nastávající v nečleněné síti

Aby zbytečně nedocházelo k takovému zatěžování sítě, je dobré síť členit. O členění sítě do oblastí jsme se zmínili v kapitole Oblasti v OSPF. Díky členění ulehčíme velikostem směrovacích tabulek a zároveň LSA bude šířeno pouze v rámci jedné konkrétní oblasti. Stejně tak tomu bude i v případě přepočtů stromů nejkratších cest. Tyto stromy budou vytvářeny pouze v rámci oblasti, ve které dojde ke změně.

Pro směrování pomocí OSPF v rámci více oblastí platí, že každá síť musí obsahovat jednu oblast páteřní, která propojuje všechny ostatní oblasti. Pokud by v síti tato páteřní oblast nebyla, nemohly by jednotlivé oblasti vzájemně komunikovat a tudíž by o sobě ani vzájemně nevěděli.

Pro tvorbu efektivních oblastí, nezatěžujících paměťové nároky směrovačů, doporučuje CISCO dodržovat tyto pravidla:

- V rámci jedné oblasti by nemělo být použito více než 50 směrovačů, pokud by měl být tento počet překročen, je lepší oblast dále členit.

- Směrovač by neměl být začleněn do více než tří oblastí. Pokud je tento počet překročen, byl by směrovač zbytečně zahlcován zprávami LSA z každé z připojených oblastí.
- Žádný směrovač by neměl mít více než 60 sousedů. Tabulka sousednosti je také ukládána v operační paměti směrovače a tudíž s velkým počtem sousedních směrovačů by dosahovala značné velikosti. Pokud však dodržíme první z uvedených pravidel, tedy nebudeme v rámci jedné oblasti používat více než 50 směrovačů, bude dodržena i tato podmínka.

Pokud nebude některá z výše uvedených podmínek splněna, nemůže být zaručena efektivnost vytvoření oblastí v rámci sítě.

Pokud používáme směrování v rámci více oblastí OSPF, setkáme se ve směrovací tabulce s dalšími typy záznamů, jelikož směrovač musí odlišit, které informace získal v rámci své oblasti a které byly získány z oblasti externí.

Ve směrovací tabulce se setkáme nově s těmito typy záznamů:

- **O*E2** (respektive O*E1) jsou cesty získané pomocí externích zpráv LSA z ostatních autonomních systémů.
- **O IA** označují mezioblastní cesty. Cesta mezi oblastmi představuje souhrnnou informaci o sítích, náležících jiné oblasti. Tyto souhrnné informace jsou získávány od hraničních směrovačů jednotlivých oblastí, které informují směrovače páteřní a ty získané informace šíří do oblastí zbývajících. Tato výměna probíhá v rámci jednoho autonomního systému.
- **O** znázorňuje cesty zjištěné v rámci jedné oblasti. Na konkrétním směrovači jsou to veškeré cesty zjištěné v rámci oblasti, do níž tento směrovač spadá.

Ať již využíváme oblast jednu, či oblastí vícero, musí docházet k výpočtům nejvýhodnějších

cest do jednotlivých sítí v rámci topologie. Tento proces označujeme jako spuštění **algoritmu SPF**, tedy vytvoření stromu nejkratších cest. Směrovač určuje cesty v následujícím pořadí:

- nejlepší cesty v rámci stejné oblasti,
- nejlepší cesty mezi jednotlivými oblastmi,
- nejlepší cesty k ostatním autonomním systémům.

Konfigurace probíhá naprosto obdobným způsobem, jako u směrování v jedné oblasti. Jediným rozdílem je nutnost předem zvážit, jaké bude rozčlenění oblastí v rámci topologie, což vyžaduje hlubší znalost nejen protokolu OSPF, ale také principů a technik návrhu architektury sítí. [1][6][7][8][11]

12.12 KONFIGURACE OSPF V RÁMCI VÍCE OBLASTÍ

Podrobná konfigurace na námi používané ukázkové topologii a to včetně ovlivnění RouterID je v příloze I.

13 OSPFV3

13.1 DŮVOD VZNIKU A ROZDÍLY OPROTI OSPFV2

Tato verze protokolu vznikla v návaznosti na protokol OSPFv2, který byl použitelný pouze v sítích využívajících směrovaný protokol IPv4. Nicméně vývoj jde zejména v rámci počítačových sítí velmi rychle kupředu a tak bylo potřeba reagovat na rozvíjející se protokol IPv6. Protokol OSPFv3 slouží k propagování směrovacích tabulek s využitím IPv6. Na směrovači je možné také využívat zároveň obě verze protokolu OSPF, ačkoliv je však princip fungování obdobný, běží pak každá z těchto verzí jako samostatný proces. Pokud na směrovači provozujeme zároveň obě verze protokolu OSPF, nazýváme tuto síť jako **dual-stacked**. [11][13]

Stejné vlastnosti obou protokolů jsou shrnuty v následujícím bloku.

- Protokoly OSPFv2 i OSPFv3 spadají do kategorie protokolů **stavu linky**, které jsou navíc beztřídní.
- Pro vytváření stromu nejkratších cest je využíván **algoritmus SPF**, který pomocí Dijkstrova algoritmu vyhledává nejkratší cesty do všech zjištěných uzlů.
- Metrika je v obou protokolech založená na hodnotě **ceny cesty**. Jelikož, jak víme z předchozího textu, je možné hodnotu této ceny měnit, dochází vždy pouze ke změnám v rámci jednoho procesu. Pokud tedy změníme pro určitou cestu, spadající do procesu OSPFv2, cenu, neovlivní tato změna nikterak proces s běžícím OSPFv3.
- Další společnou vlastností je u obou protokolů možnost využívání členění sítě do jednotlivých **oblastí.** O tom, jak síť korektně rozčlenit, či jak konfigurovat OSPFv2, pojednávaly předchozí kapitoly.
- OSPFv2 i OSPFv3 používají **stejné typy paketů.** V obou protokolech tak nalezneme hello pakety, DBD, LSR, LSU či potvrzovací LSAck.
- V prostředí obou protokolů dochází k aplikování **stejných rozhodovacích technik**, či stejnému **principu volby DR a BDR**. Nechybí zde však ani objevování nových sousedů pomocí paketů hello.
- Hodnota **router ID** je u obou protokolů o velikosti 32 b, obsahující čísla oddělená tečkou. U OSPFv2 i OSPFv3 platí, že tuto hodnotu lze měnit, či ji ponechat nastavenou na implicitní hodnotu.

Následující seznam přibližuje veškeré odlišnosti mezi OSPFv2 a OSPFv3

- Použitým směrovaným protokolem v rámci OSPFv2 je **IPv4,** zatímco u OSPFv3 je použit protokol **IPv6.**
- OSPFv3 **nepropaguje** v rámci oblastí **adresy sítí**. Propagování je prováděno pomocí IPv6 prefixů.

- Každý z protokolů využívá své **vlastní adresy pro odesílání informací**.
- **Konfigurace** probíhá u OSPFv3 odlišným způsobem, jenž bude popsán později.
- Podpora **unicast směrování** není implicitně aktivovaná jako u OSPFv2. U OSPFv3 musíme tuto možnost nejprve aktivovat.
- Zatímco v rámci protokolu OSPFv2 nebyla žádná větší podpora **autentifikace**, u protokolu OSPFv3 je podporována autentifikace na úrovni IPv6.

[10][11][13]

13.2 VOLBA ROUTER ID

Jak bylo řečeno v části o stejných i odlišných vlastnostech obou protokolů, můžeme hodnotu router ID nechat zvolit implicitně, či si ji manuálně upravit. Zajímavostí zde však je, že i když používáme směrovaný protokol IPv6, a tudíž tedy na směrovači běží proces OSPFv3, je hodnota router ID volena z nastavených adres typu IPv4. Jakým způsobem volba probíhá, bylo uvedeno v části Router ID u protokolu OSPFv2.

Jediným rozdílem zde však je, že směrovač nemusí mít nastavenou žádnou adresu rozhraní pomocí IPv4, a tudíž by nemohla být přiřazena korektní hodnota. V takovém případě IOS informuje uživatele následujícím výpisem.

```
%OSPFv3-4-NORTRID: OSPFv3 process 1 could not pick a router
ID, please configure manually.
```

Nyní je opět nutné využít ukázkovou topologii, na které je prakticky ukázána konfigurace protokolu OSPFv3 včetně ovlivnění volby RouterID.

13.3 KONFIGURACE OSPFV3

Tato problematika je umístěna v příloze J, včetně vysvětlení nejpodstatnějších částí kongfigurace a jejich vlivu na fungování směrovačů.

14 ÚVOD DO SVĚTA WAN SÍTÍ

14.1 CO JE TO WAN SÍŤ

WAN sítě, celým názvem Wide Area Network, slouží k propojování koncových stanic, obvykle na velkou vzdálenost. Samozřejmě nepropojují pouze koncové stanice, ale starají se o propojení lokálních, či metropolitních sítí, za účelem umožnění přenosu dat, videa či hlasu.

V České republice existuje mnoho WAN sítí, do nichž jsou napojeny firemní sítě velkých korporací, či univerzity. Nicméně tyto sítě mohou být soukromé, jelikož jsou vystavěny pouze pro specifickou firmu, či jsou budovány poskytovateli internetového připojení, jež tím umožňují propojení svých klientů. Pro akademickou půdu lze uvést sítě CESNET, či jeho předchůdce EARN.

Ovšem začátky WAN sítí u nás nebyly tak lehké, jelikož většina sítí nedisponovala potřebným vybavením a na svoji funkčnost kladly pouze minimální nároky. První síť, do níž bylo možné napojovat koncová zařízení, byla síť FIDO a k nám se dostala v březnu roku 1990. O dva měsíce později přichází další síť EUnet, jež sloužila pro propojení unixových počítačů.[5][9]

14.2 ZÁKLADNÍ VLASTNOSTI, METODY PŘIPOJENÍ

Následující souhrn přibližuje možné způsoby připojení sítím WAN:

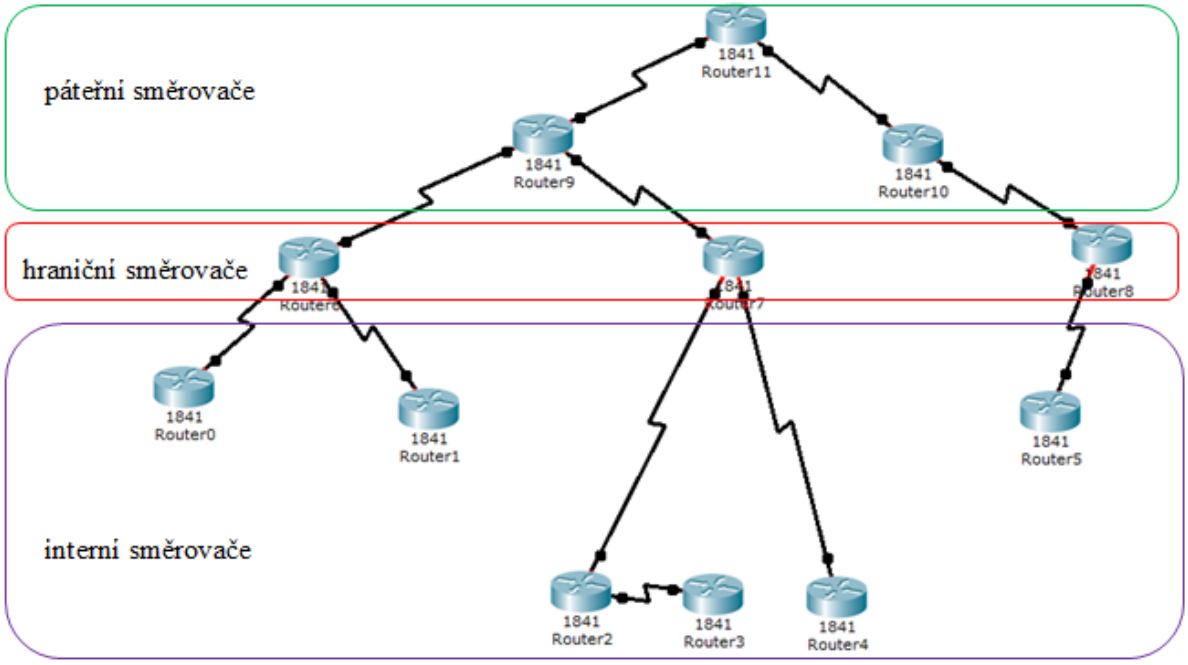
- **Pronajaté linky**, jejichž použití obvykle znamená vyšší cenu, jelikož samotným pronájem linky je velmi drahý.
- **Přepojování okruhů**, které vyhradí mezi dvěma koncovými zařízeními okruh, po němž následně probíhá komunikace. Jako příklad lze uvést vytáčené spojení.
- **Přepojování paketů** slouží k odesílání paketů v rámci trvalého virtuálního okruhu nebo může být využit přepínaný virtuální okruh.
- **Přepojování buněk**, jež je podobný předchozímu, avšak nepoužívá členění paketů, ale vytváří buňky o stejných velikostech.

Základní vlastnosti rozlehlých sítí WAN:

- Poskytují prostor pro **přenos** pomocí komunikace **se spojením**, ovšem nemusí být se nutně jednat o spolehlivý přenos.
- WAN **nepoužívají sdílené přenosové prostředky**. Jsou vzájemně propojeny.
- **Chybí** podpora **směrování na všeobecnou adresu**. Toto směrování můžeme porvést v rámci lokální sítě, avšak takto by mohlo docházet k záplavám ostatních zařízení v rámci LAN sítě a dalo by to prostor pro případné bezpečnostní útoky.
- Síť WAN **nepodporuje žádné uživatelské aplikace**, pouze pro tyto aplikace poskytuje přenosové prostředí.

V tak rozsáhlé síti, jako je síť WAN, jež obsahuje stovky i tisíce směrovačů a koncových zařízení je nemožné, aby každý směrovač znal údaje o ostatních zařízeních v rámci celé WAN sítě. Z tohoto důvodu je použita hierarchie směrovačů, v nichž každý směrovač zná informace pouze o určité množině směrovačů ostatních.[5]

Následující obrázek přibližuje hierarchii členění směrovačů v rámci WAN sítě.



Obrázek 68 WAN členění směrovačů v rámci rozsáhlé sítě

Pozn.: Při studiu se můžeme setkat i s označením brány, čili například místo páteřní směrovače je použit pojem páteřní brány. Oba pojmy jsou správné a autoři vzdělávacích materiálů, či odborných publikací budou čtenáře informovat o tom, jaký pojem budou používat.

14.3 JE SÍŤ WAN EFEKTIVNÍ?

Stejně tak jako u sítí typu LAN, můžeme i WAN sítě porovnávat podle nejrůznějších aspektů, mezi tyto aspekty řadíme:

- doba, po níž je zařízení schopno být v provozu,
- míra objemu síťového provozu,
- doba zpoždění při odesílání paketů,
- efektivnost využití síťových prostředků.

S většinou těchto kritérií jsme se již setkali při určování efektivnosti LAN sítě. Novým aspektem, jenž je ve WAN síti důležitý, je doba provozuschopnosti síťového zařízení. Toto

kritérium je důležité vzhledem k rozsahu zařízení, jež jsou závislé na konkrétním síťovém zařízenié. Výpadek takového zařízení by mohl znamenat zhroucení celé části sítě. Z tohoto důvodu jsou v sítích WAN voleny zažízení, jež mohou být v provozu takzvaně 24x7, což znamená 24 hodin 7 dní v týdnu. Samozřejmě u těchto zařízení musí být kladeny speciální nároky na chlazení i na prostory, v nichž jsou umístěny, jelikož umístění či nesprávná údržba a chlazení by mohly ovlivnit výkon daného zařízení.

Pokud se rozhodneme provozovat síť typu WAN, musíme mít na paměti, že oproti síti LAN představuje tento typ sítě značné finanční nároky na počáteční zřízení a později na pravidelné výdaje představující úhrady za pronájem linky. Do počátečních nákladů řadíme mimo finanční částky za pronájem linky i vyšší pořizovací ceny zařízení, jež jak bylo uvedeno, musí splňovat vyšší hardwarové nároky. Také prostory v nich budou zařízení umístěna, musí splňovat důraznější požadavky a tudíž i jejich provoz, jako například klimatizování, bude znamenat finanční zatížení.[5][9]

14.4 WAN A VRSTVY ISO/OSI

V předchozí části této kapitoly byly uvedeny základní vlastnosti sítí typu WAN. Nyní rozšiřme tyto vlastnosti ještě o jeden poznatek, a to, že WAN sítě a všechny jejich protokoly pracují na linké a fyzické vrstvě referenčního modelu ISO/OSI. První dvě vrstvy modelu ISO/OSI mají následující rozdělení činností:

- **Fyzická vrstva** je zodpovědna za doručovací metody, mezi které řadíme kupříkladu FrameRelay či ATM. Prvky fyzické vrstvy představují spojení mezi firmou, jenž provozuje síť typu WAN a poskytovatelem připojení k internetu, který ji pronajímá své vysílací pásmo.
- **Linková vrstva** se stará o zapouzdření a správnou adresaci rámců.

[5]

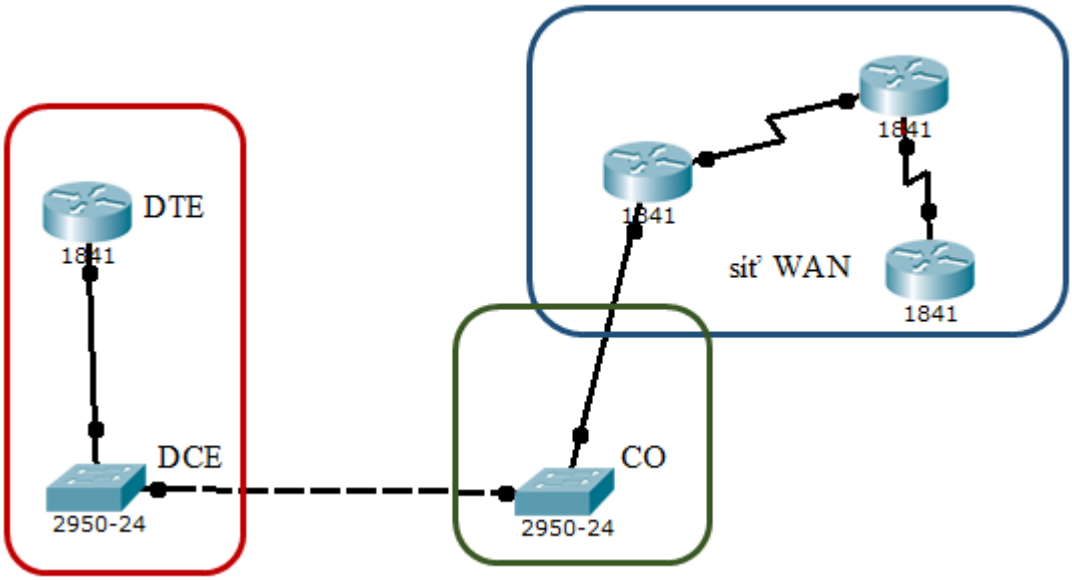
14.4.1 WAN na fyzické vrstvě

Jak bylo řečeno v předchozí části této kapitoly, dochází na fyzické vrstvě ke spojení mezi poskytovatelem a zákazníkem. V souvislosti s tímto spojením jsou používány následující pojmy a označení:

- **CPE** (customer premises equipment), jenž představuje zařízení ve vlastnictví zákazníka nebo zákazníkem pronajaté od poskytovatele připojení. Pro takovéto zařízení platí, že je napojeno na spojový kanál a tudíž je možné skrze něj využívat WAN síť.
- **CO** (central office) je označením pro sídlo poskytovatele.
- **Demarcation point** představuje hranici, rozdělující zařízení zákazníka a poskytovatele. Tedy jedná se o poslední směrovač, jenž je ve vlastnictví zákazníka. Další přeskok je již vlastnictvím poskytovatele.
- **DCE** (data circuit terminating equipment) představuje zařízení, jenž slouží jako DCE koncovka sériového spoje. Hlavním úkolem je tedy generování hodinového signálu (clock-rate) a dále se stará o odesílání dat do WAN sítě.
- **DTE** (data terminal equipment) je takové zařízení, které data neodesílá, ale pouze je připravuje na odeslání. Typickým zástupcem DTE zařízení je směrovač.
- **Poslední míle** (last mile či local loop) zastupuje připojení uživatele k poskytovateli.
- **CSU/DSU** (channel service unit/data service unit) označuje zpracování digitálního signálu. CSU se stará o zpracování přijatého signálu, zatímco DSU naopak převádí data do formátu používaného přenosovým prostředkem (světlo, …).

[5][9]

Následující obrázek přibližuje uvedené pojmy.



Obrázek 69 WAN shrnutí pojmů

Na fyzické vrstvě je používáno mnoho standardů a protokolů, jmenujme kupříkladu:

- V.35
- ISDN
- EIA/TIA-232

Tyto protokoly slouží k umožnění vzájemné komunikace na úrovni elektrických parametrů či signálů.

[5]

14.4.2 WAN na linkové vrstvě

Protokoly použité na linkové vrstvě referenčního modelu ISO/OSI se starají o navázání spojení s koncovým zařízením. Mezi jejich úkoly patří zejména definování formátu zapouzření a dále upřesnění mechanismů, jež budou použity pro přenos rámců.

Mezi nejrozšířenější protokoly linkové vrstvy používané ve WAN síti řadíme:

- **HDLC** (High-Level Data Link Protocol) představuje protokol, který podporuje synchronní komunikaci v plně duplexním režimu, jež může být jednobodová (point to point) či vícebodová (point to multipoint). HDLC je předlohou mnoha dalších spojových protokolů, jako příklad lze jmenovat protokol PPP. Protokol HDLC rozlišuje komunikující stanice jako primární a sekundární.
- **PPP** (Point to Point Protocol) je používán pro přímé spojení mezi dvěma uzly. Samotný protokol se skládá ze dvou vrstev, a to **link control protocol** a **network control protocol**. První zmiňovaný slouží k navázání spojení, jakmile je však spojení navázáno, je automaticky použit jeden nebo více protokolů typu NCP. Protokoly NCP se nestarají o navázání spojení, ale starají se o samotný přenos dat. LCP označujeme jako protokol řízení spoje a protokol NCP lze v češtině označit jako protokol řízení sítě. Protokol PPP umožňuje také šifrování a autentizaci pomocí protokolů **PAP** (Password Authentication Protocol) a **CHAP** (Challenge Authentication Protocol). Jako další vlastnost protokolu PPP lze zařadit nerozeznávání primárního a sekundárního typu stanice, a tudíž může komunikaci začít libovolná ze stanic.
- **Frame relay** je starším zástupcem protokolů spojové vrstvy ve WAN sítích. Frame relay dovoluje efektivnější využívání přenosové rychlosti WAN. Toto efektivní využití je umožněno díky interaktivnímu přizpůsobení pronajímané šířky pásma pro jednotlivé síťové aplikace.

[5][9]

ZÁVĚR

Cílem mé bakalářské práce bylo vytvoření výukových podkladů pro wiki projekt počítačových sítí. V bakalářské práci jsem se opírala zejména o strukturu nových výukových materiálů od společnosti CISCO, jež jsou v současné době základem k pochopení a objasnění principů v počítačových sítích.

Celá práce byly strukturovány do tří logicky navazujících částí tak, aby pokryly široké spektrum informací, jež jsou nezbytné pro pochopení celého fungování směrování v sítích.

První část práce se zaměřovala na obecný pohled na směrovač, jeho strukturu a využití v rámci počítačové sítě. Čtenářům byl vysvětlen základní princip, jakým směrovač zpracovává přijaté pakety a zároveň byla přiblížena vnitřní struktura tohoto zařízení. Tato část obsahovala nadstavbu oproti základnímu kurzu CCNA v podobě bližšího pohledu na jednotlivé typy rozhraní směrovače a zároveň na jejich podrobnější funkčnost.

Druhá část práce byla směřována k pochopení směrovací tabulky a procesu vyhledávání. V této části jsem se zaměřovala zejména na co nejlepší způsob zapamatování celého principu a struktury. Pro splnění tohoto bodu jsem využívala paralelu k vyhledávání ve slovníku či obecná schémata směrovací tabulky.

Poslední a nejvíce rozsáhlá část práce přibližovala směrovací protokoly a jejich nasazení v počítačových sítích. V této části výukové materiály rozebíraly podrobný popis fungování směrovacích protokolů, včetně algoritmů síťového grafu. Oproti základnímu kurzu CISCO zde byla uvedena nadstavba pro zájemce, kteří by chtěli studovat na navazujícím studiu. Tito zájemci by měli mít povědomí o použitých algoritmech síťového grafu a měli by také umět tyto znalosti aplikovat na libovolný síťový graf.

V době odevzdávání práce je text postupně implementován na portál wiki.upce.cz kde jsou texty volně dostupné všem studentům Univerzity Pardubice.

POUŽITÉ ZDROJE

[1] SPORTACK, Mark A. *Směrování v sítích IP*. Vyd. 1. Brno: Computer Press, 2004, 351 s. ISBN 80-251-0127-4.

[2] BŘEHOVSKÝ, Petr. *Praktický úvod do TCP/IP*. 1. vyd. České Budějovice: KOPP, 1994, 107 s. ISBN 80-858-2818-9.

[3] HORÁK, Jaroslav a Milan KERŠLÁGER. *Počítačové sítě pro začínající správce*. 5., aktualiz. vyd. Brno: Computer Press, 2011, 303 s. ISBN 978-80-251-3176-3.

[4] JIROVSKÝ, Václav. *Vademecum správce sítě*. 1. vyd. Praha: Grada, 2001, 428 s. ISBN 80-716-9745-1.

[5] PUŽMANOVÁ, Rita. *Moderní komunikační sítě od A do Z*. 2. aktualiz. vyd. Brno: Computer Press, 2006, 430 s. ISBN 80-251-1278-0.

[6] ODOM, Wendell. *Počítačové sítě bez předchozích znalostí*. Vyd. 1. Brno: CP Books, 2005, 383 s. ISBN 80-251-0538-5.

[7] ODOM, Wendell, Rus HEALY a Naren MEHTA. *Směrování a přepínání sítí: autorizovaný výukový průvodce*. Vyd. 1. Brno: Computer Press, 2009, 879 s. ISBN 978-80-251-2520-5.

[8] WHITE, Russ. *Optimal routing design*. Indianapolis: Cisco Press, c2005, xx, 484 s. ISBN 15-870-5187-7.

[9] *CCNA exploration accessing the WAN, version 4.0 : course booklet*. Indianapolis, IN: Cisco Press, c2010, xii, 355 p. Cisco Networking Academy Program series. ISBN 15-871-3255-9.

[10] EMPSON, Scott. *CCNA kompletní přehled příkazů: autorizovaný výukový průvodce*. Vyd. 1. Brno: Computer Press, 2009, 336 s. ISBN 978-80-251-2286-0.

[11] BILL, Chapman. *CCNA 1 and 2 Companion Guide*. Vyd. 1. Indianapolis: Cisco Press, 2004, 1043 s. ISBN 15-871-3150-1.

[12] PUŽMANOVÁ, Rita. *TCP/IP v kostce*. 2. upr. a rozš. vyd. České Budějovice: Kopp, 2009, 619 s. ISBN 978-80-7232-388-3.

[13] LAMMLE, Todd. *CCNA: výukový průvodce přípravou na zkoušku* 640-802. Vyd. 1. Brno: Computer Press, 2010, 928 s. ISBN 978-802-5123-591.

[14] MERANI, Maria Luisa, Maurizio CASONI a Walter CERRONI. *Hands-on networking: from theory to practice*. New York: Cambridge University Press, 2009, xii, 259 p. ISBN 05-218-6985-4.

[15] LU, Linyuan a Fan R CHUNG. *Complex graphs and networks*. Providence, RI: American Mathematical Society, c2006, vii, 264 p. Regional conference series in mathematics, no. 107. ISBN 08-218-3657-9.
[16] VOLEK, Josef a Bohdan LINDA. *Teorie grafů - aplikace v dopravě a veřejné správě*. Vyd. 1. Pardubice: Univerzita Pardubice, 2012, 190 s. ISBN 978-80-7395-225-9.

PŘÍLOHA A NEJPOUŽÍVANĚJŠÍ PŘÍKAZY NA PŘÍKLADECH

Uveďme zde přehled nejzákladnějších příkazů, které jsou využívány při konfiguraci směrovače.

Rozlišujeme čtyři základní režimy, ve kterých se může uživatel nacházet. Hlavními rozdíly jsou zejména v možnostech vykonávání příkazů, které jsou validní pouze ve specifikovaném módu.

Následující podkapitoly shrnují rozdělení příkazů dle režimů práce s IOS.

Pozn.: při práci s IOS se nebojte využívat tabelátor pro doplňování názvů, otazník pro zobrazení možností na pokračování příkazu, či pokud patříte mezi zkušenější uživatele, psaní příkazů zkratkovitě.

Např. místo příkazu

```
R# show ip interface brief
```

lze zapsat

```
R# sh ip int br
```

a směrovač i v tomto případě provede příslušný příkaz.

Na úrovni uživatelského režimu nelze aplikovat ochranu systému pomocí hesla. Teprve přechod do režimu privilegovaného může být zaheslována, tedy jednoduše do tohoto režimu se dostane každý.

Tento režim poznáte dle způsobu zápisu, který vypadá následovně

```
R>
```

V tomto režimu máme dále možnost provádět **kontrolu konfigurace a konektivity** sítě. Pro tyto funkce je však vhodné provést přechod do režimu privilegovaného. Tento přechod je umožněn příkazem **enable.**

Jeho použití proběhne následovně:

```
R>
```

```
R>enable
```

```
R#
```

Jak vidíme, v tomto případě by se jakýkoliv nepovolaný uživatel dostal do IOS a mohl by síť, ať již úmyslně či neúmyslně poškodit. Z tohoto důvodu je nanejvýše vhodné pro přechod do privilegovaného režimu využívat heslo, stejně tak, jak je tomu na osobních počítačích.

```
R# configure terminal
R (config)# enable secret <heslo>
R (config)# login
```

Nejprve byl proveden přechod do konfiguračního režimu a následně máme dvě možnosti zadání hesla. Prvním, avšak běžně nepoužívaným je

```
R (config)# enable password <heslo>
```

U této varianty se bohužel setkáváme s velmi **snadným prolomením** našeho hesla. Pokud použijeme tento příkaz, je heslo uloženo zcela nešifrované, a tudíž může být velmi lehce zjištěné nepovolanou osobou. Z tohoto důvodu je bezpečnější využívat variantu

```
R (config)# enable secret <heslo>
```

Která heslo nejprve zašifruje a teprve následně uloží do systému. Dalším příkazem pro zašifrování privilegovaného režimu je příkaz **login**. Pokud bychom tento příkaz nezadali, měli bychom režim opatřen heslem, avšak by systém **nevyžadoval přihlášení.**

Nepsaným pravidlem, avšak spíše dobrou zvyklostí je uživatele informovat o nutnosti přihlášení. Cisco IOS nám umožňuje informování uživatele pomocí takzvané **zprávy dne.** Tuto zprávu provedeme následujícím příkazem

```
R (config)# banner motd #text zobrazován uživateli #
```

Vždy, když provedeme konfiguraci, ať již dynamického nebo statického směrování, měli bychom provést kontrolu dostupnosti v síti. K tomuto účelu používáme příkaz **ping.** Standardně je prováděn z privilegovaného režimu, či přímo z příkazového řádku počítače. Ping pracuje na základě protokolu ICMP[5].

[5] Internet control message protokol, slouží k odesílání chybových zpráv.

Celý princip ICMP je založen na komunikaci pomocí dvou zpráv, a to:

- echo request,
- echo reply.

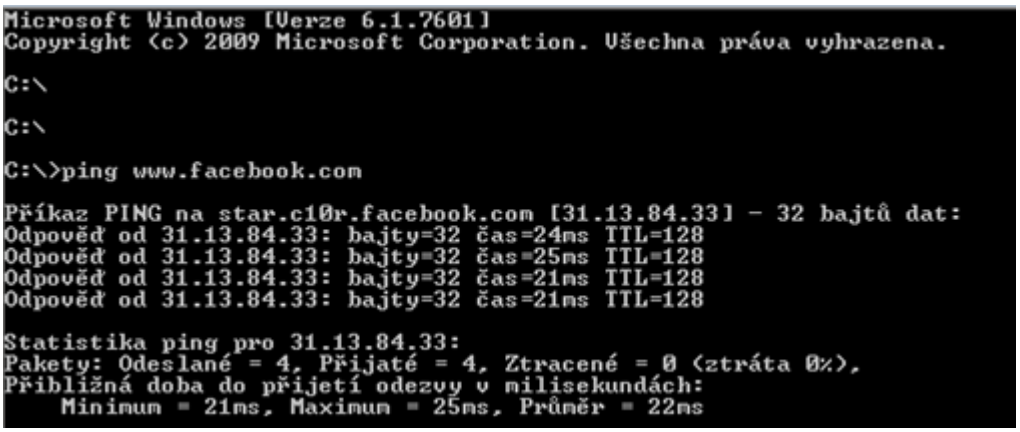
Směrovač vyšle žádost do cílového zařízení, to jej příjme a odešle na zpět odpověď znamenající, že cílové zařízení je dostupné. Takto proběhne série výměn zpráv. Obvyklá doba, po kterou odesílající zařízení čeká na přijetí odpovědi je zhruba tři sekundy, ale může se lišit.

Následně je vypsána statistika jednotlivých dvojic zpráv, tudíž je možné identifikovat, při které žádosti došlo k chybě. Tento souhrn zároveň uvádí dobu zpoždění, tedy dobu, za níž k výměně došlo. Příkaz ping provedeme následovně z jakéhokoliv režimu.

```
R1>ping 192.168.20.53

R1#ping 192.168.20.53

R1(config)#do ping 192.168.20.53

R1(config-if)#do ping 192.168.20.53
```

První dva výše uvedené způsoby nikoho nepřekvapí, zatímco u dalších přibylo slovíčko **do**. Díky tomuto přidání do zadání příkazu ping můžeme provádět ping i v režimech, ve kterých by za běžné situace nešel. Samozřejmě tomu tak není pouze u tohoto příkazu, ale u libovolného, který je dostupný pouze v privilegovaném, či uživatelském režimu.

Parametrem tohoto příkazu je **adresa cílového zařízení**, jehož dostupnost chceme ověřit. Je nutné podotknout, že není podmínkou zadávat pouze IP adresy, lze zadat i doménové jméno. Pokud však využijeme druhou variantu, je název nejprve přeložen pomocí serveru DNS na IP adresu a následně je provedena kontrola dostupnosti. Výsledný souhrn je identický pro oba způsoby. Následující obrázek ilustruje příkaz ping provedený z počítače na adresu www.facebook.com.



Obrázek 70 Příkaz ping na server www.facebook.com

Jak můžete vidět z obrázku, byl nejprve nalezen server, na němž se webová stránka nachází a následně byla určena jeho IP adresa. Lze tedy z výpisu vyčíst, že webová stránka www.facebook.com se nachází na serveru star.c10r.facebook.com s IP adresou 31.13.84.33, na níž jsme vyslali čtyři žádosti o potvrzení dostupnosti. Jednotlivé výpisy symbolizují pokusy o potvrzení. Vidíme, že konektivita zde dosahuje 100%. Poslední řádek zobrazuje dobu nejkratší, nejdelší a průměrné odezvy od tohoto serveru.

Při výpisu se můžeme setkat s těmito variantami:

- Kompletní výpis jednotlivých pokusů včetně velikosti a doby trvání.
- Informace o stavu, kdy cílový host není dostupný, nebo vypršel časový limit žádosti.

První uvedený je výpis, jenž signalizuje kompletní komunikaci, zatímco druhý znamená, že v síti došlo k chybě nebo je cílové zařízení nedostupné. V takovémto případě snad lze jen doporučit **důkladnou kontrolu** provedené **konfigurace**.

Pokud nastane v síti problém, můžeme použít příkaz **traceroute**, jenž poskytuje komplexnější pohled na průchod sítí, než ping. Zatímco ping zobrazoval pouze odezvu jednotlivých pokusů o komunikaci, zobrazuje traceroute seznam průchodů uzlů v síti.

```
R3#traceroute 192.168.20.17
```

Který zobrazí jednotlivé uzly následovně.

```
C:\Users\muffina007>tracert www.facebook.con

Tracing route to star.c10r.facebook.con [31.13.81.81]
over a maximun of 30 hops:

  1    10 ms     8 ms     8 ms  ip-89-177-162-1.net.upcbroadband.cz [89.177.162.
1]
  2     6 ms     7 ms     7 ms  ip-81-27-192-129.net.upc.cz [81.27.192.129]
  3    16 ms    16 ms    16 ms  cz-kol2-pop11-ra2-vla2133.net.upc.cz [84.116.221
.133]
  4    17 ms    17 ms    18 ms  cz-kol2-pop11-ra1-vla2132.net.upc.cz [84.116.221
.129]
  5    17 ms    17 ms    16 ms  cz-prg02a-ra2-vla2131.net.upc.cz [84.116.221.125
]
  6    38 ms    17 ms    16 ms  213.46.172.221
  7    17 ms    16 ms    17 ms  84.116.135.1
  8    17 ms    19 ms    16 ms  84.116.133.97
  9    17 ms    16 ms    16 ms  xe-1-1-1.pr01.fra2.tfbnw.net [103.4.96.53]
 10     *        *        *     Request timed out.
 11    18 ms    16 ms    18 ms  ae0.pr02.fra2.tfbnw.net [31.13.27.208]
 12    17 ms    16 ms    16 ms  po126.msw01.06.fra2.tfbnw.net [74.119.77.79]
 13    18 ms    16 ms    16 ms  edge-star-shv-06-fra2.facebook.com [31.13.81.81]

Trace complete.
```

Obrázek 71 Příkaz tracert na server www.facebook.com

Vidíme, že výsledek je stejný, avšak vidíme celou cestu paketu v rámci sítě[6].

Další velkou skupinou příkazů jsou příkazy typu **show**. S nimi se setkáváme zejména tehdy, pokud kontrolujeme směrovací tabulku, stavy jednotlivých rozhraní, verzi systému,…

Pokud přijdeme poprvé do firmy či laboratoře, využívající směrovače firmy Cisco, měli bychom se nejprve podívat na informace o používané **verzi IOS**. Tyto informace si vypíšeme následujícím příkazem.

```
R3#show version
```

Následuje poněkud obsáhlejší výpis informací, ze kterého lze například zjistit, odkud je načítán obraz IOS či jeho verze.

Často potřebujeme také zjistit, v jakém stavu jsou rozhraní na směrovači. Tento stav je možné zjistit příkazem

```
R3#show ip interface brief
```

Jenž nám vypíše přehlednou tabulku rozhraní směrovače, současně s nastavenou IP adresou a aktuálním stavem.

[6] V operačním systému Windows se nesetkáváme s příkazem traceroute, ovšem s jeho kratším označením tracert

Znázorněme si zjednodušeně získanou tabulku

Tabulka 14 Obecné schéma výpisu rozhraní

Interface	IP-Address	OK?	Method	Status	Protocol
Zobrazuje název rozhraní.	Nastavená IP adresa pro rozhraní.	YES NO	NVRAM manual …	Up Down Administratively down	Up down

První dvě pole zobrazují pouze informaci o **označení** rozhraní a jeho nastavené **adrese**. Třetím polem je **OK?** signalizuje validitu použité IP adresy. O způsobu, jakým bylo konkrétní rozhraní nastaveno, nás informuje položka **Method**, jež může nabývat celé množiny různých hodnot. Nejčastěji se setkáme s hodnotou *manual*, která symbolizuje ruční konfiguraci pomocí příkazové řádky, či *unset* pro rozhraní, která nebyla doposud konfigurována. Dalším polem, které využíváme zejména při hledání chyb v síti je pole **Status**. Zde se můžeme setkat se *třemi hodnotami*. Jejich význam si předvedeme na následující topologii.

Obrázek 72 Topologie ve výchozím nastavení

Tuto topologii tvoří dva směrovače propojené sériovým kabelem. Jako DCE byl zvolen port se0/1/0 na směrovači A, zatímco DTE port se0/1/0 na směrovači B.

Následně byla provedena konfigurace směrovače A.

```
R (config) # interface se0/1/0
R (config-if)# ip address 10.10.10.1 255.255.255.252
R (config-if)# clock rate 64000
```

```
R (config-if)#no shutdown
```

Na směrovači B byla provedena konfigurace následovně.

```
R (config) # interface se0/1/0
R (config-if)# ip address 10.10.10.2 255.255.255.252
R (config-if)#no shutdown
```

Provedeme výpis rozhraní na obou směrovačích. První obrázek představuje výpis ze směrovače A, zatímco druhý provedení stejného příkazu na směrovači B.

```
Interface              IP-Address      OK? Method Status                Protocol

FastEthernet0/0        unassigned      YES unset  administratively down down
FastEthernet0/1        unassigned      YES unset  administratively down down
Serial0/1/0            10.10.10.1      YES manual up                    up
Serial0/1/1            unassigned      YES unset  administratively down down
```

Obrázek 73 Zobrazení stavu rozhraní na směrovači A

```
Interface              IP-Address      OK? Method Status                Protocol

FastEthernet0/0        unassigned      YES unset  administratively down down
FastEthernet0/1        unassigned      YES unset  administratively down down
Serial0/1/0            10.10.10.2      YES manual up                    up
Serial0/1/1            unassigned      YES unset  administratively down down
```

Obrázek 74 Zobrazení stavu rozhraní na směrovači B

Jelikož konfigurace proběhla v pořádku a vše je nastaveno korektně, zobrazí se v poli status u obou směrovačů hodnota **up.** Tedy up značí **správně** nakonfigurované rozhraní.

Zkusme na směrovači B rozhraní se0/1/0 vypnout.

```
R (config-if)# shutdown
```

Jak se nám změnila tabulka rozhraní? V poli status bude zobrazena hodnota **administratively down**. Tato hodnota nám říká: „rozhraní bylo nakonfigurováno, avšak nebylo zapnuto.“

```
Interface              IP-Address      OK? Method Status                Protocol

FastEthernet0/0        unassigned      YES unset  administratively down down
FastEthernet0/1        unassigned      YES unset  administratively down down
Serial0/1/0            10.10.10.2      YES manual administratively down down
Serial0/1/1            unassigned      YES unset  administratively down down
```

Obrázek 75 Zobrazení rozhraní na směrovači B po vypnutí rozhraní

Poslední hodnotou, jenž může být uvedena v poli status je **down.** Down se objevuje tehdy, pokud nastane nějaký problém v síti. Rozhraní není v tomto případě vypnuto administrátorem, ale vlivem okolností. Zkusme v naší topologii jednoduše odebrat spoj mezi směrovači, čímž simulujeme výpadek v síti.

Obrázek 76 Topologie představující výpadek v síti

V tabulce se nám tato změna projeví následovně.

```
Interface              IP-Address      OK? Method Status                Protocol

FastEthernet0/0        unassigned      YES unset  administratively down down
FastEthernet0/1        unassigned      YES unset  administratively down down
Serial0/1/0            10.10.10.5      YES manual down                  down
Serial0/1/1            unassigned      YES unset  administratively down down
```

Obrázek 77 Zobrazení rozhraní po odebrání linky

Poslední položkou v tabulce výpisu rozhraní je **Protocol**. Tato položka odráží stav protokolu běžícího na rozhraní. Tedy pokud je rozhraní vypnuto, ale nastaveno, protokol na rozhraní také nepracuje. Pokud je spoj přerušen, pak také nemůže pracovat.

Pokud chceme zobrazit konfiguraci pouze konkrétního rozhraní, pak je vhodné využít příkaz

```
R # show interfaces se0/1/0
```

který zobrazí podrobné informace o daném rozhraní. Z výpisu se lze dozvědět mimo adresy a stavu také **druh zapouzdření** či **statistické informace** o průběhu odesílání paketů skrze toto rozhraní.

Posledním ze skupiny zobrazovacích příkazů je výpis směrovací tabulky. Provedeme jej následujícím příkazem

```
R # show ip route
```

Dalším, avšak více nepřehledným způsobem kontroly konfigurace je možnost spuštění režimu **debug**. Pokud programujete, tak jistě víte, že tento pojem doslova znamená *ladění.* Stejně tak, jak je tomu u programování, kdy kontrolujeme chyby v kódu, kontrolujeme i v počítačové síti vzniklé chyby. Pomocí zapnutí tohoto režimu lze vysledovat veškeré operace probíhající na směrovači. Pokud se rozhodneme pro komplexní výpis všech operací probíhajících na směrovači, musíme mít na paměti, že tím výrazně **zatěžujeme** a také **zpomalujeme** činnost směrovače. Stejně tak jako v počítači, kdy každý proces musí mít přidělen procesor, musí být procesor přidělován i ve směrovači. Implicitně jsou výpisy vypnuté a doporučuje se zapínat je pouze v případě **odhalování chyb v síti**. Výpis zapínáme a vypínáme vždy pro konkrétní proces.

```
R #debug ip rip
R # undebug ip rip
```

IOS nás informuje při zapnutí

```
RIP protocol debugging is on
```

a při vypnutí

```
RIP protocol debugging is off
```

Dalšími příkazy, které administrátor běžně používá, jsou **konfigurační příkazy.** S nimi jsme se již setkali v úvodní části této kapitoly, při nastavování hesel. Aby bylo uživateli umožněno konfigurovat směrovač a jeho rozhraní, musí přejít do **konfiguračního režimu.** Do tohoto režimu přejdeme následujícím příkazem

```
R # configure terminal
```

Úspěšný vstup do tohoto režimu signalizuje změna výpisu na terminálu.

```
R (config)#
```

Chceme-li ukončit práci v určitém režimu, využijeme příkazu exit. Tento příkaz nás vrátí vždy o úroveň zpět.

```
R # conf term
R (config)# exit
R #
```

V tomto režimu máme celou škálu možných příkazů, které lze provádět. Je dobrým zvykem si vždy směrovač nejprve **pojmenovat**. Správné pojmenování směrovače nám usnadní následnou konfiguraci směrovače.

```
R (config)# hostname A
A (config)#
```

Pokud potřebujeme konfigurovat jednotlivá rozhraní, musíme nejprve přejít do **konfiguračního režimu rozhraní**. Tento přechod provádíme v závislosti na rozhraní, které chceme nastavovat.

Při konfiguraci rozhraní nejčastěji chceme nastavit příslušnou IP adresu. Jako příklad uveďme konfiguraci IP adresy na rozhraní se0/1/0 .

```
R (config) # interface se0/1/0
R (config-if)# ip address 10.10.10.2 255.255.255.252
R (config-if) # description spoj s1
R (config-if)#no shutdown
```

O úspěšném aktivování rozhraní nás příkazová řádka ihned informuje.

Pozn.: někdy, zejména při konfiguraci sériových portů, nemusí být rozhraní nutně ihned zapnuto, jelikož je očekávána konfigurace druhé strany a teprve poté se zapnou obě rozhraní současně.

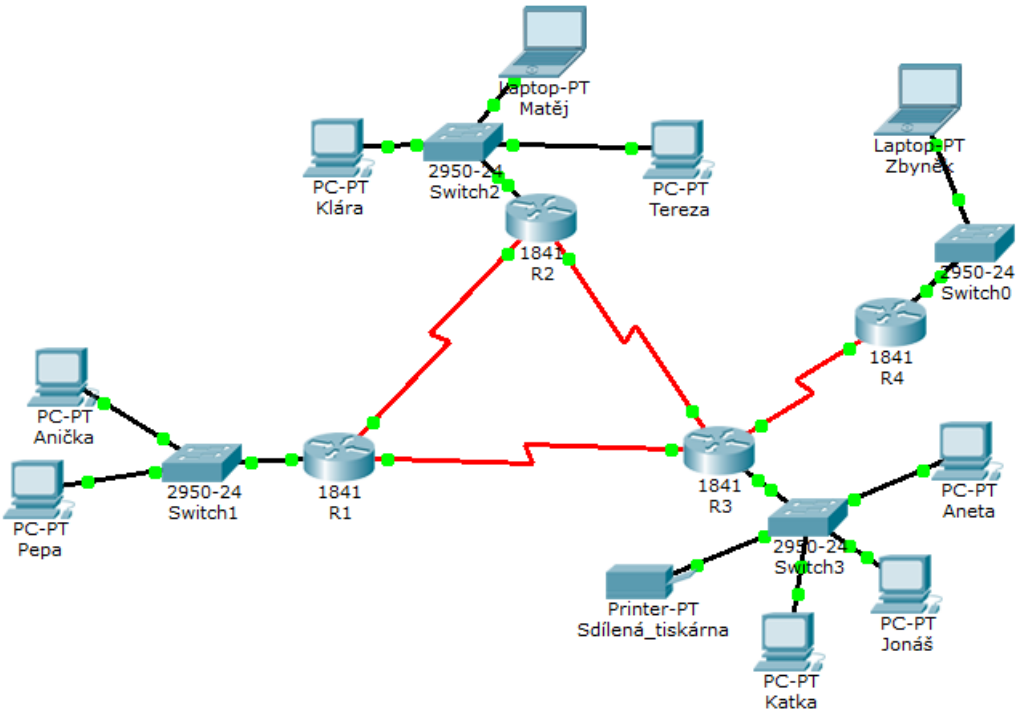
Po provedení konfigurace je vhodné změny uložit do **počáteční konfigurace** směrovače. K tomu využijeme příkaz

```
R# running-config startup-config
```

V některých případech, jako například u změny hodnot Router ID potřebujeme směrovač restartovat. K restartování směrovače slouží následující příkaz.

```
R# reload
```

PŘÍLOHA B VÝCHOZÍ MODELOVÁ TOPOLOGIE



PŘÍLOHA C SUMARIZACE NAD IPV4 A IPV6

Název sítě	Adresa
A	10.15.0.0/27
B	10.20.0.0/27
C	10.25.0.0/27
D	10.30.0.0/27

(odpověď 10.0.00/11)

Název sítě	Adresa
A	192.168.16.0/27
B	192.168.70.0/30
C	192.168.1.0/30

(odpověď 192.168.0.0/17)

Název sítě	Adresa
A	10.0.0.0/16
B	10.1.0.0/16
C	10.2.0.0/16
D	10.3.0.0/16
E	10.4.0.0/16

(odpověď 10.0.0.0/8)

Síť	Adresa
A	2001:DB8:ACAD:1000::/64
B	2001:DB8:ACAD:2000::/64
C	2001:DB8:CC1E:1::/64

(odpověď 2001:DB8:8::/33)

Síť	Adresa
A	2001:CC1E:2AB3:1A3C::/64
B	2001:CC1E:2AB3:1A4D::/64

(odpověď 2001:CC1E:2AB3:1A00::/57)

PŘÍLOHA D KONFIGURACE STATICKÉHO SMĚROVÁNÍ

Při konfiguraci statického směrování si vystačíme s tímto **jediným** příkazem:

```
R (config)# ip route <adresa_sítě> <maska_sítě> <odchozí_rozhraní>
```

Nezapomínejme, že cesta musí být nakonfigurována na každém ze směrovačů a vždy musíme zkontrolovat, zda námi navržené cesty nevytváří **smyčku**. Vždy je nejprve vhodné navrhnout si směr, kterým budou pakety procházet při cestách do koncových sítí.

Konfiguraci statických cest u modelové topologie představují následující příkazy.

```
R1> enable
R1# configuration terminal
R1 (conf)# ip route 192.168.20.0 255.255.255.240 se0/0/1
R1 (conf)# ip route 192.168.20.32 255.255.255.248 se0/0/1
R1 (conf)# ip route 192.168.20.24 255.255.255.248 se0/0/1
R1 (conf)# ip route 192.168.20.44 255.255.255.252 se0/0/1
R1 (conf)# ip route 192.168.20.52 255.255.255.252 se0/0/1

R2> enable
R2# configuration terminal
R2 (conf)# ip route 192.168.20.16 255.255.255.248 se0/1/0
R2 (conf)# ip route 192.168.20.0 255.255.255.240 se0/1/0
R2 (conf)# ip route 192.168.20.32 255.255.255.248 se0/1/0
R2 (conf)# ip route 192.168.20.40 255.255.255.252 se0/1/0
R2 (conf)# ip route 192.168.20.52 255.255.255.252 se0/1/0
R3> enable
R3# configuration terminal
R3 (conf)# ip route 192.168.20.16 255.255.255.248 se0/1/0
R3 (conf)# ip route 192.168.20.32 255.255.255.248 se0/1/0
```

```
R3 (conf)# ip route 192.168.20.24 255.255.255.248 se0/1/0
R3 (conf)# ip route 192.168.20.48 255.255.255.252 se0/1/0

R4> enable
R4# configuration terminal
R4 (conf)# ip route 192.168.20.16 255.255.255.248 se0/1/0
R4 (conf)# ip route 192.168.20.0 255.255.255.240 se0/1/0
R4 (conf)# ip route 192.168.20.24 255.255.255.248 se0/1/0
R4 (conf)# ip route 192.168.20.40 255.255.255.252 se0/1/0
R4 (conf)# ip route 192.168.20.44 255.255.255.252 se0/1/0
R4 (conf)# ip route 192.168.20.48 255.255.255.252 se0/1/0
```

Pozn.: V této ukázkové konfiguraci **nebyla provedena sumarizace**, i když by v tomto případě výrazně redukovala počet záznamů směrovací tabulky.

PŘÍLOHA E KONFIGURACE RIP

Tabulka 15 Adresové schéma pro konfiguraci RIP

Zařízení	Rozhraní	Adresa	Maska	Brána
R1	se0/1/0	192.168.26.1	255.255.255.0	n/a
	se0/1/1	192.168.24.1	255.255.255.0	n/a
	fa0/0	192.168.20.1	255.255.255.0	n/a
R2	se0/1/0	192.168.26.2	255.255.255.0	n/a
	se0/1/1	192.168.25.1	255.255.255.0	n/a
	fa0/0	192.168.23.1	255.255.255.0	n/a
R3	se0/1/0	192.168.25.2	255.255.255.0	n/a
	se0/1/1	192.168.24.2	255.255.255.0	n/a
	se0/0/0	192.168.27.1	255.255.255.0	n/a
	fa0/0	192.168.21.1	255.255.255.0	n/a
R4	se0/1/0	192.168.27.2	255.255.255.0	n/a
	fa0/0	192.168.22.1	255.255.255.0	n/a
Anička		192.168.20.2	255.255.255.0	192.168.20.1
Pepa		192.168.20.3	255.255.255.0	192.168.20.1
Tiskárna		192.168.21.2	255.255.255.0	192.168.21.1
Katka		192.168.21.3	255.255.255.0	192.168.21.1
Jonáš		192.168.21.4	255.255.255.0	192.168.21.1
Aneta		192.168.21.5	255.255.255.0	192.168.21.1
Zbyněk		192.168.22.2	255.255.255.0	192.168.22.1
Tereza		192.168.23.2	255.255.255.0	192.168.23.1
Matěj		192.168.23.3	255.255.255.0	192.168.23.1
Klára		192.168.23.4	255.255.255.0	192.168.23.1

```
R1>enable
R1#conf term
R1(config)#router rip
R1(config-router)#network 192.168.20.0
R1(config-router)#network 192.168.26.0
R1(config-router)#network 192.168.24.0

R2>enable
R2#conf term
R2(config)#router rip
R2(config-router)#network 192.168.23.0
R2(config-router)#network 192.168.26.0
R2(config-router)#network 192.168.25.0

R3>enable
R3#conf term
R3(config)#router rip
R3(config-router)#network 192.168.25.0
R3(config-router)#network 192.168.24.0
R3(config-router)#network 192.168.27.0
R3(config-router)#network 192.168.21.0

R4>enable
R4#conf term
R4(config)#router rip
R4(config-router)#network 192.168.22.0
R4(config-router)#network 192.168.27.0
```

Pro kontrolu konfigurace vyčkejte **alespoň jednu minutu**. Za tuto dobu proběhne výměna směrovacích informací pomocí protokolu RIP. Často se nám také stane, že příkaz ping obsahuje řádky, u nichž se objeví **request time out.** V takovém případě nezoufejte, jelikož pouze zatím nedošlo na některém ze směrovačů k přijetí směrovací databáze.

Kontrola konfigurace

Nejzákladnější možností kontroly směrovacího protokolu je **výpis směrovací tabulky.**

```
R# show ip route
```

V zobrazené směrovací tabulce si můžeme zkontrolovat, jaké sítě máme přímo připojeny, tedy zda ostatním směrovačům odesíláme validní informace. Dále si můžeme na tomto místě zkontrolovat přijaté informace od ostatních směrovačů.

Pokud využíváme na směrovači směrovací protokol, můžeme si zobrazit informace následovně.

```
R# show ip protocols
```

Pomocí tohoto příkazu jsou zobrazeny jednotlivé parametry běžících směrovacích protokolů a jejich aktuální stav. Přímo pro protokol RIP máme ještě jednu možnost při hledání vzniklé chyby.

```
R# show ip rip database
```

Tento příkaz vypíše celý seznam všech záznamů, které obsahuje **interní databáze** protokolu RIP. Pozor však, že jsou obsaženy i cesty na právě nedostupné přilehlé sítě.

PŘÍLOHA F KONFIGURACE RIPV2

Tabulka 16 Adresní schéma pro konfiguraci RIPv2

Adresní tabulka

zařízení	rozhraní	adresa	maska	brána
R1	se0/1/0	192.168.20.49	255.255.255.252	n/a
	se0/1/1	192.168.20.41	255.255.255.252	n/a
	fa0/0	192.168.20.17	255.255.255.248	n/a
R2	se0/1/0	192.168.20.50	255.255.255.252	n/a
	se0/1/1	192.168.20.45	255.255.255.252	n/a
	fa0/0	192.168.20.25	255.255.255.248	n/a
R3	se0/1/0	192.168.20.46	255.255.255.252	n/a
	se0/1/1	192.168.20.42	255.255.255.252	n/a
	se0/0/0	192.168.20.1	255.255.255.252	n/a
	fa0/0	192.168.20.1	255.255.255.240	n/a
R4	se0/1/0	192.168.20.54	255.255.255.252	n/a
	fa0/0	192.168.20.33	255.255.255.248	n/a
Anička		192.168.20.18	255.255.255.248	192.168.20.17
Pepa		192.168.20.19	255.255.255.248	192.168.20.17
Tiskárna		192.168.20.2	255.255.255.240	n/a
Katka		192.168.20.3	255.255.255.240	192.168.20.1
Jonáš		192.168.20.4	255.255.255.240	192.168.20.1
Aneta		192.168.20.5	255.255.255.240	192.168.20.1
Zbyněk		192.1678.20.34	255.255.255.248	192.168.20.33
Tereza		192.168.20.26	255.255.255.248	192.168.20.25
Matěj		192.168.20.27	255.255.255.248	192.168.20.25
Klára		192.168.20.28	255.255.255.248	192.168.20.25

```
R1(config)# router rip
R1(config-router)# version 2
R1(config-router)# network 192.168.20.16
R1(config-router)# network 192.168.20.40
R1(config-router)# network 192.168.20.48

R2(config)# router rip
R2(config-router)# version 2
R2(config-router)# network 192.168.20.24
```

```
R2(config-router)# network 192.168.20.44
R2(config-router)# network 192.168.20.48

R3(config)# router rip
R3(config-router)# version 2
R3(config-router)# network 192.168.20.44
R3(config-router)# network 192.168.20.40
R3(config-router)# network 192.168.20.52
R3(config-router)# network 192.168.20.0

R4(config)# router rip
R4(config-router)# version 2
R4(config-router)# network 192.168.20.52
R4(config-router)# network 192.168.20.32
```

PŘÍLOHA G KONFIGURACE RIPNG

Tabulka 17 Adresní schéma pro konfiguraci RIPng

Zařízení	Rozhraní	Adresa	Brána
R1	se0/1/0	abcd:abcd:0000:0007::0001/64	n/a
	se0/1/1	abcd:abcd:0000:0005::0001/64	n/a
	fa0/0	abcd:abcd:0000:0001::0001/64	n/a
R2	se0/1/0	abcd:abcd:0000:0007::0002/64	n/a
	se0/1/1	abcd:abcd:0000:0006::0001/64	n/a
	fa0/0	abcd:abcd:0000:0004::0001/64	n/a
R3	se0/1/0	abcd:abcd:0000:0006::0002/64	n/a
	se0/1/1	abcd:abcd:0000:0005::0002/64	n/a
	se0/0/0	abcd:abcd:0000:0008::0001/64	n/a
	fa0/0	abcd:abcd:0000:0002::0001/64	n/a
R4	se0/1/0	abcd:abcd:0000:0008::0002/64	n/a
	fa0/0	abcd:abcd:0000:0003::0001/64	n/a
Anička		abcd:abcd:0000:0001::0002/64	abcd:abcd:0000:0001::0001/64
Pepa		abcd:abcd:0000:0001::0003/64	abcd:abcd:0000:0001::0001/64
Tiskárna		abcd:abcd:0000:0002::0002/64	abcd:abcd:0000:0002::0001/64
Katka		abcd:abcd:0000:0002::0003/64	abcd:abcd:0000:0002::0001/64
Jonáš		abcd:abcd:0000:0002::0004/64	abcd:abcd:0000:0002::0001/64
Aneta		abcd:abcd:0000:0002::0005/64	abcd:abcd:0000:0002::0001/64
Zbyněk		abcd:abcd:0000:0003::0002/64	abcd:abcd:0000:0003::0001/64
Tereza		abcd:abcd:0000:0004::0002/64	abcd:abcd:0000:0004::0001/64
Matěj		abcd:abcd:0000:0004::0003/64	abcd:abcd:0000:0004::0001/64
Klára		abcd:abcd:0000:0004::0004/64	abcd:abcd:0000:0004::0001/64

Konfigurace probíhá snadněji, než u protokolů RIP a RIPv2. Jediné, co musíme udělat, je nakonfigurovat příslušná rozhraní správnými adresami a následně na těchto **rozhraních**

zapnout odesílán informací do sítě. V modelové topologii bychom tuto konfiguraci provedli následovně.

```
R1 (config)# ipv6 unicast routing
R1 (config)# int se0/1/0
R1 (config-if)# ipv6 address abcd:abcd:0000:0007::0001/64
R1 (config-if)# ipv6 rip ripng1 enable
R1 (config-if)# no sh
R1 (config-if)# exit
R1 (config)# int se 0/1/1
R1 (config-if)# ipv6 address abcd:abcd:0000:0005::0001/64
R1(config-if)# ipv6 rip ripng1 enable
R1(config-if)# no sh
R1(config-if)# exit
R1(config)# int fa0/0
R1(config-if)# ipv6 address abcd:abcd:0000:0001::0001/64
R1(config-if)# ipv6 rip ripng1 enable
R1(config-if)# no sh

R2 (config)# ipv6 unicast routing
R2 (config)# int se0/1/0
R2 (config-if)# ipv6 address abcd:abcd:0000:0007::0002/64
R2 (config-if)# clock rate 64000
R2 (config-if)# ipv6 rip ripng1 enable
R2 (config-if)# no sh
R2 (config-if)# exit
R2 (config)# int se 0/1/1
R2 (config-if)# ipv6 address abcd:abcd:0000:0006::0001/64
```

```
R2(config-if)# ipv6 rip ripng1 enable
R2(config-if)# no sh
R2(config-if)# exit
R2 (config)# int se0/0/0
R2 (config-if)# ipv6 address abcd:abcd:0000:0008::0001/64
R2 (config-if)# ipv6 rip ripng1 enable
R2 (config-if)# no sh
R2 (config-if)# exit
R2(config)# int fa0/0
R2(config-if)# ipv6 address abcd:abcd:0000:0004::0001/64
R2(config-if)# ipv6 rip ripng1 enable
R2(config-if)# no sh

R3 (config)# ipv6 unicast routing
R3 (config)# int se0/1/0
R3 (config-if)# ipv6 address abcd:abcd:0000:0006::0002/64
R3 (config-if)# clock rate 64000
R3 (config-if)# ipv6 rip ripng1 enable
R3 (config-if)# no sh
R3 (config-if)# exit
R3 (config)# int se 0/1/1
R3 (config-if)# ipv6 address abcd:abcd:0000:0005::0002/64
R3(config-if)# ipv6 rip ripng1 enable
R3(config-if)# no sh
R3(config-if)# exit
R3(config)# int fa0/0
R3(config-if)# ipv6 address abcd:abcd:0000:0002::0001/64
```

```
R3(config-if)# ipv6 rip ripng1 enable
R3(config-if)# no sh

R4 (config)# int se0/1/0
R4 (config-if)# ipv6 address abcd:abcd:0000:0008::0002/64
R4 (config-if)# clock rate 64000
R4 (config-if)# ipv6 rip ripng1 enable
R4 (config-if)# no sh
R4 (config-if)# exit
R4 (config)# int fa0/0
R4 (config-if) #ipv6 address abcd:abcd:0000:0003::0001/64
R4 (config-if)# ipv6 rip ripng1 enable
R4 (config-if)# no sh
R4 (config-if)# exit
```

Nyní se zdá, že konfigurace probíhá mnohem složitěji než u předešlých verzí tohoto protokolu. Nezapomeňme však, že u předchozích protokolů nebyla zobrazována konfigurace adres na rozhraních. Zde je však tato konfigurace jedním ze dvou kroků, které postačují ke zprovoznění protokolu RIPng v rámci sítě.

Kontrola konfigurace

Pro kontrolu obsahu, který je přeposílán slouží příkaz

```
R# show ipv6 router rip
```

který obdobně jako příkaz *show ip route* vypíše směrovací tabulku směrovače obsahující všechny IPv6 cesty.

Dále je protokol RIPng obohacen o zcela nový příkaz, sloužící ke kontrole dalších přeskoků nastavených k jednotlivým rozhraním.

```
R#show ipv6 rip <název> next-hops
```

PŘÍLOHA H KONFIGURACE EIGRP

Tabulka 18 Adresní schéma pro konfiguraci EIGRP

Adresní tabulka

zařízení	rozhraní	adresa	maska	brána
R1	se0/1/0	192.168.20.49	255.255.255.252	n/a
	se0/1/1	192.168.20.41	255.255.255.252	n/a
	fa0/0	192.168.20.17	255.255.255.248	n/a
R2	se0/1/0	192.168.20.50	255.255.255.252	n/a
	se0/1/1	192.168.20.45	255.255.255.252	n/a
	fa0/0	192.168.20.25	255.255.255.248	n/a
R3	se0/1/0	192.168.20.46	255.255.255.252	n/a
	se0/1/1	192.168.20.42	255.255.255.252	n/a
	se0/0/0	192.168.20.1	255.255.255.252	n/a
	fa0/0	192.168.20.1	255.255.255.240	n/a
R4	se0/1/0	192.168.20.54	255.255.255.252	n/a
	fa0/0	192.168.20.33	255.255.255.248	n/a
Anička		192.168.20.18	255.255.255.248	192.168.20.17
Pepa		192.168.20.19	255.255.255.248	192.168.20.17
Tiskárna		192.168.20.2	255.255.255.240	n/a
Katka		192.168.20.3	255.255.255.240	192.168.20.1
Jonáš		192.168.20.4	255.255.255.240	192.168.20.1
Aneta		192.168.20.5	255.255.255.240	192.168.20.1
Zbyněk		192.1678.20.34	255.255.255.248	192.168.20.33
Tereza		192.168.20.26	255.255.255.248	192.168.20.25
Matěj		192.168.20.27	255.255.255.248	192.168.20.25
Klára		192.168.20.28	255.255.255.248	192.168.20.25

```
R1>enable
R1#conf term
R1 (config)#router eigrp 100
R1 (config-router)#network 192.168.20.16 0.0.0.7
R1 (config-router)#network 192.168.20.40 0.0.0.3
R1 (config-router)#network 192.168.20.48 0.0.0.3


R2>enable
R2#conf term
```

```
R2(config)#router eigrp 100
R2(config-router)#network 192.168.20.24 0.0.0.7
R2(config-router)#network 192.168.20.44 0.0.0.3
R2(config-router)#network 192.168.20.48 0.0.0.3
```

V tomto bodě se nám již začnou zobrazovat informace o sousedních směrovačích. Navázání vztahu s těmito směrovači signalizuje tento výpis

```
%DUAL-5-NBRCHANGE: IP-EIGRP 100: Neighbor 192.168.20.49
(Serial0/1/0) is up: new adjacency
```

```
R3>enable
R3#conf term
R3(config)#router eigrp 100
R3(config-router)#network 192.168.20.0 0.0.0.15
R3(config-router)#network 192.168.20.40 0.0.0.3
R3(config-router)#network 192.168.20.44 0.0.0.3
R3(config-router)#network 192.168.20.52 0.0.0.3
```

```
R4>enable
R4#conf term
R4(config)#router eigrp 100
R4(config-router)#network 192.168.20.32 0.0.0.7
R4(config-router)#network 192.168.20.52 0.0.0.3
```

Kontrola konfigurace

```
R # show ip eigrp interfaces
```

Tento příkaz slouží ke kontrole rozhraní, na nichž je nakonfigurováno odesílání informací pomocí EIGRP. Dalším, nám již známým příkazem je

```
R# show ip protocols
```

který slouží k zobrazení informací o všech směrovacích protokolech a jejich nastavení na konkrétním směrovači. A konečně ještě neopominutelný příkaz

```
R# show ip route
```

který nám, jak je již důvěrně známo, zobrazí směrovací tabulku, a tudíž můžeme kontrolovat veškeré cesty, po kterých jsou odesílány pakety. Záznamy pocházející z protokolu EIGRP pak nalezneme s označením D.

PŘÍLOHA I KONFIGURACE OSPF

Směrování v rámci jedné oblasti

Konfigurace není nikterak odlišná od konfigurování jiných dynamických směrovacích protokolů. Nejprve musíme na směrovači zapnout proces OSPF.

```
R (config) # router ospf 1
```

Dalším krokem je zadávání všech přímo připojených sítí, které mají být publikovány mezi směrovači. K jejich zadávání využíváme **wildcard mask**. Hodnota této masky je vždy doplňkem oktetů masky původní do tvaru *255.255.255.255* Číslo oblasti udáváme v případě směrování v rámci jedné oblasti nula.

```
R (config-router)# network <adresa_sítě> <wildcard_mask> area
0
```

V modelové topologii probíhá konfigurace OSPF následovně.

```
R1(config)#router ospf 1
R1 (config-router)#network 192.168.20.16 0.0.0.7 area 0
R1 (config-router)#network 192.168.20.40 0.0.0.3 area 0
R1 (config-router)#network 192.168.20.48 0.0.0.3 area 0
R1 (config-router)#exit


R2 (config)#router ospf 1
R2 (config-router)#network 192.168.20.24 0.0.0.7 area 0
R2 (config-router)#network 192.168.20.44 0.0.0.3 area 0
R2 (config-router)#network 192.168.20.48 0.0.0.3 area 0
R2 (config-router)#exit
```

IOS interaktivně reaguje na navázání sousedství mezi směrovači. Ihned po navázání plného stavu přilehlosti reaguje následujícím výpisem.

```
00:14:20: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.20.49 on
Serial0/1/0 from LOADING to FULL, Loading Done
```

Tento výpis udává, **kdy** ke spojení došlo, který **proces** OSPF jej provedl. Dalším polem je Nbr, což v tomto případě představuje router ID, se kterým bylo spojení navázáno, na kterém **rozhraní** se tento **soused** nachází a nakonec výsledný stav, jichž bylo dosaženo.

```
R3 (config)#router ospf 1
R3 (config-router)#network 192.168.20.0 0.0.0.15 area 0
R3 (config-router)#network 192.168.20.40 0.0.0.3 area 0
R3 (config-router)#network 192.168.20.44 0.0.0.3 area 0
R3 (config-router)#network 192.168.20.52 0.0.0.3 area 0
R3 (config-router)# exit

R4 (config)#router ospf 1
R4 (config-router)#network 192.168.20.32 0.0.0.7 area 0
R4 (config-router)#network 192.168.20.52 0.0.0.3 area 0
R4 (config-router)#exit
```

Kontrola konfigurace

První, co doporučuji zkontrolovat je **tabulka sousedů**. Tuto tabulku zobrazíme pomocí příkazu

```
R1#show ip ospf neighbor
```

Výstupem tohoto příkazu je tabulka všech směrovačů, se kterými má aktuální směrovač navázán vztah sousedství.

```
Neighbor ID     Pri   State           Dead Time   Address         Interface
192.168.20.49     0   FULL/  -        00:00:31    192.168.20.49   Serial0/1/0
192.168.20.53     0   FULL/  -        00:00:32    192.168.20.46   Serial0/1/1
```

Obrázek 78 OSPF tabulka sousedů

První sloupec zobrazuje ID směrovače, s nímž máme navázané sousedství, jak vidíte, bylo by vhodné nastavit **router ID** na všech směrovačích, aby bylo možné je vzájemně rozlišit.

Vidíme také informaci, přes které rozhraní je sousední směrovač připojen. Další sloupec zobrazuje **stav**, ve kterém se s konkrétním sousedem nacházíme. **Dead time** určuje dobu, před kterou naposledy došlo k přijetí paketu od sousedního směrovače, kterého se záznam týká. Předposlední sloupec obsahuje **IP adresu rozhraní** sousedního směrovače, kterým je tento směrovač propojen s aktuálním. Poslední sloupec pak pouze zobrazuje označení rozhraní, přes které jsme se sousedním směrovačem propojeni.

Další informací, která se může hodit při kontrole konfigurace je výpis **link-state databáze.**

```
R1#sh ip ospf database
```

Výsledkem tohoto příkazu je zobrazení celé link-state databáze.

```
            OSPF Router with ID (192.168.20.50) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
192.168.20.49   192.168.20.49   360         0x80000006 0x009494 5
192.168.20.50   192.168.20.50   360         0x80000006 0x0051c5 5
192.168.20.54   192.168.20.54   361         0x80000004 0x00336c 3
192.168.20.53   192.168.20.53   356         0x80000008 0x008cf8 7
```

Obrázek 79 OSPF výpis link-state databáze

Z tohoto výpisu můžeme určit **ID směrovače**, na kterém byl tento výpis proveden. Dalšími údaji je **seznam všech směrovačů**, o kterých se směrovač dozvěděl pomocí procesu OSPF. V této tabulce vidíme i směrovač, na kterém se aktuálně nacházíme. Jako kontrolu správnosti lze porovnat počet záznamů v této tabulce, spolu s reálným počtem směrovačů v síti. Tedy pokud naše síť obsahuje čtyři směrovače, bude i tato tabulka obsahovat čtyři záznamy.

V rozsáhlejších sítích použijeme k tomuto účelu například dokumentaci, ve které by měl být uveden počet zařízení v síti.

Dalším příkazem, který nám může usnadnit hledání chyb v rámci protokolu OSPF je příkaz

```
R1#show ip protocols
```

```
Routing Protocol is "ospf 1"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Router ID 192.168.20.50
  Number of areas in this router is 1. 1 normal 0 stub 0 nssa
  Maximum path: 4
  Routing for Networks:
    192.168.20.24 0.0.0.7 area 0
    192.168.20.44 0.0.0.3 area 0
    192.168.20.48 0.0.0.3 area 0
  Routing Information Sources:
    Gateway         Distance      Last Update
    192.168.20.49         110      00:23:31
    192.168.20.50         110      00:23:31
    192.168.20.53         110      00:23:27
    192.168.20.54         110      00:23:32
  Distance: (default is 110)
```

Obrázek 80 OSPF výpis informací o protokolu

Pomocí tohoto výpisu můžeme lehce zjistit **ID směrovače**, **číslo procesu** OSPF, **směrovače**, z nichž přijímáme informace a **všechny sítě**, které byly zjištěny pomocí protokolu OSPF. Lze zjistit informaci o **počtu oblastí** a celkový **počet cest**.

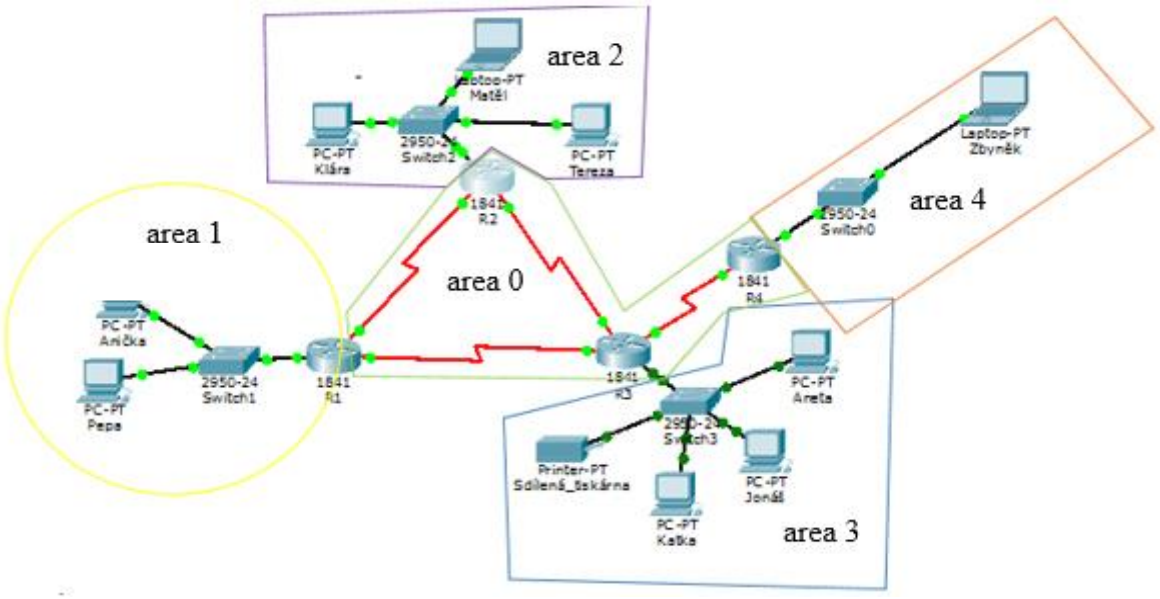
Směrování v rámci více oblastí

Oproti směrování v rámci jedné oblasti rozčleníme modelovou topologii do oblastí. Je vhodné rozepsat přehledný výpis rozhraní, spolu s oblastí, do které bude spadat.

Tabulka 19 OSPF rozčlenění topologie do oblastí

Směrovač - rozhraní	Oblast
R1 - se0/1/0	0
R1 – se0/1/1	0
R1 – fa0/0	1
R2 – se0/1/0	0
R2 – se0/1/1	0
R2 – fa0/0	2
R3 – se0/1/0	0
R3 – se0/1/1	0
R3 – fa0/0	3
R4 – se0/1/0	0
R4 – fa0/0	4

Následující obrázek znázorňuje grafické uspořádání oblastí.



Obrázek 81 OSPF grafické rozčlenění oblastí

Obecný postup konfigurace

Zapneme proces OSPF označený libovolným číslem.

```
R (config)# router ospf <číslo_procesu>
```

A dále použijeme příkaz pro všechny přímo připojené sítě.

```
R (config-router)# network <adresa_sítě> <wildcard_mask> area
<číslo_oblasti>
```

V modelové topologii by konfigurace probíhala následujícím způsobem.

```
R1(config)#router ospf 1
R1(config-router)#network 192.168.20.16 0.0.0.7 area 1
R1(config-router)#network 192.168.20.40 0.0.0.3 area 0
R1(config-router)#network 192.168.20.48 0.0.0.3 area 0
R1(config-router)#exit

R2(config)#router ospf 1
```

```
R2(config-router)#network 192.168.20.24 0.0.0.7 area 2
R2(config-router)#network 192.168.20.44 0.0.0.3 area 0
R2(config-router)#network 192.168.20.48 0.0.0.3 area 0
R2(config-router)#exit

R3(config)#router ospf 1
R3(config-router)#network 192.168.20.0 0.0.0.15 area 3
R3(config-router)#network 192.168.20.40 0.0.0.3 area 0
```

V tento okamžik již můžeme vidět, že mezi směrovači se začíná formovat sousedství. IOS nás o této skutečnosti informuje následujícím výpisem. Není zde žádný rozdíl oproti výpisu z OSPF provozovaného v rámci jedné oblasti. Opět zde můžeme vidět informace o tom, který proces OSPF sousedství zformoval, s jakým sousedem a přes jaké rozhraní. Dále vidíme stav FULL, tedy indikátor plného sousedského stavu. Dva směrovače si tímto okamžikem začínají vyměňovat informace.

```
00:06:57: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.20.49 on
Serial0/1/1 from LOADING to FULL, Loading Done

R3(config-router)#network 192.168.20.44 0.0.0.3 area 0
R3(config-router)#network 192.168.20.52 0.0.0.3 area 0
R3(config-router)# exit

R4(config)#router ospf 1
R4(config-router)#network 192.168.20.32 0.0.0.7 area 4
R4(config-router)#network 192.168.20.52 0.0.0.3 area 0
R4(config-router)#exit
```

Ke kontrole konfigurace využijeme stejné příkazy jako u využití jedné oblasti.

Tip: Pokud chcete zobrazit pouze záznamy směrovací tabulky získané pomocí protokolu OSPF, můžete použít příkaz na výpis směrovací tabulky spolu s parametrem ospf. Zápis by vypadal následovně.

```
R1# show ip route ospf
```

Router ID

Již několikrát jsme se v rámci tohoto textu setkali s pojmem **router ID**. Možná si teď říkáte, jakým způsobem je směrovači přiřazena hodnota ID, jelikož ta musí být v rámci sítě jedinečná a přitom o sobě mnohdy všechny směrovače v síti ani nevědí. Hodnota ID směrovače je přiřazována následujícím způsobem.

1. Pokud byla nastavena hodnota ID směrovače v rámci OSPF pomocí příkazu

   ```
   R (config) # router ospf 1
   R (config-router) # router-id <přiřazená_adresa>
   <maska>
   ```

   potom je použita tato hodnota. Pokud bychom při ruční změně hodnoty ID směrovače zadali adresu, jenž je již použita v rámci sítě, informoval by nás IOS následujícím výpisem

   ```
   %OSPF-4-DUP_RTRID1: Detected router with duplicate
   router ID
   ```

   Pozn.: Pokud jakou budeme provádět nějaké změny v ID směrovače nebo v síti, mohly by tyto změny ovlivnit celý výpočet v rámci OSPF, a tudíž je dobré po provedení takovýchto úprav na směrovači uložit běžící konfiguraci a restartovat příkazem

   ```
   R # reload
   ```

   Pokud jsme neměnili žádná rozhraní, pak stačí vymazat běžící proces OSPF a tím znovu spustit veškeré výpočty v síti. Vymazání procesu provedeme následujícím příkazem.

   ```
   R# clear ip ospf process
   ```

2. Pokud směrovač nemá nastavenou hodnotu ID, použije se nejvyšší IP adresa ze všech rozhraní typu loopback, které směrovač obsahuje.
3. Pokud směrovač nemá nastaveny rozhraní typu loopback, potom se vezme nejvyšší IP adresa přiřazená fyzickým rozhraním tohoto směrovače.

Ukažme si na konfiguraci z předchozí kapitoly, jak se projevují změny router ID. Jelikož žádný ze směrovačů v modelové topologii neměl nastavenou hodnotu router ID, ani neobsahoval rozhraní typu loopback, jehož adresa by mohla být použita, je použita nejvyšší

adresa aktivního rozhraní. Zjištění aktuální přiřazené hodnoty router ID můžeme zjistit například výpisem stavové databáze, tento výpis provedeme

```
R # show ip ospf database
```

Jehož výstupem je následující tabulka

```
            OSPF Router with ID (192.168.20.50) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
192.168.20.50   192.168.20.50   1243        0x80000007 0x004fc6 5
192.168.20.49   192.168.20.49   1244        0x80000007 0x009295 5
192.168.20.54   192.168.20.54   1244        0x80000005 0x00316d 3
192.168.20.53   192.168.20.53   1244        0x80000009 0x008af9 7
```

Obrázek 82 OSPF implicitní router ID

Z celého výpisu obsahu stavové databáze nás bude v této části kapitoly zajímat řádek

Ve kterém hodnota 192.168.20.50 udává nejvyšší adresu aktivního rozhraní a je použita jako **router ID**. Že je právě tato adresa tou nejvyšší nastavenou se lze přesvědčit výpisem přehledu rozhraní směrovače.

```
Router#sh ip int br
```

```
Interface              IP-Address      OK? Method Status                Protocol

FastEthernet0/0        192.168.20.25   YES manual up                    up
FastEthernet0/1        unassigned      YES unset  administratively down down
Serial0/1/0            192.168.20.50   YES manual up                    up
Serial0/1/1            192.168.20.45   YES manual up                    up
Vlan1                  unassigned      YES unset  administratively down down
```

Obrázek 83 OSPF ověření implicitního nastavení router ID

V tomto výpisu se můžeme snadno přesvědčit, že naše hodnota router ID opravdu odpovídá nejvyšší nastavené hodnotě rozhraní. Konkrétně se tedy jedná o IP adresu přiřazenou rozhraní serial0/1/0.

Abychom se přesvědčili, že vyšší prioritu při volbě router ID má hodnota rozhraní loopback, přidejme si na směrovač rozhraní lo1, kterému přiřaďme adresu kupříkladu 1.1.1.1. Na hodnotě, která tomuto rozhraní přidělena nezáleží, jelikož nám bude sloužit pouze pro nastavení hodnoty router ID.

```
R2 (config)# int lo1

R2 (config-if)# ip add 2.2.2.2 255.255.255.255

R2 (config-if)# exit

R2 (config)# exit

R2# copy running-config startup-config

Destination filename [startup-config]?

Building configuration...

[OK]

R2 # reload
```

Po opětovném načtení IOS provedeme příkaz

```
R2 # show ip ospf database
```

Jehož výsledkem je následující hodnota ID pro směrovač R2

```
OSPF Router with ID (2.2.2.2) (Process ID 1)
```

Nejvyšší prioritu při určování router ID má přímá konfigurace této hodnoty. Nechme si tedy nastaveno rozhraní lo1 a ručně nastavme hodnotu ID na 1.1.1.1

```
R2 #conf term

R2 (config)#router ospf 1

R2 (config-router)#router

R2 (config-router)#router-id 1.1.1.1

R2 (config-router)#Reload or use "clear ip ospf process"
command, for this to take effect
```

Vidíme, že sám směrovač nás upozorní, abychom buď restarovali směrovač anebo alespoň restartovali proces OSPF. Pokud měníme hodnotu router ID takto ručně, vystačíme si s druhou variantou.

```
R2 # clear ip ospf proces

Reset ALL OSPF processes? [no]: y

Router#
```

```
00:05:52: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.20.50 on
Serial0/1/0 from FULL to DOWN, Neighbor Down: Adjacency forced
to reset

00:05:52: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.20.50 on
Serial0/1/0 from FULL to DOWN, Neighbor Down: Interface down
or detached

00:05:52: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.20.53 on
Serial0/1/1 from FULL to DOWN, Neighbor Down: Adjacency forced
to reset

00:05:52: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.20.53 on
Serial0/1/1 from FULL to DOWN, Neighbor Down: Interface down
or detached

00:06:00: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.20.53 on
Serial0/1/1 from LOADING to FULL, Loading Done

00:06:00: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.20.50 on
Serial0/1/0 from LOADING to FULL, Loading Done
```

Vidíme, že nejprve se nám přeruší veškeré vazby a následně je spuštěn celý proces vytváření spojení. Nicméně nás bude zajímat, jaký vzala tato úprava efekt na hodnotu router ID.

Použijme tedy opět příkaz

```
R2 # show ip ospf database
```

Jehož výsledkem je tabulka obsahující pro nás důležitý řádek

```
OSPF Router with ID (1.1.1.1) (Process ID 1)
```

Tedy vidíme, že nejvyšší prioritu při volbě router ID má manuálně nastavená hodnota.

PŘÍLOHA J KONFIGURACE OSPFV3

Tabulka 20 Adresní schéma pro konfiguraci OSPFv3

Zařízení	Rozhraní	Adresa	Brána
R1	se0/1/0	abcd:abcd:0000:0007::0001/64	n/a
	se0/1/1	abcd:abcd:0000:0005::0001/64	n/a
	fa0/0	abcd:abcd:0000:0001::0001/64	n/a
R2	se0/1/0	abcd:abcd:0000:0007::0002/64	n/a
	se0/1/1	abcd:abcd:0000:0006::0001/64	n/a
	fa0/0	abcd:abcd:0000:0004::0001/64	n/a
R3	se0/1/0	abcd:abcd:0000:0006::0002/64	n/a
	se0/1/1	abcd:abcd:0000:0005::0002/64	n/a
	se0/0/0	abcd:abcd:0000:0008::0001/64	n/a
	fa0/0	abcd:abcd:0000:0002::0001/64	n/a
R4	se0/1/0	abcd:abcd:0000:0008::0002/64	n/a
	fa0/0	abcd:abcd:0000:0003::0001/64	n/a
Anička		abcd:abcd:0000:0001::0002/64	abcd:abcd:0000:0001::0001/64
Pepa		abcd:abcd:0000:0001::0003/64	abcd:abcd:0000:0001::0001/64
Tiskárna		abcd:abcd:0000:0002::0002/64	abcd:abcd:0000:0002::0001/64
Katka		abcd:abcd:0000:0002::0003/64	abcd:abcd:0000:0002::0001/64
Jonáš		abcd:abcd:0000:0002::0004/64	abcd:abcd:0000:0002::0001/64
Aneta		abcd:abcd:0000:0002::0005/64	abcd:abcd:0000:0002::0001/64
Zbyněk		abcd:abcd:0000:0003::0002/64	abcd:abcd:0000:0003::0001/64
Tereza		abcd:abcd:0000:0004::0002/64	abcd:abcd:0000:0004::0001/64
Matěj		abcd:abcd:0000:0004::0003/64	abcd:abcd:0000:0004::0001/64
Klára		abcd:abcd:0000:0004::0004/64	abcd:abcd:0000:0004::0001/64

```
R1(config)#int se0/1/0
R1(config-if)#ipv6 address abcd:abcd:0000:0007::0001/64
R1(config-if)#no sh
```

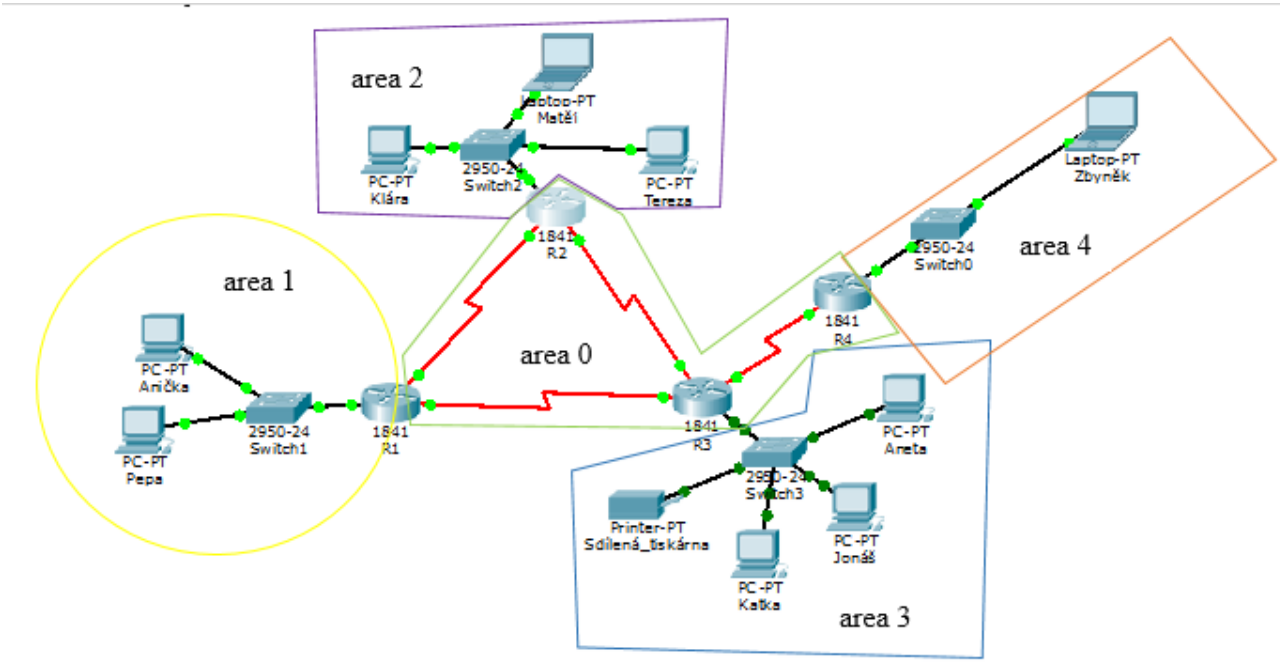
```
R1(config-if)#int se0/1/1
R1(config-if)#ipv6 address abcd:abcd:0000:0005::0001/64
R1(config-if)#clock rate 64000
R1(config-if)#no sh
R1(config-if)#int fa0/0
R1(config-if)#ipv6 address abcd:abcd:0000:0001::0001/64
R1(config-if)#no sh

R2(config)#int se0/1/0
R2(config-if)#ipv6 address abcd:abcd:0000:0007::0002/64
R2(config-if)#clock rate 64000
R2(config-if)#no sh
R2(config-if)#exit
R2(config)#int se0/1/1
R2(config-if)#ipv6 address abcd:abcd:0000:0006::0001/64
R2(config-if)#no sh
R2(config-if)#exit
R2(config)#int fa0/0
R2(config-if)#ipv6 address abcd:abcd:0000:0004::0001/64
R2(config-if)#no sh
R2(config-if)#exit

R3(config)#int se0/1/0
R3(config-if)#ipv6 address abcd:abcd:0000:0006::0002/64
R3(config-if)#clock rate 64000
R3(config-if)#no sh
R3(config-if)#exit
```

```
R3(config)#int se0/1/1
R3(config-if)#ipv6 address abcd:abcd:0000:0005::0002/64
R3(config-if)#no sh
R3(config-if)#exit
R3(config)#int se0/0/0
R3(config-if)#ipv6 address abcd:abcd:0000:0008::0001/64
R3(config-if)#no sh
R3(config-if)#exit
R3(config)#int fa0/0
R3(config-if)#ipv6 address abcd:abcd:0000:0002::0001/64
R3(config-if)#no sh
R3(config-if)#exit

R4(config)#int se0/1/0
R4(config-if)#clock rate 64000
R4(config-if)#ipv6 address abcd:abcd:0000:0008::0002/64
R4(config-if)#no sh
R4(config)#int fa0/0
R4(config-if)#ipv6 address abcd:abcd:0000:0003::0001/64
R4(config-if)#no sh
```

Dalším krokem je pouze přiřazení jednotlivých rozhraní do zvolených oblastí. V tomto kroku tedy máme dvě možnosti:

- Všechna rozhraní přidáme do jedné oblasti, budeme tedy provádět **single-area OSPF**.
- Rozhraní rozčleníme do jednotlivých oblastí, budeme tedy aplikovat **multi-area OSPF**.

Pro rozdělení rozhraní do oblastí vyžijeme následující rozčlenění.



Obrázek 84 OSPFv3 rozčlenění oblastí

Přiřazení konkrétního rozhraní do oblasti provedeme následujícím příkazem

```
R (config) # <označení_rozhraní>
R (config-if) # ipv6 ospf <číslo procesu> area <číslo oblasti>
```

Pro úspěšnou konfiguraci OSPFv3 musíme zapnout proces OSPF a zároveň zapnout unicastovou komunikaci pomocí IPv6, která je implicitně vypnutá.

```
R (config) # ipv6 unicast-routing
R (config) # ipv6 ospf <číslo procesu>
```

V tomto kroku, pokud máme nastavenu hodnotu router ID, můžeme zobrazený režim opustit. Pokud bychom však chtěli tento režim opustit a na směrovači by nebyla nakonfigurována hodnota pro router ID, požadoval by po nás směrovač zadání hodnoty router ID prostřednictvím varování IOS.

```
%OSPFv3-4-NORTRID: OSPFv3 process 1 could not pick a router-
id,please configure manually
```

Rozhraní na směrovačích R1 až R4 zařadíme do oblastí protokolu OSPFv3 následovně.

```
R1(config)#ipv6 router ospf 1
% IPv6 routing not enabled
```

```
R1(config)#ipv6 unicast-routing
R1(config)#ipv6 router ospf 1
R1(config-rtr)#router-id 1.1.1.1
R1(config-rtr)#exit
R1(config)#int se0/1/0
R1(config-if)#ipv6 ospf 1 area 0
R1(config-if)#exit
R1(config)#int se0/1/1
R1(config-if)#ipv6 ospf 1 area 0
R1(config-if)#exit
R1(config)#int fa0/0
R1(config-if)#ipv6 ospf 1 area 1
R1(config-if)#exit
```

Zvýrazněná položka konfigurace představuje hlášku prostředí IOS, který nás informuje, že nemáme zapnuté unicastové směrování nad IPv6. Nejprve musíme provést příkaz

```
R1(config)#ipv6 unicast-routing
```

a následně teprve spustit proces OSPFv3.

```
R1(config)#ipv6 router ospf 1
```

Na ostatních směrovačích přiřadíme rozhraní následovně.

```
R2(config)#ipv6 unicast-routing
R2(config)#ipv6 router ospf 1
R2(config-rtr)#router-id 2.2.2.2
R2(config-rtr)#exit
R2(config)#int se0/1/0
R2(config-if)#ipv6 ospf 1 area 0
R2(config-if)#exit
```

V tento okamžik nás začne IOS informovat o navázaných spojeních se sousedními směrovači.

```
00:38:17: %OSPFv3-5-ADJCHG: Process 1, Nbr 1.1.1.1 on
Serial0/1/0 from LOADING to FULL, Loading Done

R2(config)#int se0/1/1

R2(config-if)#ipv6 ospf 1 area 0

R2(config-if)#exit

R2(config)#int fa0/0

R2(config-if)#ipv6 ospf 1 area 2

R2(config-if)#exit


R3(config)#ipv6 unicast-routing

R3(config)#int se0/1/0

R3(config-if)#ipv6 ospf 1 area 0

R3(config-if)# exit

R3(config)#int se0/1/1

R3(config-if)#ipv6 ospf 1 area 0

R3(config-if)#exit

R3(config)#int se0/0/0

R3(config-if)#ipv6 ospf 1 area 0

R3(config-if)#exit


R4(config)#ipv6 unicast-routing

R4(config)#ipv6 router ospf 1

R4(config-rtr)#router-id 4.4.4.4

R4(config-rtr)#exit

R4(config-if)#ipv6 ospf 1 area 0

R4(config-if)#exit

R4(config)#int se0/1/0

R4(config-if)#ipv6 ospf 1 area 0
```

```
R4(config-if)#exit
R4(config)#int fa0/0
R4(config-if)#ipv6 ospf 1 area 4
R4(config-if)#no sh
```

PŘÍLOHA K TABULKA VLSM

Počet bitů potřebných po hosty	Maximální počet hostů	Formát masky	Délka masky
0	1	255.255.255.255	32
1	2	255.255.255.254	31
2	4	255.255.255.252	30
3	8	255.255.255.248	29
4	16	255.255.255.240	28
5	32	255.255.255.224	27
6	64	255.255.255.192	26
7	128	255.255.255.128	25
8	256	255.255.255.0	24
9	512	255.255.254.0	23
10	1024	255.255.252.0	22
11	2048	255.255.248.0	21
12	4096	255.255.240.0	20
13	8192	255.255.224.0	19
14	16384	255.255.192.0	18
15	32768	255.255.128.0	17
16	65536	255.255.0.0	16
17	131072	255.254.0.0	15
18	262144	255.252.0.0	14
19	524288	255.248.0.0	13
20	1048576	255.240.0.0	12
21	2097152	255.224.0.0	11
22	4194304	255.192.0.0	10
23	8388608	255.128.0.0	9
24	16777216	255.0.0.0	8