Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 812 19 Bratislava

Simulátor komunikácie v počítačovej sieti

Tím č. 5
Red Dwarf

Odbor: Počítačové systémy a siete
Vedúci tímového projektu: Ing. Katarína Jelemnská, PhD

Bc. Martin Hornáček
Bc. Michal Jánoš
Bc. Milan Melicherčík

Máj 2005

Zadanie projektu

Navrhnite a zrealizujte programový systém pre simuláciu sieťovej komunikácie na druhej a tretej vrstve sieťovej architektúry RM OSI. Systém má umožňovať:

- definovanie topológie simulovanej siete,
- simuláciu rôznych prepájacích zariadení (napr. prepínač, smerovač, firewall ...),
- simuláciu komunikácie medzi prepájacími zariadeniami.

Funkčnosť navrhnutého systému overte v sieti so simulovanými zariadeniami pomocou komunikácie medzi koncovými zariadeniami.

OBSAH:

ZOZNAM OBRÁZKOV:

ZOZNAM TABULIEK:

1 Úvod

Predkladaný dokument je dokumentáciou k riešeniu projektu v rámci tímového projektu v školskom roku 2005/2006 tímom 19 – Red Dwarf. Našou úlohou bolo riešenie úlohy Simulátor komunikácie v počítačovej sieti. Dokument obsahuje analýzu, špecifikáciu a návrh riešenia.

1.1 Prehľad dokumentu

Analýza problému je popísaná v kapitole 2. Prehľadom problematiky a rozdelením základných sieťových technológií sa zaoberá kapitola 2.1. V kapitole 2.2 je analyzované riešenie podobného projektu iným tímom.

Špecifikácia požiadaviek sa nachádza v kapitole 3. V kapitole 3.1 je špecifikácia používateľského rozhrania, v kapitole 3.2 implementácia jednotlivých zariadení a v kapitole 3.2 sú prípady použitia.

Kapitola 4 obsahuje návrh systému.

1.2 Použité skratky

Táto kapitola obsahuje vysvetlenie skratiek použitých v tomto dokumente.

AS -	autonómny systém
BPDU -	dátová jednotka prepínacieho protokolu (*Bridge Protocol Data Unit).*
BDR -	záložný vyhradený smerovač (*Backup Designated Router*)
CIDR -	beztriedne smerovanie v rámci domény
CLI -	interfejs príkazového riadku
CRC -	kontrolný medzi súčet
CSMA/CD	metóda prístupu na linku a detekcie kolízií
DDP -	paket na komunikáciu s databázou
DR -	vyhradený smerovač (*Designated Router*)
DVA -	algoritmus na výpočet najkratšej cesty
FDDI -	optické rozhranie pre distribuované dáta
ID -	identifikátor
IGP -	protokol v rámci jednej domény

ISO -	medzinárodná štandardizačná organizácia
LAN -	lokálna sieť
LSA -	algoritmus na výpočet najkratšej cesty
MAC -	jedinečná adresa sieťového rozhrania
MAN -	miestna sieť
Mbps -	prenosová rýchlosť udávajúca počet mega bitov za sekundu
RFC -	odporúčania
RM-OSI -	simulačný model siete
STP -	protokol vetviaceho sa stromu
TCN -	upozornenie na zmenu topológie
VLSM -	variabilná dĺžka masky podsiete
WAN -	siete geografického rozsahu

1.3 Použitá notácia

Opis notácie použitej pri vytváraní diagramov uvedených v dokumente.

Diagram prípadov použitia a aktivít

Používateľ

Asociácia, väzba

Prípad použitia

2 Analýza

2.1 Úvod do problematiky

Počítačová sieť je systém vzájomne prepojených a spolupracujúcich počítačov. Medzi nimi možno prostredníctvom siete prenášať informácie.

2.1.1 Druhy počítačov v sieti

- **Pracovné stanice**- slúžia na spracovanie údajov používateľom. Je to samostatný počítať pripojený do siete, využívajúci jeho služby.
- **Server**- zabezpečuje chod siete. Realizuje funkcie siete a poskytuje ostatným používateľom svoje prostriedky (pamäť, tlačiareň...)

V sieti môže byť ľubovoľný počet serverov a pracovných staníc. Ak je serverov v sieti viac, môžu sa navzájom v poskytovaní služieb a prostriedkov doplňovať.

2.1.2 Základné časti siete

Sieť pozostáva z týchto základných častí:

- **Hardware** - zahrňuje všetky technické prostriedky počítača. Patria sem aj prostriedky, ktorými je realizované vlastné prepojenie siete (sieťové adaptéry...)
- **Software** - programové vybavenie, ktoré v spolupráci s hardware-om siete zabezpečuje funkcie siete. U niektorých operačných systémov sú tieto funkcie už jeho súčasťou.

2.1.3 Význam počítačových sietí

- **Zdieľanie údajov** - vďaka tomu, že dátové súbory sú uložené na serveroch siete a pripojení používatelia majú k nim prístup, môže potrebné dátové súbory spracovávať viac používateľov siete súčasne.
- **Zdieľanie prostriedkov** - umožňuje pracovným staniciam spoločne používať prostriedky siete, ktoré ponúkajú servery siete. Najčastejšie ide o zdieľanie diskov, keď lokálne disky pracovných staníc nemajú kapacitu a zdieľanie tlačiarní.

- **Zvýšenie spoľahlivosti systému** - v súvislosti so zdieľaním prostriedkov je možné v prípade poruchy zdieľaného prostriedku nahradiť tento prostriedok iným (tlačiareň...) a systém môže pracovať ďalej.

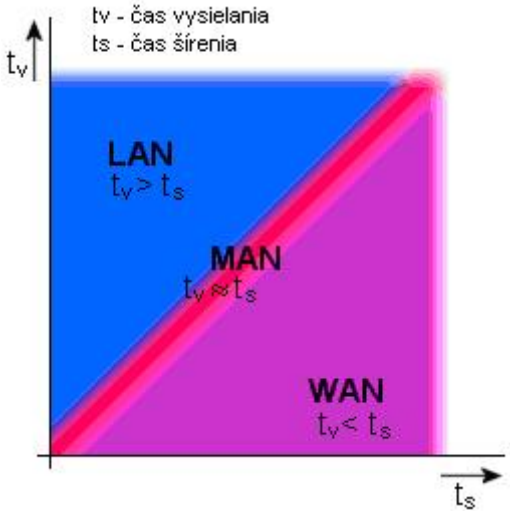
2.1.4 Rozdelenie podľa rozlohy

Siete sa rozdeľujú podľa pomeru doby vysielania a prijímania dát.

Local Area Network – **LAN –** Tieto siete sú rozsiahle od 10 m až do 1000 m. Sú to väčšinou siete v jednej budove alebo viacerých susediacich budovách. V rámci budovy sa používa štruktúrovaná kabeláž kombinujúca UTP a optické káble . Pre spojenie budov sa používajú optické káble alebo bezdrôtové spoje. Tieto siete môžu byť prepojené do ďalších väčších sietí. U LAN je doba vysielania **tv** vyššia ako doba šírenia signálu **ts** po prenosovom médiu (**tv > ts**).

Metropolitan Area Network – **MAN –** Verejná sieť pracujúca vysokou rýchlosťou a schopná prenášať dáta na vzdialenosť až 80 km. Tato sieť je menšia než WAN ale väčšia než LAN. Pre klasifikáciu pre ňu platí približne to isté čo v sieťach LAN. Sieť MAN má približne rovnakú dobu vysielania ako šírenia signálu (**tv = ts**).

Widea Area Network – WAN – S rastom geografického dosahu sietí pripojovanie užívateľov v rôznych mestách alebo štátoch prerastá sieť LAN a MAN do siete WAN (Wide Area Network). Počet užívateľov v takej sieti môže byť od desiatich do niekoľko tisíc užívateľov.
Doba vysielania je menšia než doba šírenia (**tv < ts**).



Obrázok č. 1: Rozdelenie sietí podľa rozlohy

2.1.5 Rozdelenie podľa topológie

Všetky návrhy siete vychádzajú z troch základných topológií:

- **Zbernicová topológia siete** - ak sú zapojené za sebou pozdĺž jediného kábla (segmentu).
- **Hviezdicová topológia siete** - ak sú počítače zapojené k segmentom, ktoré vychádzajú z jediného bodu (rozbočovača).
- **Prstencová topológia siete** - ak sú počítače zapojené ku káblu, ktorý tvorí prstenec.

2.1.6 Typy sietí podľa technológie

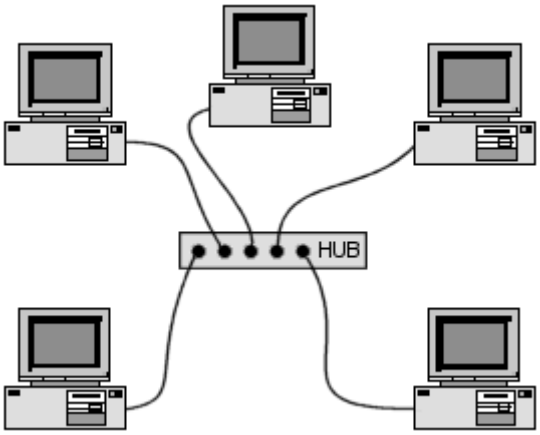
Siete sa dajú rozdeliť na 5 základných skupín, podľa použitej technológie:

- ArcNet
- Token-ring
- 100VG-AnyLAN
- FDDI
- Ethernet

ArcNet

Skratka slovného spojenia "Attached Resource Computer Network" (počítačová sieť s prepojenými prostriedkami). Ide o počítačovú sieť vyvinutou spoločnosťou Datapoint Corporation v roku 1977, ktorá umožňuje prepojiť množstvo osobných počítačov a pracovných staníc. Maximálny počet je 255.

Prenosovým médiom je koaxiálny kábel RG-62 A/U s impedanciou 93 ohmov. ArcNet je ale možno prevádzkovať aj na krútenej dvojlinke alebo optickom kábli. S použitím koaxiálneho kábla je maximálna dĺžka kábla od pracovnej stanice k rozbočovaču 610 metrov.



Obrázok č. 2: Sieť typu ArvNet

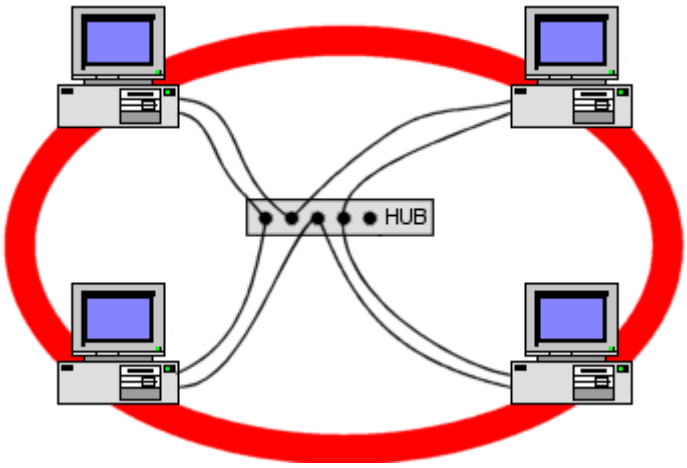
Predstavená sieť využíva prístupovú metódu založenú na predávaní známky a má prenosovú rýchlosť až 2,5 Mbps. Novšia verzia ArcNet Plus podporuje prenosovú rýchlosť až 20 Mbps. Maximálny priemer siete je 6,5 km. Fyzické zapojenie je hviezda, ale logická komunikácia je kruh.

Token-ring

Tato sieť bola v roku 1984 predstavená spoločnosťou IBM, ako súčasť riešenia prepojiteľnosti všetkých tried počítačov IBM.

V novších verziách bola prenosová rýchlosť 16Mb/s. Maximálna dĺžka závisí od počtu koncových zariadení, použitých káblov a zosilňovačov. V sieti token ring sú stanice prepojené do kruhu. Právo vysielať sa odovzdáva postupne v poradí pomocou špeciálneho rámca token. Aj keď je táto technológia založená na kruhové topologi, sieť Token-ring používa

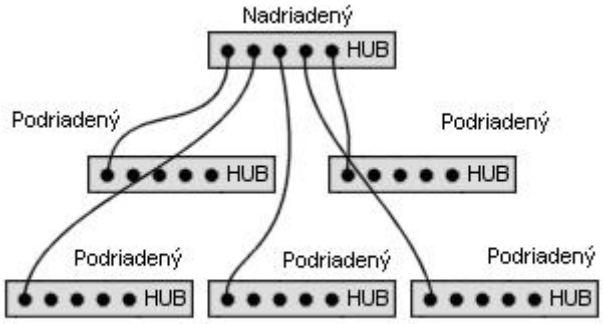
hviezdicové skupiny až ôsmich pracovných staníc, ktoré môžu byť napojené na hlavný kruh. Maximálny počet staníc je až 260 na jeden koncentrátor.



Obrázok č. 3: Sieť typu Token-ring

100VG-AnyLAN

Je to riešenie od firmy Hewlett-Packard. Rýchlosť tejto siete je minimálne 100 Mbps. Maximálny priemer siete je 7,7 km. Maximálny počet staníc nie je obmedzený, záleží od počtu rozbočovačov. Médiom je krútená dvojlinka a optický kábel. Je tu použitá bezkolízna prístupová metóda, umožňujúca dve úrovne priority (nízku a vysokú). Na 7,7 km je jeden rozbočovač. Za každý druhý rozbočovač sa musí odčítať 1,1 km.

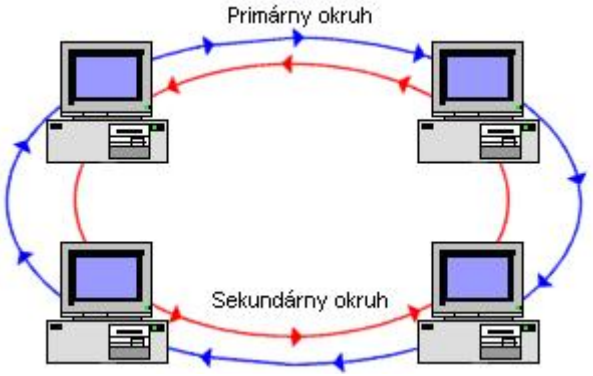


Obrázok č. 4: Sieť typu 100VG-AnyLAN

FDDI

Skratka slovného spojenia "Fiber Distributed Data Interface" (optické rozhranie pre distribuované dáta). Bola vytvorená v roku 1986.

Rýchlosť prenosu je 100 Mbps používajúca dvojitú protismernú kruhovú topológiu, podporujúcu až 500 počítačov. Jeden kruh sa označuje ako primárny a druhý ako sekundárny. Prenos dát prebieha väčšinou v primárnom okruhu. Pokiaľ nastane porucha v primárnom prstenci, FDDI automaticky prekonfiguruje sieť tak, aby mohli byť dáta posielané v druhom okruhu. Vďaka tejto redundancii je zaistená vysoká spoľahlivosť.

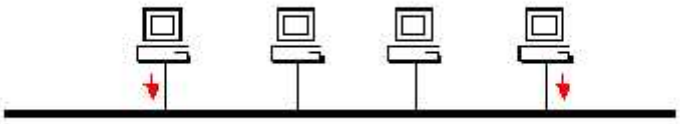


Obrázok č. 5:Sieť typu FDDI

Ethernet

Ethernet bol vyvinutý firmou Xerox v roku 1976. Ethernet používa prístupovú metódu CSMA/CD. Má svoj typ rámca. Pôvodne používal zbernicovú topológiu a umožňoval pripojiť na hlavný segmenty až 1024 počítačov a pracovných staníc. Jednotlivé stanice sú prepojené pomocou koaxiálneho kábla, optickým káblom či krútenou dvojlinkou.

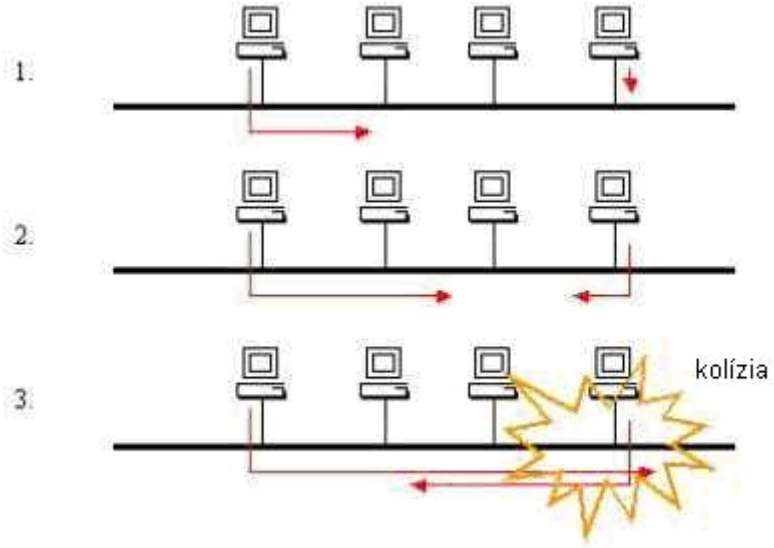
CSMA (Carrier Sense Multiple Access) - stanica pripravená vysielať dáta sleduje, či prenosové médium nepoužíva iná stanica. V prípade, že áno, stanica skúsi pristúpiť neskôr. Ak sa médium uvoľní začne stanica vysielať svoje dáta.



Obrázok č. 6: Metóda CSMA

CD (Collision Detection) – stanica počas vysielania sleduje, či je na médiu signál odpovedajúci vysielaným úrovniam. Prípad, kedy nastane interakcia signálov z viacerých staníc sa nazýva kolízia. V prípade detekcie kolízie stanica generuje signál JAM a všetky

stanice ktoré v danom okamžiku vysielali generujú náhodnou hodnotu času, po ktorej sa pokúsia vysielanie zopakovať.



Obrázok č. 7: Metóda CD

Vďaka tejto jednoduchosti boli dosiahnuté nízke ceny sieťových adaptérov a aktívnych prvkov, čo viedlo k značnému rozšíreniu Ethernetu. Jednoduchosť riešenia avšak priniesla aj jednu významnú nevýhodu – s narastajúcim počtom uzlov narastá aj počet kolízií a tým klesá teoretická priepustnosť siete. Súbor uzlov, ktorých vzájomná činnosť môže vygenerovať kolíziu sa nazýva kolízna doména.

Vedľa výrazu kolízna doména existuje výraz broadcastová doména. V počítačovej sieti sa vyskytujú principiálne dva typy paketov – tzv. unicasty a nonunicasty. Unicasty sú pakety, ktoré majú konkrétneho adresáta vyjadreného regulárnou sieťovou adresou. Nonunicasty používajú skupinovú adresu a sú určené, buď všetkým užívateľom (broadcasty) alebo vybranej skupine (multicasty). Problém je v tom, že nonunicastu sa musí počítač venovať aj keď nie je určený preň. S nárastom počtu uzlov v broadcastovej doméne narastá i množstvo nonunicastov. Z tohto dôvodu je nutné udržať veľkosť broadcastovej domény v rozumnej miere. Používané aktívne prvky majú k broadcastovej doméne rozdielny vzťah a preto ich voľbou sa dá priepustnosť siete ovplyvniť.

Formát paketu

Ako bolo už povedané, všetky rýchlostné modifikácie Ethernetu používajú rovnakú komunikačnú metódu CSMA/CD. Používajú však aj rovnaký formát a veľkosť paketu. Ethernetový paket je definovaný na 1. a 2. vrstve OSI.

Základnou častou paketu je hlavička linkovej vrstvy, ktorú nasledujú dáta (vrátane hlavičiek vyšších vrstiev). Hlavičky sú 4 typov a sú vzájomne nekompatibilné. Tieto typy sú :

- Ethernet_II
- Ethernet_802.3
- Ethernet_802.2
- Ethernet_SNAP

Formát – Ethernet_II.

Preambula	cieľová adresa (DA)	zdrojová adresa (SA)	typ paketu	dáta	CRC
8 byte	6 byte	6 byte	2 byte	46 až 1500 byte	4 byte

Každý paket je začína preambulou, ktorá slúži k synchronizácii vysielacej a prijímacej stanice. Nasleduje cieľová a zdrojová adresa MAC, číslo označujúce typ paketu, dátová časť a kontrolný súčet (CRC).

Používané média

Ethernet je dnes štandardizovaný v týchto verziách:

1. "Klasický" Ethernet s prenosovou kapacitou 10 Mbit/s:

10Base-2

– používa ako prenosové médium tienený koaxiálny kábel označovaný ako Thin Ethernet s impedanciou 50 ohm

- dĺžka segmentu môže byť maximálne 185 m
- na jednom segmentu môže byť maximálne 25 staníc
- segment musí byť na oboch koncoch ukončený pomocou tzv. terminátorov

10Base-5

– používa ako prenosové médium tienený koaxiálny kábel ozn. ako Thick Ethernet s impedanciou 50 ohm

- dĺžka segmentu môže byť maximálne 500 m
- segment musí byť na oboch koncoch ukončený pomocou tzv. terminátorov

10Base-T

– používa ako prenosové médium krútenú dvojlinku s impedanciou 100 ohm (min. Cat 3)
- dĺžka kábla medzi uzlom a aktívnym prvkom môže byť max. 100 m

10Base-FL

- používa ako prenosové médium optický kábel
- dĺžka medzi uzlami môže byť max. 2 km

2. Fast Ethernet s prenosovou kapacitou 100 Mbit/s:

100Base-TX

– používa ako prenosové médium krútenú dvojlinku s impedanciou 100 ohm (min. Cat 5)
– dĺžka kábla medzi uzlom a aktívnym prvkom môže byť max. 100 m

100Base-T4

– používa ako prenosové médium krútenú dvojlinku s impedanciou 100 ohm (min. Cat 3)
– dĺžka kábla medzi uzlom a aktívnym prvkom môže byť max. 100 m
– technológia nie je príliš rozšírená

100Base-FX

- používa ako prenosové médium optický kábel
- dĺžka medzi uzlami môže byť max. 2 km

3. Gigabit Ethernet s prenosovou kapacitou 1000 Mbit/s:

1000Base-SX

- používa ako prenosové médium optický kábel
- dĺžka medzi uzlami a aktívnym prvkom je ovplyvnená parametrami kábla

1000Base-LX

- používa ako prenosové médium optický kábel
- dĺžka medzi uzlami a aktívnym prvkom je ovplyvnená parametrami kábla

2.1.7 Model RM-OSI

Model RM-OSI je referenčný komunikačný model označený skratkou slovného spojenia "Open Systen Interconnection" (Prepojenie otvorených systémov). Je to doporučený

model siete definovaný organizáciou ISO v roku 1983, ktorý rozdeľuje vzájomnú komunikáciu medzi počítačmi do siedmich súvisiacich vrstiev.

Úlohou každej vrstvy je poskytovať služby nasledujúcim vyšším vrstvám a nezaťažovať vyššiu vrstvu detailmi o tom ako je služba v skutočnosti realizovaná. Uvedený model obsahuje nasledujúce vrstvy (každá vyššia vrstva využíva funkcie vrstvy nižšej).

1. Fyzická vrstva

Definuje prostriedky pre komunikáciu s prenosovým médiom a s technickými prostriedkami rozhrania. Ďalej definuje fyzické, elektrické, mechanické a funkčné parametre týkajúce sa fyzického spojenia jednotlivých zariadení. Je hardwarová.

2. Linková vrstva

Zaisťuje integritu toku dát z jedného uzlu siete na druhý. V rámci tejto činnosti je vykonávaná synchronizácia blokov dát a riadenia ich toku. Je hardwarová.

3. Sieťová vrstva

Definuje protokoly pre smerovanie dát, prostredníctvom ktorých je zaistený prenos informácií do požadovaného cieľového uzla.

4. Transportná vrstva

Definuje protokoly pre štruktúrované správy a zabezpečuje bezchybnosť prenosu. Rieši napríklad rozdelenie súboru na pakety a potvrdzovanie.

5. Relačná vrstva

Koordinuje komunikáciu a udržuje reláciu tak dlho, pokiaľ je potrebná. Ďalej zaisťuje zabezpečovacie, prihlasovacie funkcie.

5. Prezentačná vrstva

Špecifikuje spôsob, akým sú dáta formátované, prezentované, transformované a kódované. Rieši napríklad CRC, kompresiu a dekompresiu, šifrovanie dát.

6. Aplikačná vrstva

Je to v modeli najvyššia vrstva. Definuje spôsob, akým komunikujú so sieťou aplikácie, napríklad databázové systémy, elektronická pošta alebo programy pre emuláciu terminálov.

2.1.8 IP adresa

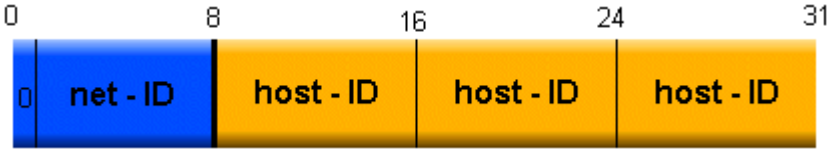
Ak chceme v rámci siete naviazať spojene s iným počítačom, musíme poznať jeho IP adresu. IP adresu musí mať každý počítač inú, pretože inak by nebolo možné rozlíšiť s akým počítačom chceme komunikovať.

IP adresy prideľuje je medzinárodná autorita poverená správou IP adries. V súčasnej dobe sa používa 32 bitová verzia IPv4 a verzia IPv6, ktorá je 128 bitová.

IPv4 adresa má veľkosť 4 byte = 32 bitov. Najčastejšie sa zapisuje v desiatkovej sústave, kde je jednotlivý byte oddelený bodkou. Každý byte môže nadobudnúť hodnotu od 0 - 255.

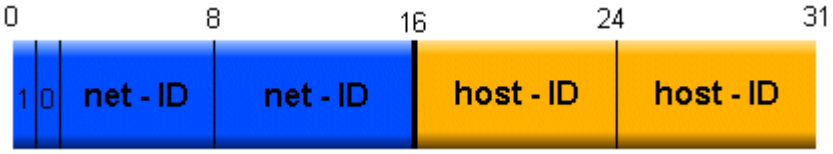
IP adresa sa skladá z dvoch častí net - ID (adresa siete) a host - ID (adresa počítača). Podľa toho ako sú jednotlivé siete rozľahlé rozlišujeme tri hlavné triedy IP adries - **A**, **B** a **C**.

Trieda A



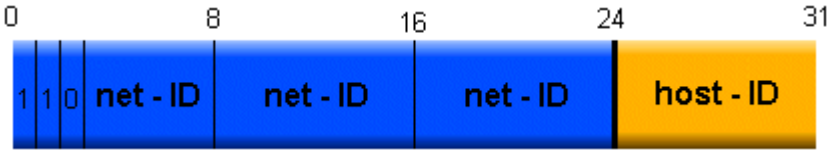
Dovoľuje adresovanie iba 126 sietí, ale v každej z nich maže byť až 16 miliónov počítačov. Rozsah hodnôt IP adries je: 0.0.0.0 až 127.255.255.255.

Trieda B



Trieda B umožňuje adresovať už 16 tisíc sietí a 65 tisíc počítačov v každej z nich. Prvé dva byte je adresa siete a ďalšie dva adresa počítače. Rozsah hodnôt v triede B je: 128.0.0.0 až do 191.255.255.255.

Trieda C



IP adresou triedy C dokážeme adresovať až 2 milióny sietí. V každej môže byť 254 počítačov. Prvé tri byte sú adresou siete a jeden byte adresou počítača. Rozsah je: 192.0.0.0. až 223.255.255.255

Špeciálne IP adresy

Niektoré IP adresy sú vyhradené pre špeciálne účely:

Rozsah od **224.0.0.0** do **239.255.255.255** je zaradený do triedy D. Tato trieda je využívaná pre multicasting.

Rozsah od **240.0.0.0** do **247.255.255.255** patrí do triedy E. Tieto hodnoty sú rezervované pre ďalšie použitie a pre experimentálne účely.

127.0.0.0 nebo **127.0.0.1** sú určené k testovacím účelom. Nazývajú sa tiež loopback adresy.

Broadcast adresa, **255.255.255.255** je určená všetkým počítačom v danej sieti. Používajú sa k hromadnému rozosielaniu paketov.

2.1.9 Aktívne prvky

Podľa počtu uzlov použitých v počítačovej sieti a v závislosti na jej topológii by mali byť volené aktívne prvky. V LAN sieťach sú používané nasledujúce typy aktívnych prvkov.

1. vrstva – fyzická vrstva

Opakovač (repeater) a **Rozbočovač** (hub) – aktívny prvok zaisťujúci spojenie dvoch a viacerých segmentov siete tím, že signál obdŕžaný na jednom porte zopakuje do ostatných portov, pričom signálu obnoví ostré vzostupné a zostupné hrany; rozširuje kolíznu i broadcastovú doménu.

Prevodník (Media Converter) – zariadenie, ktoré zaisťuje konverziu signálu z jedného typu média na iné

2. vrstva – linková vrstva

Most (bridge) – dvojportové zariadenie ktoré oddeľuje prevádzku na dvoch segmentoch siete na základe učenia fyzických (MAC) adries uzlov na oboch portoch. Na základe týchto adries most buď dáta na druhou stranu prepustí alebo neprepustí; most pracuje na druhej vrstve modelu OSI (linková vrstva) a preto je nezávislý od protokolu, ale je závislý na použitej sieťovej technológii. Most oddeľuje kolíznu doménu, ale rozširuje broadcastovú doménu; filtračná schopnosť sa vzťahuje len na Unicast pakety.

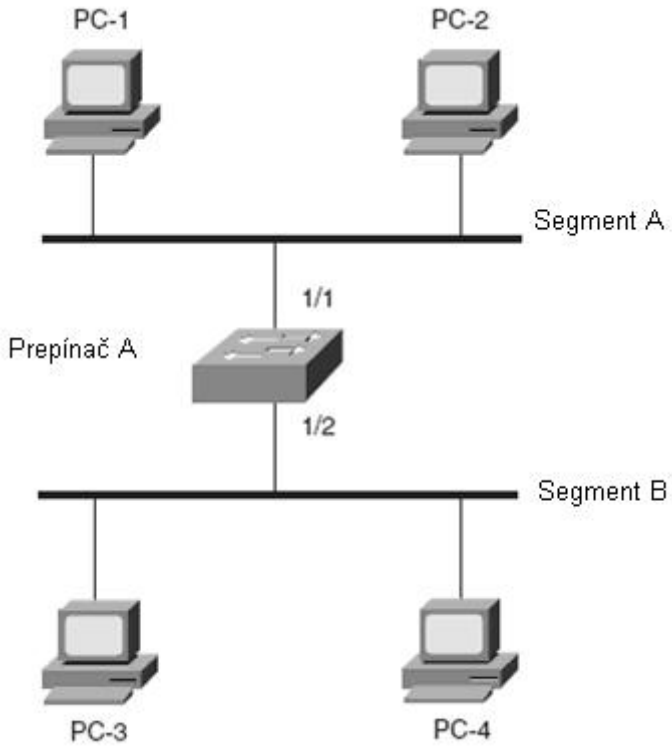
Prepínač (switch) – vysokorýchlostný multiportový most ktorý umožňuje:
– paralelnú komunikáciu medzi portami (tzn. napr. dvojice portov 2-3, 5-9, 6-4, … môžu komunikovať súčasne)
– popri štandardnej polovične duplexnej prevádzke prináša teoreticky dvakrát rýchlejší plne duplexní prenos

Prepínač oddeľuje kolíznu domény, ale rozširuje broadcastovú doménu.

2.1.10 Mechanizmus STP

Veľké siete nie sú navrhované len aby rýchlo a efektívne prenášali rámce alebo pakety, ale musia zvažovať ako sa rýchlo zotaviť z prípadnej chyby v sieti. Na tretej vrstve si smerovacie protokoly uchovávajú v tabuľke nadbytočné cesty do cieľovej siete, a tak sa môžu rýchlo zotaviť z poruchy, ak na primárnej ceste nastane chyba. Smerovanie tiež umožňuje využívať viaceré cesty, a tým rozdeľovať zaťaženie. Na druhej vrstve sa však žiadne smerovacie protokoly nepoužívajú a nadbytočné linkové spojenia nie sú dovolené, preto sa na druhej vrstve využíva „The Spanning-Tree Protocol“ (STP, protokol vetviaceho sa stromu), ktorý poskytuje nadbytočné linky a vyváženie záťaže, a tak sa môže zotaviť z poruchy bez včasného zásahu.

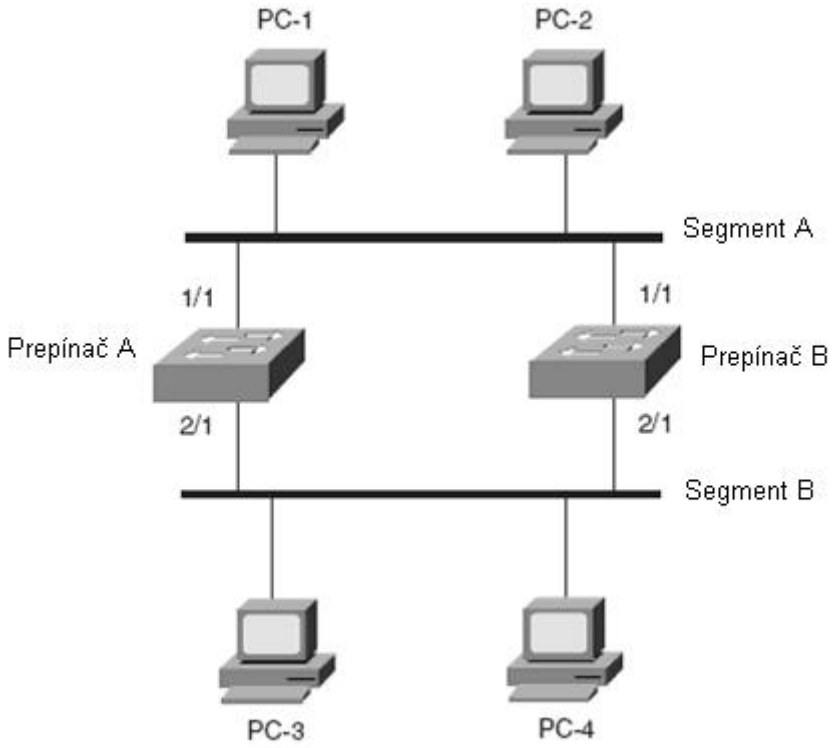
Na obrázku č. 8 je znázornená jednoduchá sieť, kde prepínač funguje ako most. Táto sieť ale neposkytuje žiadne nadbytočné linky a v prípade, ak by prepínač alebo jedna z jeho liniek zlyhala, bola by sieť nefunkčná.



Obrázok č. 8: Sieť s jedným prepínačom

Spoľahlivosť a odolnosť siete môžeme zvýšiť pridaním viacero liniek a ďalšieho prepínača, ako vidíme na obrázku č. 9. Ale z dôvodu, že o sebe prepínače nevedia, vznikajú

tzv. „*bridging loop*“ (premosťovacie slučky), kedy duplikujú rámce a preposielajú si ich stále medzi sebou.



Obrázok č. 9: Sieť s dvomi prepínačmi

Riešením tohto problému je fyzické prerušenie liniek a vypnutie prepínača alebo použitie STP protokolu, ktorý zabraňuje vytvoreniu týchto slučiek, a tým môžeme využívať výhody nadbytočných liniek a prepínačov v sieti.

Pomocou STP protokolu prepínače medzi sebou komunikujú, vyjednávajú medzi sebou bezslučkové cesty do jednotlivých segmentov v sieti. Slučky sú objavené skôr ako sú cez linky vysielané dáta a nadbytočné linky sú vypnuté. V prípade zlyhania jednej linky, prepínač pohotovo zareaguje a zapne jej záložnú (nadbytočnú) linku.

Vďaka STP komunikujú všetky prepínače zapojené v sieti a na základe informácií, ktoré si medzi sebou vymieňajú, každý prepínač vykonáva „Spanning-Tree“ algoritmus. Algoritmus zvolí referenčný bod v sieti a k nemu vypočítava cenu cesty každej linky. Ak algoritmus nájde nadbytočné linky, cez jednu posiela rámce a druhú vypne. Algoritmus spája prepínače do stromovej štruktúry a zabraňuje tak vytváraniu slučiek. Ak jedna z aktívnych liniek zlyhá, prepočíta sa celý strom znova, aby mohla byť jedna nadbytočná linka aktivovaná.

Spanning Tree algoritmus a protokol 802.1D

Komunikácia

Prepínače komunikujú medzi sebou a vymieňajú si dáta pomocou *Bridge Protocol Data Units (BPDUs, dátová jednotka prepínacieho protokolu).* Prepínač vysiela BPDU rámec cez každý port, kde zdrojová MAC adresa je jedinečná MAC adresa portu a cieľová adresa je multicastová adresa 01-80-c2-00-00-00, na ktorej počúvajú všetky prepínače v sieti.

Existujú dva typy BPDU:

- Konfiguračná BPDU, ktorá slúži na výpočet stromu,
- TCN (*Topology Change Notification, oznámenie zmeny topológie)* BPDU, ktorou sa upozorňuje na zmenu topológie siete

Cieľom výmeny BPDU je vytvorenie jednotného a stabilného stromu topológie. Konfiguračné BPDU obsahujú informácie o identifikácii prepínača, cenu cesty a nastavenia časovačov. Všetky tieto údaje sa používajú pri výpočte všeobecného stromu a zvolenia referenčného bodu v sieti tzv. *Root Bridge (koreňový prepínač).*

Popis	Veľkosť v bajtoch
ID protokolu	2
Verzia	1
Typ správy	1
Príznak	1
ID koreňového prepínača	8
Cena cesty ku koreňu	4
ID odosielateľa (prepínača)	8
ID portu na prepínači	2
Vek správy	2
Maximálny vek správy	2
Interval vysielania BPDU	2
Oneskorenie	2

Tabuľka č. 1: Obsah konfiguračného BPDU

Voľba koreňového prepínača

Aby sa všetky prepínače v sieti dohodli na bezslučkovej typológii, musia si zvoliť referenčný bod v sieti. Na voľbe tohto bodu sa podieľajú všetky pripojené prepínače. Každý prepínač má svoje jedinečné identifikačné číslo – ID, ktorého veľkosť je dva bajty a pozostáva z nasledujúcich častí:

• **Priorita prepínača (2 bajty) -** môže nadobúdať hodnotu od 0 do 65 535. Prednastavená hodnota je 32 768.

• **MAC Adresa (6 bajty) –** jedinečná adresa prepínača

Na začiatku, pri zapnutí prepínača, nevie nič o okolitých prepínačoch a za koreňový prepínač si zvolí sám seba. Voľba prebieha nasledovne:

- Každý prepínač vysiela BPDU, kde ID koreňového prepínača a ID odosielateľa je jeho vlastné ID.
- Prijaté BPDU ja analyzované a ak hodnota ID koreňového prepínača je menšia ako jeho aktuálna, nahradí svoju vlastnú hodnotu ID koreňového prepínača prijatou.
- Prepínač opäť preposiela BPDU, ale už so zmenenou hodnotou ID koreňového prepínača.
- Prepínače sa skôr či neskôr dohodnú a za referenčný bod siete si zvolia prepínač s najnižšou prioritou.

Voľba koreňového prepínača je pretrvávajúci proces, ktorý sa spúšťa zmenou ID koreňového prepínača každé dve sekundy.

Voľba koreňových portov

Po voľbe koreňového prepínača musí každý prepínač definovať svoj vzťah k nemu. Tento proces sa tiež nazýva voľba koreňového portu. Tento port sa vyberá na základe najmenšej ceny cesty ku koreňu. Cenu cesty tvorí súčet cien jednotlivých liniek, ktoré vedú ku koreňu stromu. Jej predvolené hodnoty sú vypísané v tabuľke č. 2. Všeobecne platí, čím väčšia priepustnosť linky, tým je menšia jej cena. Pôvodná norma IEEE 802.1D definovala cenu cesty ako podiel 1000 Mbps / priepustnosť linky v Mbps, ale z dôvodu rozvoja

moderných technológií a používania gigabitového ethernetu zaviedla IEEE novú nelineárnu škálu pre cenu cesty. Porovnanie je znázornenie v tabuľke č. 2.

Priepustnosť linky	Stará norma	Nová norma
4 Mbps	250	250
10 Mbps	100	100
16 Mbps	63	62
45 Mbps	22	39
100 Mbps	10	19
155 Mbps	6	14
622 Mbps	2	6
1 Gbps	1	4
10 Gbps	0	2

Tabuľka č. 2: Cena cesty

Voľba portov na posielanie dát

V tomto stave sú zatiaľ všetky linky aktívne a STP algoritmus musí zvoliť jeden port na posielanie dát pre každý segment siete tzv. *Designated Port*. Algoritmus vyberá port na základe najmenšej ceny cesty ku koreňovému prepínaču. Ak má susediaci prepínač na zdieľanom segmente siete rovnakú cenu cesty, berie sa ďalej do úvahy najnižšie ID prepínača na segmente a potom najmenšie ID portu prepínača.

Typy časovačov

STP protokol používa tri typy časovačov, aby sa uistil, že strom bol vytvorený správne a nevznikli žiadne slučky

• „**Hello Time“** – Interval posielania konfiguračných BPDU
• „**Forward Delay“ –** Čas, ktorý strávi port v stavoch počúvajúci a učiaci. Predvolená hodnota je 15 sekúnd.

• „**Maximum (max) Age“ –** Čas, ako dlho prepínač udržiava informácie o prijatom BPDU v pamäti. Po uplynutí tejto doby , prepínač informuje o zmene topológie v sieti. Predvolená hodnota je 20 sekúnd.

Tieto hodnoty môžu byť zmenené len na koreňovom prepínači a sú ďalej preposielané pomocou konfiguračných BPDU. Takto sa zaisťuje jednotná hodnota časovačov v celej sieti.

Zmena topológie

Ak nastane zmena v topológii, prepínač, ktorý túto zmenu zistí, posiela cez koreňové porty TCN BPDU, pokiaľ nedostane potvrdenie od koreňového prepínača. Ten nastaví v nasledujúcich správach príznak o zmene v topológii. Každý prepínač, ktorý prijme taký príznak, skráti čas uloženia MAC adries vo svojej tabuľke z 300 sekúnd na 15. Vďaka tomu, zostávajú prepínacie tabuľky aktuálne, lebo v nej zostanú len aktívne stanice.

Popis	Veľkosť v bajtoch
ID protokolu	2
Verzia	1
Typ správy	1

Tabuľka č. 3: Obsah TCN BPDU

Stav portu

V procese výpočtu stromu, musí každý port prechádzať niekoľkými stavmi aby sa stal aktívnym a mohol preposielať dáta. Protokol definuje nesledujúce stavy:

Blokujúci

Po inicializácii portu, sa nachádza port v blokujúcom stave, a preto nemôžu vznikať žiadne prepínacie slučky. V tomto stave port nemôže prijímať, vysielať dáta a ani vytvárať prepínaciu tabuľku na základe prijatých rámcov. Portu je dovolené len prijímať BPDU rámce od susediacich prepínačov.

Počúvajúci

Port prechádza do tohto stavu ak ho prepínač môže určiť za koreňový port alebo port na posielanie dát. V tomto stave port stále nemôže posielať a prijímať dátové rámce, ale má dovolené prijímať a posielať BPDU, a tak sa aktívne zúčastniť na tvorbe stromu topológie. Až teraz sa určuje, či daný port bude prenášať dáta alebo sa vráti späť do blokujúceho stavu.

Učiaci

Po čase strávenom v stavoch blokujúci a počúvajúci, ktorý určuje časovač „Forward Delay“, môže prejsť do stavu učiaci. V tomto stave si tvorí MAC tabuľku pre daný port.

Preposielajúci

Po ďalšom čase „Forward Delay“ port môže preposielať dátové rámce, tvorí MAC tabuľku a prijíma a posiela BPDU. Port je teraz plne funkčný v rámci topológie

3. vrstva – sieťová vrstva

Smerovač (router) – je to dva a viac portové zariadenie, ktoré pracuje na podobnom princípe ako prepínač, ale na tretej vrstve modelu OSI – pracuje teda s logickými adresami a je protokolovo závislý, ale relatívne nezávislý na použitej sitovej technológii (pre každú technológiu musí mať patričný adaptér). Smerovače sú v LAN sieťach používané prevažné pre oddelenie broadcastových domén – túto oblasť však opúšťajú lebo začínajú byť nahradzované smerovacími prepínačmi. Vedľa použitia v sieťach LAN, našli smerovače dôležité uplatnenie vo WAN sieťach, kde sú používané pre prepojovanie vzdialených lokalít

Prepínač pracuje s jednou tabuľkou a to s tabuľkou, kde sú relácie medzi MAC adresou a portom zariadenia. Smerovač pracuje s dvomi tabuľkami. V prvej je relácia medzi MAC adresou, logickou adresou a portom (tabuľka obsahuje údaje iba o priamo pripojených uzloch). V druhej tabuľke je zoznam sietí (častí logických adries) s portom kadiaľ vedie najlepšia cesta do danej siete.

Smerovací prepínač (routing switch) – je to relatívne nové zariadenie, ktoré pracuje s rýchlosťami obvyklými pre druhú vrstvu i s informáciami tretej vrstvy.

2.1.11 Základy smerovania v IP prostredí

Stanice v rámci jednej logickej siete komunikujú priamo (s použitím mechanizmu ARP). Pokiaľ však chce komunikovať stanica z jednej siete (napr. 192.168.1.x) s uzlom z inej siete (napr. 192.168.2.x), je potrebné siete prepojiť zariadením pracujúcim na 3. vrstve OSI.

Smerovače si udržiavajú prehľad o tom, na ktorom porte je aká sieť. Tieto informácie sú do zariadenia zadávané staticky alebo je používaný určitý mechanizmus pre ich dynamickú výmenu. Dynamických smerovacích protokolov je pomerne veľa napr. RIP, IGRP EIGRP, OSPF, BGP, EGP, ...

V závislosti na implementácii môžu byť súčasťou smerovacej tabuľky i masky cieľových sietí a typ protokolu, pomocou ktorého sa smerovač o sieti dozvedel.

Smerovacie protokoly

RIP

Smerovací protokol, ktorý využíva DVA algoritmus a ako metriku cesty používa počet skokov. Slúži na smerovanie v rámci autonómneho systému. RIP si vymieňa aktualizácie v pravidelných intervaloch a ak nastane zmena v sieti.

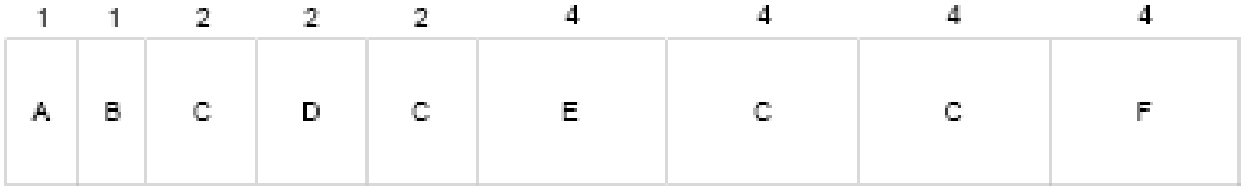
Metrika

Ak prijatá aktualizácia obsahuje nové informácie o novej ceste, priradí ju do svojej tabuľky a zahrnie ju do svojich aktualizácii, pričom jej zvýši metriku o jedna. Ako adresa ďalšieho skoku sa použije IP adresa odosielateľa danej aktualizácie. RIP uchováva len cestu s najlepšou metrikou do danej siete. Rip využíva počet skokov na meranie vzdialenosti medzi zdrojovou a cieľovou sieťou. Každý smerovač na ceste od zdroja k cieľu má priradenú hodnotu skoku, ktorá je väčšinou 1. Aby sa RIP vyhol smerovacím slučkám, používa maximálny počet skokov 15. Ak po zvýšení metriky je hodnota 16, označí danú sieť za nedosiahnuteľnú.

RIP vo všeobecnosti posiela aktualizácie každých 30 sekúnd. Každá cesta má svoj časovač, ktorý ak uplynie, vyhlási ju za nedosiahnuteľnú.

Formát paketu

RIP je enkapsulovaný v IP protokole. Jeho formát je zobrazený na obrázku č. .



Obrázok č. 10: Formát dátovej časti IP - RIP paketu

A – požiadavka / odpoveď, určuje typ správy.
B – verzia
C – nevyužíva sa
D – obsahuje informácie o smerovanom protokole, pre IP má hodnotu 2
E – IP adresa
D – metrika danej adresy

RIP podporuje „load balancing“ až cez 6 ciest s rovnakou metrikou. Prednastavené je použitie 4 ciest a ich výber sa vykonáva na základe algoritmu „round robin“.

RIP v1 má nasledujúce obmedzenia:

- „Classful Routing Protocol“ – triedny smerovací protokol.
- Vo svojich aktualizáciách neposiela údaj o maske.
- Aktualizácie posiela ako broadcast (255.255.255.255).
- Nepodporuje autentifikáciu.
- Nepodporuje VLSM a CIDR.

IGRP

Smerovací protokol vyvinutý firmou CISCO. Využíva DVA algoritmus, každému susediacemu smerovaču posiela celú a lebo časť svojej smerovacej tabuľky. Pri výpočte metriky zohľadňuje priepustnosť, záťaž, oneskorenie a spoľahlivosť linky, pričom správca siete môže určiť váhu jednotlivých parametrov. Metrika nadobúda hodnoty od 1 po 255. Slúži na smerovanie v rámci autonómneho systému.

Aktualizácie sa posielajú broadcastom každých 90 sekúnd

OSPF

- Ide o beztriedny smerovací protokol vnorený do IP protokolu
- Používa sa na IGP (Interior gateway protocol) smerovanie v rámci jedného autonómneho systému (AS)
- Veľké prepojovacie siete umožňuje rozdeliť do viacerých samostatných AS pripojených na chrbticový AS (zníženie záťaže smerovačov a prenosu medzi nimi)
- Na určenie cesty využíva metódu LSA (Link State Algorithm)
- Výberové kritériá: oneskorenie, priepustnosť, pripojiteľnosť
- Smerovaciu tabuľku generuje pomocou Dijkstrovho algoritmu

Smerovače používajúce OSPF smerovací protokol môžeme rozdeliť do 4 kategórií:

- Internal router – smerovač vnútri AS
- Area border router – smerovač na hranici AS a chrbticového AS
- Backbone router – smerovač patriaci iba do chrbticového AS
- AS boundary router – router vnútri AS, ktorý zdieľa smerovacie informácie s inými smerovačmi v iných AS

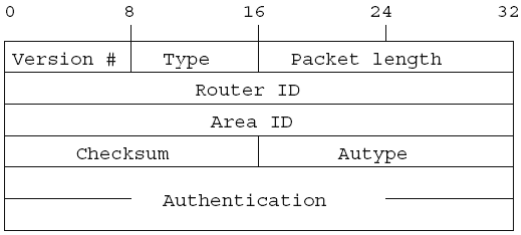
Vlastnosti smerovačov:

- V každej oblasti je jeden určený DR (designated router) prípadne jeden záložný BDR (backup DR), ktoré sa určia pre každý AS automaticky na základe vyššej priority a vyššieho ID routra (ID je vlastne IP adresa routra v AS)
- DR a BDR je logicky priľahlý ku všetkým smerovačom v rámci AS, ale vo všeobecnosti priľahlý smerovač je iba priamo susediaci smerovač
- Každý smerovač má jednu tabuľku priľahlých smerovačov a jednu smerovaciu tabuľku, ale area border router a AS boundary router má pre každý AS má samostatné pre každý AS. Každý si ešte udržiava vlastnú tabuľku (LSD) opisujúcu topológiu AS.
- Informácie medzi smerovačmi sa šíria hromadne záplavou „flood“ pomocou multicast adries (adresou 224.0.0.5 medzi priľahlými smerovačmi a adresou 224.0.0.6 medzi DR a BDR smerovačmi)
- OSPF protokol implementuje 5 typov paketov:
  - HELLO pakety
  - DDP pakety – Database Description packet
  - LSR – Link State Request

- LSU – Link State Update
- LSAck – Link State Acknowledge

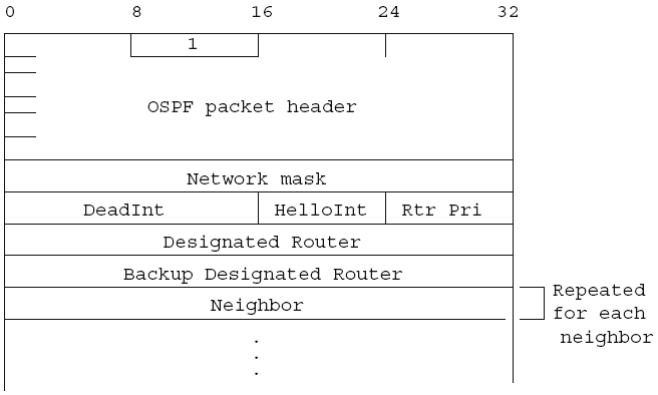
Pakety OSPF protokolu:

Pakety protokolu sú zapuzdrené do IP protokolu (pole protokol IP protokolu obsahuje hodnotu 89). Hello pakety sa slúžia na objavovanie a udržiavanie spojenia so susedmi. DDR a LSR pakety sa používajú na formovanie priľahlostí. LSU a LSAck pakety zabezpečujú obnovu LSD databázy. Každý LSU paket nesie niekoľko nových LSA (Link State Advertisements) informácií o jeden skok ďalej od ich bodu vzniku. Každá LSA obsahuje identifikátor smerovača, ktorý ju vytvoril a kontrolnú sumu. Všetky typy protokolu majú rovnaký formát hlavičky. Jeho podobu môžeme vidieť na nasledujúcom obrázku.

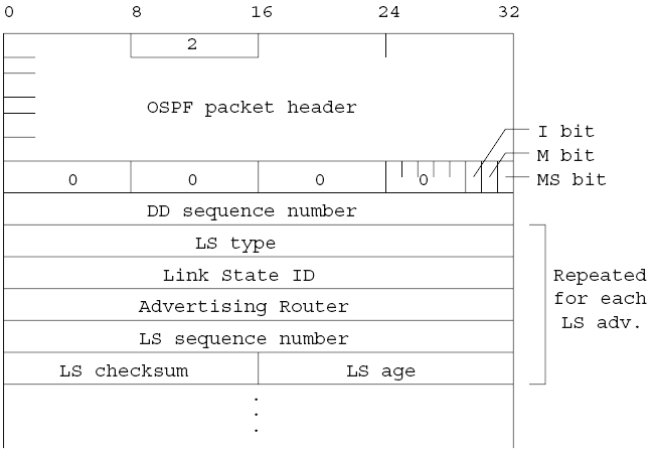


Obrázok č. 11: Formát hlavičky OSPF paketu

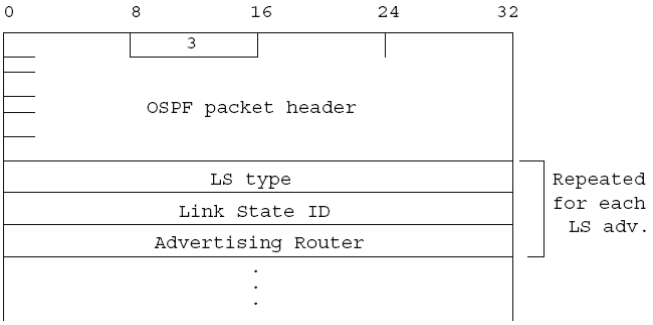
Jednotlivé typy OSPF paketu sa rozlišujú na základe poľa Type v spoločnej hlavičke. Podrobný formát zvyšných častí OSPF paketu je možné vidieť na nasledujúcich obrázkoch. Popis jednotlivých polí, ako aj ostatné informácie je možné získať z RFC dokumentácie OSPF protokolu (RFC 1131).



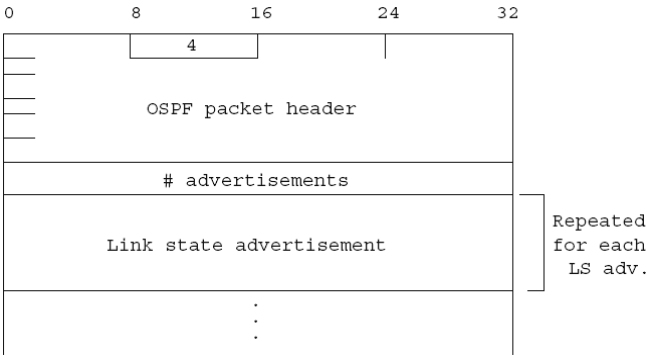
Obrázok č. 12: Formát HELLO paketu



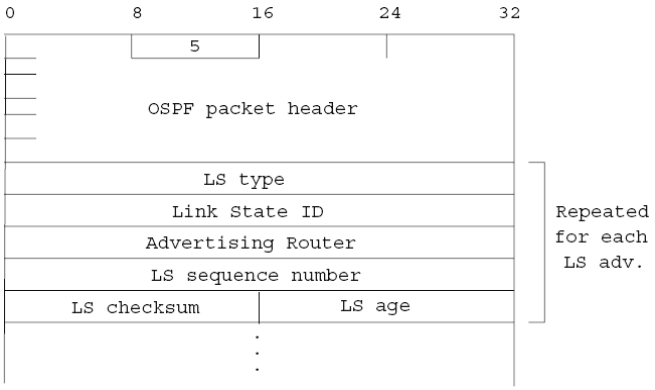
Obrázok č. 13: Formát DDR paketu



Obrázok č. 14: Formát LSR paketu



Obrázok č. 15: **Formát LSU paketu**



Obrázok č. 16: **Formát LSAck paketu**

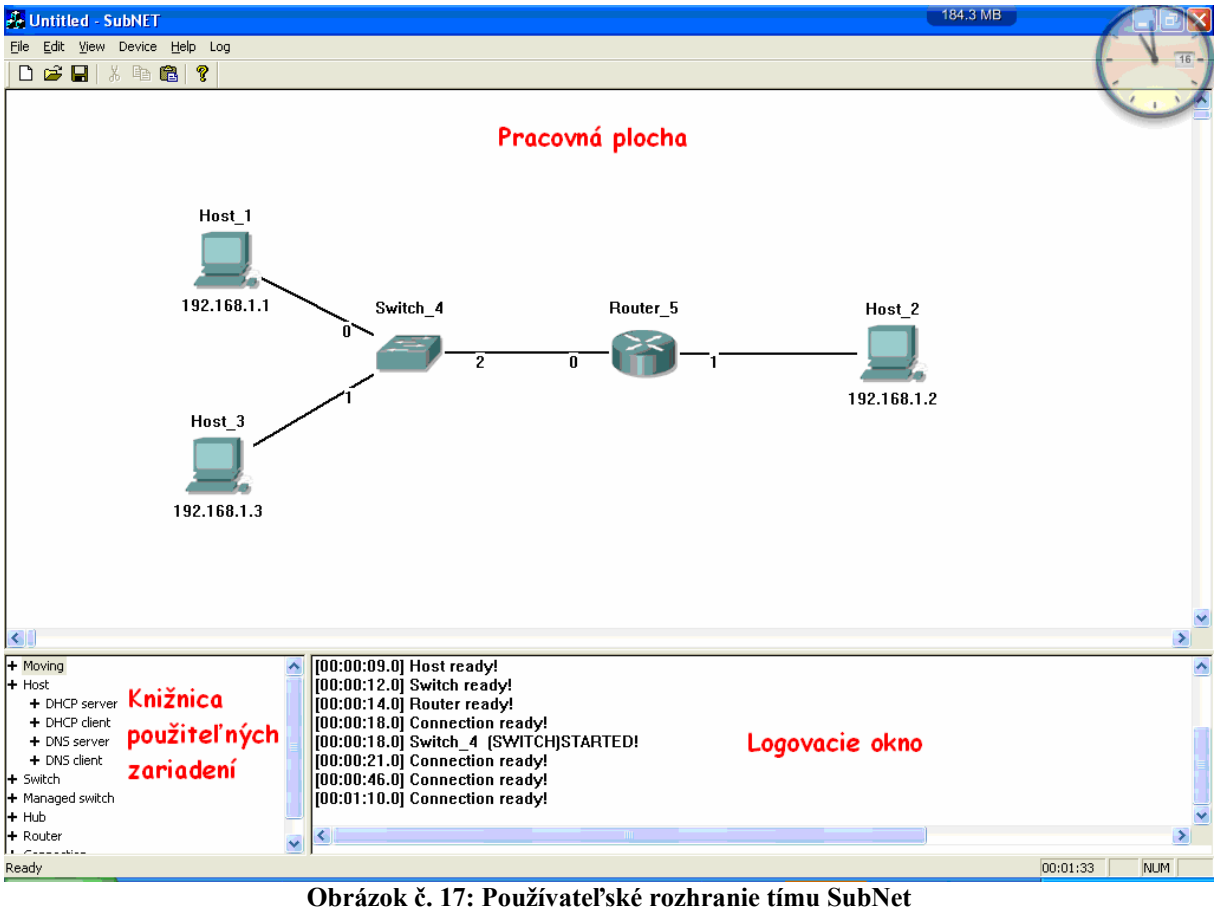
2.2 Analýza konkurenčných produktov

Pri príprave na projekt sme sa nevyhli ani štúdiu už existujúcich riešení. Jednalo sa o riešenie od tímu, ktorý riešil podobný problém minulý rok na predmete Tímový projekt na FIIT STU, ako aj o riešenia od komerčných subjektov.

2.2.1 SUBNET

SUBNET je výtvorom tímu, ktorého zadanie projektu bolo totožné s našim. Aplikácia je použiteľná pre výukový proces a pre simuláciu počítačovej siete.

2.2.1.1 Používateľské rozhranie



Obrázok č. 17: Používateľské rozhranie tímu SubNet

Na prvý pohľad je aplikácia používateľsky príjemná, avšak počas práce s ňou sa vyskytujú určité nedostatky:

Pre vytváranie topológie je na ľavej časti obrazovky strom, ktorý obsahuje jednotlivé prvky siete. Veľkosť tohto stromu je nedostatočná a je nutné ho posúvať, čo spomaľuje tvorbu topológie.

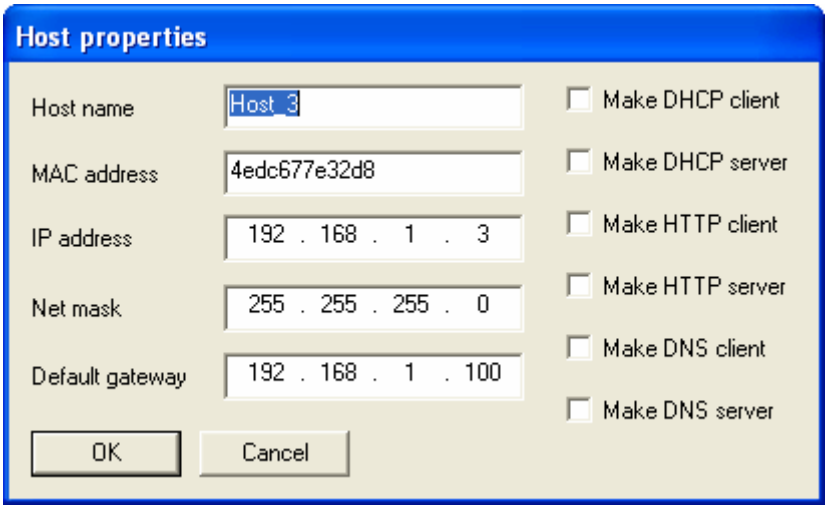
Rýchlosť kreslenia stromu siete je obmedzená a po pridaní nejakého prvku siete je nutné čakať asi 1 sekundu, kým je možné pridať ďalší prvok (dôvod je popísaný v časti implementácia).

Aplikácia umožňuje pridávanie rôznych typov sieťových zariadení ako opakovač, prepínač, smerovač a linkové spojenie. Umožňuje tiež vytvárať užívateľské prvky siete odvodené od základných prvkov a tiež ukladanie a načítanie topológie do / zo súboru.

Klady a zápory jednotlivých prvkov siete:

Connection : + Dá sa spustiť „Sniffer“ na každom spojení
– Neumožňuje žiadne nastavenia pre dané spojenie
– Nedá sa spustiť na dvoch a viacerých spojeniach súčasne

Host: + Každý host má konzolu, kde si môžeme otestovať základné príkazy: ping, ipconfig, arp, tracert
– Možnosti DHCP klient server, DNS klient server, HTTP klient server boli nefunkčné a spôsobovali pád aplikácie



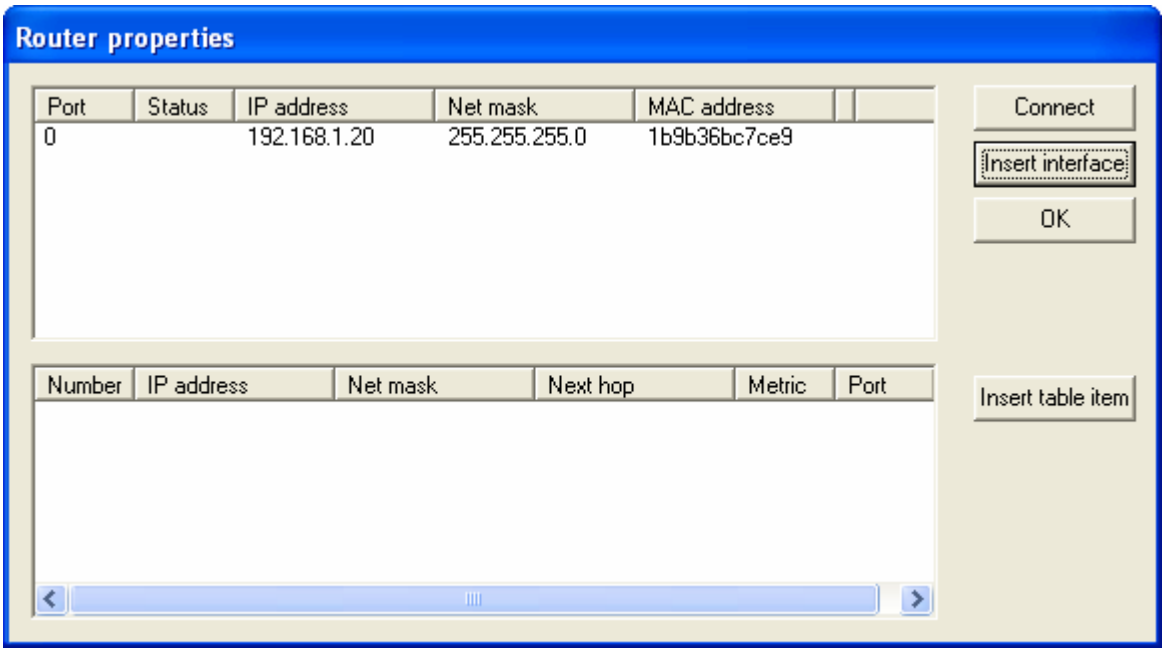
Obrázok č. 18: Nastavenie stanice

Switch: – Možnosť nastavenia len MAC adresy, žiadny rozdiel medzi manažovateľným a obyčajným switchom
– Nedokážu zobraziť tabuľku MAC adries a portov

Router: + Dá sa zmeniť metrika ale vzhľadom na RIP je to zbytočné
+ Príkazy konzoly: enable, helo, ping, show (history, route), tracert
– Nutnosť manuálneho pridania interfejsu s MAC, IP, maskou, portom

– Smerovacia tabuľka je len statická

– Čo sa raz vytvorí, nedá sa odstrániť (položka smerovacej tabuľky, interfejs)



Obrázok č. 19: Nastavenie smerovacej tabuľky smerovača

Použité protokoly:

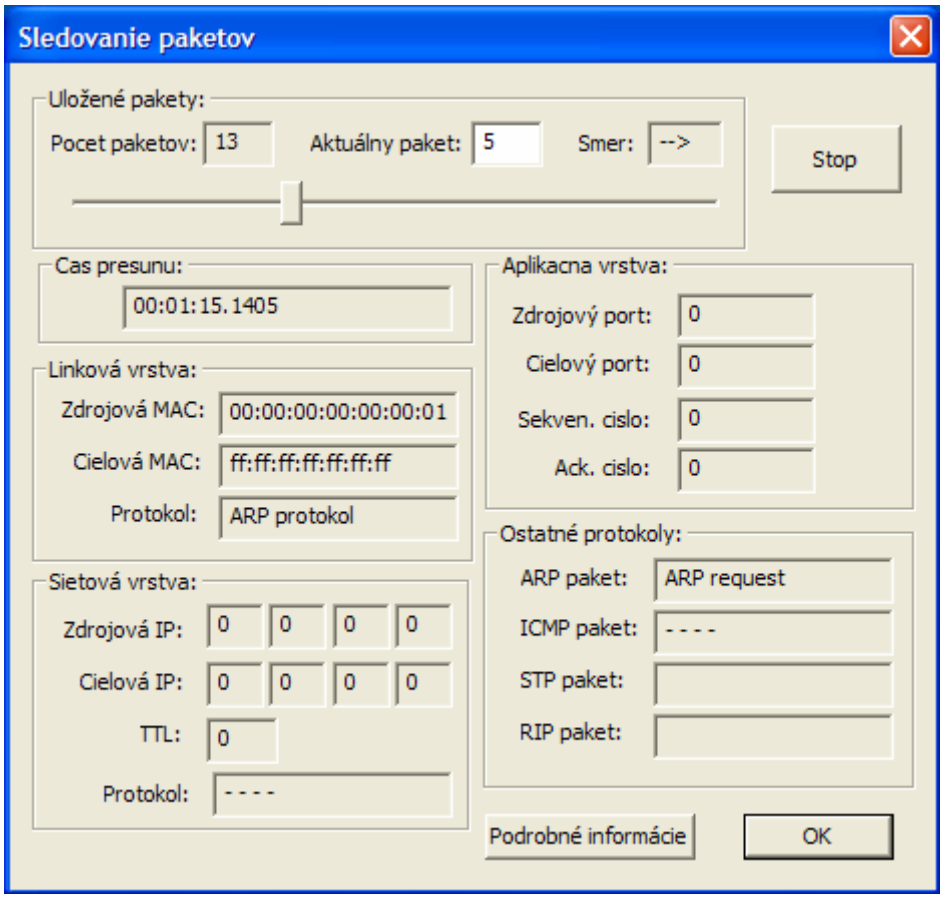
STP	-	nefunkčné
ARP	+	funkčné
ICMP	+	funkčné
RIP	-	nie celkom funkčné

Konzola:

Na každom zariadení je možné spustiť konzolu s príkazovým riadkom, kde je možné zadávať vopred definované príkazy. Vzhľadom na to že niektoré funkcie aplikácie nie sú funkčné, tak ani tieto príkazy konzola nerozoznáva.

Ďalšie vlastnosti:

V analyzátore paketov sa pri icmp paketoch nezobrazuje zdrojová a cieľová IP adresa. Pri vytvorení viacerých spojení medzi dvomi rovnakými zariadeniami nie je vidieť, či to predchádzajúce spojenie sa odstránilo alebo ostalo zapojené. Aplikácia neumožňuje zobrazovanie akýchkoľvek štatistík.



Obrázok č. 20: Analyzátor paketov

Pretože sme sa rozhodli pokračovať v zdokonaľovaní tejto aplikácie, ktorej základ tvorí silný a prepracovaný model simulovania siete, uvádzame analýzu implementačnej časti tejto aplikácie.

2.2.1.2 Analýza Implementácie

Použité knižnice

Tvorcovia tohto programu použili knižnicu TINY na modelovanie procesných udalostí. Táto knižnica obsahuje rôzne generátory náhodných čísel. Umožňuje výpisy štatistík a má v sebe implementované rôzne typy dátových zásobníkov ako FIFO rady a obslužné miesta.

Táto knižnica je primárne určená na modelovanie systémov pomocou udalostí avšak disponuje aj podporou procesného modelovania. V plnej miere implementuje model simulačného času ako aj programátorské rozhranie (API) v jazyku C++. Je použiteľná na platforme UNIX aj MS Windows. Disponuje aj generátormi náhodných čísel s rôznym typom

rozdelení, ktoré autori použili na implementáciu stochastických udalostí (chyby pri prenose). Ďalej umožňuje zbierať z odsimulovaného systému rôzne štatistické údaje, potrebné pre ďalšiu analýzu správania sa systému.

Na implementovanie CLI využívajú autori knižnicu Readline. Táto umožňuje programátorom vytvárať funkcie na dopĺňanie ľubovoľných informácií nielen názvov súborov. Knižnica Readline obsahuje viaceré funkcie, ktoré boli pre tento projekt potrebné. Zahŕňa prácu s rôznymi terminálmi, nízko úrovňovú správu prijímaných znakov a umožňuje programátorovi vytvárať" vlastné funkcie na doplňovanie príkazov a prácu s históriou príkazov.
Túto knižnicu autor využil na :

- vytvorenie' funkcie na doplňovanie príkazov,
- kontextovú nápovedu a prácu s históriou.

Vo všeobecnosti knižnica Readline ponúka sadu príkazov pre manipuláciu s textom, ktorý je vpísaný do príkazového riadku, umožňujúc jeho opravu v prípade chyby, namiesto celého prepísania riadku.

V knižnici sú implementované funkcie pre:

- pohyb po riadku – k týmto príkazom patria aj prislúchajúce klávesové skratky. Napr. presun na začiatok riadku – Ctrl + a. Presun na koniec riadku – Ctrl + e atď.
- tzv. killing príkazy – tie umožňujú zmazanie textu z riadku s tým, že sa uloží a neskôr môže opäť vložiť. Čiže metóda – copy paste. Taktiež sú nadefinované klávesové skratky napr. Ctrl+k znamená kopírovanie a Ctrl + y je vloženie textu.
- príkazy pre hľadanie histórie
- príkazy pre menenie textu
- príkazy pre dopĺňanie – pomocou tlačidla TAB
- je možné posielať do príkazov numerické argumenty, ktoré špecifikujú zopakovanie príkazu
- vytvorenie tzv. inicializačného súboru, ktorým sa nadefinujú určité vlastnosti správania sa knižnice

Triedy

Kvôli prehľadu použitých tried uvádzame diagram tried, z ktorého budeme vychádzať. Z diagramu je vidieť, že boli implementované knižnice pre prácu s protokolmi Ethernet, IP, ARP, ICMP, UDP, TCP smerovací protokol RIP a protokol STP.

Sú to triedy :

C_Protocol - abstraktná trieda, zahrňuje potrebné atribúty a metódy pre všetky protokoly

C_ProtSTP - umožní používanie spanning-tree algoritmu pre switch-e

C_ProtIP - trieda zabezpečujúca komunikáciu pomocou IP protokolu

C_ProtICMP - trieda zabezpečujúca komunikáciu pre službu ping

C_ProtRIP - každý router môže túto triedu používať na realizáciu RIP protokolu

C_ProtARP - trieda, pomocou ktorej je možné používať ARP protokol

C_ProtTCP - trieda zabezpečujúca komunikáciu pomocou TCP protokolu

C_ProtUDP - trieda zabezpečujúca komunikáciu pomocou UDP protokolu

C_Packet - trieda ktorá definuje paket, jeho dĺžku a dáta

C_ProtEthernet - objekt protokolu ethernet poskytuje rozhranie pre prácu s ethernetovým rámcom

V týchto triedach sú implementované metódy pre jednoduchú prácu s paketmi jednotlivých protokolov, pre získavanie konkrétnych dát z týchto paketov a pre vytváranie paketov s konkrétnymi dátami a parametrami. Veľmi dobre je implementované vytváranie rámcov od horných vrstiev k nižším a tiež naopak.

Pre zariadenia boli navrhnuté tieto triedy:

C_ArpCache - objekt C_ArpCache poskytuje metódy pre prácu s tabuľkou Arp

C_CPU - objekt spracovania - facility

C_Device - všeobecná trieda pre zariadenia:

C_Host

C_Hub

C_Router

C_ManagedSwitch

C_Switch

Zariadenia odvodené od všeobecnej triedy **C_Device**. Každé zariadenie má svoje konkrétne parametre ako počet portov, interfejsov ,svoje obslužné miesto **CFacilty** a svoje služby. Obslužné miesto je proces, ktorý je budený iným procesom, napríklad procesom z interfejsu ak naň prišli nejaké pakety. Tento zobudený proces potom môže napr. vybrať všetky pakety z interfejsov, ktoré mali niečo vo vstupných FIFO radoch svojich potov a poslať ich po spracovaní na iné interfesjsi (výstupné FIFO rady portov pre dané interfejsi).

C_Interface - všeobecná trieda pre rozhranie na niektorom zariadení :

C_InterfaceHost

C_InterfaceHub

C_InterfaceMSwitch

C_InterfaceRouter

C_InterfaceSwitch

Každé zariadenie má svoj vlastný typ interfejsu odvodený od jedného typu C_Interface. Interfejs má svoj port CPort, svoje vlastné obslužné miesto CFacility, ktoré vykonáva určitú činnosť ak do vstupných FIFO radov jeho portov prišli nejaké pakety. (Napr. zobudí obslužný proces zariadenia). Každý interfejs má tiež FIFO rad pre pakety prichádzajúce z IP vrstvy.

C_Port - miesto pre pripojenie spojenia

C_RouteTable - poskytuje metódy pre prácu so smerovacou tabuľkou

Príkazy pre konzolu:

C_Tracert

C_Ping

Pre služby boli navrhnuté tieto triedy:

C_AppService - všeobecná trieda pre aplikačné služby

C_Service - všeobecná trieda pre služby:

C_ServiceDHCP

C_ServiceDNS

C_ServiceHTTP

C_ServiceICMP

C_ServiceIPHost

C_ServiceIPRouter

C_ServiceRIP

C_ServiceSTP

C_ServiceTCP

C_ServiceUDP

C_TcpSession

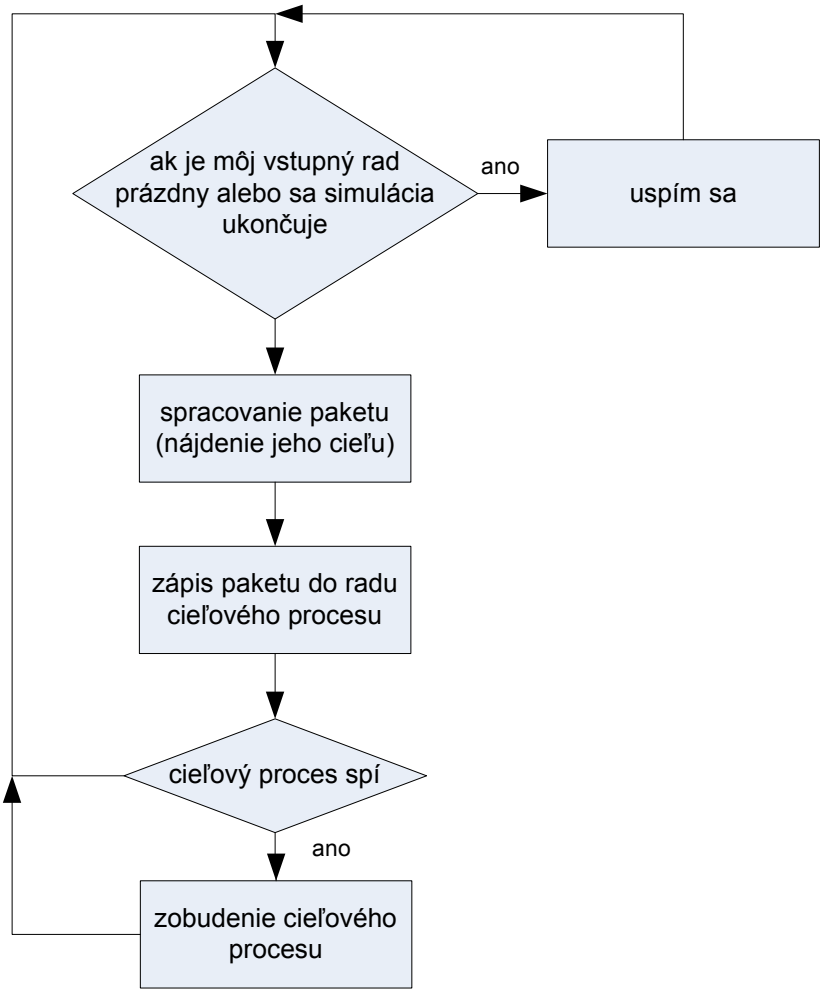
Služby sú odvodené od všeobecnej triedy **C_Service.** Každá služba má jedno obslužné miesto **CFacility,** jeden proces spracovania, a dva rady jeden, do ktorého procesy dávajú pakety na výstup a jeden, do ktorého procesy dávajú pakety na vstup. Obslužný proces vyberá pakety zo vstupného radu, vykonáva potrebné činnosti ako napr. smerovanie a posiela vytvorený paket do výstupného radu. (ak je nutné tak budí niektorý iný obslužný proces napr. IP proces).

Procesy

Pre potreby aplikácie sú vytvorené dva procesy Refresher a Loader. Úlohou Refreshera je každú sekundu kontrolovať, či neboli pridané nejaké zariadenia a ak boli tak ich inicializuje pridá ich stromu. Po tom ako skontroloval prítomnosť nových zariadení, pozdrží vykonávanie samého seba na 1 sek. a potom pokračuje v nekonečnom cykle dokola.Úlohou Loadera je uskutočnenie zmeny v prípade pridania interfejsu nejakého zariadenia, pridaním smerovacieho pravidla poprípade pri zmene nejakej služby.

Skelet procesu

Na obrázku č 10 je vidieť kostru, z ktorej pozostáva každý proces (s výnimkou procesu Refresh). Ide o „nekonečnú“ slučku, v ktorej proces kontroluje svoje vstupné rady, spracováva požiadavky a zapisuje odpovede do radov iných procesov, ktoré následne aj „budí“ ak je to potrebné.



Obrázok č. 21: Skelet procesu

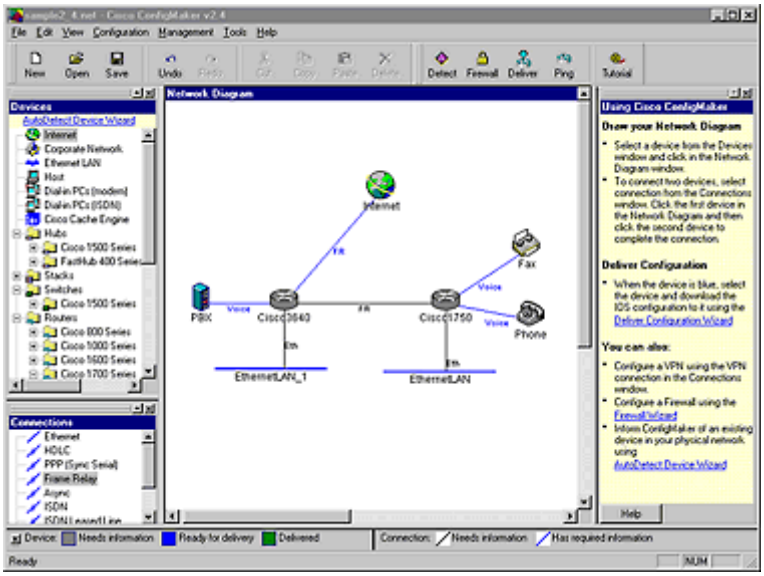
2.3 Komerčné riešenia

Komerčné riešenia, ktoré sme analyzovali mali rôzne zameranie. Niektoré z nich boli voľne prístupné (aj keď po registrácii na stránke firmy, ktorá ich produkuje), iné mali dostupné len demo verziu a z iných bol k dispozícii len produktový list. Tak isto bolo rôzne zameranie produktov – niektoré z nich sú zamerané na konfiguráciu siete, pričom neumožňujú jej simuláciu, kým iné nemajú také možnosti konfigurácie zariadení, ale umožňujú simuláciu. Testované produkty boli: Cisco ConfigMaker, Cisco Network Assistant, Gambit Virtual Lab a RouterSim Network Visualizer.

2.3.1 Cisco ConfigMaker a Cisco Network Assistant

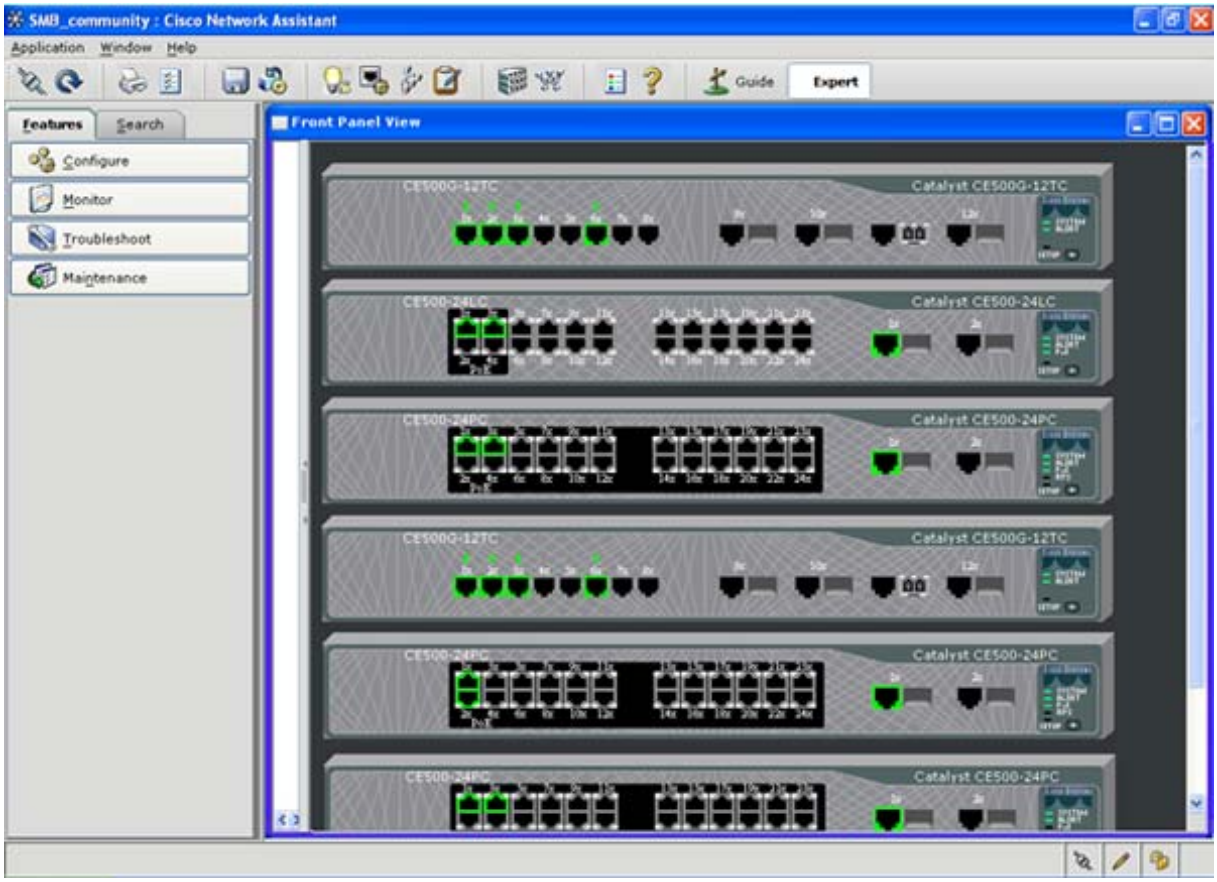
Oba tieto produkty pochádzajúce od jednej firmy (Cisco Systems, Inc.) majú veľmi podobné zameranie, a preto sú popísane spolu. Obe aplikácie sú dostupné po registrácii na firemnej stránke a sú zamerané na podporu vytvorenia a konfigurácie malých až stredných sietí vybudovaných za použitia Cisco zariadení. V programoch sa dá (za použitia grafického

užívateľského prostredia) vybudovať požadovaný model siete pozostávajúci z počítačov, opakovačov, prepínačov a smerovačov (podporované sú Cisco zariadenia zo sérií 800, 1000, 1600, 1700, 2500, 2600, 3600 a 4000). Programy umožňujú vytvorenú konfiguráciu zariadení uložiť do súboru a následne načítať do Cisco zariadenia (čo uľahčí následnú konfiguráciu zariadení), resp. v prípade Cisco Network Assistant rovno nahrať po sieti do zariadenia.



Obrázok č. 22: Ukážková obrazovka programu Cisco ConfigMaker

Program pozostáva z hlavného okna, v ktorom sa nachádza model siete. V ľavo od neho sú dve okná v ktorých sa nachádzajú rôzne typy zariadení, ktoré sa dajú pridať do modelu siete a taktiež rôzne druhy spojení.



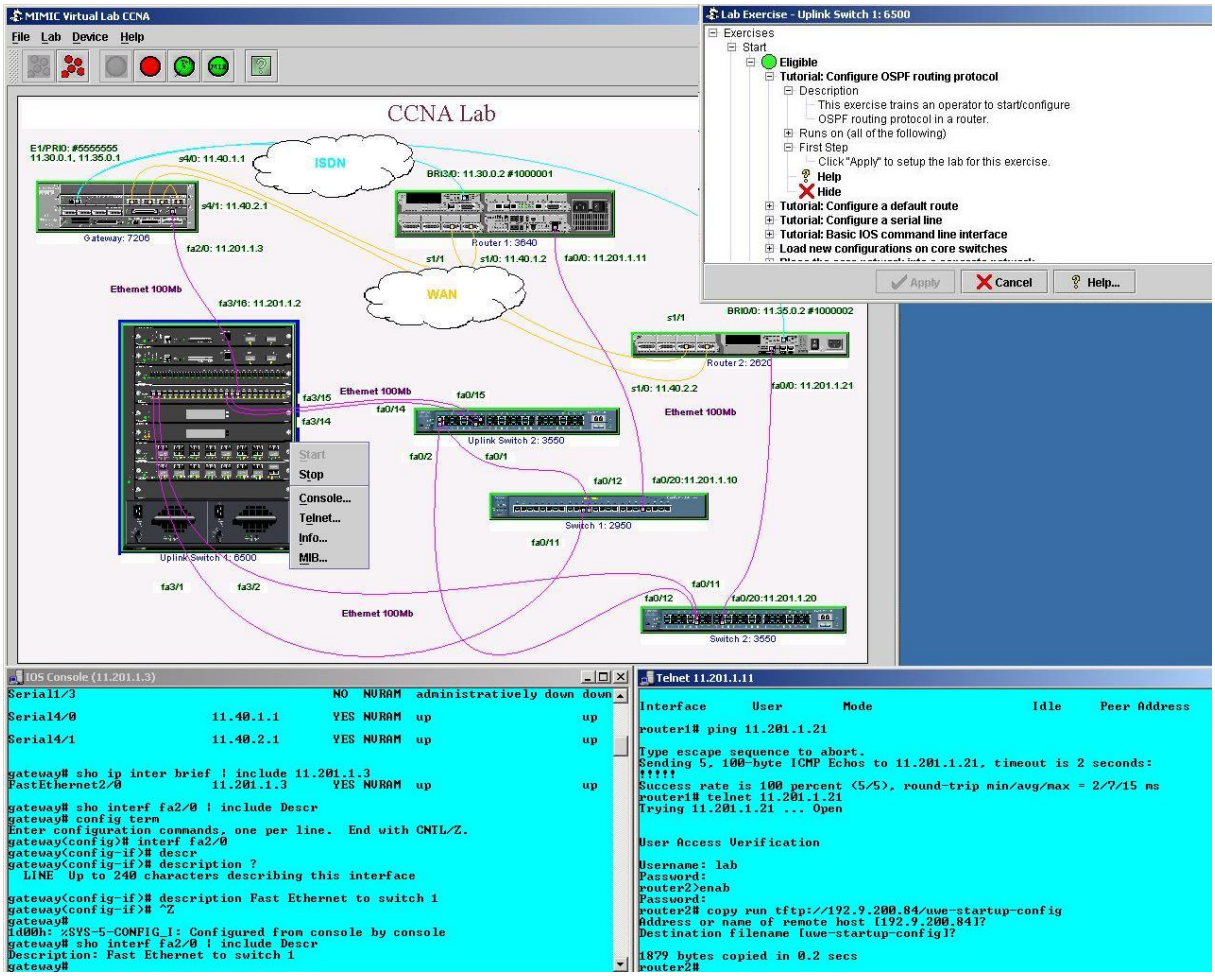
Obrázok č. 23: Ukážková obrazovka programu Cisco Network Assistant.

2.3.2 Gambit Virtual Lab

Tento produkt je určený na výuku v programe Cisco Network Academy (CCNA), a preto sa snaží maximálnej miere zachovávať správanie Cisco produktov. Umožňuje vytvorenie modelu siete pozostávajúcej z maximálne 7 sieťových zariadení (Cisco smerovače (2620, 3640 a 7206) a prepínače (2950, 3550 a 6500), ktoré môžu byť navzájom pospájané spojeniami používanými v LAN, WAN a ISDN. V demo verzii nie je možné si vytvárať vlastné modely sietí, ale len načítať niektorý z predpripravených modelov dodávaných spolu s produktom. Konfigurácia zariadení sa uskutočňuje pomocou konzolového rozhrania, pričom simulátor podporuje plnú sadu príkazov Cisco IOS. Produkt umožňuje nasledovné operácie:

- práca v rôznych módoch – užívateľský, privilegovaný, configuračný a mód nastavovania rozhraní
- nastavenie hesla, IP adries, meno, šírka prenosového pásma
- nastavenie smerovacích protokolov – RIP, IGRP, EIGRP, BGP, OSPF, IS-IS
- „ping“-ovanie zariadení v sieti
- uloženie/načítanie konfigurácie
- štartovanie zariadenia z „flash“ pamäte alebo z TFTP serveru

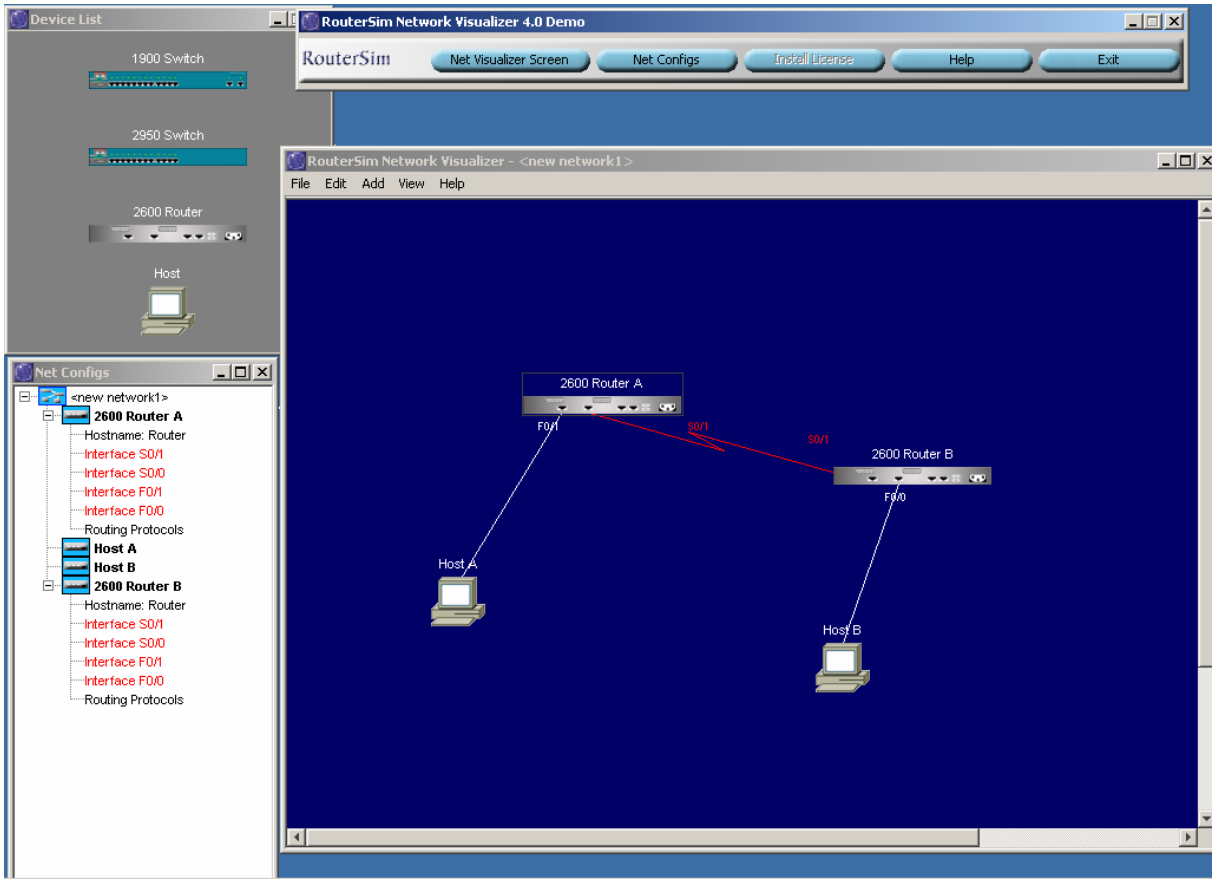
- vzdialené „logovanie“ cez SNMP
- konfigurácia ISDN, CDP, PPP, Frame Relay, ACL a NAT



Obrázok č. 24: Ukážková obrazovka programu znázorňujúca model siete a konzoly prepínačov.

2.3.3 RouterSim Network Visualizer

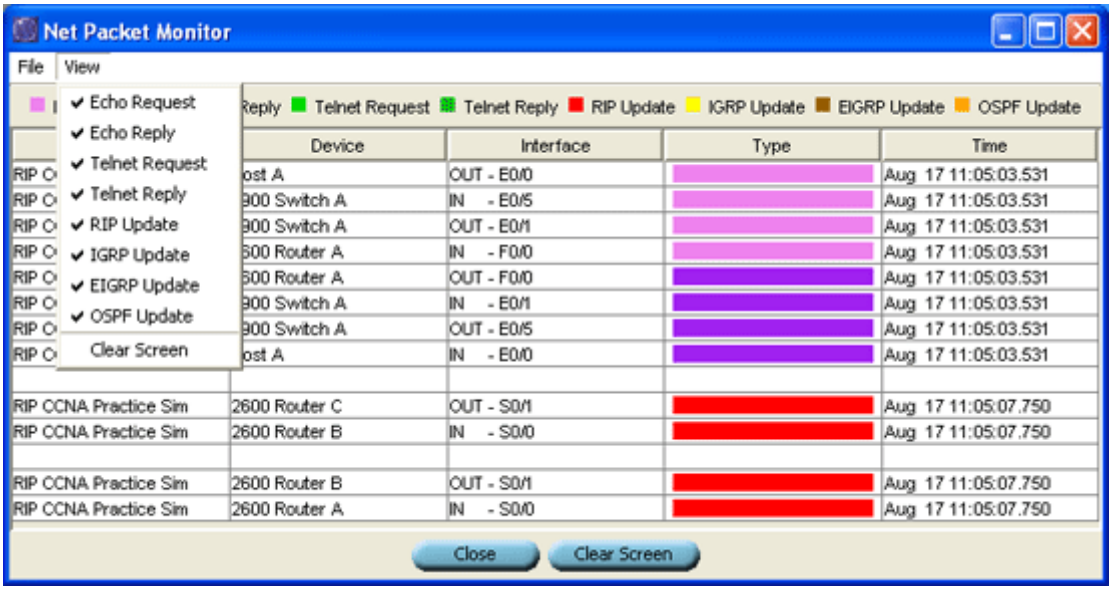
Tento program je dostupný v podobe demo verzie a je zameraný taktiež na výuku v CCNA kurzoch. V tejto demo verzii je k dispozícii len možnosť pridania dvoch smerovačov (dve sériové a 2 ethernet rozhrania) a dvoch počítačov, ktoré je možné konfigurovať pomocou konzoly (pričom sa ale konfigurácia zobrazuje aj v špeciálnom okne).



Obrázok č. 25: Ukážkové okno z programu RouterSim Network Visualizer 4.0 Demo s pridaným maximálnym počtom zariadení.

Pri zakúpení je k dispozícii už verzia 5.0, ktorá má podľa výrobcu množstvo vylepšení a má nasledovné vlastnosti:

- neobmedzený počet modelov sietí ako aj zariadení v modeli (obmedzené len dostupnými zdrojmi počítača)
- podporované zariadenia: Cisco 2621 smerovač s Enterprise edition 12.x softwéru; Cisco 2950 prepínač s 12 10/100 portami; Cisco Catalyst 1912EN prepínač s Enterprise edition softwéru; Cisco 3550 prepínač s 10 10/100 portami; koncové počítače
- umožňuje sledovať komunikáciu zariadení po sieti – pomocou Net Packet Monitoru (viď nasledujúci obrázok)



Obrázok č. 26: Zobrazenie zachytených paketov

- plná emulácia IOS-u simulovaných Cisco zariadení (napr. podpora smerovacích protokolov RIP, RIP v.2, OSPF, EIGRP), podpora NAT, …)

2.3.4 Zhrnutie

Program RouterSim Network Visualizer (spolu s programom Gambit MIMIC Simulator Suite, čo je rozšírená verzia programu Virtual Lab, ku ktorému ale bol k dispozícii len produktový list, a preto je veľmi ťažké hodnotiť jeho funkcie) spĺňajú požiadavky kaldené na simulátor siete (použiteľný napr. pri výučbe). Ale ich zásadná nevýhoda je značná cena, ktorú je potrebné zaplatiť za licencie.

3 Špecifikácia požiadaviek

Na základe vyhotovenej komplexnej analýzy a analýzy predchádzajúceho riešenia tímu SUBNET z minulého roku, sme si stanovili zoznam požiadaviek, ktoré sa budeme snažiť splniť podľa možností v čo najväčšej miere. Vo všeobecnosti ide o dopracovanie a prípadné rozšírenie predchádzajúceho riešenia. Nasledovná špecifikácia preto vychádza zo špecifikácie aplikácie SUBNET a obsahuje iba body, ktoré je podľa nás vhodné oproti pôvodnému projektu doriešiť, pozmeniť alebo doplniť, ako aj body, o ktoré plánujeme predchádzajúce riešenie rozšíriť. Stanovené požiadavky sú rozdelené do nasledovných kategórií:

3.1 Používateľské prostredie

- Ošetrenie uloženia a načítania modelu do a zo súboru
- Pridanie stromu aktuálne použitých zariadení v sieťovom modeli
- Vytvorenie rýchleho konfiguračného bloku
- Optimalizácia užívateľského prostredia a jeho ovládania
- Predvolená automatická konfigurácia zariadení pridávaných do modelu siete

3.2 Implementácia zariadení

Spojenie:

- Nastavenie parametrov prenosu prepojenia (chybovosť – BER, typ – serial, ethernet)
- Sniffovací dialóg – prepracovanie obsahu okna
- Sniffovací dialóg – ošetriť možnosť sledovať viaceré spojenia súčasne
- Vyznačenie prenosovej trasy sledovaných paketov
- QOS štatistiky prenosovej linky

Koncové zariadenie (host):

- Implementácia telnet spojenia medzi hostami (server a klient)

Prepínač (switch):

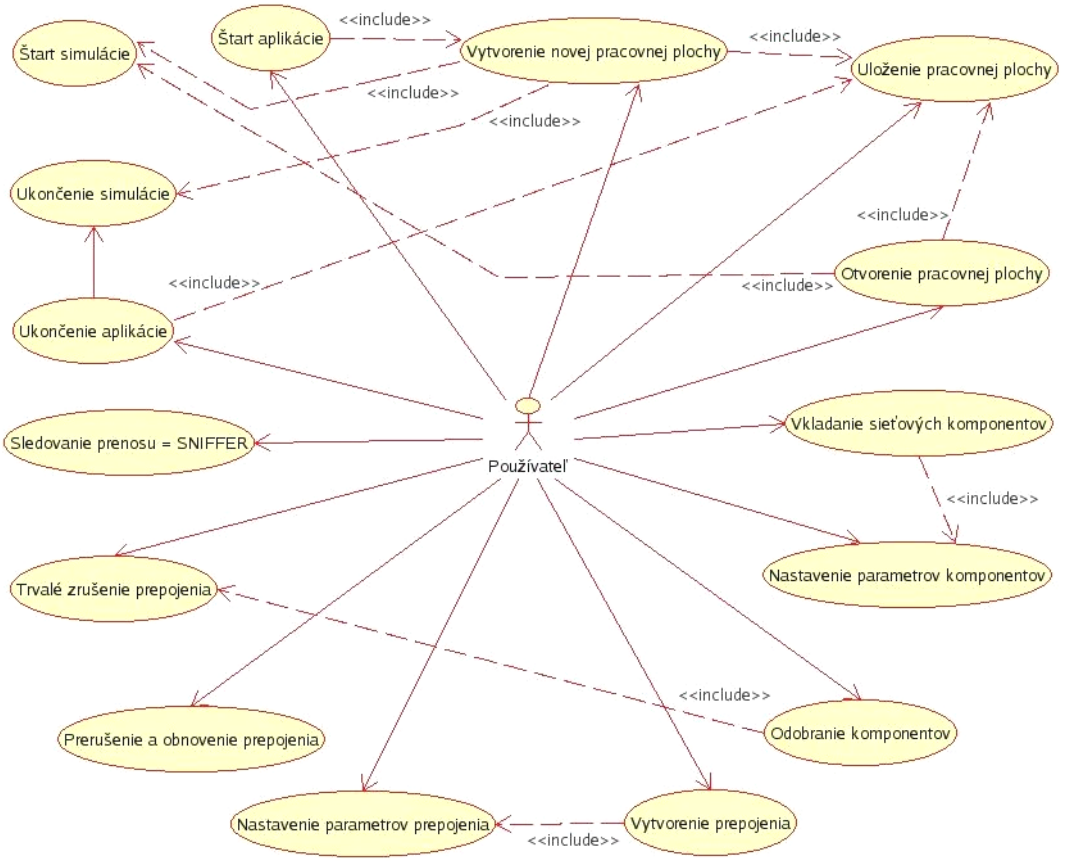
- Zobrazenie smerovacej tabuľky
- Implementácia manažovateľného prepínača (VLANy, port security)
- Implementácia STP protokolu

Smerovače (router):

- Ošetrenie sieťových rozhraní smerovačov – preddefinovanie počtu rozhraní
- Implementácia smerovacích protokolov RIP, IGRP, OSPF

3.3 Prípady použitia

Diagram prípadov použitia ako aj samotné prípady použitia sa od pôvodne navrhovanej špecifikácii líšia z dvoch dôvodov. Prvý dôvod je, že predchádzajúca implementácia obsahuje rozdiely oproti pôvodnej špecifikácii a druhý je doplnenie a aplikácia nami navrhovaných zmien do funkcionálnych vlastností programu. Výsledná podoba diagramu prípadov použitia je zobrazená na obrázku č. 26.



Obrázok č. 27: **Aktualizovaný a upravený diagram prípadov použitia**

Keďže sa jednotlivé prípady použitia viac či menej líšia od pôvodných a niektoré boli vypustené, uvádzame podrobný popis jednotlivých udalostí s ich fungovaním.

3.3.1 Štart aplikácie

Spustením aplikácie sa vykoná proces štartu aplikácie. Ten v sebe zahŕňa načítanie a spustenie programu a následné vytvorenie novej pracovnej plochy (viď Vytvorenie novej pracovnej plochy). Tým sa zároveň spustí simulácia pre aktuálnu pracovnú plochu.

3.3.2 Vytvorenie novej pracovnej plochy

Zvolením možnosti „Nový“ v časti „Súbor“ hlavného menu sa vytvorí nové pracovné prostredie, v ktorom je možné vytvoriť model siete, ktorý bude simulovať sieťový prenos. Táto možnosť automaticky zahŕňa funkciu spustenia simulácie (viď Štart simulácie). Ak v aktuálna pracovná plocha obsahovala nejaký vytvorený, alebo otvorený sieťový model, pred vytvorením prázdnej pracovnej plochy sa vykonajú prípady použitia Ukončenie simulácie a Uloženie pracovnej plochy.

3.3.3 Štart simulácie

V tomto prípade sa v pozadí aplikácie vytvorí nová simulácia, ktorá bude obsahovať modely zariadení a spojení z aktuálne otvoreného sieťového modelu v pracovnom prostredí aplikácie a následne sa začne jej vykonávanie.

3.3.4 Uloženie pracovnej plochy

Tento prípad môže nastať dvomi spôsobmi. V prvom sa predpokladá, že ho vyvolá samotný používateľ zvolením možnosti „Uložiť“ v časti „Súbor“ hlavného menu aplikácie. V druhom prípade je vyvolaný iným prípadom použitia, kedy po jeho aktivácii sa používateľ upozorní na fakt, že pracovná plocha bola zmenená a ponúkne sa mu možnosť, či ju chce používateľ uložiť, alebo zahodiť zmeny. Ak užívateľ ignoruje zmeny, pokračuje vykonávanie situácie, ktorá tento prípad vyvolala. Ak používateľ si zvolí uloženie, prípad pokračuje spoločne tým, že sa otvorí okno so stromovou štruktúrou, kde si užívateľ zvolí miesto a názov súboru, do ktorého sa uloží pracovná plocha. Ak pracovná plocha bola už vopred uložená, používateľovi sa neotvorí možnosť zvoliť miesto, kde sa plocha uloží, ale automaticky sa uloží do súboru s pôvodným názvom a umiestnením na disku.

3.3.5 Otvorenie pracovnej plochy

Používateľ si v hlavnom menu aplikácie v časti „Súbor“ zvolí možnosť „Otvoriť“, čím sa začne proces otvárania uloženej pracovnej plochy. V rámci neho sa testuje obsah aktuálnej

pracovnej plochy. Ak aktuálna plocha obsahuje nejaký model, aktivuje sa prípad Uloženie pracovnej plochy. Po prípadnom uložení sa otvorí dialógové okno, v ktorom si používateľ v súborovej štruktúre nájde súbor s uloženou pracovnou plochou. Po výbere a stlačení tlačidla „Otvoriť“ sa v pracovnej ploche zobrazí načítaný model a aktivuje prípad Štart simulácie.

3.3.6 Ukončenie simulácie

V tomto prípade sa ukončia všetky simulačné procesy, prípadne vyhodnotia globálne štatistiky. Zároveň sa zo simulácie odstránia všetky mechanizmy používané pri simulácii sieťového modelu.

3.3.7 Ukončenie aplikácie

Tento prípad nastane, keď používateľ ukončí beh samotnej aplikácie. Jeho úlohou je zastavenie simulácie a v prípade, že aplikácia obsahuje vytvorený model siete, ponúkne používateľovi možnosť uložiť sieťový model do súboru.

3.3.8 Vkladanie sieťových komponentov

Výberom sieťového komponentu v bloku knižnice sieťových komponentov má používateľ možnosť pridať rôzne druhy sieťových zariadení do modelu siete, ktorý slúži na simulovanie prenosu. Pridaním komponentu do modelu sa zároveň vytvoria údajové štruktúry slúžiace na simuláciu prenosu a uloženie nastavení. Pridaným zariadeniam sa automaticky nastavia preddefinované nastavenia podľa svojho druhu aktiváciou prípadu použitia Nastavenie parametrov komponentu.

3.3.9 Nastavenie parametrov komponentu

Ak tento prípad použitia vyvolal prípad použitia Vkladanie sieťových komponentov, nastavenia sa aplikujú automaticky podľa preddefinovaných možností. Ak tento prípad vyvolal používateľ kliknutím pravého tlačidla myši, pre zariadenia manažovateľný prepínač alebo smerovač sa otvorí konzola pre podrobné nastavenie zariadení, inak sa otvorí dialógové okno, v ktorom sa nastavia základné nastavenia zariadenia. Tento prípad zároveň zastrešuje možnosť otvorenia konzoly na zariadení, v ktorej sa zadávajú príkazy sieťovej komunikácie.

3.3.10 Odobratie komponentu

Týmto prípadom použitia sa zastrešuje operácia odobratia sieťového prvku z modelu siete. Tým sa zo simulačného modelu odstránia všetky štruktúry vlastné odstraňovanému modelu. Ak bolo na komponent napojené sieťové spojenie, pre toto spojenie sa zároveň vyvolá prípad použitia Trvalé zrušenie prepojenia.

3.3.11 Vytvorenie prepojenia

Výberom sieťového prepojenia v bloku knižnice sieťových komponentov má používateľ možnosť prepojiť dve sieťové zariadenia. Typ prepojenia sa detekuje automaticky. Tento prípad zároveň aktivuje prípad použitia Nastavenie parametrov prepojenia pre aktuálne vytvárané prepojenie.

3.3.12 Nastavenie parametrov prepojenia

Ak tento prípad použitia vyvolal prípad použitia Vytvorenie prepojenia, nastavenia sa aplikujú automaticky podľa preddefinovaných možností. Ak tento prípad vyvolal používateľ kliknutím pravého tlačidla myši na sieťovom spojení, otvorí sa dialógové okno, v ktorom užívateľ nastaví základné vlastnosti prenosu cez dané spojenie.

3.3.13 Prerušenie a obnovenie prepojenia

Tento prípad zastrešuje dve operácie nad sieťovým prepojením. Po kliknutí pravého tlačidla myši má používateľ možnosť prerušiť fungujúce prepojenie a tým pozastaviť na ňom prenos, resp. obnoviť prenos na tomto spojení. Deaktiváciou prepojenia ostane prepojenie pripojené, avšak žiadna komunikácia ním nebude prechádzať.

3.3.14 Trvalé zrušenie prepojenia

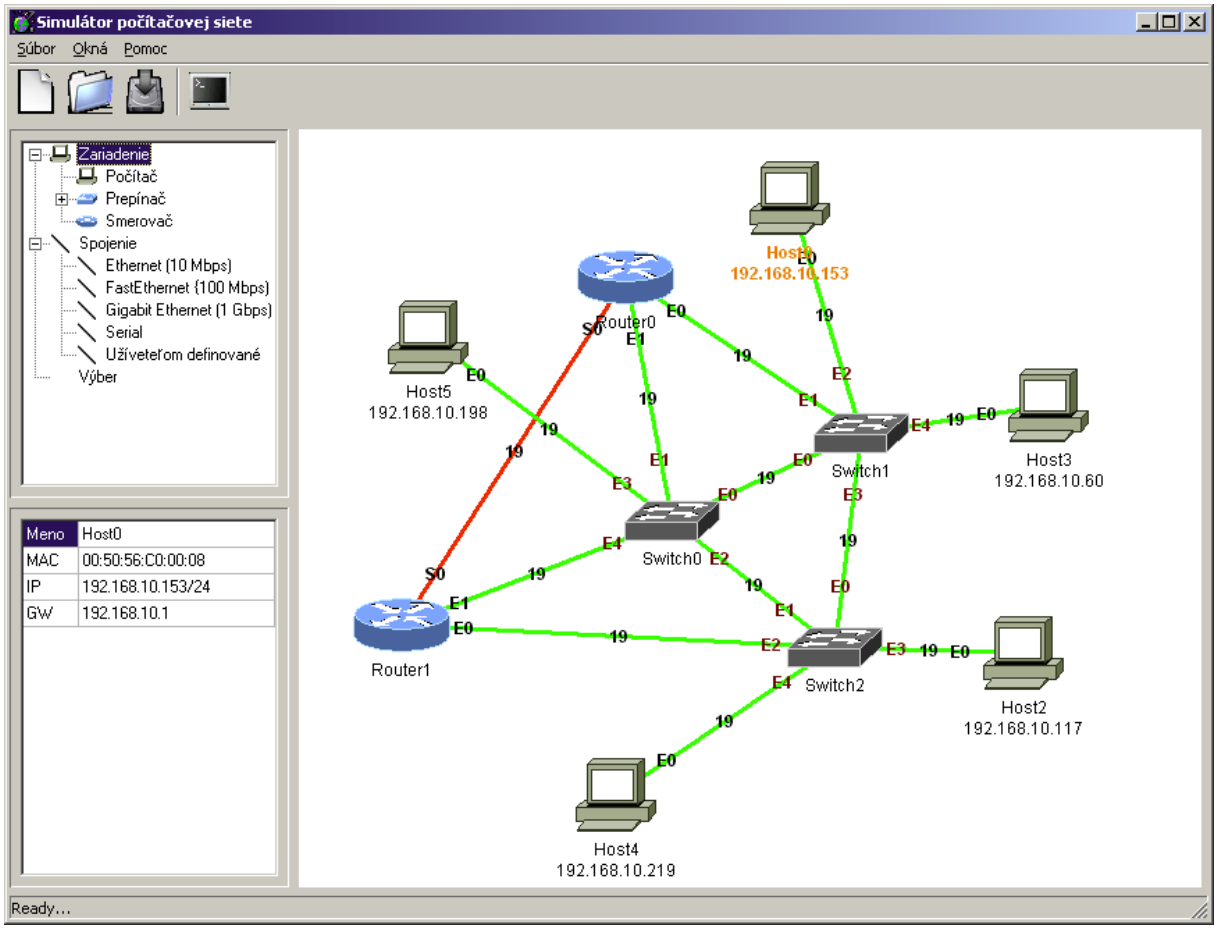
Operáciu úplného zrušenia prepojenia dvoch zariadení zastrešuje tento prípad použitia. Kliknutím pravého tlačidla myši na spojenie má používateľ možnosť v zobrazenom menu zrušiť celé prepojenie a tým aj všetky údajové štruktúry slúžiace na simuláciu prenosu. Po úplnom zrušení prepojenia nebude možné zobraziť trasu prenosov, ktoré boli prenášané týmto prepojením.

3.3.15 Sledovanie prenosu – SNIFFER

Sledovanie prenosu je možné aktivovať na existujúce prepojenia dvoch sieťových komponentov. Funkcia sniffera je rozdelená na niekoľko častí. Prvá obsahuje časovo usporiadaný zoznam prenášanej komunikácie. Označením konkrétneho preneseného údajového rámca je možné v časti s detailmi zobraziť hlbšiu analýzu rámca. V bloku štatistík sú zobrazené rozličné štatistické informácie získané z prenosu, ako aj parametre linky hovoriace o kvality prenosu (QoS).

4 Návrh

4.1 Návrh GUI



Obrázok č. 28: Návrh obrazovky

Hlavné okno aplikácie sa skladá z dvoch častí – v ľavej časti sú to zariadenia, ktoré je možné pridať do modelu siete (vľavo hore) a nastavené parametre jednotlivých existujúcich zariadení (vľavo dole). Najväčšiu časť obrazovky zaberá plocha zobrazujúca model siete. Okrem toho je možné si zobraziť konzolu zariadenia (na konfiguračné alebo testovanie účely) a okno zobrazujúce rámce prenesené sieťou (a z nich následne vypočítané štatistiky).

4.2 Hrubý návrh

Keďže náš návrh vychádza z už hotového systému, musíme pri návrhu zohľadniť jeho doterajšiu implementáciu. Preto uvádzame diagramy tried, v ktorých sú doplnené nami navrhované triedy.

4.2.1 Diagram tried

Diagram tried predstavuje celkový pohľad na architektúru navrhovaného systému. Pozostáva z dvoch častí:

- Funkcionálny diagram tried – predstavuje triedy vykonávajúce základné funkcie systému a reprezentujú základne objekty systému.
- Diagram tried pre služby – triedy, ktoré sú využívané funkcionálnymi triedami

Vo funkcionálnom grafe sú zobrazené (normálnym písmom) základné triedy, ktoré už sú implementované a sú celkovo funkčné. Hrubým písmom sú vyznačené triedy, ktoré neboli plne funkčné, alebo ktoré neboli vôbec implementované.

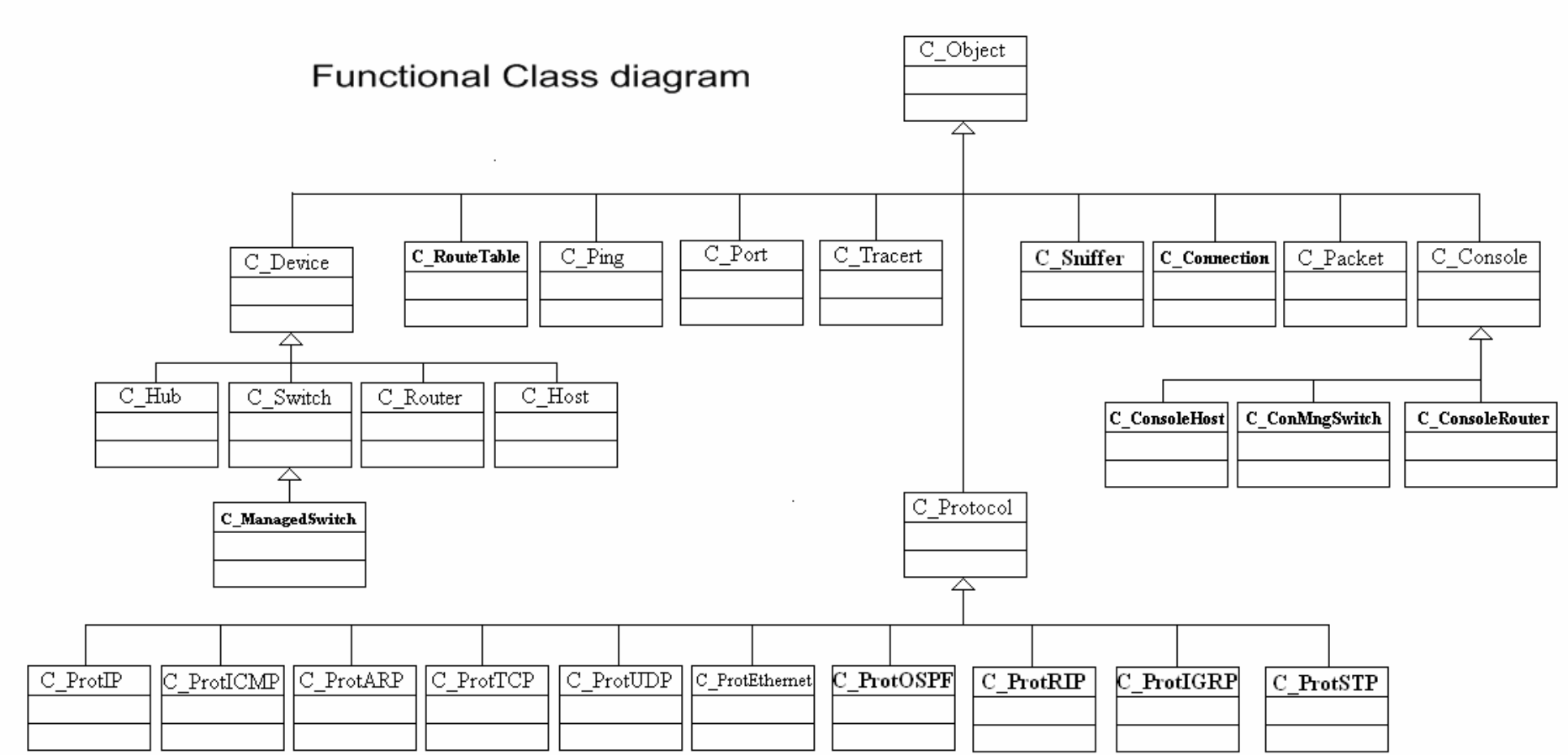
Medzi ne patria:

C_ProtSTP - trieda bude implementovať základné funkcie pre prácu s paketmi STP protokolu, vytváranie paketu, získavanie dát z paketu.

C_ProtIGRP - trieda bude implementovať základné funkcie pre prácu s paketmi IGRP protokolu, vytváranie paketu, získavanie dát z paketu.

C_PrtoOSPF - trieda bude implementovať základné funkcie pre prácu s paketmi OSPF protokolu, vytváranie paketu, získavanie dát z paketu.

C_ProtRIP - trieda bude implementovať základné funkcie pre prácu s paketmi RIP protokolu, vytváranie paketu, získavanie dát z paketu.



Obrázok č. 29: Funkcionálny diagram tried

C_ManagedSwitch - trieda odvodená od triedy C_Switch, ide o zariadenie konfigurovateľného prepínača, na ktorom sa budú dať nastaviť vlan-y a switch port security.

C_Sniffer - trieda, v ktorej je implementovaný sniffer, ktorý je možné súčasne spustiť na viacerých linkách

C_Connection - implementácia spojenia medzi zriadeniami, umožňuje rôzne typy káblov a rýchlostí, half a full duplex.

C_RouteTable - implementácia dynamickej a statickej smerovacej tabuľky pre router

C_ConsoleHost -

C_ConMngSwitch -

C_ConsoleRouter - implementovaná konzola pre zadávanie príkazov ako iná alternatíva k nastavovaniu cez grafické rozhranie.

V diagrame služieb sú tak isto hrubým písmom vyznačené služby, ktoré musíme upraviť alebo vytvoriť.

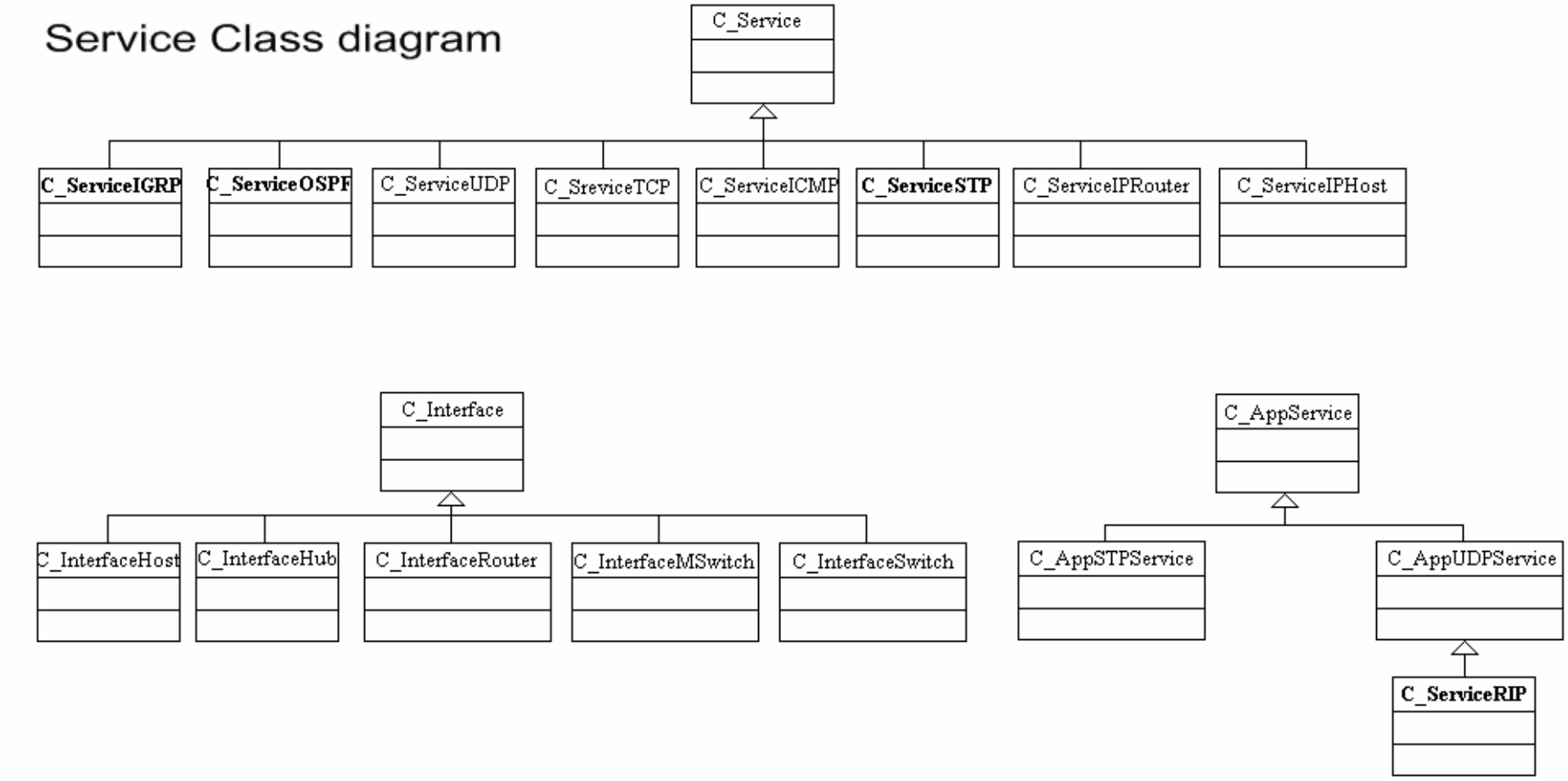
Medzi ne patria:

C_ServiceSTP - služba zabezpečí chod spanning tree algoritmu a teda odstránenie logických slučiek v topológii.

C_ServiceOSPF - služba zabezpečí fungovanie smerovacieho protokolu OSPF

C_ServiceIGRP - služba zabezpečí fungovanie smerovacieho protokolu IGRP

C_ServiceRIP - služba zabezpečí fungovanie smerovacieho protokolu RIP



Obrázok č. 30: Diagram tried pre služby

5 Podrobný návrh

5.1 Rip2

Smerovacia tabuľka

Smerovacia tabuľka uchováva cesty, ktoré dané zariadenie pozná – či už priamo pripojené alebo vzdialené. Pozostáva minimálne z týchto položiek (stĺpcov):

- IP adresa cieľovej destinácie
- maska
- IP adresa, cez ktorú je destinácia prístupná (next hope)
- číslo výstupného rozhrania
- metrika tejto cesty
- administratívna vzdialenosť

Cesty musia byť zoradené v prvom rade podľa cieľovej adresy, dĺžky masky a následne podľa metriky. Napr.:

192.168.1.0	255.255.255.0	192.168.2.2	0	3	1
10.1.1.0	255.255.0.0	10.1.2.3	0	2	1
10.1.1.0	255.0.0.0	10.1.3.4	1	1	1

Trieda musí obsahovať funkcie pre vkladanie, mazanie a hľadanie ciest. Usporadúvanie zabezpečuje automaticky. Tabuľka bude podporovať VLSM.

RIP2 protokol

Do smerovacej tabuľky pridáva tieto položky:

- čas pridania položky do smerovacej tabuľky
- informácia pre smerovací protokol, že sa zmenila alebo pridala cesta
- príznak neplatnej cesty (nastaví sa po uplynutí časovača danej cesty)

Formát paketu a teda aj dátovej štruktúry:

8	**16**	**32 bit**
Command	Version	Unused
Address family identifier		Route tag (only for RIP2; 0 for RIP)
IP address		
Subnet mask (only for RIP2; 0 for RIP)		
Next hop (only for RIP2; 0 for RIP)		
Metric		

Štruktúra hlavičky je nasledovná:

```
BYTE Command        / typ paketu – request, response
BYTE Version        /typ verzie RIP – v našom prípade 2
BYTE RouteTag       / identifikátor domény
BYTE AFI            / identifikátor protokolu pre ktorý RIP beží, pre IP = 2
```

Ďalej budú nasledovať hodnoty IP adries a ich masky, metrika, a next hop

```
SIPAddress  IP      /cieľová IP
SIPAddress  Mask    /maska siete
SIPAddress NextHop  /prednastavený next hop, smeruje sa priamo naň
BYTE    Metric      /metrika
```

Okrem uvedeného obsahuje RIP2 aj proces posielania update paketov. Tento periodicky konštruuje paket, do ktorého vkladá všetky cesty zo smerovacej tabuľky s administratívnou vzdialenosťou 120 spolu s maskou a priamo pripojené cesty, ktoré boli používateľom povolené.

RIP2 v sebe implementuje Triggered Updates, Split Horizont a Route Poisoning.

Triggered Updates a Route Poisoning

Túto funkciu RIP2 budeme implementovať tak, že v prípade zmeny v topológie (odstránenie kábla na lokálnom rozhraní) sa automaticky na daných routroch, ktorých sa zmena týka, upraví smerovacia tabuľka. Cesta sa nastaví ako nedostupná (hop count 16)a zobudí sa proces rozposielania update paketov.

Split Horizont

Cesty ktoré sa naučil z nejakého rozhrania nebude zahŕňať do Update-ov, ktoré posiela na toto rozhranie. Tým sa zabráni vzniku logických slučiek v smerovaní.

Multicast

Paket je spracovaný iba r**Error! Style not defined.**outrom, pretože je poslaný ako multicast 224.0.0.9, Ostatné stanice paket nespracovajú.

Autonómne systémy

RIP je možné použiť vo viacerých autonómnych systémoch (v jednej LAN) s rôznym autonómnym číslom. Update-y prijaté smerovačom, ktorý je v inom autonómnom systéme ako číslo v Routing Domain poly, sú zahadzované. Jeden router môže byť nastavený ako rozhranie medzi dvomi autonómnymi systémami, na neho budú smerované všetky cesty, ktoré idú do druhého autonómneho systému.

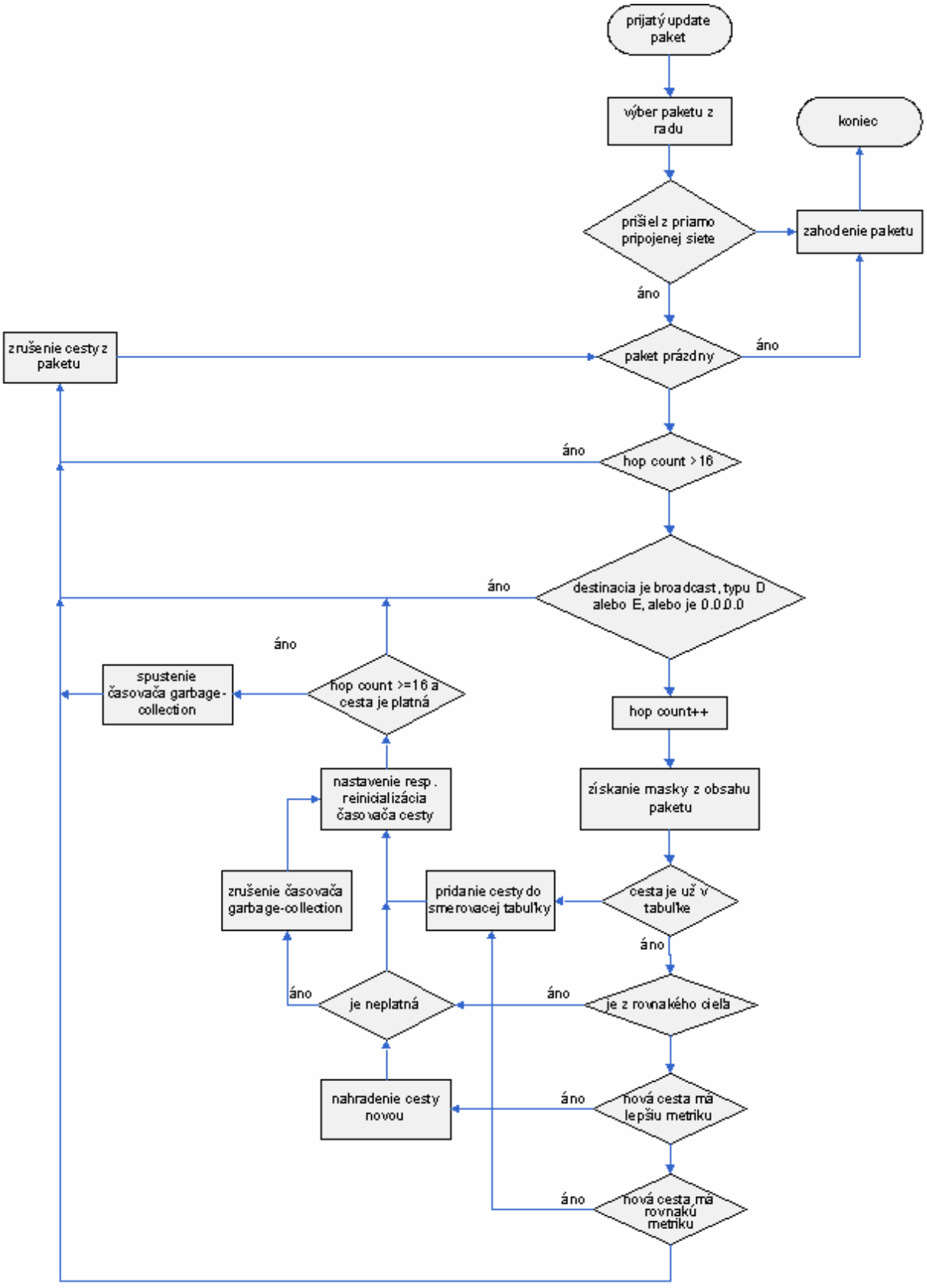
Update-y sa posielajú každých 30sekún, ak nedostane router update o nejakej konkrétnej ceste po dobu 180s, označí ju ako nedostupnú a bude ju posielať v update-och s hopcount 16. To spôsobí, že ostatné routre už nebudú posielať na túto adresu cez tento router. Ak nedostane update po dobu 120s , záznam úplne vymaže a prestane posielať o nej updaty. To spustí proces, garbage-collection, ktorý ju odstráni z tabuľky.

V prípade pripojenia nového routra do siete a je zapnuté triggered updates, kde už beží RIP (1 alebo 2) musí router poslať request paket na každé pripojené rozhranie (multicast). Okolité routre mu odpovedia a on si naplní tabuľku. Súčasne ale posiela on im nové upravené cesty pričom je stále zapnutý split horizont.

Existujú dva typy requestu. Prvý je request na celú tabuľku, ten je identifikovaný poľom Adress Family id = 0, a Metrikou = 16. Druhým typom je požiadavka na konkrétnu položku v tabuľke. Tento typ však nebude implementovaný.

Jedným z parametrom RIP2 je možnosť odpovedať na RIP1 request pakety. To sa bude dať nastaviť ako parameter služby. Ak bude táto možnosť aktívna, služba sa bude pri posielaní vyžiadaných updatov, správať ako RIP1. Sumarizácia adries nebude implementovaná.

Smerovací protokol bude implementovaný podľa štandardu RFC 2453 a vývojový diagram je nasledovný:



Obrázok č. 31: Vývojový diagram RIP2 protokolu

5.2 IGRP

Protokol IGRP je jednou z možností, aby smerovače koordinovali smerovanie paketov. Pri jeho špecifikácii boli kladené nasledovné ciele:

- stabilné smerovanie aj vo veľkých a zložitých sieťach
- rýchla odpoveď na zmeny topológie siete
- nízka vlastná réžia
- rozdelenie záťaže medzi paralelnými linkami
- vplyv chybovosti linky a jej záťaže na smerovanie

Súčasná verzia podporuje smerovanie len TCP/IP, aj keď je protokol vo všeobecnosti robený pre smerovanie ľubovoľných protokolov. Patrí do rodiny protokolov založených na DVA (distant vector algorithm). Periodicky každý smerovač vysiela svoju smerovaciu tabuľku (po vynechaní ciest obmedzených algoritmom „split horizon“) na všetky okolité smerovače. Smerovač po prijatí smerovacej informácie od iného smerovača porovná jeho cesty so svojimi. V prípade nájdenia novej, alebo vhodnejšej ako pozná, si ju zapamätá. Výhodnosť cesty sa vyhodnocuje pomocou metriky, na ktorej hodnotu vplýva prenosová kapacita cesty, oneskorenie pri prenose, spoľahlivosť a záťaž cesty. Metriku je možné vypočítať pomocou nasledovného vzorca: $\left(\frac{K_1}{Be} + (K_2 * Dc)\right)r$, kde *K1* a *K2* sú konštanty; *Be* je efektívna priepustnosť: priepustnosť nezaťaženej siete*(1-zaťaženie linky); r je spoľahlivosť (% úspešne prenesených paketov) a *Dc* je oneskorenie: $Dc = Ds + Dcir + Dt$, kde *Ds* je oneskorenie v prepínači; *Dcir* je oneskorenie linky a *Dt* je oneskorenie pri vysielaní. Metrika je uchovávaná v 24-bitovej premennej, pričom 0 znamená priamo pripojené rozhranie a 224-1 zas nedostupný cieľ.

Proces budovania smerovacej tabuľky je popísaný Bellman-Fordovým algoritmom, pričom platia nasledujúce pravidlá:

- namiesto jednoduchej metriky sa vypočítava pomocou vyššie uvedeného vzťahu
- pakety sa smerujú cez viacero ciest s podobnou metrikou
- na dosiahnutie stability sú implementované niektoré techniky, ktoré zabezpečujú, aby nedochádzalo k cyklom pri výpadku spojenia – *holddown*, spúšťané aktualizácie (*triggered updates*), rozdelenie horizontu (*split horizon*) a „otrávenie“ ciest (*route poisoning*)

Nasleduje popis algoritmov ktoré určujú správanie sa smerovača po výskyte niektorej udalosti významnej pre IGRP systém.

Spracovanie prichádzajúcich paketov

Údajové pakety, ktoré prídu na rozhranie R

Urči protokol použitý v pakete. Ak nie je protokol podporovaný, tak zahoď paket

Ak za zhoduje cieľová adresa s niektorou z adries brán, alebo je to broadcast adresa, tak spracuj protokol spôsobom špecifickým pre neho

Ak je cieľová adresa v priamo pripojenej sieti, tak pošli paket.

Ak nie je známa cesta k cieľovej adrese, alebo všetky cesty vedú v protismere, tak pošli chybovú správu a zahoď paket.

Vyber cestu. Ak existuje viac ciest, tak vyber jednu z nich procesom round-robin, pričom metrika má nepriamoúmerný vplyv. Pošli paket na ďalší smerovač.

Spracovanie prichádzajúcich aktualizácií

Zo zdroja Z príde aktualizácia

Pre každý typ služby podporovaný smerovačom použi smerovaciu údaje asociované s danou službou

Pre každý cieľ C v aktualizácii

Ak je C nedosiahnuteľné alebo v stave „holddown“, ignoruj záznam a pokračuj s ďalším cieľom C

Vypočítaj metriku pre cestu S k C cez Z

Ak nie je cieľ C v smerovacej tabuľke, tak:

pridaj cestu S do tabuľky a nastav posledný čas aktualizácie pre S a C na aktuálny

vyvolaj aktualizáciu

nastav metriku pre C a S na novú metriku vypočítanú vyššie

inak (C je v smerovacej tabuľke)

porovnaj novú metriku pre S s najlepšou existujúcou metrikou pre C

ak je nová menšia

ak je C nedosiahnuteľná v aktualizácii alebo (sú povolené stavy „holtdown“ a nová metrika > existujúca metrika * V (parameter)) alebo („holddown“-stavy sú zakázané a S má väčší počet skokov ako v starom prípade), tak

odstráň S zo smerovacej tabuľky (ak sa nachádza)

ak bola S poslednou cestou k C

pokiaľ sú povolené „holddown“stavy, tak nastav holddown čas pre C na aktuálny čas + „holddown“ čas a spusti aktuaizáciu

inak

vypočítaj novú metriku pre C

ulož informáciu o novej metrike pre cestu S do smerovacej tabuľky

vlož cestu S do smetovacej tabuľky (ak sa tam už nenachádza)

nastav čas poslednej aktualizácie pre S a C na aktuálny čas

inak (nová je menšia alebo rovná starej)

nastav metriku pre C a S na novú vypočítanú hodnotu

ak existujú iné cesty do C, ktoré sú mimo povolenej odchýlky, tak ich vymaž

ulož novú metriku do záznamu o ceste S do smerovacej tabuľky

nastav posledný čas aktualizácie pre S a C na aktuálny

Periodické spracovanie

Proces je aktivovaný hodinami (napr. raz za sekundu)

pre každú cestu S v smerovacej tabuľke (s výnimkou priamo pripojených)

ak je aktuálny čas < posledný čas aktualizácie S + „invalid“ čas

pokračuj ďalšou cestou

odstráň S zo smerovacej tabuľky

ak bola S poslednou cestou k C

nastav metriku pre C na nedosiahnuteľné

pokiaľ sú povolené „holddown“-stavy, tak

spusti „holddown“ časovač pre C a

spusti aktualizáciu

inak spusti výpočet najlepšej metriky pre C

koniec „for“

pre každý cieľ C v smerovacej tabuľke

ak je metrika D „nedosiahnuteľné“

zmaž všetky cesty do D

ak je aktuálny čas >= poslednému času aktualizácie D + čas pre vymazanie

odstráň záznam pre C

koniec „for“

pre každé rozhranie R pripojené na smerovač

```
  prepočítaj zaťaženie kanálu a chybovosť
  ak sa vyťaženie kanálu alebo chybovosť zmenila, tak prepočítaj metriky
V čase vyslania aktualizácie
 spusti aktualizáciu
```

Všeobecná aktualizácia

```
Proces vyvolaný „vyvolanou aktualizáciou“
 pre každé sieťové rozhranie R pripojené na smerovač
  vytvor prázdnu aktualizačnú správu
  pre každý typ podporovanej služby S
   použi cestu/cieľ pre S
   pre každý cieľ C
    ak má niektorá cesta ku C ide k rozhranie R na svoj ďalší skok
     pokračuj ďalším cieľom
    ak je niektorá cesta do C s menšou metrikou v správe
     pokračuj ďalšou cestou
    vytvor záznam pre C v správe, použi metriku z cestu s najmenšou metrikou
   koniec „for“
  koniec „for“
  ak sú nejaké záznamy v aktualizačnej správe, tak vyšli cez rozhranie R
 koniec „for“
```

IGRP paket

IGRP používa na prenos svojich informácií IP datagramy typu 9 (IGP). Paket začína s hlavičkou, za ktorou nasledujú údajové položky o cestách.

Štruktúra hlavičky je nasledovná:

```
  unsigned version: 4;  /* protocol version number */
  unsigned opcode: 4;   /* opcode */
  uchar edition;        /* edition number */
  ushort asystem;       /* autonomous system number */
  ushort ninterior;     /* number of subnets in local net */
  ushort nsystem;       /* number of networks in AS */
  ushort nexterior;     /* number of networks outside AS */
```

```
ushort checksum;      /* checksum of IGRP header and data */
```

V súčasnosti musí byť číslo verzie (version) nastavené na 1, v poli opcode znamená aktualizáciu, 2 vyžiadanie. Edition je sériové číslo, ktoré sa zväčšuje, keď sa zmení smerovacia tabuľka. Pomocou neho môžu ostatné smerovače určiť, či došlo k zmene a teda je potrebné prepočítať smerovaciu tabuľku. Asystem je číslo autonómneho systému, pretože jeden smerovač môže patriť do viacerých autonómnych systémov, ktoré majú nezávislé smerovacie tabuľky. Ninterior, nsystem a nexterior udávajú počet položiek v každej z troch sekcií v aktualizačnej správe. Checksum je kontrolný súčet je počítaný z IGRP hlavičky a zo smerovacích informácií, ktoré nasledujú, pričom je pole checksum vynulované. Kontrolný súčet sa počíta rovnakým algoritmom, ako v prípade UDP.

IGRP požiadavka (opcode = 2) spôsobí, že dotyčný smerovač vyšle svoju smerovaciu tabuľku.

V odpovedi sa nachádza toľko smerovacích položiek, koľko sa zmestí do dotyčného paketu (resp. menej, pokiaľ je ich menej na poslanie). Každá položka pozostáva z nasledovných polí:

```
uchar number[3];      /* 3 significant octets of IP address */
uchar delay[3];       /* delay, in tens of microseconds */
uchar bandwidth[3];   /* bandwidth, in units of 1 Kbit/sec */
uchar mtu[2];         /* MTU, in octets */
uchar reliability;    /* percent packets successfully tx/rx */
uchar load;           /* percent of channel occupied */
uchar hopcount;       /* hop count */
```

Typy uchar[2] a uchar[3] sú obyčajné 16- a 24.bitové celé čísla, usporiadané v normálnom IP poradí. Number definuje IP adresu. Kvôli šetreniu miestom, sú prenášané len najvyššie 3 B, okrem interior sekcie, kde sú prenášané najnižšie 3 B. Vzhľadom na toto usporiadanie, nie je možné v prípade iných ciest, ako sú interior používať podsiete, kým interior cesty sú vždy v podsieti, a preto je možné doplniť 1. B adresy. Oneskorene má ako jednotku 10 ms, takže maximálne oneskorenie je až 168 sekúnd. Bandwith je inverzná šírka prenosového zoškálovaná faktorom 1010. MTU (maximal transfer unit) je udávaná v B a spoľahlivosť ako zlomok s menovateľom 255 (max. je hodnota 255). Tak isto je udávaná aj záťaž, kým počet skokov je jednoduché číslo.

Návrh implementácie

Protokol bude implementovaný v samostatnej triede C_ServiceIGRP. Na svoju činnosť potrebuje protokol zabezpečiť pravidelné spúšťanie (predvolene každých 90 s, pričom je tento interval upraviteľný). V týchto intervaloch sa bude spúšťať metóda zabezpečujúca vyslanie smerovacej tabuľky pre všetky okolité smerovače, pričom (v súlade s technikou split horizon) nebude posielať informácie o cestách, ktoré sú smerované na dotyčný smerovač, aby sa zabránilo tvorbe slučiek. Na spracovanie prichádzajúcich paketov bude slúžiť ďalšia metóda, ktorá zabezpečí správne nastavenie smerovacej tabuľky podľa údajov z prišlého paketu. Po zistení zmeny v topológii siete v bezprostrednom okolí je aktualizovaná smerovacia tabuľka a následne je vyvolané okamžité rozposlanie smerovacej tabuľky okolitým smerovačom (trigger update). Algoritmy, podľa ktorých metódy pracujú sú popísané vyššie. Okrem nich budú ešte metódy na spustenie a ukončenie aktualizácie smerovacej tabuľky pomocou IGRP prtokolu a tak isto aj na vynulovanie IGRP podsystému.

5.3 OSPF

Každý smerovač si pre všetky svoje rozhrania, ktoré používajú OSPF smerovací protokol, udržiava nasledovné informácie:

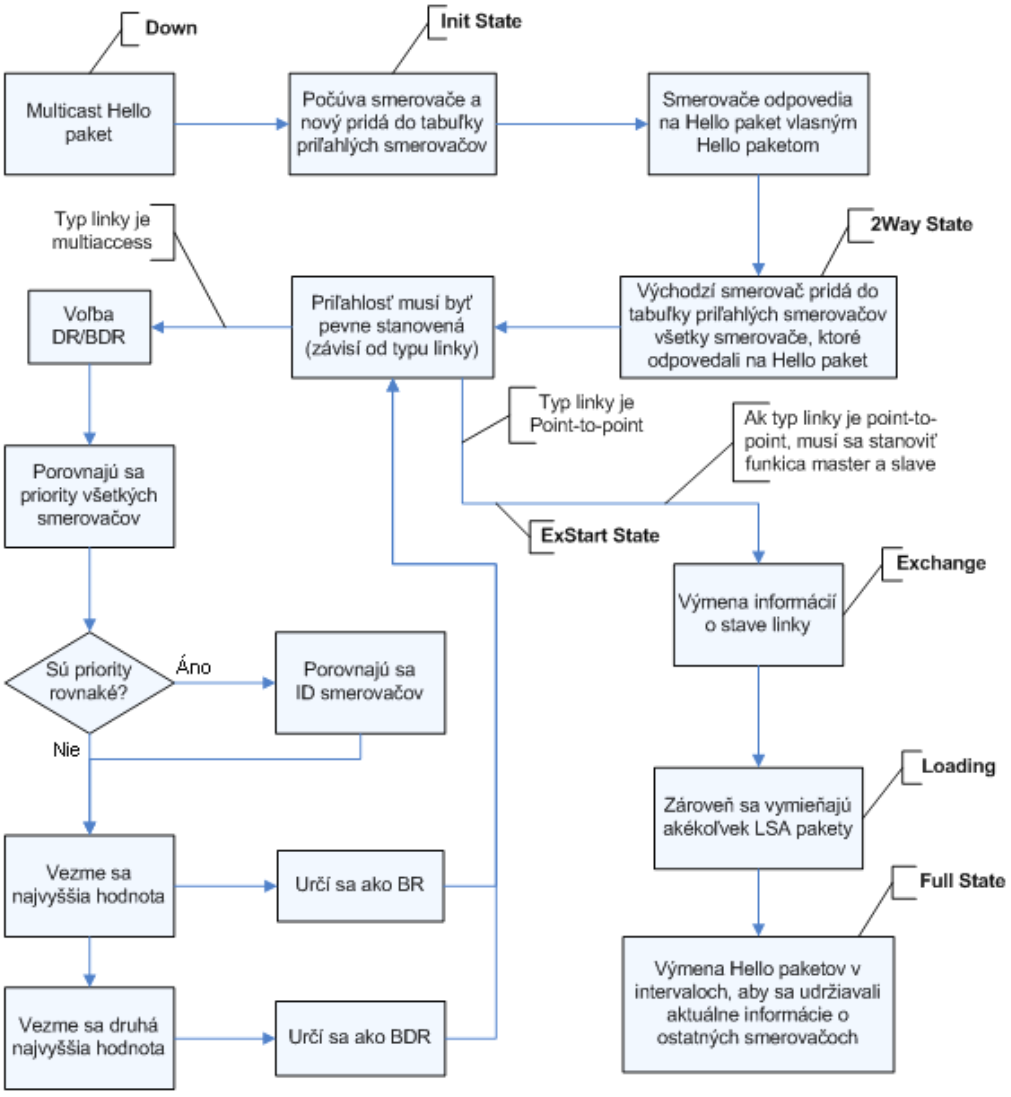
typ linky, stav linky, IP adresa rozhrania, IP maska rozhrania, ID oblasti, Hello interval, Router Dead interval, Inf Trans Delay, priorita smerovača, Hello časovač, Waiting časovač, zoznam susedných smerovačov, DR smerovač, BDR smerovač, Cena vyslania paketu, Rxmt interval, Autentifikačný kľúč.

Na základe týchto konfiguračných údajov smerovač vytvára a vyhodnocuje pravidelné udalosti (napríklad Hello pakety) resp. ich pravidelne aktualizuje podľa vzniknutej situácii. Smerovače medzi sebou komunikujú pomocou OSPF paketu, ktorého formát bol popísaný v predchádzajúcich kapitolách, kde bola vykonaná analýza tohto protokolu. Protokol bude implementovaný podľa štandardu RFC 1131.

Protokol využíva priľahlosti susedov na získanie celkovej topológie siete. Robí sa to v siedmich krokoch, ktoré určujú stav protokolu. Ide o nasledovné stavy: Down State, Init State, 2Way State, Ex Start State, Exchange State, Loadnikg State, Full State. Prvotný stav, pred tým, ako sa protokol snaží spoznať susedov, je Down State.

- Init State: smerovač príde do tohto stavu, ked prvý krát dostane Hello paket . Každý Hello paket obsahuje zoznam priľahlých smerovačov, ktoré sú známe vysielajúcemu smerovaču. Nasledujúce Hello pakety obsahujú už obsahujú objavených susedov.
- 2Way State: smerovač sa doň dostane, keď prijme Hello paket so svojim ID. Vtedy môže smerovač začať vysielať DBD a LSAck pakety.
- ExStart State: je ďalším stavom, do ktorého sa smerovač dostane. V tomto stave začne byť platný vzťah priľahlosti s priľahlými smerovačmi. V sieťach s mulstiaccess prístupom sa v tomto stave zvolí DR a BDR smerovač a v sieťach point-to-point sa určí master a slave smerovač.
- Exchange State: V tomto stave si smerovače pomocou paketov DBD potvrdených LSAck paketom vymenia navzájom svoje tabuľky stavov liniek. Ak niektorá linka v DBD pakete je nová, stav smerovača sa zmení na Loading State, inak sa zmení na Full State.
- Loading State: smerovač, ktorý dostal informáciu o zmene, vyšle LSR paket, ktorým si vyžiada dodatočné informácie o novej linke. Cieľový smerovač odpovie LSU paketom, ktorý je následne potvrdený LSAck paketom.
- Full State: je posledným stavom smerovača, v ktorom majú všetky smerovače rovnaký obsah tabuliek so stavmi liniek. Dijkstrovým algoritmom sa vypočítajú smerovacie cesty a aktualizuje sa smerovacia tabuľka.

Celý proces vykonaných operácií je bližšie vyobrazený v nasledovnom diagrame.



Obrázok č. 32 Vývojový diagram operácií OSPF protokolu

Protokol OSPF je zapuzdrený do IP protokolu. Každý smerovač má vlastnú IP adresu ale zároveň počúva aj na multicast adresách, ktoré slúžia na komunikáciu záplavou. Implementovaná trieda OSPF protokolu teda na svoju komunikáciu bude používať triedu IP protokolu a čo sa týka komunikácie bude sa podobať na protokoly TCP resp. UDP vnorené do IP protokolu.

5.4 Spanning tree protocol

Spanning tree algoritmus využíva pri svojej činnosti posielanie BPDU rámcov. Tieto sa rozlišujú na konfiguračné správy a TCN (Tree Change Notification) správy. Každý switch musí mať jedinečnú mac adresu. Všetky BPDU sa posielajú na spoločnú cieľovú multicast adresu 01-80-C2-00-00-00. Na tejto adrese počúvajú všetky prepínače, ktoré majú spanning tree protokol.

Keďže nemáme implementovaný 802.3 rámec, BPDU rámec sa teda balí do Ethernet II rámca. Typ protokolu 1111(hex) pre BPDU je vybraný z neobsadených typov protokolov.

Formáty paketov

Konfiguračný BPDU:

popis	počet bytov
Protocol identifier	2 - Identifikuje spanning tree algoritnus a protokol. Vždy = 0.
Protocol version identifier	1 - Identifikuje verziu protokolu. Vždy = 0.
BPDU type	1 - Identifikuje typ BPDU: 00000000 = konfiguračný, 10000000 = TCN
Flags	1 - 10000000 = potvrdenie zmeny topológie root prepínačom, 00000001 = oznam o zmene topologie
Root identifier	8 - Identifikátor root switchu. Skladá sa z Mac adresy prepínača (6 Bajtov) a priority (2 Bajty). Prioritu nastavuje administrátor. (default 32 768 pre Cisco)
Root path cost	4 – nižšie popísané
Bridge identifier	8 - Identifikátor switchu, ktorý vysiela BPDU. Skladá sa z Mac adresy prepínača (6 Bajtov) a priority (2 Bajty). Prioritu nastavuje administrátor. (default 32 768 pre Cisco)
Port identifier	2 – Určuje prioritu portu.
Message age	2 - Vek BPDU rámca.
Max age	2 - Max čas uchovania najlepšieho BPDU. (default 20s)
Hello time	2 - Perióda posielania hello BPDU. (default 2s)

Forward delay 2 - Max čas čakania v stave listening a learning. (default 15s)

Root path cost - Cena cesty odpovedá šírke pásma.

šírka pásma		cena
4	Mbps	250
10	Mbps	100
16	Mbps	62
45	Mbps	39
100	Mbps	19
155	Mbps	14
622	Mbps	6
1000	Mbps	4
10000	Mbps	2

TCN BPDU:

popis	počet bytov
Protocol identifier	2
Protocol version identifier	1
BPDU type	1

Stavy portov

Porty sa môžu nachádzať v nasledujúcich stavoch:

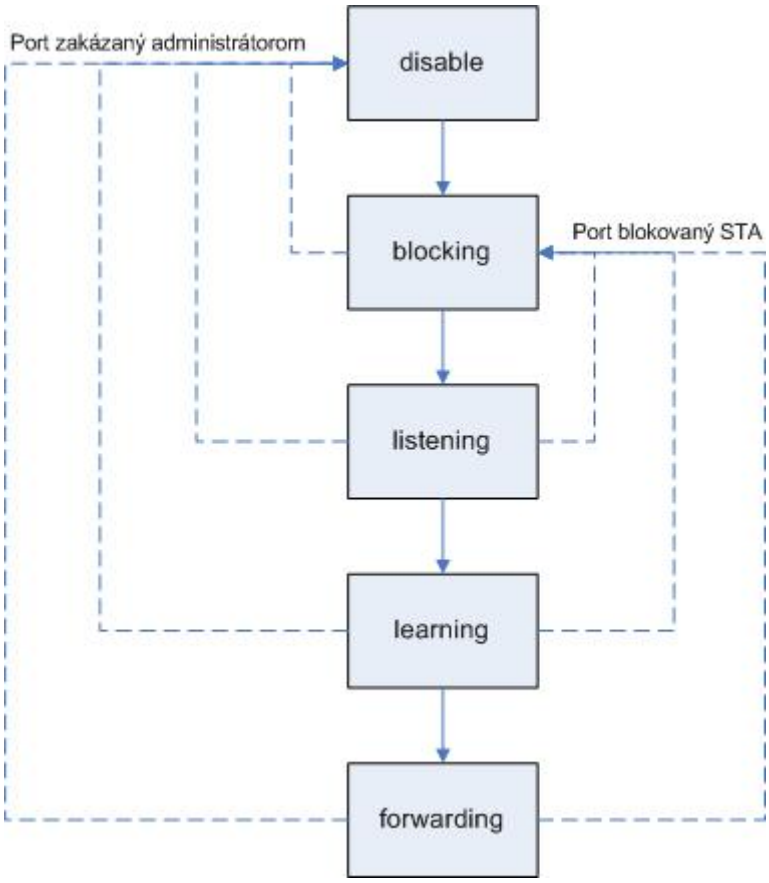
Disable

- BPDU príjem NIE
- BPDU vysielanie NIE
- Učenie sa MAC adries NIE
- Preposielanie rámcov NIE

Port je administrátorom zakázaný. Zmeniť stav sa môže ak administrátor povolí port.

Blocking

- BPDU príjem ÁNO
- BPDU vysielanie NIE
- Učenie sa MAC adries NIE
- Preposielanie rámcov NIE



Obr. 1: Stavy portov zariadenia switch

Port je administrátorom povolený. Do tohto stavu sa dostane po inicializácii, alebo po povolení portu zo stavu disable administrátorom. Do tohto stavu sa môže dostať aj z learning, listening a forwarding stavu s použitím Spanning tree algoritmu. A to v tom prípade, že iný prepínač bude vybratý ako predurčený (designated) na určitom porte, alebo sa objaví nový root prepínač.

Z tohto stavu sa port dostane, ak sa príjme konfiguračný BPDU a vstúpi do stavu listening. Zo stavu blocking sa port dostane do stavu disable, ak administrátor zakáže daný port.

Listening

- BPDU príjem ÁNO
- BPDU vysielanie ÁNO
- Učenie sa MAC adries NIE
- Preposielanie rámcov NIE

Port je administrátorom povolený. Do tohto stavu sa port dostane z bloking stavu pokiaľ prepínač dostane konfiguračný BPDU. Z tohto stavu sa port dostane, ak uplynie časovač (forward dealy), do stavu learning. Z tohto stavu sa port dostane aj ak sa príjme konfiguračný BPDU a s použitím Spanning tree algoritmu vstúpi do stavu blocking. Zo stavu listening sa port dostane do stavu disable, ak administrátor zakáže daný port.

Learning

- BPDU príjem ÁNO
- BPDU vysielanie ÁNO
- Učenie sa MAC adries ÁNO
- Preposielanie rámcov NIE

Port je administrátorom povolený. Do tohto stavu sa dostane zo stavu listening po uplynutí časovača. Po uplynutí časovača port sa dostane do stavu forwarding. Zo stavu learning sa port dostane aj ak sa príjme konfiguračný BPDU a s použitím Spanning tree algoritmu vstúpi do stavu blocking. Zo stavu learning sa dostane port do stavu disable, ak administrátor zakáže daný port.

Forwarding

- BPDU príjem ÁNO
- BPDU vysielanie ÁNO
- Učenie sa MAC adries ÁNO
- Preposielanie rámcov ÁNO

Port je administrátorom povolený. Do tohto stavu sa dostane ak v stave learning vyprší časovač. Z tohto stavu sa dostane prijatím BPDU rámca. Zo stavu forwarding sa dostane port do stavu disable, ak administrátor zakáže daný port.

5.5 Telnet

Pôvodný produkt bude rozšírený o ďalšiu službu, ktorou je telnet. Funkcia telnetu je jednoduchá. Číta znaky zo štandardného vstupu a sieťovým prenosom ich prenesie na druhý počítač, kde sa spracujú rovnako, ako keby boli zadané na štandardnom vstupe vzdialeného počítača. Výstup programov zo vzdialeného počítača sa opäť sieťovým prenosom prenesie na lokálny počítač a vypíše sa na štandardný výstup. Prenášané údaje sú obyčajné znaky, ktoré sa prenášajú telnet protokolom.

Na strane servera načúva komunikácii služba telnet servera, ktorá nezávisle od prijímania dokáže zaslať text na stranu klienta. Rovnako aj klient nezávisle od zadávania údajov na vstupe dokáže posielať údaje serveru. Mechanizmy čítania a posielania údajov sieťového prenosu sú v obidvoch prípadoch rovnaké. Rozdielne je ich spracovanie na obidvoch stranách.

Služby servera a klienta budú implementované v triedach C_ServiceTELNETServer a C_ServiceTELNETClient, ktoré budú odvodené od triedy C_AppTCPService, pretože sa jedná o služby prenášané pomocou TCP spojenia. Strana klienta komunikuje priamo s otvoreným príkazovým riadkom sieťového uzla. Na strane servera sa v prípade aktívneho telnet spojenia zablokuje príkazový riadok ale vykonávanie príkazov bude možné sledovať aj v tomto príkazovom riadku.

Server bude schopný prijať iba jedno spojenie a ďalšie pokusy o viacnásobné pripojenie budú zamietnuté. Rovnako počítač, z ktorého bude vytvorené telnet spojenie, nebude schopný prijať telnet spojenie, aby sa zabránilo vzniku slučiek a zablokovaniu príkazových riadkov na všetkých uzloch v slučke.

6 Opis riešenia

6.1 Implementácia tried

C_RouteTable

Je implementovaná triedou C_RouteTable. Táto obsahuje objekt CPtrList knižnice MFC, ktorý implementuje zreťazený zoznam. V tomto zozname sa cesty uchovávajú.
Poskytuje funkcie:

- InsertRoute – vloženie cesty
- DeleteRoute – zmazanie cesty podľa IP adresy siete a jej masky
- DeleteRouteByPortNum – zmazanie cesty podľa čísla portu
- FindOutPort – vráti číslo portu podľa IP adresy
- FindRouteByIP – vráti celú položku cesty podľa IP adresy
- DeleteProtokolRoutes – zmaž z tabuľky cesty podľa admin. vzdialenosti
- FindRouteByIPandMask – vráti celú položku cesty podľa IP adresy a masky – pre protokol RIP
- IvalidateDirectRoute – priamo pripojenej sieti nastaví metriku 16 a flag, že je nedostupná

Usporadúvanie sa vykonáva vo funkcii InsertRoute a to tak, že sa prehľadáva od začiatku zoznamu a ak sa nájde IP, ktorá je menšia ako vkladaná, tak sa vloží pred ňu. Ak sa nájde IP rovnaká, potom sa ďalej prehľadáva zoznam rovnakým spôsobom, ale porovnáva sa dĺžka masky. Obdobne je to potom pre metriku cesty.

Prehľadávanie tabuľky je sekvenčné. Napr. funkcia FindRouteByIPandMask je implementovaná nasledovne:

```
if( m_table.GetCount() == 0 ) return NULL;

	pos = m_table.GetHeadPosition();

	for( i = 0; i < m_table.GetCount(); i++ )
	{
		p = ( SRouteItem* ) m_table.GetNext(pos);

		if ( p->m_destIP == dest && p->m_maskIP == VLSmask )
		{
			return p;
		}
	}
```

```
return NULL; //nenasla sa ziadna cesta
```

C_RouterInterface

Tento je implementovaný v triede `C_InterfaceRouter`. Bol doplnený o nasledovné premenné a funkcie:

```
BOOL  m_RIPpassive;          //! ak je TRUE tak sa nim neposielaju
                             smerovacie informacie
BOOL  m_SplitHorizont;       // default je TRUE, split je zapnuty

BOOL  SplitHorizontEnabled(); - vracia hodnotu m_SplitHorizont
void  SetSplitHorizont(BOOL Enable);  - nastavuje hodnotu m_SplitHorizont
void  PassiveInterfaceRIP( BOOL val ); - nastavuje hodnotu m_RIPpassive
BOOL  IsRIPPassive();        - vracia hodnotu m_RIPpassive

void  IpRipSendVersion( int ver ); - nastavuje verziu RIP pre posielanie
void  IpRipReceiveVersion( int ver ); - nastavuje verziu RIP pre prijímanie
                                    podľa dole uvedených hodnôt


CString GetSendVersion();
CString GetReceiveVersion();


int   m_Send;                // 1 - vysiela len verziu RIP
                             // 2 - vysiela len verziu RIP2
                             // 0 - vypnute vysielanie
                             // -1 - prikaz vypnuty - defaultne
int   m_Receive;             // 1 - prijima len verziu RIP
                             // 2 - prijima len verziu RIP2
                             // 12 -prijima verziu RIP aj RIP2
                             // 0 - vypnute prijimanie
                             // -1 - prikaz vypnuty - defaultne
```

Tieto premenné budú slúžiť na nastavenie protokolu RIP na jednotlivých rozhraniach smerovača. Pretože pôvodná implementácia nie je navrhnutá tak, aby sa dalo zistiť, z ktorého rozhrania paket prišiel, musel som využiť pole priorita v štruktúre CTrans, od ktorej je

oddedená štruktúra CPacket na uchovanie čísla portu, na ktorom som daný paket prijal. Toto pole bolo doteraz nevyužité. Tým sme predišli predčasnému rozbaľovaniu peketu na rozhraní.

Ďalej boli implementované všetky funkcie potrebné na nastavovanie a konfiguráciu rozhraní, ako v konzole smerovača, tak aj v konfiguračnom dialógu.

Ďalej bola upravená funkcia `Shutdown()` nasledovne:

```
void C_InterfaceRouter::Shutdown( BOOL shut )
{
	SIPAddress myIP, maskIP;
	SRouteItem* route = NULL;

	m_adminDown = shut;
	GetIPAddress( &myIP, &maskIP);

	if ( shut == TRUE )
	// ak je administrative down tak aj protocol je down
	{
		m_protocolState = eDown;

		//odstran priamo pripojenu linku z tabulky ak je down
		m_device->m_routeTable->IvalidateDirectRoute(myIP & maskIP ,
maskIP);


	}
	else if ( IsIPSet() == TRUE && m_state == eUp )
	{
		// vlozim cestu do smerovacej tabulky

		route = m_device->m_routeTable->FindRouteByIP(myIP);
		if ( route == NULL ) m_device->m_routeTable->InsertRoute( myIP
& maskIP, maskIP, myIP, this->m_myNum, 0, 0 );
		else if ( route->m_metrics == 16 && route->m_validFlag == FALSE
)
		{
			route->m_metrics = 0;
			route->m_validFlag = TRUE;
		}
```

```
            m_protocolState = eUp;
      }
//triggered RIP2
      if ( ((C_Router*)m_device)->m_RIP2Service->m_Triggered == TRUE )
      {
            (((C_Router*)m_device)->m_RIP2Service-
>GetRip2UpdateTimeoutProcess())->cancel();
            (((C_Router*)m_device)->m_RIP2Service-
>GetRip2UpdateTimeoutProcess())->activateAfterDelayT( 0 );
      }

}
```

Hlavná zmena sa týka odstránenia priamo pripojených liniek zo smerovacej tabuľky, ak prejdú do stav admindown, a ich opätovného pridania pri ich znovuzapojení.

Druhou zmenou je spustenie procesu posielania updatov ak je nastavená služba triggered updates. V závislosti na používanom protokole.

Implicitne po vytvorení nového rozhrania na smerovači sú nastavené nasledovné hodnoty:

```
m_adminDown = TRUE;
m_state = eDown;
m_protocolState = eDown;
m_RIPpassive = TRUE;
m_SplitHorizont = TRUE;
m_Receive   = -1;
m_Send      = -1;
```

Proces obsluhujúci rozhranie smerovača bol nasledovne upravený:

Zmena stavu rozhrania:

Ak došlo k zmene stavu rozhrania do stavu DOWN, v smerovacej tabuľke zruším všetky cesty súvisiace s týmto rozhraním (ak také sú), pretože je to moje lokálne rozhranie a viem že cez neho nemôžem smerovať.

V prípade, že je aktívna služba triggered updates, spustí sa proces posielania updatov.

Zmena IP adresy rozhrania:

Ak na rozhraní dôjde ku zmene IP adresy, musí sa zmeniť aj položka v smerovacej tabuľke pre príslušný protokol.

V prípade, že je aktívna služba triggered updates, spustí sa proces posielania updatov.

C_ServiceIPRouter

V procese CProcessIPRouterExecution v C_ServiceIPRouter obsluhujúcom pakety prichádzajúce a odchádzajúce do/z rozhrania smerovača bola odstránená podmienka, ktorá zahadzovala pakety ak bolo rozhranie v stave passive. V tomto stave rozhranie prijíma všetky updaty, avšak žiadne neposiela.

C_Router

Zariadenie smerovača bolo doplnené o službu RIP

```
C_ServiceRIP2*          m_RIP2Service;                  //! RIP2 sluzba
```

C_ProtRIP2

Nová trieda realizujúca prácu s paketmi protokolu RIP2 a RIP1. Pretože formát paketu je rovnaký, iba vo verzii 1 sa nevyužívajú niektoré položky, sú jednou triedou implementované oba protokoly.

Vychádza z protokolu RIP1 a bola doplnená o nasledujúcu funkciu :

```
BYTE GetVersion(C_Packet *packet); - ktorá vráti verziu protokolu
```

Formát protokolu RIP

```
typedef struct
{
      BYTE countOfItem;
      ERip2PacketType type;
      BYTE Version;       -verzia protokolu
      SRip2TableItem items[25];
} SRIP2;
```

Položka cesty v pakete má nasledovný formát.

```
typedef struct
{
	SIPAddress address;
	SIPAddress VLSMask; – rozdiel oproti RIP1,
	int hopCount;
} SRip2TableItem;
```

C_ServiceRIP2

Služba zabezpečuje vytváranie, spracovávanie updatov, a správu smerovacej tabuľky. Bola doplnená o premennú m_Triggered, ktorej implicitná hodnota po spustení smerovacieho protokolu je FALSE.

Boli vykonané nasledovné zmeny vo funkciách:

```
C_ServiceRIP::AddDirectNetwor( SIPAddress net )

// povolim prijem aj vysielanie RIP paketov
((C_InterfaceRouter*)this->m_device->GetInterface(port))->
PassiveInterfaceRIP( FALSE );
```

Po pridaní siete príkazom network sa nastaví passive na rozhraní na FALSE.

Boli pridané nasledovné funkcie:

`bool C_ServiceRIP::DelDirectNetwork( SIPAddress net )`, ktorá má funkciu ako príkaz no network, a teda odstráni sieť zo zoznamu priamo pripojených sietí, ktoré sa smerujú.

`BOOL C_ServiceRIP::IsInterfaceInDirectList( int port )` – funkcia zistí, či dané rozhranie sa už nachádza v zozname priamo pripojených sietí pre smerovanie.

`void C_ServiceRIP::Triggered(BOOL shut)` – funkcia zapína alebo vypína triggered updaty.

```
void C_ServiceRIP::Vesion(int ver) - prepnutie z RIP1 na RIP2. Táto funkcia plní funkciu príkazu version. Hodnoty môžu byť nasledovné:
      0 - (default) prijímanie RIP1 aj RIP2, posielanie RIP1
      1 - prijímanie aj vysielanie len RIP1
      2 - prijímanie aj vysielanie len RIP2


void C_ServiceRIP::StopRIP() - zastaví smerovací protokol, zmaže zo smerovacej tabuľky všetky cesty súvisiace s admin. vzdialenosťou 120.
```

Zmeny voči RIP2 voči RIP1:

Zmeny nastali v posielaní paketov na multicast adresu 224.0.0.9 a prijímaní paketov len z tejto adresy. Ďalšia zmena je v tom, že sa nevypočítava štandardná maska, ale zoberie sa maska poslaná v update.

CProcessRIP2Execution

Tento proces zabezpečuje spracovávanie prijatých updatov na rozhraniach smerovača. Bol doplnený o triggered updaty a route poisoning.

Route Poisoning

V prípade, že som prijal cestu s hopcount väčším alebo rovným 16, overím si, či to nie je práve moje lokálne rozhranie. Ak nie je tak si cestu označím ako nedostupnú tak, že jej nastavím hopcount = 16 a validflag = FALSE a spustím jej garbage timer. Túto cestu naďalej posielam v updatoch do vypršania časovača. Pokiaľ mi príde cesta s lepšou metrikou ako 16, časovač zruším a nastavím jej nový hopcount a validflag = TRUE.

Triggered updaty

Ak dôjde k akejkoľvek zmene v smerovacej tabuľke, spúšťa sa proces `CRIPUpdateTimeoutProcess`.

Ak služba prijala update, tak skontroluje položku priorita, kde je uložené číslo rozhrania, z ktorého paket prišiel. Ak sa zhodujú čísla verzií rozhrania a prijatého updatu, tak

sa paket pošle ďalej na spracovanie. Inak sa zahadzuje. Potom sa skontroluje globálne nastavenie smerovacieho protokolu. Ak sa nezhodujú verzie, update je zahodený.

```
//v priorite paketu je ulozene cislo, z ktoreho interfejsu mi prisiel paket
// 1 - prijima len verziu RIP
// 2 - prijima len verziu RIP2
// 12 -prijima verziu RIP aj RIP2
// 0 - vypnute prijimanie
// -1 - prikaz vypnuty - defaultne

if ( m_service->m_protRIP2->GetVersion(tmpPacket) == 1 )
{
      int     ver     =     atoi(((C_InterfaceRouter*)m_service->GetDevice()-
      >GetInterface(tmpPacket->priority))->GetReceiveVersion());
      if (  ver != 1 && ver != 12 && ver !=-1 )
      {
            delete tmpPacket;
            continue;
      }
}
if ( m_service->m_protRIP2->GetVersion(tmpPacket) == 2 )
{
      int     ver     =     atoi(((C_InterfaceRouter*)m_service->GetDevice()-
      >GetInterface(tmpPacket->priority))->GetReceiveVersion());
      if (  ver != 2 && ver != 12 && ver !=-1 )
      {
            delete tmpPacket;
            continue;
      }
}

//ak prisiel update a nie je to aktualne nastavena verzia, tak ho zahodim,
ale iba ak je zpanuta verzia
if      (      m_service->m_RIPversion      !=      m_service->m_protRIP2-
>GetVersion(tmpPacket) && m_service->m_RIPversion !=0 )
{
      delete tmpPacket;
      continue;
}
```

CRIP2UpdateTimeoutProcess

Tento proces vykonáva vytváranie smerovacích paketov a ich rozposielanie do výstupných bufferov jednotlivých rozhraní smerovača.

Tento proces bol doplnený o funkciu split horizont a route poisoning.

Proces bol prepracovaný tak, aby na každé rozhranie posielal iné updaty a teda vynechával v nich informáciu o cestách, ktoré sa naučil cez dané rozhranie.

Ďalej bol upravený tak, aby posielal aj cesty s hopcount = 16 a validfag = FALSE, čo je route poisoning, aby okolité smerovače dostali informáciu o nedostupnosti siete.

Updaty sa posielajú len na rozhrania, ktoré nie sú v stave passive. Verzia protokolu RIP sa nastavuje najprv na základe nastavení verzie na konkrétnom rozhraní. Ak nie je na rozhraní nič nastavané (-1), berie sa do úvahy globálne nastavenie smerovacieho protokolu.

```
tmpPacket = new C_Packet();
//v priorite paketu je ulozene cislo, z ktoreho interfejsu mi prisiel paket

int port = ((C_Router*)m_service->GetDevice())->GetPortByHostIP(routeItem-
>m_destIP);
int ver = atoi(((C_InterfaceRouter*)m_service->GetDevice()-
>GetInterface(port))->GetSendVersion());
if ( ver == 1 ) m_service->m_protRIP2->GetRip1Packet( tmpPacket,
eRip2Response );
else if ( ver == 2 ) m_service->m_protRIP2->GetRip2Packet( tmpPacket,
eRip2Response );
else if (ver == -1 && m_service->GetVersion() == 2)
                              m_service->m_protRIP2->GetRip2Packet(
tmpPacket, eRip2Response );
else if (ver == -1) m_service->m_protRIP2->GetRip1Packet( tmpPacket,
eRip2Response );
else if ( ver == 0 )
{
      delete tmpPacket;
      continue;
}
```

CRIP2GarbageTimeoutProcess

Proces zabezpečuje zmazanie položky v smerovacej tabuľke po uplynutí časovača garbage_collection. Proces bol prepracovaný, pretože spôsoboval veľa chýb a pádov aplikácie. Hlavné zmeny sa týkali ošetrenia stavov po uplynutí časovača v hľadania cesty na vyhodenie po jeho uplynutí.

6.2 Manažovateľný prepínač

Trieda C_ManagedSwitch poskytuje metódy pre realizáciu manažovateľného prepínača . Je odvodená od spoločnej triedy C_Device. Spolupracuje z triedou C_InterfaceMSwitch, ktorá implementuje porty manažovateľného prepínača.

Metódy posielania paketov, práca z tabuľkou a uspávania procesu sú rovnaké ako v prípade implementácie jednoduchého prepínača.

Proces je v nasledovných častiach odlišný.

V prípade zmeny topológie, keď sa pripojí alebo odpojí linka, sa budí proces STP, implementovaný v triede C_ServiceSTP.

V prípade ak príchodzí paket má multicast adresu pre medziprepínačovú komunikáciu, budí sa proces STP a paket sa vkladá do radu pre STP. Číslo portu sa vloží do premennej priority.

```
if ( destMac == SwMultiCastMac ){
      paket->priority = index;
( m_Sw->m_STPService->GetIncommingBuffer() )->insert( (CTrans *)paket );
      if( ( m_Sw->m_STPService->GetExecutionProcess() )->isIdle() )
( m_Sw->m_STPService->GetExecutionProcess() )->activateAfterProcess(this);
            continue;
}
```

V manažovateľnom prepínači sú implementované dva časovače. Prvý časovač sa spúšťa v prvotnej inicializácii prepínača po pripojení linky. Počas tohto času zostávajú všetky aktívne porty prepínača v blokujúcom stave. Tento časovač môže byť zrušený po prijatí bpdu rámca na hociktorom porte.

```
void C_MaxTimer::body()
{
	SNLogMessage(m_proc->m_Sw->GetName() + " " + " (M SWITCH Max
	Timer)STARTED!");

	for(;;)
	{
		if (g_exitSimulation==TRUE) passivate();
		m_proc->m_Sw->m_max_cancel = FALSE;

		Delay(g_simulation,(TIMETYPE)( (TIMETYPE)m_proc->m_Sw-
		>m_Max_Age_Timer * 1000000000.0 ),this);
		hold( (TIMETYPE)( (TIMETYPE)m_proc->m_Sw->m_Max_Age_Timer *
		1000000000.0 ) );

		if ( m_proc->m_Sw->m_max_cancel == FALSE )
		{
			m_proc->m_Sw->m_max_time = TRUE;
			if ( m_proc->m_Sw->m_STPService->GetExecutionProcess()-
			>isIdle() )
			{
				m_proc->m_Sw->m_STPService->GetExecutionProcess()-
				>activateAfterDelayT(0);
			}
		}
		passivate();
	}
}
```

Druhým časovač slúži na časové odstupy medzi posielaním Hello BPDU rámcov.

```
void C_HelloTimer::body(){
	SNLogMessage(m_proc->m_Sw->GetName() + " " + " (M SWITCH Hello
	Timer)STARTED!");
	for(;;){
		if (g_exitSimulation==TRUE) passivate();
		m_proc->m_Sw->m_hello_cancel = FALSE;

		Delay(g_simulation, (TIMETYPE)( (TIMETYPE)m_proc->m_Sw-
		>m_Hello_Timer * 1000000000.0 ) , this);

		hold( (TIMETYPE)( (TIMETYPE)m_proc->m_Sw->m_Hello_Timer *
		1000000000.0 ) );

		if ( m_proc->m_Sw->m_hello_cancel == FALSE ){
			m_proc->m_Sw->m_hello_time = TRUE;

			if ( m_proc->m_Sw->m_STPService->GetExecutionProcess()-
			>isIdle() )
			{
				m_proc->m_Sw->m_STPService->GetExecutionProcess()-
				>activateAfterDelayT(0);

				passivate();
			}
		}
	}
}
```

Každé rozhranie prepínača má implementované tri časovače, ktoré slúžia na prechod stavmi STP procesu.

- **Spanning tree algoritmus**

Spanning tree algoritmus využíva pri svojej činnosti posielanie BPDU rámcov. Tieto sa rozlišujú na konfiguračné správy a TCN (Tree Change Notification) správy. Každý switch musí mať jedinečnú mac adresu. Všetky BPDU sa posielajú na spoločnú cieľovú multicast adresu 01-80-C2-00-00-00. Na tejto adrese počúvajú všetky prepínače, ktoré majú spanning tree protokol.

Keďže nemáme implementovaný 802.3 rámec, BPDU rámec sa teda balí do Ethernet II rámca. Typ protokolu 1111(hex) pre BPDU je vybraný z neobsadených typov protokolov.

Fázy STA

Vždy keď sa prepínač zapne vysiela BPDU rámce o tom, že on je root prepínač. Keďže root prepínač posiela Hello pakety každých HelloTime sekúnd, prepínač ktorý bol práve zapojený, sa hneď dozvie, či má byť on root prepínač. Ak novo pripojený prepínač má nižšiu ID, ako aktuálny root prepínač, začína sa fáza výberu root prepínača.

Ak novo pripojený prepínač má vyššiu ID ako, aktuálny root prepínač, začína sa fáza zmeny topológie.

Fáza výberu root prepínača

Každý prepínač prepne porty do stavu blocking. Po uplynutí časovača a prepnutí portu do stavu lisening posiela hello BPDU rámce každých HelloTime sekúnd. Pri príchode každého BPDU si prepínač poznačí, ktorý port má pripojený ku ďalšiemu prepínaču. Potom sa porovná Root identifier BPDU s tým ktorý má uložený. Ak BPDU má Root identifier menší, uloží si ho a ďalej posiela BPDU s novým Root identifier. Takto to pokračuje až kým sa všetky nezhodnú na tom, ktorý je root prepínač.

Táto voľba root prepínača trvá určitý čas Forward delay. V prípade, že prepínač je sám v sieti tak všetky porty prepne do stavu forwarding. Ak prepínač dostane hello BPDU aspoň od jedného prepínača, nie je v sieti sám a prejde do fázy výberu portu a súčasne do fázy výberu predurčeného (designated) portu.

Root port následne posiela Hello pakety každých HelloTime sekúnd.

Pri porovnávaní Root identifier sa berie do úvahy najskôr, čo najnižšia priorita a až potom, čo najnižšia MAC adresa.

Fáza výberu root portu

Root prepínač začne posielať konfiguračné BPDU, kde nastaví Root path cost na hodnotu 0. Každý prepínač nastaví porty do listening módu. Keď prepínač príjme BPDU prepošle ho na ostatné porty, pričom pripočíta cenu cesty z ktorej prišiel. Najlepší BPDU už s pripočítanou cenou cesty si uchová s identifikátorom, z ktorého portu prišiel. Najlepší BPDU vyberá na základe:

1. najnižší Root identifier
2. najnižšia Root path cost
3. najnižší Bridge identifier
4. najnižší Port identifier

Po vypršaní času Forward delay sa nastaví ako root ten port, z ktorého prišiel najlepší BPDU a port prejde do stavu learning. Root port musí byť iba jeden na jednom prepínači.

Voľba root portu:

```
BOOL C_InterfaceMSwitch::ElectRoot()
{
	C_InterfaceMSwitch *intTemp;
	C_ManagedSwitch *MStemp =(C_ManagedSwitch*) m_device;

	//bool bIsRoot = TRUE;
	//bool bIsIn = FALSE;

	if(strcmp(m_bpdu->m_Root_MAC,MStemp->m_bpdu->m_Root_MAC)==0)
	{
		if( m_bpdu->m_Root_priority == MStemp->m_bpdu->m_Root_priority)
		{

			if(m_bpdu->m_Root_path_cost == MStemp->m_bpdu-
				>m_Root_path_cost)
				return TRUE;
```

```
		}
	}
		return FALSE;

	}
```

Fáza výberu predurčeného (designated) portu

Je obdobná ako fáza výberu root portu pričom predurčený port sa vyberá rovnakým spôsobom a podľa rovnakých kritérií. Rozdiel je v tom že BPDU uchováva bez pripočítania ceny cesty a porovnáva s najlepšou BPDU volenou pri výbere root portu. Pokiaľ BPDU bude menšia ako najlepšia BPDU volená pri výbere root portu, daný port sa nastaví do blocking módu. V opačnom prípade sa označia ako predurčené. Predurčených portov môže byť na prepínači viac. Po uplynutí času Forward delay port prejde do stavu learning.

Voľba predurčeného portu:

```
BOOL C_InterfaceMSwitch::ElectDesignated()
{
	if(m_bHaveBPDU==FALSE) return true;


	if(m_state == eUp && strcmp(((C_ManagedSwitch*) m_device)->m_bpdu-
	>m_Root_MAC,m_bpdu->m_Root_MAC) < 0)
		return true;
	else if (m_state == eUp && strcmp(((C_ManagedSwitch*) m_device)-
		>m_bpdu->m_Root_MAC,m_bpdu->m_Root_MAC) == 0 && strcmp(m_port-
		>m_addressMAC,m_bpdu->m_Bridge_MAC)<0 )
		return true;


	return FALSE;
	}
```

Všetky root porty a predurčené porty sa nakoniec zo stavu learning po vypršaní času Forward delay, nastavia do stavu forwarding.

6.3 Sledovanie komunikácie (sniffer)

Z dôvodu existencie spoločného snifferu pre každé spojenie bolo rozhodnuté, že sa celý tento podsystém prebuduje tak, aby bol jediný sniffer pre všetky spojenia. Pakety sa uchovávajú v triede odvodenej z mapy (za použitia knižnice STL), kde ako vyhľadávací kľúč

slúži číslo paketu. Okrem samotného paketu (položka *SStoredPacket::packet*) je ešte ukladaná informácia, ktorými spojeniami príslušný paket prešiel. Táto sa ukladá v podobe bitového vektoru (položka *SStoredPacket::connections*).

```
class SStoredPacket           //! predstavuje strukturu pre ulozenie paketu
{
public:
      C_Packet    packet;           //! samotny paket
      std::vector<bool> connections;      //! uchovava cestu ramca

      SStoredPacket(const C_Packet& src);
      SStoredPacket(const SStoredPacket &src);
      operator=(SStoredPacket src);
} ;

typedef std::map<unsigned int,SStoredPacket> SPacket;
```

Samotné ukladanie paketov prebieha pri ich vyslaní cez rozhranie zariadenia na spojenie (metóda `typedef std::map<unsigned int,SStoredPacket> SPacket;`). Najprv sa overí, či sa daný paket (má rovnaké číslo ako niektorý už zaznamenaný) už vyskytol v systéme. V prípade, že je to úplne nový paket, pridá sa do mapy, pričom sa už aj nastaví príznak jeho existencie na príslušnom spojení.

```
      SPacketIterator m_iter = m_PacketBuffer.find(packet->GetNo());
      if(m_iter == m_PacketBuffer.end()) {
            SStoredPacket *c = new SStoredPacket(*packet);
            c->connections.at(conNo-1) = 1;
            std::pair<unsigned int,SStoredPacket> p(packet->GetNo(), *c);
            m_PacketBuffer.insert(p);
            m_currentPacketCount++;
      }
```

V prípade, že sa už dotyčný paket vyskytol na sieti a bol poslaný na iné rozhranie za pomoci opakovača alebo prepínača, tak bude iterátor *m_iter* nastavený na záznam uchovávajúci informácie o ňom. V tomto prípade postačuje zmeniť príznak jeho existencie na spojení.

```
      m_iter->second.connections.at(conNo-1) = 1;
```

Okrem samotného uloženia sa informácie o novo prijatom pakete pridajú aj do dialógu snifferu.

```
    m_currentListener->OnSniffEvent(eSniffEventNewPacket,this,packet->GetN
o() );
```

Základné informácie o jednotlivých odchytených paketoch sú zobrazené v dialódu snifferu. V prípade, že dôjde k výberu niektorého paketu, je podľa poradového čísla

vyhľadaný medzi uloženými paketami. Následne sa zavolá metóda `int C_Sniffer::GetPacketInfo(unsigned int packetNumber, SPacketInfo* packetInfo )`, ktorá na základe tohto čísla *packetNumber* vyčíta informácie a uloží ich do premennej *packetInfo*. Na základe týchto vrátených údajov následne vyplní sniffer príslušné položky v dialógu.

```
this->m_editSrcMac.SetWindowText( GetFullMacAddress( tmpInfo.srcMac ) );
this->m_editDstMac.SetWindowText( GetFullMacAddress( tmpInfo.dstMac ) );

switch( tmpInfo.ethProtType )
{
	case eProtocolIP:
		this->m_editEthProt.SetWindowText( "IP protokol" );
		break;
	case eProtocolARP:
		this->m_editEthProt.SetWindowText( "ARP protokol" );
		break;
	case eProtocolRARP:
		this->m_editEthProt.SetWindowText( "RARP protokol" );
		break;
	case eProtocolDHCPReq:
		this->m_editEthProt.SetWindowText( "DHCP ziadost" );
		break;
	case eProtocolDHCPRes:
		this->m_editEthProt.SetWindowText( "DHCP odpoved" );
		break;
	case eProtocolSTP:
		this->m_editEthProt.SetWindowText( "SpanningTree protokol" );
		break;
	}
	this->m_editSrcIP_A.SetWindowText( GetIPPart( tmpInfo.srcIP, 1 ) );
	this->m_editSrcIP_B.SetWindowText( GetIPPart( tmpInfo.srcIP, 2 ) );
	this->m_editSrcIP_C.SetWindowText( GetIPPart( tmpInfo.srcIP, 3 ) );
	this->m_editSrcIP_D.SetWindowText( GetIPPart( tmpInfo.srcIP, 4 ) );
	this->m_editDstIP_A.SetWindowText( GetIPPart( tmpInfo.dstIP, 1 ) );
	this->m_editDstIP_B.SetWindowText( GetIPPart( tmpInfo.dstIP, 2 ) );
	this->m_editDstIP_C.SetWindowText( GetIPPart( tmpInfo.dstIP, 3 ) );
	this->m_editDstIP_D.SetWindowText( GetIPPart( tmpInfo.dstIP, 4 ) );

	tmpStr.Format( "%d", tmpInfo.ttl );
	m_editTTL.SetWindowText( tmpStr );

	switch( tmpInfo.ipProtType )
	{
		case eProtocolICMP:
			this->m_editProtNet.SetWindowText( "ICMP protokol" );
			break;
		case eProtocolIGMP:
			this->m_editProtNet.SetWindowText( "IGMP protokol" );
			break;
		case eProtocolTCP:
			this->m_editProtNet.SetWindowText( "TCP protokol" );
			break;
		case eProtocolUDP:
			this->m_editProtNet.SetWindowText( "UDP protokol" );
			break;
		default:
			this->m_editProtNet.SetWindowText( "- - - -" );
```

```
            break;
}

tmpStr.Format( "%d", tmpInfo.srcPort );
m_editSrcPort.SetWindowText( tmpStr );

tmpStr.Format( "%d", tmpInfo.dstPort );
m_editDstPort.SetWindowText( tmpStr );

tmpStr.Format( "%d", tmpInfo.seqenceNumber );
m_editSeqNum.SetWindowText( tmpStr );

tmpStr.Format( "%d", tmpInfo.acknowledgeNumber );
m_editAckNum.SetWindowText( tmpStr );

switch( tmpInfo.arpPacketType )
{
      case eArpRequest:
            this->m_editArpType.SetWindowText( "ARP request" );
            break;
      case eArpResponse:
            this->m_editArpType.SetWindowText( "ARP response" );
            break;
      case eRarpRequest:
            this->m_editArpType.SetWindowText( "RARP request" );
            break;
      case eRarpResponse:
            this->m_editArpType.SetWindowText( "ARP response" );
            break;
      default:
            this->m_editArpType.SetWindowText( "- - - -" );
            break;
}

switch( tmpInfo.icmpPacketType )
{
      case eIcmpEchoResponse:
            this->m_editIcmpType.SetWindowText( "Echo Response" );
            break;
      case eIcmpUndelivered:
            this->m_editIcmpType.SetWindowText( "Undelievered" );
            break;
      case eIcmpEchoRequest:
            this->m_editIcmpType.SetWindowText( "Echo Request" );
            break;
      case eIcmpTimeExceeded:
            this->m_editIcmpType.SetWindowText( "Time Exceeded" );
            break;
      default:
            this->m_editIcmpType.SetWindowText( "- - - -" );
            break;
}
```

Ukážky funkčnosti snifferu sa nachádzajú v používateľskej príručke.

6.4 Ukladanie konfigurácie

Pretože boli pridané niektoré ďalšie funkcie a stavy zariadení, musela byť doplnená aj funkcia zapisovania schémy do súboru ako aj funkcia čítania schémy zo súboru. (`CWSElement::Serialize )`

Pretože po načítaní štruktúr zo súboru sa spúšťa proces `C_RefreshProcess`, ktorého funkcia spočíva vo vytvorení inštancií tried a nastavenie ich parametrov podľa načítaných štruktúr, musel byť aj tento proces doplnený o nové funkcie, ktoré správne nastavia sieťové prvky.

6.5 Doplnenie GUI

Pre nastavovanie smerovacích protokolov a jednotlivých rozhraní boli pridané do existujúceho konfiguračného rozhrania tlačidlá a dialógy na umožnenie jednoduchej manipulácie. Tie sú popísané v používateľskej príručke.

6.6 Doplnenie konzoly

Pre nastavovanie smerovacích protokolov a jednotlivých rozhraní, boli do konzoly smerovača pridané príkazy ktoré sú popísané v používateľskej príručke.

7 Používateľská príručka

7.1 Požiadavky, inštalácia a spustenie

Aplikáciu je možné používať iba na počítačoch s operačným systém Windows 2000 a Windows XP. Minimálna potrebná konfigurácia je procesor Pentium 133 MHz a 32 MB pamäte RAM. Optimálna konfigurácia je procesor Pentium II 266MHz a 128 MB pamäte RAM, avšak nároky na procesor, a taktiež na pamäť vo veľkej miere závisia od veľkosti vytvorenej topológie, a od aktivity vytvorenej siete.

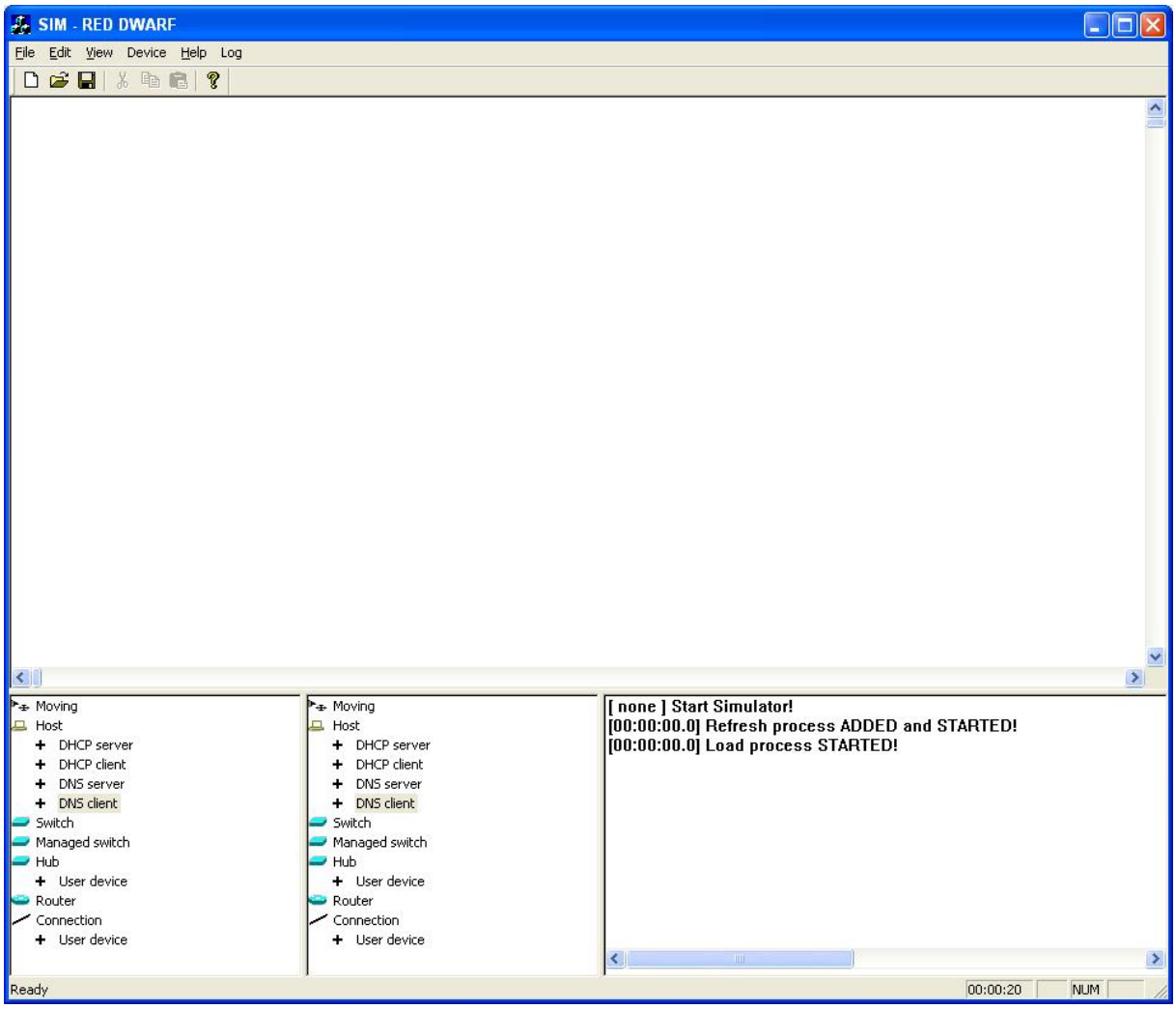
Program sa následne spúšťa klasickým spôsobom poklepaním po jeho ikone alebo z príkazového riadku.

7.2 Vytvorenie sieťovej topológie a jej editovanie

V tejto časti vysvetlíme spôsob práce s pracovnou plochou aplikácie. Podrobne opíšeme spôsob vytvárania sieťovej topológie, jej zapísania do súboru, načítania a editácie.

7.2.1 Pracovná plocha

Pracovná plocha aplikácie je rozdelená na štyri hlavné časti: plocha na ktorú je vykresľovaná topológia siete, knižnica použiteľných zariadení, knižnica zariadení na pracovnej ploche a okno pre výpis logovacích informácií. Na ploche sa tiež nachádza panel nástrojov a hlavná ponuka aplikácie.



Obrázok č. 33: Pracovná plocha

7.2.2 Vytvorenie topológie

Po spustení aplikácie sa na ploche nenachádza žiadna topológia, a preto môžeme okamžite začať s pridávaním nových prvok do topológie. V prípade, že už máme vytvorenú nejakú topológiu, v editovaní ktorej už nechceme pokračovať, a chceme vytvoriť novú, vykonáme to buď kliknutím na ikonu , príp. vybratím položky „File → New“ v hlavnom menu, príp. klávesovou kombináciou „Ctrl+N“.

Pridanie nového sieťového prvku do vytváranej topológie sa deje nasledovne: Zariadenie, ktoré chceme pridať do topológie si vyberieme buď z knižnice použiteľných zariadení, a to kliknutím na dané zariadenie, alebo kliknutím na položku zariadenia v menu „Device“. Po vybratí zariadenia sa zmení kurzor, a to na obrázok daného zariadenia. Vybrané zariadenie môžeme teraz kliknutím umiestniť na kresliacu plochu. Takto môžeme vytvoriť aj viacero zariadení rovnakého typu za sebou. Ak chceme zmeniť typ zariadenia, tak

postupujeme podobne ako keď sme zariadenie vyberali po prvý krát. Ak už nechceme pridávať nové prvky do topológie, tak klikneme pravým tlačidlom myši, čím sa kurzor vráti do pôvodnej podoby. V tomto okamžiku môžeme kliknutím ľavým tlačidlom a jeho podržaním presúvať jednotlivé zariadenia po ploche.

Špecifickým zariadením je linka. Jej vybratím z ponuky zariadení sa kurzor myši navonok nezmení, ale zmení sa jeho funkcia. Vytvorenie spojenia spočíva v kliknutí ľavým tlačidlom myši na spojované zariadenie a pretiahnutím spojenia na zariadenie ktoré chceme pripojiť. Nad týmto zariadením pustíme ľavé tlačidlo, čím sa spojenie vytvorí. V prípade, že pustíme tlačidlo nie nad zariadením, tak sa spojenie nevytvorí.

Používateľ môže zariadenie z vytvorenej topológie vymazať dvoma spôsobmi. Buď si kliknutím na zariadenie pravým tlačidlom myši vyvolá kontextové menu a následne vyberie „Delete“, alebo označí zariadenie kliknutím ľavým tlačidlom myši a vyberie z hlavného menu položku „Edit → Delete“. Vymazaním zariadenia sa automaticky vymažú aj všetky prípadné spojenia.

7.2.3 Uloženie topológie

Používateľom môže vytvorenú topológiu uložiť do súboru kliknutím na ikonu  v panely nástrojov, príp. vybratím položky „File → Save“ alebo „File → SaveAs“ z hlavného menu, príp. klávesovou kombináciou „Ctrl+S“.

V prípade, že používateľ zvolí spôsob „File → SaveAs“ alebo v prípade, že vytvorená topológia ešte nebola nikdy uložená, aplikácia vyvolá dialóg, v ktorom používateľ môže zadať meno, pod ktorým bude daná topológia uložená. Tiež v ňom môže vybrať adresár, do ktorého bude súbor uložený.

V prípade, že sa pokúsime načítať nový súbor bez toho aby sme pôvodný uložili, budem na to vyzvaný.

7.2.4 Načítanie uloženej topológie

Používateľ môže načítať vytvorenú topológiu kliknutím na ikonu  v panely nástrojov, príp. vybratím položky „File → Open“ z hlavného menu, príp. klávesovou kombináciou „Ctrl+O“. Tým sa vyvolá dialóg, v ktorom si používateľ môže vybrať súbor, ktorý chce otvoriť.

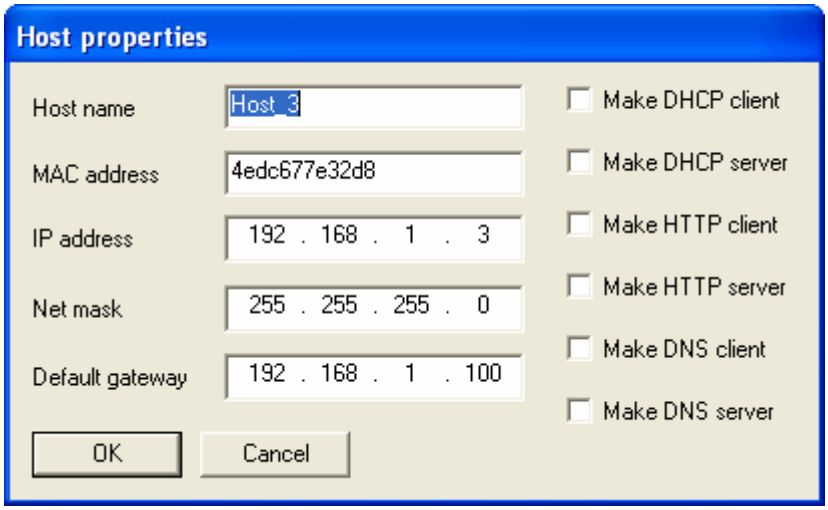
7.2.5 Vytvorenie nových knižničných zariadení

Používateľ si môže preddefinovať niektoré vlastné zariadenia. V hlavnom menu vyberie položku „Device → Insert new device“, čím otvorí dialógové okno, v ktorom si môže vybrať akému zariadeniu chce priradiť svoje vlastné počiatočné vlastnosti, a toto zariadenie pomenovať. Používateľ si napríklad môže upraviť zariadenie Host pridaním služby DHCP klient. Po pridaní takého zariadenia sa používateľovi toto zariadenie objaví v zozname knižničných zariadení. Ak takéto zariadenie vloží na plochu, tak sa začne hneď správať ako DHCP klient, a teda pokúsi sa automaticky získať IP adresu pomocou protokolu DHCP.

7.3 Konfigurovanie sieťových zariadení

7.3.1 Nastavenie vlastností zariadenia typu Host

Vlastnosti zariadenia Host je možné nastaviť v dialógu, ktorý používateľ môže vyvolať buď pravým kliknutím na zariadenie Host a následne vybratím položky „Properties“ z kontextového menu, alebo označením zariadenia ľavým tlačidlom myši a vybratím položky „Device → Properties“.



Obrázok č. 34: Konfiguračný dialog zariadenia HOST

K nastaviteľným vlastnostiam zariadenia Host patrí nastavenie:

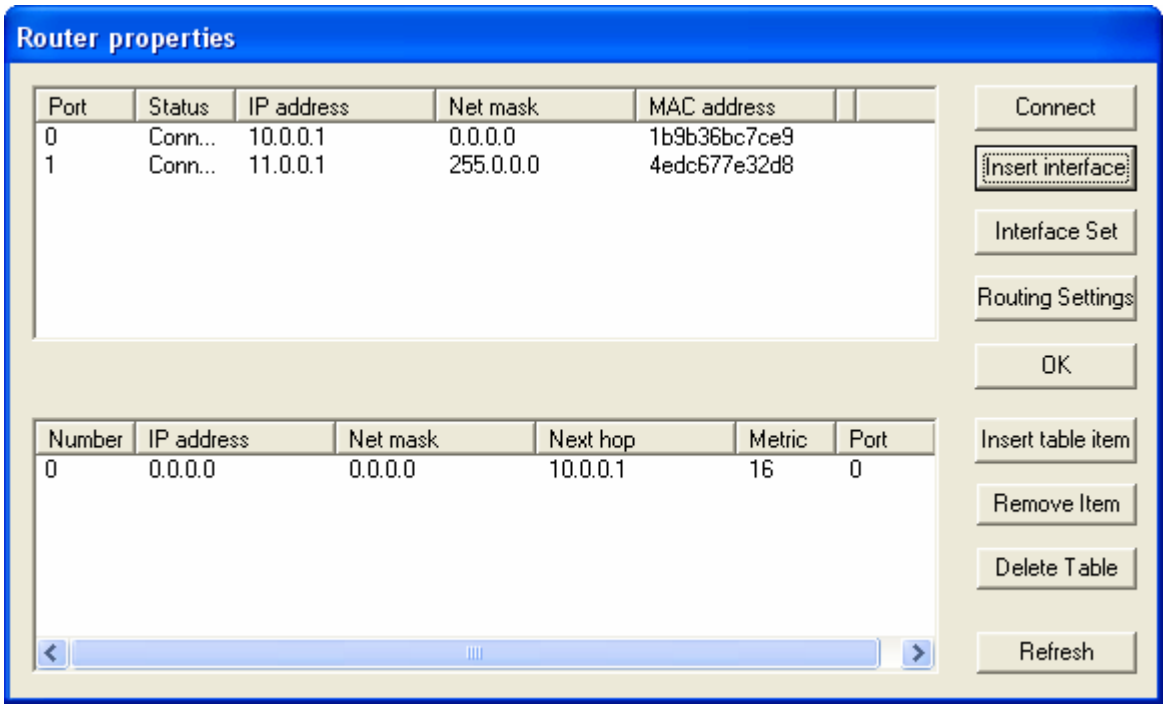
- Mena zariadenia
- MAC adresy
- IP adresy

- Sieťovej masky
- IP adresy defaultnej brány

V dialógu je tiež možné zariadeniu Host priradiť úlohu DNS servera, DNS klienta, DHCP servera, DHCP klienta, HTTP servera a HTTP klienta.

7.3.2 Nastavenie vlastností zariadenia typu Router

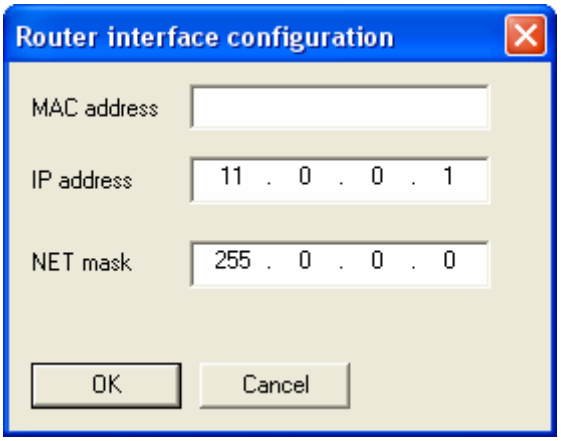
Vlastnosti zariadenia Router je možné nastaviť podobne ako pri zariadení typu Host v dialógovom okne. Toto dialógové okno sa vyvolá rovnakým spôsobom ako pri zariadení Host. Keďže pre nastavenie vlastností zariadenia Router je primárne určený simulátor CISCO IOS konzoly, tak v dialógu sa dajú nastaviť len niektoré vlastnosti.



Obrázok č. 35: Konfiguračný dialóg zariadenia ROUTER

Patrí sem hlavne možnosť priradenia nových rozhraní smerovaču a vytvorenie statických smerovacích ciest. Priradenie nových rozhraní používateľ vykoná kliknutím na tlačidlo „Insert interface“. Tým vyvolá dialóg, v ktorom je mu umožnené nastaviť MAC adresu IP adresu a sieťovú masku pre dané rozhranie. Novú statickú smerovaciu cestu používateľ pridá kliknutím na tlačidlo „Insert Table Item“. Vo vyvolanom dialógu používateľ určí IP adresu cieľovej siete, jej sieťovú masku a IP adresu nasledujúceho smerovača na ceste.

Interface set – umožní nastavenie a prenastavenie parametrov rozhrania ako IP adresu a masku. Treba mať vybrané konkrétne rozhranie v zozname, kliknutím na jeho číslo.



Obrázok č. 36: Zmena nastavení rozhrania

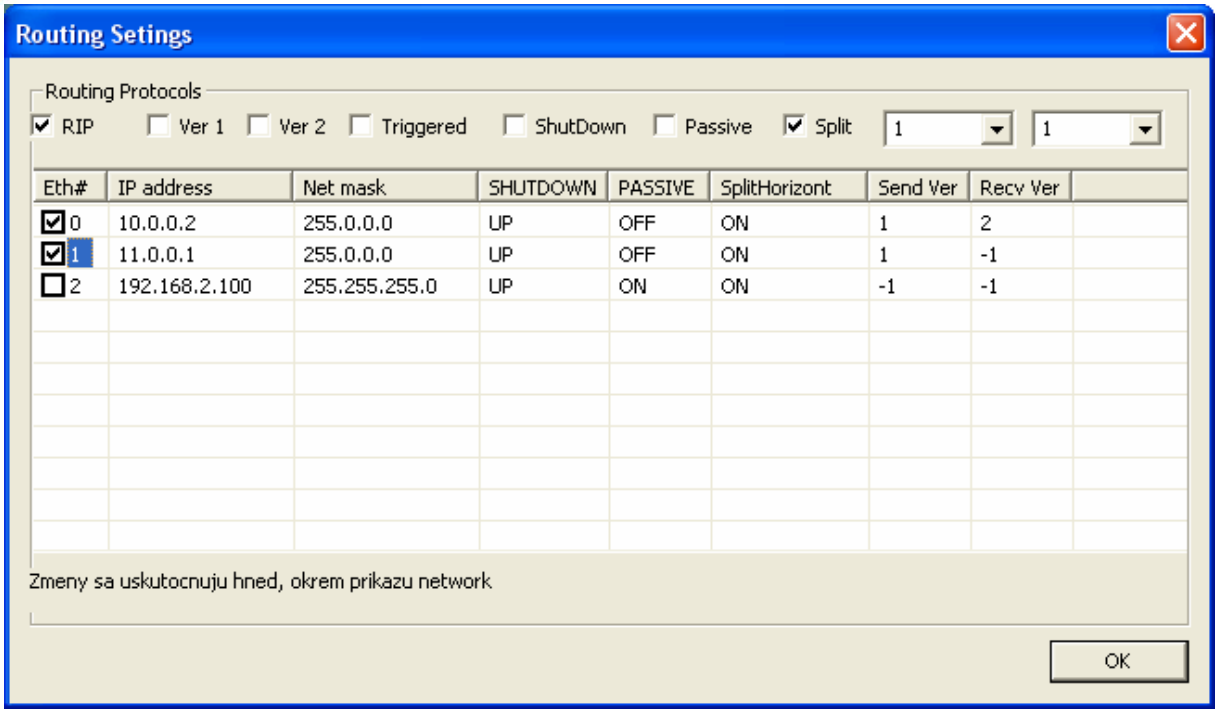
Remove Item – odstráni vybratú cestu zo smerovacej tabuľky. (vyberá sa podobne ako rozhrania)

Delete Table – zmaže celú smerovaciu tabuľku

Mazanie sa týka všetkých ciest okrem priamo pripojených ciest, tie sa z tabuľky nedajú vymazať. Odstránia sa až zhodením rozhrania akýmkoľvek spôsobom.

Refresh – znovu načítanie obsahu tabuľky do okna. Pre prípad, že prišiel nejaký update.

Routing Settings – otvorí konfiguračný dialóg pre smerovacie protokoly



Obrázok č. 37: Konfiguračný dialóg zariadenia ROUTER pre smerovacie protokoly

V dialógu si môžeme vybrať jeden z dvoch dostupných smerovacích protokolov a tiež môžeme zapnúť službu pre okamžité spúšťanie updatov pri nejakej zmene (triggered).

V zozname sú vypísané všetky rozhrania na smerovači a ich aktuálne nastavenia. Po výbere niektorého rozhrania je možné zmeniť niektorý z ich parametrov Shutdown, Passive, Split. Zmeny v konfigurácii sa uskutočnia okamžite.

Zaškrtávacie políčko pri čísle rozhrania je zaškrtnuté ak je rozhranie zahrnuté do zoznamu sietí pre smerovanie (príkaz network). Je možné takto pridávať a odoberať rozhrania zo smerovacieho procesu. Táto zmena sa však prejaví až po stlačení tlačidla OK.

V prípade , že chceme vypnúť smerovanie úplne, musíme odznačiť RIP.

Ver1 a Ver2 sú identické s príkazom Version. Ak sú obe odznačené, tak je prijíma aj RIP1 aj RIP2, a odosiela len RIP1.

SendVer a RecvVer sú verzie protokolov na jednotlivých rozhraniach. -1 značí základné nastvenie(neblokuje nič), 12 pre príjem oboch verzií, 0 pre blokovanie.

7.3.3 Konzola CLI

Konzola sa používa na konfiguráciu a sledovanie nastavenia daného zariadenia. Fungovanie konzoly predstavuje jednoduchý príkazový riadok, ktorý spúšťa príkazy zadané

používateľom. Jej spustenie sa vyvolá dvojitým poklepaním na zariadenie. Konzola pre každé zariadenie je do istej miery kópiou reálnej konzoly zariadení.

Základné správanie konzoly kopíruje správanie sa konzoly pre CISCO IOS:

- Zadanie „?“ – ak je tento príkaz zadaný samostatne v riadku, do konzoly sa vypíše zoznam príkazov, danej konfiguračnej úrovne, ktoré sú k dispozícii. Nie všetky príkazy sú implementované. Zoznam funkčných príkazov je uvedený v ďalších kapitolách.
- Zadanie „príkaz?“ – zobrazí zoznam príkazov, ktoré sa zhodujú so zadaným príkazom.
- Zadanie „príkaz ?“ – zobrazí zoznam parametrov prislúchajúcich k zadanému príkazu.
- Zadanie „príkaz + TAB“ – doplní zadaný reťazec príkazom, ak taký existuje.
- Spustenie správne zadaného príkazu sa vykoná po stlačení klávesu „ENTER“

Jedným z možných príkazov je príkaz „telnet“. Za týmto príkazom nasleduje IP adresa cieľového zariadenia. Po pripojení sa na iné zariadenie môžeme zadávať príkazy priamo na lokálnom zariadení. Zadávané príkazy sa posielajú na vzdialený počítač, tam sa vykonajú a výsledky sa odošlú späť.

Ďalej sú uvedené len nové doplnené príkazy.

7.3.3.1 Smerovanie

Router(config)#router rip	- zapnutie smerovacieho protokolu
Router(config)# no router rip	- vypnutie smerovacieho protokolu
Router(config-router)# netowork	- pridanie priamo pripojenej siete do smerovacieho procesu
Router(config-router)# no network	- odobratie priamo pripojenej siete do smerovacieho procesu
Router(config-router)# version <1 alebo 2>	- zvolenie verzie smerovacieho protokolu
Router(config-router)#no version	- základné nastavenie smerovacieho protokolu
Router(config-router)# triggered	- zapnutie okamžitého posielania updatov
Router(config-router)#no triggered	- vypnutie okamžitého posielania updatov

Router(config-router)#timers <basic, flush, invalid, update> <cr> - nastavenie hodnoty časovačov smerovacieho protokolu

Router(config-router)#passive-interface ethernet <cr> - vypnutie posielania updatov na konkrétnom rozhraní

Router(config-router)#no passive-interface ethernet <cr> - zapnutie posielania updatov na konkrétnom rozhraní

7.3.3.2 Rozhrania

Router(config-if)#shutdown - vypnutie rozhrania

Router(config-if)#no shutdown - zapnutie rozhrania

Router(config-if)#ip split-horizont - zapnutie split-horizontu

Router(config-if)#no ip split-horizont - vypnutie split-horizontu

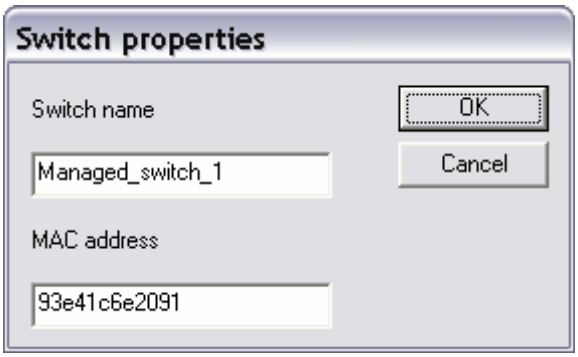
Router(config-if)# ip rip {send | receive} vesion {1 | 2 | 12 | off} -nastavenie smerovacieho protokolu RIP na rozhraní

7.3.3.3 Všeobecné príkazy

Router#clear ip route <A.B.C.D network> <A.B.C.D mask> | * pre všetky cesty - príkaz na mazanie smerovacej tabuľky alebo len jednotlivých ciest

7.3.4 Nastavenie vlastností zariadenia typu prepínač

Vlastnosti zariadenia Prepínač je možné nastaviť podobne ako pri zariadení typu Host v dialógovom okne. Toto dialógové okno sa vyvolá rovnakým spôsobom ako pri zariadení Host. Dialógové okno umožňuje zmenu názvu a mac adresy prepínača.



Obrázok č. 38: Konfiguračný dialóg zariadenia HOST

Jednotlivé fázy stavu portov sú znázornené rôznymi farbami.
Fáza blocking je znázornená ružovou farbou.

Obrázok č. 39: Fáza blocking

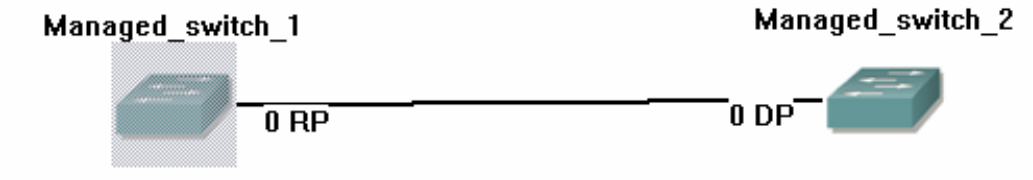
Fáza listening je znázornená žltou farbou.

Obrázok č. 40: Fáza listening

Fáza learning je znázornená modrou farbou.

Obrázok č. 41: Fáza learning

Fáza forwarding je znázornená čiernou farbou.



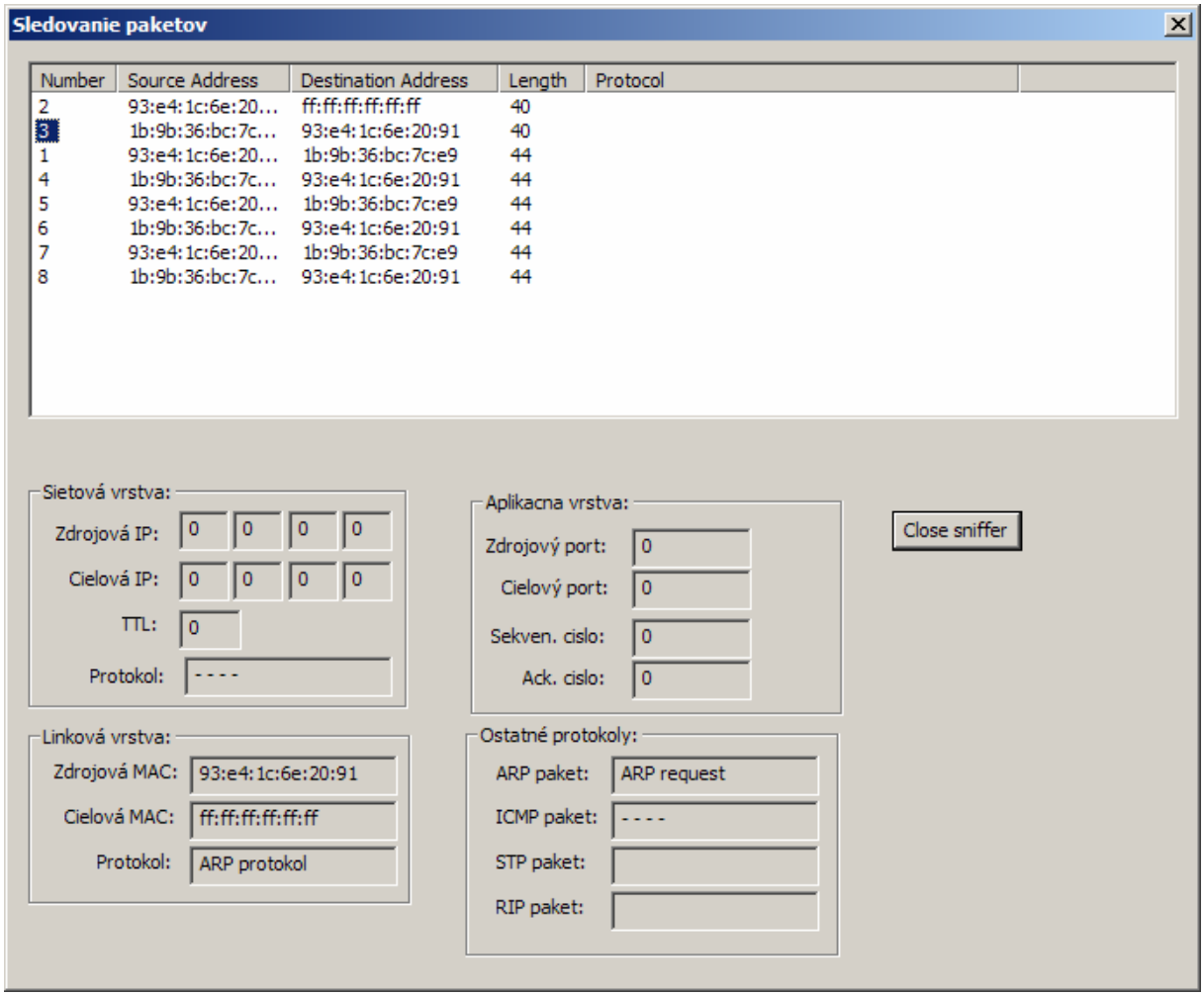
Obrázok č. 42: Fáza forwarding

Označenie portu obsahuje číslo rozhrania na danom zariadení a typ portu v zmysle STP. Kde označuje

- „RP“ - koreňový port,
- „DP“ – predurčený port a
- „BP“ – zablokovaný port.

7.3.5 Sniffer

Informácie o všetkých paketoch, ktoré prešli určitým spojením môžu byť pre pozorovateľa siete veľmi zaujímavé. Práve pre tieto účely bol vytvorený objekt sniffer (sledovač), pomocou ktorého môže používateľ sledovať celkovú komunikáciu cez určité prepojenie v sieti. Na nasledujúcom obrázku je možné vidieť dialóg sledovania paketov.



Vo vrchnej časti sú informácie o zachytených paketoch, pričom sa ukazujú základné informácie o každom pakete ako sú MAC adresa odosielateľa a príjemcu a taktiež aj jeho

dĺžka. Ostatnú časť okna predstavujú informačné textové údaje, ktoré zobrazujú detailné informácie o aktuálnom pakete.

V prípade požiadavky na zobrazenie podrobnejších informácií o dotyčnom pakete je potrebné kliknúť naň a v spodnej časti dialógu sa zobrazia informácie, ako sú IP adresa odosielateľa a príjemcu, typ paketu, čísla portov (v prípade použitia TCP alebo UDP protokolov) a pod.

Objekt sniffer, v prípade že je aktivovaný, ukladá všetky pakety, pričom ich množstvo nie je obmedzené (len pamäťovými schopnosťami počítača). V prípade veľkej aktivity siete a dostatočne dlhého behu simulátora môže dôjsť k zaplneniu voľnej pamäte, čo má za následok fakt, že simulátor nebude plne schopný svojej funkcie (pretože nebude mať kam uchovávať ďalšie pakety).

8 Uskutočnené testy

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Smerovacia tabuľka
Opis testu:	Test spočíval v napĺňaní tabuľky rôznymi cestami, pričom sa očakávalo potrebné usporiadavanie podľa destinácie, podľa dĺžky masky a podľa metriky. Ďalej sa testovalo mazanie cesty, hľadanie cesty.
Požiadavka:	Úspešne vykonané všetky spomenuté operácie.
Výsledok:	OK
Opravná činnosť:	žiadna

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Update Timer
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi a hostami.
Požiadavka:	Úspešné naplnenie smerovacích tabuliek na všetkých hostoch a ich obnova po vypršaní časovača
Výsledok:	OK
Opravná činnosť:	žiadna

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Invalide Timer
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi a hostami. Po naplnení smerovacích tabuliek sa zmazala nejaká cesta.
Požiadavka:	Úspešné naplnenie smerovacích tabuliek na všetkých hostoch a ich obnova po vypršaní časovača. Po zrušení nejakej cesty sa daná sieť nastaví ako nedostupná (šíri sa s metrikou 16), a spustí sa jej garbage timer (route poisoning)
Výsledok:	OK
Opravná činnosť:	žiadna

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Garbage timer
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi a hostami. Po naplnení smerovacích tabuliek sa zmazala nejaká cesta.
Požiadavka:	Úspešné naplnenie smerovacích tabuliek na všetkých hostoch a ich obnova po vypršaní časovača. Po zrušení nejakej cesty sa daná sieť nastavý ako nedostupná, a spustí sa garbage timer, po jeho uplynutí sa cesta zmaže z tabuľky
Výsledok:	OK
Opravná činnosť:	žiadna

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Zmena hodnoty garbage časovača
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi a hostami. Po naplnení smerovacích tabuliek sa zmazala nejaká cesta. Hodnota časovača zmenená zo 120s na 30s čo je hodnota updatov
Požiadavka:	Úspešné naplnenie smerovacích tabuliek na všetkých hostoch a ich obnova po vypršaní časovača. Po zrušení nejakej cesty sa daná sieť nastavý ako nedostupná, a spustí sa garbage timer, po uplynutí tohto časovača sa musí cesta zmazať.
Výsledok:	Zastavenie simulácie
Opravná činnosť:	Spôsobené zápornou hodnotou času vo funkcii Delay. Ošetrené tak, že ak je hodnota záporná tak sa nastaví na hodnotu časovača. Tým sa zabezpečí okamžité zrušenie cesty ak je čas od vloženia do tabuľky väčší ako hodnota časovača.

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Zmena hodnoty Invalid časovača
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi a hostami. Hodnota časovača zmenená zo 180s na 15s čo je menej ako hodnota updatov
Požiadavka:	Úspešné naplnenie smerovacích tabuliek na všetkých hostoch. Po uplynutí 15s po prijatí updatu, sa všetky nelokálne cesty musia zmazať z tabuľky. A po prijatí ďalšieho updatu sa musia znova objaviť.
Výsledok:	Zastavenie simulácie
Opravná činnosť:	Spôsobené veľmi malým rozdielom v časoch vypršania timera a vloženia cesty. Ošetrené pripočítaním 1ms k časom aby bola splnená potrebná podmienka.

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Ping
Opis testu:	Test spočíval vo vytvorení topológie s smerovačmi a hostami a prepínačmi
Požiadavka:	Úspešné naplnenie smerovacích tabuliek na všetkých hostoch a potom úspešné pingnutie najvzdialenejších hostov.
Výsledok:	Pád aplikácie
Opravná činnosť:	Pri posielaní paketu bol jeden pointer = NULL; ošetrené, trieda CManagedSwitch

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2

Modul:	Sniffer
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi ,manažovateľnými prepínačmi a hostami. Zapnutie sniffera, a preskúmanie funkcionality.
Požiadavka:	Správne vypisovanie údajov
Výsledok:	Zlé vypisovanie obsahu, prepisovanie vybratej položky s príchodom nového paketu
Opravná činnosť:	Pridaná globálna premenná na udržanie indexu zvolenej položky. Oprava vypisovania údajov (spôsobené zlým indexom)

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Príkaz Network
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi ,manažovateľnými prepínačmi a hostami a zadaním príkazu „no network“ na ľubovoľnom rozhraní, a jeho opätovnom zapnutí (po čase)
Požiadavka:	Po zadaní príkazu „no network“ , spustenie invalid časovačov na ostatných routroch pre siete cez toto rozhranie, vrátane časovačov na lokálnom smerovači pre všetky cesty naučené cez toto rozhranie.
Výsledok:	Rozhranie bolo odstránené z network listu, avšak iný účinok to nemalo, nespustili sa žiadne časovače.
Opravná činnosť:	Pridané podmienky do smerovacieho protokolu, aby sa cesty mimo network listu neposielali v updatoch, neprijímali sa na nich updaty a ani cez ne sa žiadne updaty neposielali.

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Dialóg RoutingSetting
Opis testu:	Testovaná bola funkcionalita a jednoduchosť ovládania nastavení
Požiadavka:	Jednoduchosť ovládania
Výsledok:	Po výbere rozhrania na smerovači a zmene jedného parametru bolo potrebné znovu vybrať rozhranie, čo pri zmene viacerých parametrov viedlo k nadmernému klikaniu.
Opravná činnosť:	Pridanie globálnej premennej na zapamätanie si indexu posledného zvoleného rozhrania

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Triggered Updates
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi ,manažovateľnými prepínačmi a hostami zapnutím okamžitých updatov na ľubovoľných smerovačoch.
Požiadavka:	Okamžité posielanie updatov zo smerovačoch so zapnutou funkciou, ak prijali update alebo došlo k zmene stavu ich rozhrania.
Výsledok:	OK

Opravná činnosť:	Žiadna

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Split Horizont
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi ,manažovateľnými prepínačmi a hostami, tak aby smerovače tvorili slučku. Nechali sme skonvergovať smerovací protokol. A potom sme vypli Split horizont na týchto routroch. A odstránili jedno prepojenie.
Požiadavka:	Po spustení simulácie je očakávané skonvergovanie smerovacích tabuliek do 60s, po vypnutí horizontu a spojenia, sa bude hopcount ciest pre odstránenú cestu neustále zvyšovať až po 16 a až potom sa cesta označí ako nedostupná.
Výsledok:	OK
Opravná činnosť:	Žiadna

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Passive Interface
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi ,manažovateľnými prepínačmi a hostami a zadaním príkazu „passive interface“ na ľubovoľnom rozhraní, a jeho opätovnom zapnutí (po čase)
Požiadavka:	Po zadaní príkazu „passive interface“ , spustenie invalid časovačov na ostatných smerovačoch pre siete cez toto rozhranie. Lokálny router by sa mal ďalej učiť siete cez toto rozhranie.
Výsledok:	OK
Opravná činnosť:	žiadna

Vykonávateľ:	Michal Jánoš
Dátum:	6.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Spojene Smerovač prepínač
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi ,manažovateľnými prepínačmi a hostami so spusteným RIP.
Požiadavka:	Správne naplnenie tabuliek a po skonvergovaní STP a RIP
Výsledok:	Na rozhraní smerovača spojenom s prepínačom sa metrika zväčšila na 1
Opravná činnosť:	Ošetrenie smerovacieho protokolu, ak prijme update, ktorý vyslal sám z toho istého rozhrania tak ho zahodí. Ďalšia chyba bola v procese posielania broadcastov na prepínači, kedy sa najprv vložil paket do radu a až potom sa testoval stav rozhrania, či je možné na neho posielať. Vkladanie presunuté do vnútra podmienky.

Vykonávateľ:	Michal Jánoš
Dátum:	19.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Odstránenie linky
Opis testu:	Test spočíval vo vytvorení topológie so smerovačmi ,manažovateľnými prepínačmi a hostami so spusteným RIP.
Požiadavka:	Správne naplnenie tabuliek a po skonvergovaní STP a RIP a odtránení spojenia
Výsledok:	Rozhranie na opačnej strane spojenie ostalo aktívne
Opravná činnosť:	Doplnenie zhodenia rozhrania na opačnej strane spojenia

Vykonávateľ:	Martin Hornáček
Dátum:	14.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Dialóg RoutingSetting
Opis testu:	Testovanie jednotlivých tlačidiel a overenie ich funkčnosti
Požiadavka:	Správna odpoveď systému
Výsledok:	Ak nebolo na smerovači priradené žiadne rozhranie a bolo stlačené tlačidlo *Connect* nastal pád aplikácie.
Opravná činnosť:	Ak nebolo vybrané žiadne rozhranie, automaticky sa vybralo rozhranie 1, ktoré však nebolo definované. Ak sa teraz vyberie nedefinované rozhranie, nevykoná nič.

Vykonávateľ:	Martin Hornáček
Dátum:	14.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Konvergencia STP stromu
Opis testu:	Test spočíval vo vytvorení topológie s manažovateľnými prepínačmi.
Požiadavka:	Po spustení sa mal na obrazovke zobraziť vytvorený strom a odstrániť slučky.
Výsledok:	OK
Opravná činnosť:	Žiadna

Vykonávateľ:	Martin Hornáček
Dátum:	14.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Blokovanie spojení medzi manažovateľnými prepínačmi
Opis testu:	Test spočíval vo vytvorení topológie s manažovateľnými prepínačmi a pripojenými hostami, ktoré sa medzi sebou vykonávali príkaz *ping*.
Požiadavka:	Príkaz *ping* medzi stanicami by mal byť úspešný po až po vytvorení STP stromu a mali by byť využité len linky v stave *forwarding*
Výsledok:	OK
Opravná činnosť:	Žiadna

Vykonávateľ:	Martin Hornáček
Dátum:	15.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Konvergencia STP stromu

Opis testu:	Test spočíval v postupnom pridávaní manažovateľných prepínačov a vytváraní slučiek.
Požiadavka:	Prepínače mali zablokovať vzniknuté linky a postupne vytvoriť nový STP strom
Výsledok:	OK
Opravná činnosť:	Žiadna

Vykonávateľ:	Martin Hornáček
Dátum:	15.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Príkaz *ping*
Opis testu:	Vytvorenie topológie a vykonávanie príkazu *ping*
Požiadavka:	Prijatie správnej odpovede.
Výsledok:	Niekedy sa stanice nedokázali „pingnúť“ aj keď boli spojené obyčajným prepínačom.
Opravná činnosť:	Chyba bola spôsobená v zlom algoritme generovania MAC adries, niekedy algoritmus vygeneroval rovnakú MAC adresu, ktorá už bola v topológii použitá. Teraz sa MAC adresa generuje postupným zvyšovaním čísla o jedna.

Vykonávateľ:	Milan Melicherčík
Dátum:	17.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Sniffer
Opis testu:	Vytvorenie topológie, zapnutie snifferu a pozeranie informácií o paketoch
Požiadavka:	Zobrazenie správnych informácií
Výsledok:	Sniffer zobrazoval informácie o nesprávnych paketoch
Opravná činnosť:	Chyba bola spôsobená neusporiadavaním paketov v okne snifferu podľa ich čísla, pričom sa číslo paketu bralo z „kliknutého“ riadku

Vykonávateľ:	Milan Melicherčík
Dátum:	17.6.2006
Operačný systém:	Windows XP, SP2
Modul:	Sniffer
Opis testu:	Vytvorenie topológie, zapnutie snifferu a pozeranie informácií o paketoch
Požiadavka:	Zobrazenie správnych informácií
Výsledok:	Sniffer zobrazoval informácie niekedy nezmyselné informácie
Opravná činnosť:	Spôsobené načítavaním informácie z dialógového okna, pričom nie vždy došlo k úplnému prevodu. Odstránené použitím inej funkcie na načítanie parametrov.

9 Použitá literatúra:

[1] Timovy projekt 2004/2005, tim SubNet

[2] Počítačová sieť
http://sk.wikipedia.org/wiki/Po%C4%8D%C3%ADta%C4%8Dov%C3%A1_sie%C5%A5

[3] Understanding Spanning-Tree Protocol
http://www.cisco.com/univercd/cc/td/doc/product/rtrmgmt/sw_ntman/cwsimain/cwsi2/cwsiug2/vlan2/stpapp.htm

[4] Routing Information Protocol
http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/rip.htm

[5] Interior Gateway Routing Protocol
http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/igrp.htm

[6] OSPF Design Guide
http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/ospf.htm

[7] The OSPF Specification
http://www.rfc.net/rfc1131.html

[8] Todd Lammle, Sean Odom, Kevin Wallace: CCNP: Routing Study Guide, SYBEX Inc., Alameda, CA, 2001

[9] An Introduction to IGRP
http://www.cisco.com/warp/public/103/5.html