ŽILINSKÁ UNIVERZITA V ŽILINE
Fakulta riadenia a informatiky

28360320102045

DIPLOMOVÁ PRÁCA

Študijný program: 9.2.6 Informačné systémy

Bc. Martin Majerský

Simulačný model siete Metro Ethernet
s MPLS chrbticovou sieťou v OPNET Modeler

Vedúci: Ing. Jana Uramová, PhD.
Stupeň kvalifikácie : Inžinier

Reg. č. 45/2009 Máj 2010

Žilina
2010

ABSTRAKT

MAJERSKÝ, Martin: *Simulačný model siete Metro Ethernet s MPLS chrbticovou sieťou v OPNET Modeler* [diplomová práca] – Žilinská univerzita v Žiline, Fakulta riadenia a informatiky; Katedra informačných sietí. – Vedúci: Ing. Jama Uramová, PhD. – Stupeň odbornej klasifikácie: Inžinier v programe Informačné systémy. Žilina: FRI ŽU v Žiline, 2010. – 69 s.

Diplomová práca sa zaoberá vytvorením simulačného modelu siete Metro Ethernet s MPLS chrbticovou sieťou v OPNET Modeler. Model vystihujúci podstatné vlastnosti technológií z pohľadu mechanizmov pre garanciu služby. Práca obsahuje informácie o modelovacom a simulačnom nástroji OPNET Modeler. Poskytuje postupy na vytvorenie simulačného modelu v tomto nástroji. Cieľom práce je zaznamenať vyhodnotiť štatistiky o oneskoreniach a stratách v sieti vzhľadom na Triple-play služby (hlas, video, internet).

Kľúčové slová: Ethernet, MPLS, kvalita služby, oneskorenie, straty paketov.

ABSTRACT

MAJERSKÝ, Martin: Simulation model of Metro Ethernet with MPLS core network in OPNET Modeler. Department of InfoComm networks. Instructor: Ing. Jana Uramová, PhD. –Qualification level: Engineer in field Information systems. Žilina: FRI – ŽU in Žilina, 2010. – 69 p.

The diploma thesis deals with the creation of a network simulation model of Metro Ethernet with MPLS backbone in OPNET Modeler. The model describes the essential characteristics of technology in terms of quality of service mechanisms. The work contains information on modeling and simulation tool OPNET Modeler. Provides procedures to create simulation model in this tool. The aim of this thesis is to record and evaluate statistics on delays and losses in the network view of the triple-play services (voice, video, internet).

Key words: Ethernet, MPLS, quality of service, delay, paket loss.

Čestné vyhlásenie

Vyhlasujem, že som celú diplomovú prácu vypracoval samostatne s použitím uvedených zdrojov.

V Žiline, 14. Mája 2010

.....

vlastnoručný podpis

OBSAH

Zoznam ilustrácií a tabuliek

Zoznam skratiek

CFI – Canonical Format Indicator

CQ – Custom Queuing

CSMA/CD – Carrier Sense with Multiple Access and Collision Detection

DSCP – Differentiated Services Code Point

FEC – Forwarding Equivalce Class

FIFO – Firts In First Out

FQ – Fair Queuing

FTP – File Transfer Protocol

GSM – Global System for Mobile communication

HD – High Definition

HTTP – Hypertext Tranfer Protocol

IEEE – Institute of Electrical and Electronics Engineers

IP – Internet Protocol

IPTV – Internet Protocol Telivision

ISDN – Integrated Services Digital Network

ITU – International Telecommunication Union

JVT – Join Video Team

LAN – Local Area Network

LER – Label Edge Router

LSP – Label Switched Path

LSR – Label Switch Router

MAC – Media Access Control

MEN – Metro Ethernet

MPEG – Moving Picture Experts Group

NGN – Next Generation Network

PCM – Pulse Code Modulation

PCP – Priority Code Point - kód priority

PHB – Per Hop Behavior

PQ – Priority Queuing

PSTN – Public switched telephone network

QoS – Quality of Service

RED – Random Early Detection

RSVP – Resource ReSerVation Protocol

RTP – Real-time Transport Protocol

TE – Traffic Engineering

TPID – Tag Protocol Identifier

TTL – Time To Live

UDP – User Datagram Protocol

VID – VLAN Identifier

VLAN - Virtual Local Area Network

VoIP – Voice over Internet Protocol

WFQ – Weighted Fair Queuing

WRED – Weighted Random Early Detection

Úvod

Komunikačné siete prešli za posledné roky značnými zmenami. Upustilo sa od modelu, kde pre každý typ služby sa budovala samostatná sieť so špecifikovanými parametrami presne pre danú službu. Tieto siete mali výhodu v tom, že služba poskytovaná po tejto sieti spĺňala kvalitu požadovanú od zákazníkov. Ak však bola požiadavka na novú službu, bolo potrebné vybudovať novú sieť. Pretože ak aj bolo možné prevádzkovať novú službu po niektorej z existujúcich sietí, služba nebola poskytovaná v požadovanej kvalite. Súčasný trend vo vývoji komunikačných sietí je konvergencia existujúcich sietí do jednej univerzálnej siete, ktorá by zabezpečila poskytovanie rôznorodých služieb s garanciou požadovanej kvality. Základom tejto siete sa stali počítačové paketové siete. Avšak ich hlavnou úlohou bolo prenášanie dát, bez nejakého zabezpečenia kvality. Jediná garancia spočívala v prístupe do siete bez odmietnutia. Sieť nerozlišovala medzi tokmi, ktoré prenášala. Ku všetkým sa správala rovnako. Keďže nie všetky služby majú rovnaké požiadavky na sieť, bolo potrebné tieto dátové toky jednotlivých služieb od seba odlíšiť a poskytnúť im požadovaný výkon siete. Preto sa vyvíjali mechanizmy ako sú klasifikácia tokov, následne ich prioritizácia, systémy riadenia frontov a pod., slúžiace na zabezpečenie týchto požiadaviek.

Pri návrhu takýchto rozsiahlych a zložitých sietí, ich funkcionality a vlastností, často siahajú návrhári takýchto sietí po pomerne mladom vednom obore zaoberajúcom sa modelovaním a simuláciou. Vytvorenie modelu siete a zistenie správania sa tohto modelu simuláciou, umožňuje odstrániť nedostatky a zaviesť vylepšenia ešte pred fyzickým vybudovaním danej siete. To mnohokrát ušetrí čas a hlavne peniaze.

Medzi takéto siete môžeme zaradiť aj metropolitné siete pokrývajúce mestské časti alebo celé mestá, v ktorých provideri poskytujú tzv. TriplePlay služby. Tieto zahŕňajú dátové služby, hlasové služby a televíziu.

1 Základné pojmy v modelovaní a simulácii

Modelovanie a simulácia sa venuje štúdiu skúmaných objektov, pričom tieto objekty buď už v realite existujú (výrobný podnik, nemocnica, burza, železničná stanica, organizmus živočícha, informačný systém a pod.) alebo by exitovať mohli (továreň po rekonštrukcii, novo projektovaný terminál kontajnerovej dopravy, chorobou zasiahnutý organizmus atď.) (Kavička, a iní, 2005).

Tieto skúmané objekty (tak existujúce, ako aj projektované alebo myslené) nemožno v ich úplnej zložitosti popísať, resp. racionálne pochopiť a zvládnuť. Preto sú na nich zavádzané abstrakcie, ktoré zanedbávajú niektoré aspekty týchto objektov, ktoré nie sú z pohľadu konkrétneho typu skúmania dôležité. Nezanedbané aspekty sa vyberajú tak, aby boli príslušným vedeckým, technickým či spoločenským odporom zvládnuteľné. Uvedené abstrakcie sa v modelovaní a simulácií nazývajú systémy (Kavička, a iní, 2005)

Rôzne druhy štúdia jedného daného objektu budú viesť k vytvoreniu odlišných systémov. Napríklad uzly komunikačnej siete ako sú smerovač, prepínač alebo server. Odborníci môžu študovať tieto zariadenia z pohľadu ich špecializácie. Iný systém vymedzí odborník zaoberajúci sa hardwarom ako odborník špecializujúci sa na prevádzku v sieti.

Okolím skúmaného objektu nazývame tie objekty reálneho sveta, ktoré neboli vybraté na skúmanie, ale napriek tomu potrebujeme uvažovať ich existenciu a vlastnosti kvôli ich vzťahom so skúmaným objektom. Abstrakciu okolia skúmaného objektu nazývame potom okolím systému (Kavička, a iní, 2005).

Ak je skúmaným objektom smerovač, okolie môžu tvoriť napríklad iné smerovače, klienti alebo servery pripojené k tomuto smerovaču. Tieto s ním komunikujú a ovplyvňujú tým dátový tok prechádzajúci týmto smerovačom.

Pri vymedzovaní systému na objekte skúmania sa môže, ale nemusí zanedbať význam času. Systém, ktorý od významu času abstrahuje, sa nazýva statickým systémom. Naproti tomu systém, ktorého časovú stránku nezanedbávame sa v odbore modelovania a simulácie nazýva dynamickým systémom. (Kavička, a iní, 2005)

Množina okamihov, v ktorých dynamický systém existuje, sa nazýva (časová)

existenciou dynamického systému, pričom táto existencia je daná abstrakciou a môže ju predstavovať akákoľvek podmnožina reálnych čísel (Kavička, a iní, 2005).

Skúmanie vymedzeného systému na objekte smerovača, ktoré je zamerané na proces spracovania paketov, môže definovať existenciu, ktorá zahŕňa iba okamihy, v ktorých dochádza k začiatku, respektíve ukončeniu činností zahrnutých do systému: prijatie paketu, zaradenie paketu do fronty, vybratie paketu z fronty, odoslanie paketu a pod.

Systém je zložený z prvkov (entít, ktoré predstavujú určité fyzické alebo logické elementy objektu skúmania. Rozlišujeme permanentné a temporárne entity, pričom prvé z nich sú v dynamickom systéme počas celej jeho existencie, zatiaľ čo druhé nie. Exogénne prvky vznikajú mimo systému (v jeho okolí) a endogéne vznikajú v samotnom systéme. Analogicky, temporárne prvky môžu zaniknúť tým, že sú presunuté do okolia systému (a tým pre systém prestanú existovať) alebo zaniknú priamo v systéme.

Za permanentné prvky v smerovači môžeme považovať procesor, porty, pamäť. Príkladom temporárnych prvkov sú pakety, ktoré sú taktiež exogénnymi prvkami. Rozlišujeme tiež stabilné prvky a mobilne prvky, ktoré majú schopnosť sa premiestňovať alebo byť premiestňované (Kavička, a iní, 2005).

V komunikačných sieťach je typickým mobilným prvkom paket. Do mobilných prvkom môžeme tiež zaradiť mobilné typy prijímačov alebo vysielačov. Medzi stabilné prvky v sieťach môžeme zaradiť ostatné statické uzly v komunikačnej sieti.

Prvky systému majú svoje vlastnosti (atribúty), ktorých hodnoty sa môžu v prípade prvkov dynamického systému v čase meniť. Atribúty môžeme ďalej členiť na štandardné a referenčné. Stav dynamického systému v čase t je určený prvkami, ktoré sú v čase t v systéme prítomne a hodnotami ich atribútov v tomto čase (Kavička, a iní, 2005).

Medzi štandardné atribúty môžeme zaradiť napríklad veľkosť paketu v bajtoch, rýchlosť linky v bitoch za sekundu, adresu odosielateľa a príjemcu paketu a pod.

Podstatou modelovania v zmysle výskumnej techniky/metódy je náhrada skúmaného systému (originálu) jeho modelujúcim systémom (modelom), s cieľom získať pomocou pokusov s modelom informácie o modeli.

Simulácia je výskumná technika/metóda, ktorej podstatou je náhrada skúmaného dynamického systému (originálu) jeho simulujúcim systémom (stimulačným modelom), s ktorým sa experimentuje s cieľom získať informácie o pôvodnom skúmanom dynamickom systéme (Kavička, a iní, 2005).

2 Model prevádzky siete MEN

V minulosti bolo príznačné pre komunikačné siete, že pre každý typ služby (alebo skupinu služieb podobného typu) sa budovali špeciálne siete. Tieto siete boli navrhované s parametrami potrebnými na zabezpečenie dostatočnej kvality tej ktorej služby. V súčasnosti vývoj komunikačných sietí smeruje k vybudovaniu jednej univerzálnej siete pre rôzne druhy poskytovaných služieb. Týmto integrovaným sieťam tiež hovoríme „siete novej generácie“ z anglického Next Generation Networking (NGN) (Jančo, 2008).

Pri zaradení novej služby do portfólia poskytovaných, nie je potrebné vybudovať novú sieť, ale stačí implementovať danú službu do NGN siete. Tu ale vzniká problém. Každá z implementovaných služieb v sieti má iné požiadavky na prenosové parametre. Preto je obtiažne zabezpečiť garantovanú kvalitu služby.

Keďže siete novej generácie sú budované na základe súčasných paketových sietí, je potrebné odstrániť nedostatky, ktoré v týchto sieťach existujú. Boli budované s cieľom poskytovať prenosovú službu, bez zabezpečenia parametrov spojenia, ako napríklad doručenie paketov do určitej doby, prípadne zaistenie dopredu definovanej šírky pásma počas doby trvania prenosu, a pod.

Kvalita služby (Zeman, 2006) je definovaná ako výsledok výkonnosti služby, ktorý určuje stupeň spokojnosti používateľa. Kvalita služby v IP sieťach charakterizuje výkonnosť toku paketov jednou alebo viacerými sieťami. Snahou je doručiť pakety medzi koncovými uzlami podľa určených kritérií. Základné pravidlá pre prevádzku v Internete definované už v 70. rokoch sú:

- žiadnemu paketu nebude odoprený prístup do siete,
- so všetkými paketmi bude zaobchádzané rovnako,
- jediná garancia prenosu paketu je, že bude prenesený čo najlepším spôsobom (Best Effort) v závislosti na dostupných prostriedkoch siete, tzn. že nebude zbytočne a umelo dochádzať k oneskoreniu alebo stratám paketov.

Z týchto dôvodov sa do súčasných integrovaných komunikačných sietí implementujú mechanizmy zabezpečujúce kvalitu pre daný typ služby.

2.1 Typy prevádzky v komunikačných sieťach

V závislosti od charakteristiky dátového toku sa líšia aj požadované parametre siete, vďaka ktorým bude zabezpečená požadovaná kvalita služby. Pre rozdelenie sieťovej prevádzky sa najčastejšie stretneme s delením do troch základných skupín (Jančo, 2008):

- **Real-Time** – do tejto triedy sú zahrnuté typy interaktívnych služieb, kde sú aktívne obe strany: IP telefónia (VoIP), video konferenčné hovory a pod. Pri týchto typoch služieb je požadovaná veľmi nízka stratovosť paketov (typicky do 0,5%), malé oneskorenie (150 ms) a kolísanie oneskorenia (jitter, okolo 10 ms). Ďalšími typmi služieb, kde už je však jedna strana pasívna sú napríklad digitálna televízia (IPTV) alebo streamované rozhlasové vysielanie. Pri týchto službách sú nároky na oneskorenie a jitter nižšie a aj vyššie oneskorenie nespôsobuje zníženie kvality služby.

- **Bussiness** – táto trieda zahŕňa rôzne interaktívne aplikácie: rôzne firemné aplikácie, telnet, vzdialená práca s databázami, webové aplikácie a pod. Veľké straty paketov alebo oneskorenia v doručení vplývajú na kvalitu a efektívnosť práce s touto aplikáciou. Požadované oneskorenie je na úrovni do 250 ms, strata paketov do 1%. Kolísanie oneskorenia nemá na tento typ služby vplyv.

- **Best Effort** – do tejto kategórie spadá zvyšná sieťová prevádzka nezahrnutá v predchádzajúcich dvoch. Táto trieda je definovaná iba mierov strát paketov, ktorá je určená dostupnou šírkou pásma. Oneskorenie a jitter na kvalitu tohto typu služby nemá vplyv a je povolené zdržanie paketov v jednotlivých uzloch siete. Náklady na prevádzku týchto služieb sú najnižšie.

2.2 Hlasová prevádzka

Hlasová prevádzka je tvorená dátovým tokom, ktorý vznikol odoberaním hlasových vzoriek s frekvenciou 8000 za sekundu. Tieto vzorky sú, na základe odporúčania ITU-T G.711, kvantované do 8 bitového slova, čo vytvára bitový tok

s intenzitou 64 kbit/s. Po pripočítaní veľkosti hlavičky paketu RTP/UDP/IP protokolu sa hodnota toku zväčšuje na 80 kbit/s. Ak sa do jednotlivých paketov balí 10 alebo 20ms reči vzniká prúd s intenzitou 50 alebo 100 paketov za sekundu.

Ak by sme uvažovali o využití kompresie podľa odporúčania ITU-T G.729.A s potlačením ticha, viedlo by to k zníženiu intenzity toku zo 64 kbit/s na 8 kbit/s. Pri zachovaní veľkosti hlasového rámca vkladaného do paketu, sa telo paketu zmenší zo 160 bajtov na 20bajtov. Po pripočítaní hlavičiek prenosových protokolov je výsledná hodnota toku 20 kbit/s. Treba však podotknúť, že za zníženie intenzity toku platíme zhoršením kvality telefónnej služby. Preto operátori prechádzajú na kódovanie hlasovej prevádzky podľa odporúčania G.711 bez kompresie a potlačenia ticha.

2.3 Video prevádzka

Veľkosť dátového toku prenášaného videa je závislá od rozlíšenia zdrojového signálu a kódovacieho algoritmu použitého na spracovanie video signálu.

Nový štandard pre kódovanie videa, známy ako odporúčanie H.264 od ITU-T je posledným štandardom kódovania videa v poradí. H.261 (1990), MPEG-1 Video (1993), MPEG-2 Video (1994), H.263 (1995, 1997), MPEG-4 Visual or part 2 (1998). Tieto predchádzajúce štandardy reflektovali technologický vývoj v kompresii videa a adaptácií kódovania videa v rôznych aplikáciách a sieťach. Aplikácie od video telefónie (H.261), cez spotrebiteľské video na CD (MPEG-1) až po vysielanie štandardnej alebo HD televízie (MPEG-2). Siete používané pre video komunikáciu, vrátane prepínaných sietí ako PSTN (H.263, MPEG-4) alebo ISDN (H.262); paketové siete ako ATM (MPEG-2, MPEG-4), Internet (H.263, MPEG-4) alebo mobilné siete (H.263, MPEG-4). H.264/AVC prináša najväčší algoritmický skok v celej evolúcii štandardizovaných video kodekov. Tento progres bol možný vďaka video expertom z ITU-T a MPEG, ktorý vytvorili Joint Video Team (JVT) v Decembri 2001, kvôli vývoju nového H.264/AVC štandardu. H.264/AVC bol sfinalizovaný v marci 2003, keď bol aj schválený ITU-T.

2.4 Dátová prevádzka

Dátová prevádzka má väčšinou elastický charakter. Relácie sú charakterizované predovšetkým strednou dĺžkou súboru. Z hľadiska kvality služby je potrebné určiť stredný čas potrebný na doručenie súboru. Tento čas sa začína počítať od okamihu vyslania prvého bitu vysielačom a končí prijatím posledného bitu prijímačom. Štúdium tejto prevádzky ukázalo, že intervaly medzi paketmi majú subexponenciálne rozdelenie s „dlhým chvlostom“ (napr. Weibullovo rozdelenie).

3 Parametre hodnotenia kvality služby

Základnými atribútmi (Jančo, 2008), podľa ktorých môžeme hodnotiť kvalitu služby sú:

- **Paket loss** (strata paketov)
- **Delay** (oneskorenie)
- **Jitter** (kolísanie oneskorenia)

3.1 Paket loss

K strate paketov v sieti dochádza z viacerých príčin. Tými menej častými sú hardwarové chyby v sieťových uzloch alebo rušenie na prenosovom médii. Častejšie sa môžeme stretnúť so stratou paketov zapríčinenou stavom v sieti. Teda zahltením sieťového uzla, kedy smerovače alebo prepínače nestíhajú odbavovať prichádzajúce dostatočne rýchlo. Fronty vo vyrovnávacích pamätiach sa preplnia a ďalšie prichádzajúce pakety sú zahodené. Za stratený sa tiež považuje paket, ktorý je doručený s väčším oneskorením ako je požadované a jeho obsah už je pre danú aplikáciu/službu neaktuálny.

Rôzne druhy služieb a aplikácii sa s týmto nedostatkom vysporiadavajú odlišne. Real-time služby, ktoré používajú transportné protokoly bez potvrdenia doručenia, túto stratu jednoducho ignorujú alebo majú mechanizmy na nahradenie dát zo strateného paketu. Služby, využívajúce na prenos dát protokoly s potvrdením o doručení, pri strate paketu požiadajú odosielaciu stranu o opätovné zaslanie paketu.

3.2 Delay

Oneskorenie doručenia paketu je počítané od okamihu odoslania posledného bitu paketu odosielateľom po okamih prijatia posledného bitu príjemcom. Toto oneskorenie je tvorené z viacerých dielčích oneskorení:

- Oneskorenie pri prenose (odvodené od rýchlosti šírenia signálu a vzdialenosti)
- Oneskorenie kódovaním a serializáciou (príprava paketov na prenos)

- Oneskorenie spôsobené čakaním no fronte sieťového uzla
- Oneskorenie pri prepínaní (vyhľadanie ďalšej cesty v sieti)

3.3 Jitter

Kolísanie oneskorenia doručenia paketov je spôsobené zmenou stavu v sieti. Oneskorenie môže narásť vďaka väčšiemu počtu paketov vo vyrovnávacej pamati. Taktiež odlišná cesta prenosu po sieti môže zapríčiniť dĺžku oneskorenia.

Jitter má veľký dopad napríklad na službu VoIP. Pri odosielaní paketov v 20 ms intervaloch sa na druhej strane taktiež očakávajú príchody v daných intervaloch. To však nemusí byť vždy dodržané kvôli dynamickým zmenám zaťaženia v sieti. Oneskorený paket by narušil časový sled paketov, takže spracovanie takéhoto paketu sa nekoná a paket sa považuje za stratený. Tento problém je riešený vyrovnávacími pamäťami v telefónoch alebo bránach.

4 Modely zabezpečenia kvality služby

Poznáme dva modely (Zeman, 2006) pre poskytovanie rôznych úrovní sieťových služieb v sieťach schopných prenášať dáta, hlas aj video.

- Integrated service – IntServ
- Differentiated service – DiffServ

4.1 Integrated service

Model integrovaných služieb sa snaží dodržať garantovanú úroveň obsluhy pre vybrané služby po celej trase prenosu. Aplikácia vyžaduje od siete, aby zabezpečila určitú úroveň parametrov prenosu, ktoré potrebuje na to aby pracovala správne. Aplikácia musí vedieť aká je charakteristika toku, ktorý generuje a signalizovať uzlom v sieti na celej trase prenosu. Prenos začne až vtedy, keď mechanizmus IntServ bol schopný zarezervovať potrebné sieťové prostriedky. Vyžiadanie si sieťových prostriedkov aplikáciou môže IntServ poskytnúť za pomoci protokolu RSVP (Resource ReSerVation Protocol). RSVP nie je smerovací protokol. Pri hľadaní najlepšej trasy spolupracuje so smerovacími protokolmi.

Model IntServ podporuje dva rozdielne typy služieb:

Služba s riadenou záťažou – pre zaistenie spoľahlivého prenosu medzi dvoma bodmi riadením záťaže.

Garantovaná služba – pre garantované maximálne oneskorenie pri prenose v danom pásme.

4.2 Differentiated service

DiffServ predstavuje metódu, ktorá pomocou množiny klasifikačných nástrojov a mechanizmov pre prácu s frontami poskytuje široké možnosti pri zabezpečovaní kvality služby dátovým prenosom v sieti. Táto metóda je založená na tom, že okrajové smerovače klasifikujú rôzne druhy paketov prechádzajúcich cez sieť. Rôzne toky sa dajú klasifikovať na základe sieťovej adresy, protokolu, vstupného portu. Tok je tiež možné klasifikovať pomocou základných alebo

rozšírených access listov. Potom sa zaraďujú do tried. Každej triede sa priradí DiffServ hodnota (DSCP). V jadre siete sú potom pakety preposielané na základe nadefinovaného správania sa sieťových uzlov k toku s príslušnou hodnotou DSCP.

Keďže žiadna z metód (intServ a DiffServ) neponúka komplexné riešenie, využíva sa kombinovanie týchto dvoch metód. Najnovší trend v zabezpečovaní kvality služby je zjednodušenie a automatizácia s ohľadom na jednoduchosť a schopnosť zabezpečiť garanciu kvality služby v IP sieťach. Technológie zabezpečujúce kvalitu služby ponúkajú množstvo možností a môžu byť využité pri výstavbe veľmi dômyselných sietí.

5 Mechanizmy zabezpečenia kvality služby

Pri zabezpečovaní kvality obsluhy dát sú používané rozličné nástroje ponúkané zariadeniami, ktoré sa podieľajú na riadení toku týchto dát cez sieť. V tejto kapitole si popíšeme kategórie, do ktorých spadajú:

- Classification and marking (klasifikačné a značkovacie nástroje)
- Policing and shaping (nástroje pre dozor a obmedzovanie)
- Congestion Management (manažment presýtenia siete)
- Congestion Avoidance (ochrana pred zahltením)

5.1 Classification and marking

Prvou operáciou pri definovaní QoS politiky je identifikácia dátového toku, ktorá je zabezpečená pomocou klasifikácie a značkovania.

Klasifikácia (Classification) – klasifikačné mechanizmi identifikujú pakety. Na základe tejto identifikácie môže dôjsť k ďalším činnostiam ako sú značkovanie, zaradovanie do frontov, policing, shaping a pod. HĺbkovÚ klasifikáciu paketov nie je potrebná vykonávať v každom uzle siete. Nástrojom pre klasifikáciu je takzvaný klasifikátor. Ten kontroluje určité polia paketu kvôli identifikácii typu dátového toku, ku ktorému patrí.

Značkovanie (Marking) – aplikovanie značiek, na základe ktorých závisí správanie sa rozhodovacích nástrojov. Nástroje používané na značkovanie sa nazývajú značkovače. Zapisujú polia do vnútra paketu, rámca, alebo návestia pre uchovanie informácie o výsledkoch analýzi pri klasifikácii.

5.2 Policing and shaping

Dohľad nad tokom a tvarovanie toku patria medzi najstaršie formy QoS mechanizmov. Oba tieto mechanizmy majú podobnú úlohu, ktorou je identifikovať a odpovedať na priestupky toku dát. Zvyčajne priestupky toku dát identifikujú rovnako, avšak rozdiel sa s touto situáciou vysporiadavajú.

Policers pri detekcii množstva dát, ktoré prevyšuje preddefinovanú šírku pásma, tieto „dáta mimo dohody“ zahodia.

Shapers sú nástroje, ktoré pri svojej činnosti využívajú mechanizmy frontov. Pri presiahnutí dohodnutej hranice sa dáta nezahadzujú, ale sú pozdržané vo vyrovnávacej pamäti. Po uvoľnení kapacít linky pre daný tok sú tieto dáta odoslané. Hovoríme tomu tiež vyhladenie dátového toku.

5.3 Congestion Management

Ak nastane zahltenie na niektorom z rozhraní uzla, sú aktivované nástroje zaraďovania do frontov. Aby nedošlo pri zahltení k strate paketov, sú tieto uložené do frontu vo vyrovnávacej pamäti.

Queuing (radenie do frontu) je logické zaraďovanie paketov vo vyrovnávacej pamati.

Sheduling (planovanie) je proces pri rozhodovaní ktorý paket má byť odoslaný ako ďalší. Pre určenie toho čo má byť odoslané najskôr slúžia viaceré typy plánovacích algoritmov.

5.4 Congestion Avoidance

Tento mechanizmus je navrhnutý pre prevenciu pred zahltením rozhrania alebo frontu.

Tail drop – jedná sa o najjednoduchšie zahadzovanie paketov. Majme front plniaci sa FIFO princípom. Keď sa front zaplní, ďalší prichádzajúci paket je zahodený. Prichádzajúce pakety sú zahadzované, až kým sa vo fronte neuvoľní miesto a paket môže byť zaradený do frontu.

Weighted Random Early Detection (WRED) – táto technika priebežne sleduje zaplnenie frontu. Ak sa blíži ku stavu zahltenia, začne selektívne zahadzovať pakety. Pri tomto zahadzovaní rešpektuje nastavené hodnoty, čím je zabezpečené, že pakety s vyššou prioritou sú zahadzované neskôr.

6 Systémy riadenia frontov

V tejto kapitole sa oboznámime so základnými vlastnosťami systémov riadenia frontov. Podľa (Jančo, 2008).

6.1 First-In First-Out (FIFO)

FIFO patrí medzi najjednoduchšie algoritmy riadenia frontov. Zahŕňa skladovanie a posielanie paketov na základe poradia ich príchodu. Tento algoritmus je výpočtovo nenáročný. Neumožňuje však zohľadňovanie priority paketov. Nezabezpečuje ochranu proti aplikáciam zahlcujúcim sieť. Náhle špičkové zaťaženia môžu spôsobiť veľké oneskorenia pri doručovaní časovo citlivých dát. FIFO bol jedným z prvých krokov v kontrolovaní sieťovej prevádzky, ale dnešné siete vyžadujú sofistikovanejšie algoritmy. Nežiadúcim efektom je zahadzovanie paketov, zapríčinené preplneným frontom. Zahodené pakety môžu byť pakety s vysokou prioritou.

6.2 Priority Queuing (PQ)

Algoritmus prioritného frontovania zabezpečuje čo najrýchlejšie spracovanie dôležitých dát. Poskytuje striktné uprednostňovanie dôležitej prevádzky na základe protokolu, zrojovej/cieľovej adresy, vstupného rozhrania atď. Každý prichádzajúci paket je vložený do jedného z frontov podľa príslušnej priority. Počas spracovania dáva algoritmus frontom s vyššou prioritou absolútnu prednosť pred frontami s nižšou prioritou. Pre každý front je možné nastaviť veľkosť – počet paketov, ktoré môže daný front obsahovať. Technika PQ je určená pre zaistenie správneho spracovania dôležitých dátových tokov prechádzajúcich cez viaceré odlišné linky.

6.3 Custom Queuing (CQ)

Vlastné frontovanie je navrhnuté tak, aby umožňovalo rôznym typom aplikácii alebo skupinám podobných aplikácii s určitými požiadavkami na minimálnu šírku pásma zdieľať sieť. Toto prostredie vyžaduje proporcionálne

delenie šírky pásma medzi aplikácie a používateľov. Každému druhu prevádzky je priradená určitá dĺžka frontu. Tieto sú následne cyklicky obsluhované. Plánovač obsluhuje fronty, vyberajúc predkonfigurovaný počet bytov pre danú frontu v každom cykle. Podobne ako PQ, je aj CQ staticky nakonfigurované a neprispôsobuje sa automaticky požiadavkám siete.

6.4 Fair Queueing (FQ)

Algoritmus férovej obsluhy zaraďuje prichádzajúce pakety do frontov podľa typu. Tieto fronty sú potom mechanizmom „round robin“ cyklicky obslukované. Z každého frontu sa vyberie vždy jeden paket určený na odoslanie. Tým je výstupná linka rozdelená medzi fronty kvázi rovnomerne, v závislosti na veľkosti paketov. Tento algoritmus neumožňuje uprednostnenie niektorého z tokov.

6.5 Weighted fair queueing (WFQ)

Algoritmus váženej férovej obsluhy je rozšírením FQ. Pridanie váhy spočíva v tom, že každému frontu je vyhradená iná šírka pásma výstupnej linky. V každom cykle je vyberaný počet bitov pomerne k pridelenej veľkosti výstupnej linky. Ak algoritmus zistí, že bol vybraný posledný bit paketu, paket je odoslaný. Táto implementácia je pomerne náročná, preto sa pristupuje k takým, kde sa pakety nerozkladajú po bitoch. Pri zaraďovaní paketu do frontu je vypočítaný čas kedy by mal paket opustiť front. Plánovač potom vyberá na odoslanie pakety s najmenším týmto časom.

6.6 Leaky Bucket

Algoritmus „Leaky Bucket“ (deravé vedro) slúži na tvarovanie toku. Charakterizujú ho dve hodnoty. Veľkosť frontu vyrovnávacej pamäte a rýchlosť výstupnej linky. Jeho správanie sa dá prirovnať k deravému vedru. Ak na port príde väčšie množstvo dát ako je linka schopná odoslať, sú tieto dáta pozdržané a odoslané neskôr, keď sú na to prostriedky. Dáta z frontu sú odosielané

konštantnou rýchlosťou bez ohľadu na ich množstvo. Tieto vlastnosti umožňujú lepšie využiť kapacity linky a regulovať veľkosť výstupného toku.

6.7 Token bucket

Token Bucket (vedro značiek) je metóda používaná pre dohľad nad tokom. Parametre charakterizujúce túto metódu sú: maximálne množstvo tokenov (značiek) a počet tokenov, ktoré pribudnú za jednotku času. Aby paket prichádzajúceho toku bol poslaný ďalej je potrebné aby mu bol pridelený token (alebo niekoľko tokenov). Ak nie je k dispozícii dostatočný počet tokenov, paket je zahodený. Tokeny pribúdajú konštantnou rýchlosťou ale iba do maximálneho určeného množstva.

7 Sieťové technológie

7.1 Ethernet

Ethernet (Jančo, 2008) bol vyvíjaný spoločnosťou XEROX v sedemdesiatych rokoch 20. storočia. V roku 1980 spoločnosť IEEE štandardizovala Ethernet a pozname ho pod označením IEEE 802.3. Je založený na myšlienke komunikácie počítačov v sieti, ktorá sa podobá na rádiovú komunikáciu. Počítače komunikujú na spoločnom médiu, v tomto prípade spoločné metalické vedenie.

Prvé Ethernet siete mali zbernicovú topológiu a ako prenosové médium používali koaxiálny kábel. V súčasnosti koaxiálny kábel nahradila krútená dvojlinka, umožňujúca duplexný prenos. Najčastejšie používaná topológia je hviezda. Každá stanica v sieti je identifikovaná jedinečnou hardwarovou 48 bitovou adresou MAC (Media Access Control).

Prístup k prenosovému médiu je riadený metódou CSMA/CD (Carrier Sense with Multiple Access and Collision Detection – mnohonásobný prístup s detekciou kolízií). Každá stanica počúva na médiu. Ak je médium voľné, stanica môže začať vysielať. Vďaka oneskoreniu spôsobeného šírením signálu sa môže stať, že voľné médiu detekujú viaceré stanice a dôjde ku kolízii. Stanica, ktorá počas vysielania zistí kolíziu, preruší vysielanie a odošle špeciálnu správu – jam. Táto správa vyvolá indikáciu kolízie aj u ostatných staníc. Tie rovnako prerušia vysielanie a odmlčia sa na náhodne dlhý čas. Po tomto čase sa opäť pokúsia o komunikáciu.

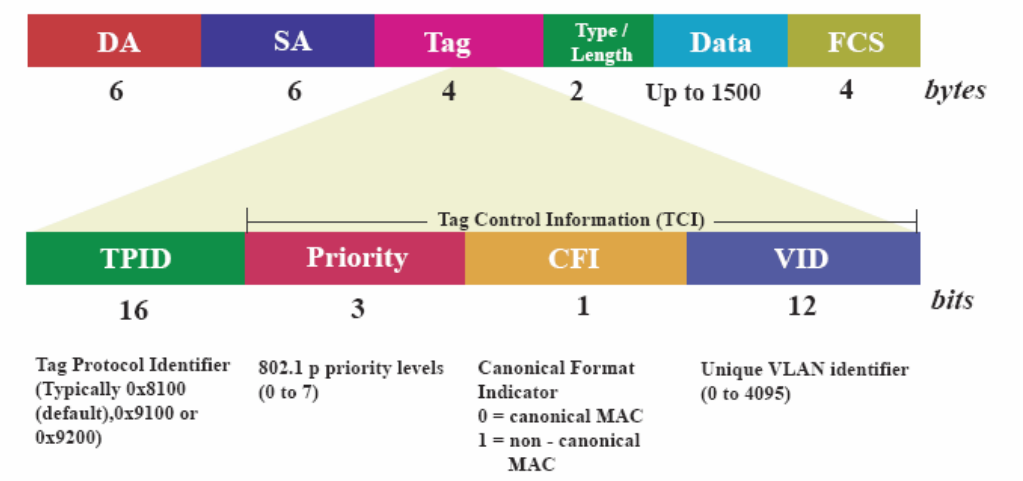
Vývojové špecifikácie štandardu Ethernet z pohľadu dosahovaných rýchlostí:

- **Ethernet** (IEEE 802.3) - rýchlosť 10Mb/s; v súčasnosti sa používa iba na zariadeniach, ktoré nepodporujú FastEthernet
- **Fast Ethernet** (IEEE 802.3u) – rýchlosť 100Mb/s; najrozšírenejší, používaný na pripájanie koncových zariadení
- **Gigabit Ethernet** (IEEE 802.3z) – rýchlosť 1Gb/s; kompatibilný s predchádzajúcou verziou

- **10 Gigabit Ethernet** (IEEE 802.3ae) – rýchlosť 10Gbit/s; podporuje iba full duplex linky, rôzne špecifikácie pre medené vodiče a optické vlákna

7.2 802.1Q (VLAN)

Sieťový štandard 802.1Q alebo tiež Virtuálne LAN siete (Jančo, 2008) bol spracovaný pracovnou skupinou IEEE 802.1. Umožňuje rozdelenie veľkých LAN s množstvom uzlov na menšie fragmenty požadovanej veľkosti. Taktiež je možné z rôznych segmentov viacerých LAN vytvoriť jednu logickú sieť. Správnym rozvrhnutím týchto logických podsietí sa dosiahne vyššej bezpečnosti v sieti. Vytvorením virtuálnych sietí sa tiež celá sieť rozdelí do viacerých broadcastových domén. Pri väčšom počte staníc v sieti môže broadcast výrazne zaťažiť sieť a znížiť jej výkonnosť. Keďže prepojenie uzlov v jednotlivých virtuálnych sieťach je iba logické, sú veľmi flexibilné a ľahko manažovateľné. Každá VLAN má svoje unikátne označenie, ktoré je reprezentované 12 bitovým číslo. To predstavuje 4094 možných virtuálnych sietí.



Obrázok 1. Zobrazenie TCI ethernet rámca.

Do ethernetového rámca sa vkladá 32 bitový blok. Je umiestnený medzi poľom *Adresa odosielateľa* a poľom *Typ/Veľkosť*. Tento blok obsahuje nasledujúce záznamy:

- **TPID** – Tag Protocol Identifier (značka identifikátora protokolu) – 16 bitový záznam. Pre identifikovanie rámca ako IEEE 802.1Q je nastavená hodnota

0x8100. Toto pole sa nachádza na pozícii bloku *Typ/Veľkosť*, takže je možné určiť či sa jedná jedná o obyčajný rámec alebo rámec s VLAN.

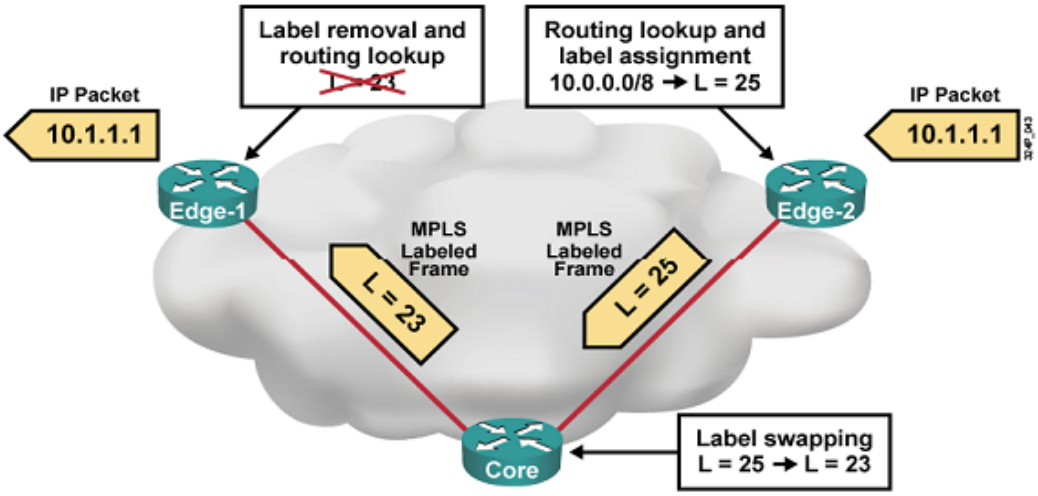
- **PCP** – Priority Code Point (kód priority) – 3 bitový záznam obsahujúci prioritu podľa IEEE 802.1p. Označuje prioritu rámca na úrovni 0 (najnižšia) až 7 (najvyššia), ktorá môže byť použitá na stanovenie priority rôznych tied prevádzky (dáta, hlas, video)
- **CFI –** Canonical Format Indicator – 1 bitový záznam. Označuje či je MAC adresa v kánonickom tvare. Je nastavená vždy na 0. Používa sa iba pre kompatibilitu medzi ethernetovými a token ring sieťami.
- **VID –** VLAN Identifier – 12 bitový záznam. Určuje pre akú VLAN je daný rámec určený. Hodnota 0 značí, že rámec nepatrí do žiadnej VLAN. Hodnota 0xFFF je rezervovaná.

7.3 Multiprotocol Label Swtching

V čase, keď MPLS vznikalo bola väčšina smerovačov softwarová. Smerovanie v IP sieťach je výpočtovo náročné. Vyhľadávanie predčíslia siete neznámej dĺžky. Na softwárových smerovačoch je procesor neraz úzkym hrdlom. Technológia MPLS bola teda vyvíjaná ako cesta k urýchleniu smerovania, ktoré by nebolo výpočtovo náročné. V súčastnosti sa však vynára využitie, ktoré ponúka nové možnosti pre širokú škálu IP sietí. Ak je MPLS použité spolu s diferencovanými službami (DiffServ), vynúteným smerovaním a plánovaním prevádzky môžeme dosahovať požadované parametre kvality služby. Riadením prevádzky je možné predchádzať zahlteniu siete. Optimálne rozloženie záťaže prispieva k efektívnejšiemu využitiu prenosových médií. Zavádzanie MPLS je práve kvôli optimalizácii IP sietí. Potom je ľahšie dosiahnuť požadované hodnoty oneskorenia a strát paketov.

Idea MPLS (GABAUER, 2007) spočíva v označovaní prenášaných paketov prídavným návestím (labelom), ktoré má konštantnú dĺžku a jednoznačnú hodnotu. Paket je pri vstupe do siete označený okrajovým smerovačom návestím. Žiadny so smerovačov už nesmeruje pakety podľa IP adresy ale podľa daného návestia. Po prijatí označeného paketu sa smerovač pozrie do tabuľky návestí a podľa nájdeného záznamu odpovedajúcemu návestiu daného paketu ho odošle

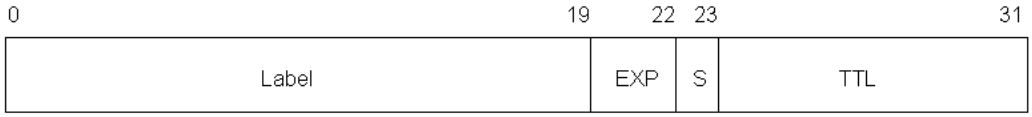
na príslušné výstupné rozhranie. Jednotlivé smerovače si návesita pre konkrétne siete vytvárajú samočinne a nemusia mať tú istú sieť označenú rovnakou hodnotou. Preto si smerovače tieto hodnoty návestí medzi sebou preposielajú. Potom smerovač pred odoslaním paketu môže zmeniť návestie podľa toho, na ktorý smerovač daný paket smeruje. Na výstupe paketu zo siete okrajový smerovač odoberie návestie a paket je ďalej smerovaný pôvodnými metódami použitého protokolu ako pred vstupom do MPLS siete.



Obrázok 2. Príklad smerovania v MPLS sieti.

MPLS hlavička má veľkosť 32 bitov. Je súčasťou rámca druhej vrstvy a vkladá sa medzi hlavičku rámca a hlavičku IP paketu. Skladá sa z nasledovných častí:

- Label (návestie) – 20 bitov. Obsahuje hodnotu návestia.
- EXP – 3 bity. Experimentálne bity využívané pre potreby klasifikácie paketu v rámci definovania QoS.
- S (Bottom of stack) – označuje prvé návestie vložené do paketu
- TTL (Time To Live) – 8 bitov. Podobne ako v IP určuje počet hopov, po ktorých paket zanikne.



An MPLS label is composed of the following parts:

20-bit Label Value
3-bit Experimental Field
1-bit bottom-of-stack indicator
8-bit Time-to-Live field

Obrázok 3. Zobrazenie MPLS hlavičky.

8 OPNET Modeler

OPNET Modeler (OPNET Technologies, Inc., 2008) vyvinula spoločnosť OPNET Technologies, Inc. Umožňuje modelovanie, analýzu a dizajnovanie sietí. Taktiež je možné vytváranie vlastných modelov zariadení, protokolov a aplikácií. Možno porovnávať vplyv rôznych technológií, modelovať rôzne typy prevádzky.

Hlavnou doménou tohto simulačného nástroja je rýchla a efektívna práca vďaka jeho grafickému prostrediu. Pri simulovaní namodelovanej siete je možne zaznamenávať rôzne druhy štatistík. Po ukončení simulácie je možné výsledky zobraziť v grafoch a spracovať.

Ďalšou nenahraditeľnou súčasťou Opnetu sú jeho knižnice. Obsahujú obrovské množstvo hotových modelov sieťových zariadení rôznych výrobcov, najpoužívanejšie transportné protokoly a typy liniek. Ku každému modelu zariadenia je prístupný zdrojový kód a je možné ho podľa potreby upravovať. Samozrejmosťou je aj široká škála parametrov podľa typu zariadenia, ktoré je možné nastaviť.

8.1 Základné prvky v OPNET Modeler

Štruktúra modelov sa v OPNETe skladá z troch základných úrovní prvkov.

- subnet model (model podsiete)
- node model (model uzla)
- process model (model procesu)

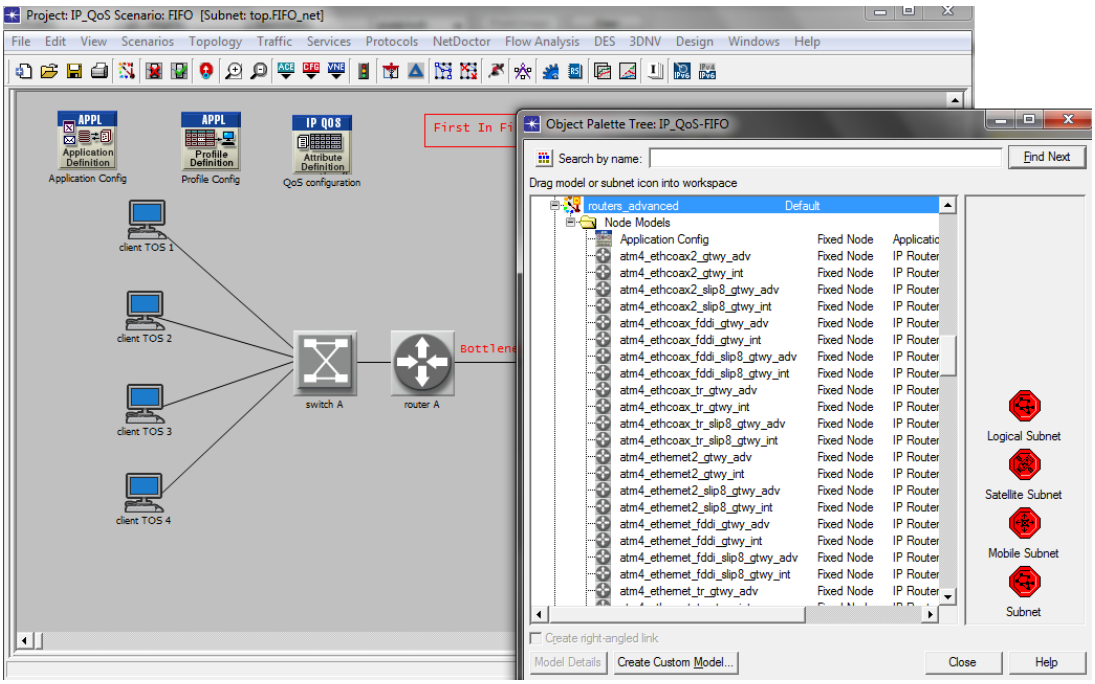
Subnet model je prvok najvyššej úrovne. Obsahuje topológiu siete. Nachádzajú sa tu poprepájané modely uzlov staníc, serverov, smerovačov, switchov a pod. **Node model** je model konkrétneho sieťového zariadenia. Obsahuje všetky procesy, ktoré su pre dané zariadenie špecifické a vzťahy medzi nimi. **Process model** modeluje činnosť procesu pomocou konečno-stavového automatu. Sú tu definované stavy a ich prechody. Jednotlivé stavy sú implementované v jazyku C/C++.

8.2 Editory

OPNET má pre každú úroveň modelu vlastný editor, v ktorom je možné vytvárať alebo upravovať daný model.

8.2.1 Editor projektu

Editor projektu je grafický editor modelujúci topológiu a komunikáciu v sieti. Sieť obsahuje modely uzlov a moduly na určené konfiguráciu modelovanej siete. Všetky tieto komponenty je možné do projektu vložiť z palety objektov (Object palette) pomocou akcie *drag and drop* (ťahaj a pusť).



Obrázok 4. Ukážka editoru projektu a palety objektov.

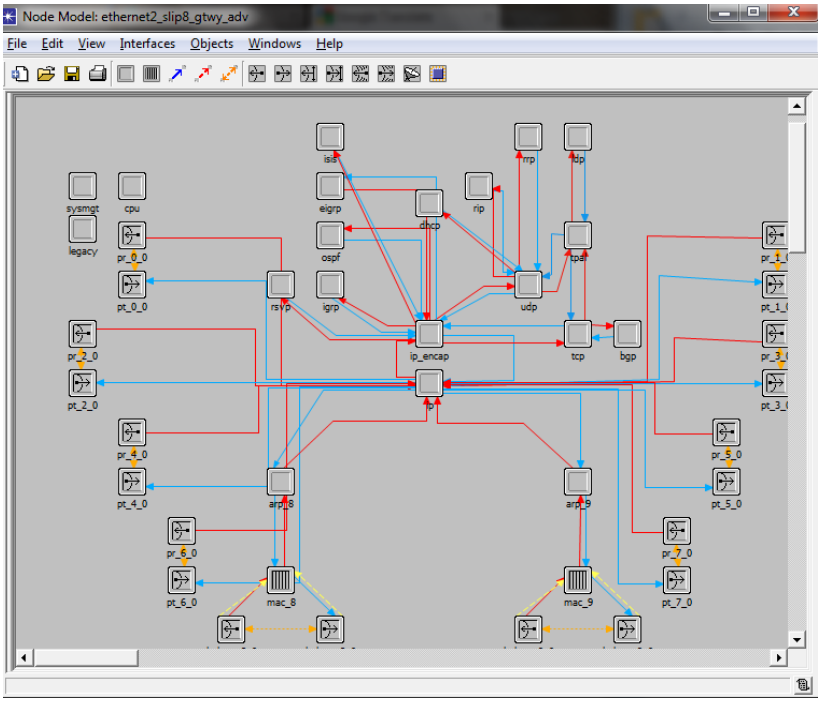
Vytváraný projekt môže obsahovať niekoľko scenárov. Každý z týchto scenárov môže modelovať odlišné topológie s vlastnou konfiguráciou. Taktiež pri spúšťaní simulácie je možné zvoliť, ktoré scenáre sa odsimulujú. Výsledné štatistiky sú osobitné pre jednotlivé scenáre.

Nástrojová lišta projektového editoru okrem iného umožňuje: otvorenie palety objektov, prechod na subsieť vyššej úrovne, priblíženie alebo vzdialenie pohľadu, otvorenie dialógu na spustenie simulácie, zobrazenie výsledkov, zobrazenie grafov a i.

8.2.2 Editor uzla

Po dvojitom kliknutí na niektorý z uzlov vložených do projektu sa dostaneme do nižšej úrovne. Node Editor ponúka prostriedky potrebné na modelovanie vnútorného fungovania uzlov. V rámci Node Editora môžeme pristupovať v rôznym modulom. Každý druh modulov slúži na modelovanie vnútorných aspektov správania sa uzla, ako sú vytváranie dát, uchovávanie dát, spracovanie dát a smerovanie, prenos dát a pod. Jeden model uzla sa obvykle skladá z viacerých modulov – niekedy desiatky alebo dokonca stovky modulov.

Prepojenie modulov je realizované ako *Packet Streams* (toky paketov) alebo ako *Statistic wires* (štatistická linky). Packet streams prenášajú pakety s dátami medzi modulmi v uzle. Statictic wires poskytujú možnosť jednému modulu sledovať rôzne hodnoty v inom module. Kombinované použitie modulov, a uvedených spôsobov prepojenia poskytuje možnosť vytvárať veľmi realistické simulácie správania uzla. Okrem toho je možné s konkrétnymi prijímačmi a vysielačmi zaobchádzať ako so spojenou dvojicou modulov.



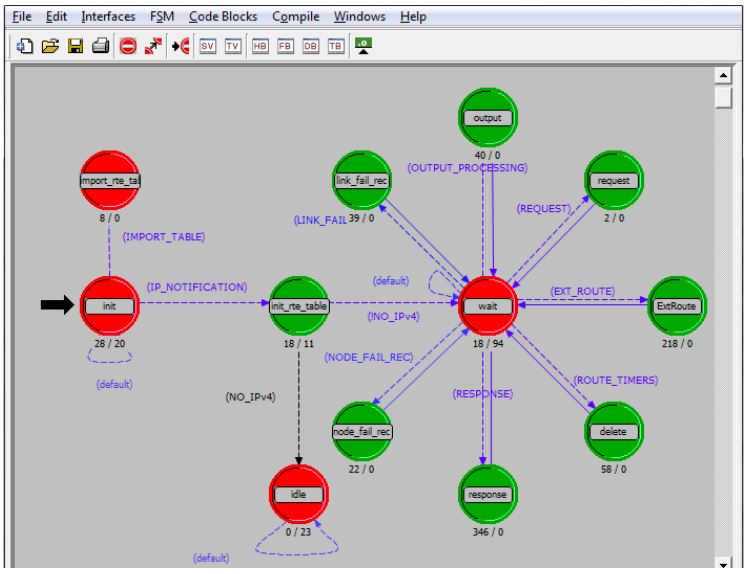
Obrázok 5. Ukážka editora uzla.

8.2.3 Editor procesu

Proces v rámci počítačových systémov a komunikačných sietí možno považovať za sériu logických operácií vykonávaných na dátach a podmienok, ktoré tieto operácie vyvolávajú. Procesy môžu implementovať vlastnosti oboch, hardwarových aj softwarových komponentov. Procesné modely v OPNETe popisujú logické procesy reálneho sveta, ako napríklad:

- komunikačné protokoly a algoritmy
- manažment zdieľaných zdrojov
- frontovanie
- špecializované generátory prevádzky
- mechanizmy na zber štatistík
- operačné systémy

Editor procesu poskytuje potrebné funkcie pre vytváranie procesných modelov, ktoré sa skladajú z grafickej a textovej zložky. Pre znázornenie logickej organizácie modelu procesu využíva diagramy prechodov stavov. Ikony reprezentujú logické stavy a ich prepojenia určujú prechody medzi stavmi. Operácie vykonávané procesom sú popísané vo výrazoch písaných v jazyku C/C++. Tieto výrazy môžu byť asociované so stavmi, prechodmi alebo v špeciálnych blokoch procesného modelu. Na vytváranie výrazov (programový kód) je určený jednoduchý editor.



Obrázok 6. Ukážka editora procesu.

Kombinácia grafických komponentov a komponentov na vkladanie textu má dve hlavné výhody:

- grafické prvky umožňujú vizualizovať módy procesov a kontrolu prechodov medzi týmito módmi. Táto reprezentácia pomáha napríklad aj pri vytváraní dokumentácie.
- písanie kódu v jazyku C/C++ zabezpečuje minimálne obmedzenia na zložitosť alebo vernosť modelu.

9 Modelovanie siete

Vytváranie simulačného modelu siete bolo so snahou maximálne využiť prostriedky ponúkané simulačným nástrojom OPNET Modeler. Zostavenie modelu, konfigurácia zariadení a definícia aplikácii bude podrobne vysvetlená ďalej.

9.1 Návrh siete

Vytvoríme si v OPNETe nový projekt. Na začiatku do projektu, na pracovnú plochu, vložíme objekty potrebné na definovanie a konfiguráciu aplikácií, vytváranie profilov, definovania QoS parametrov a nastavení potrebných pre MPLS:

- Application config
- Profile config
- QoS Attribute config
- MPLS Config

V lište nástrojov v editore projektu klikneme na  **Object Palette**. V palete vyberieme skupinu **utilities** a vložíme dané objekty na plochu.

Ďalej pokračujeme vkladaním modelov sieťových uzlov. Z palety objektov vložíme na plochu modely uzlov nachádzajúce sa v palete s názvom **MPLS** a majú nasledovné označenie:

- **ethernet2_slip8_ler**
- **ethernet2_slip8_lsr**

Model ethernet2_slip8_ler predstavuje okrajový smerovač, ktorý je zodpovedný za značkovanie vstupných tokov MPLS návestím a takto označené ich posiela ďalej do siete. Model ethernet2_slip8_lsr predstavuje chrbticový smerovač, ktorý na základe značky, pridelenej okrajovým smerovačom, posiela pakety ďalej v sieti. Do projektu sme vložili osem LSR smerovačov a šesť LER smerovačov.

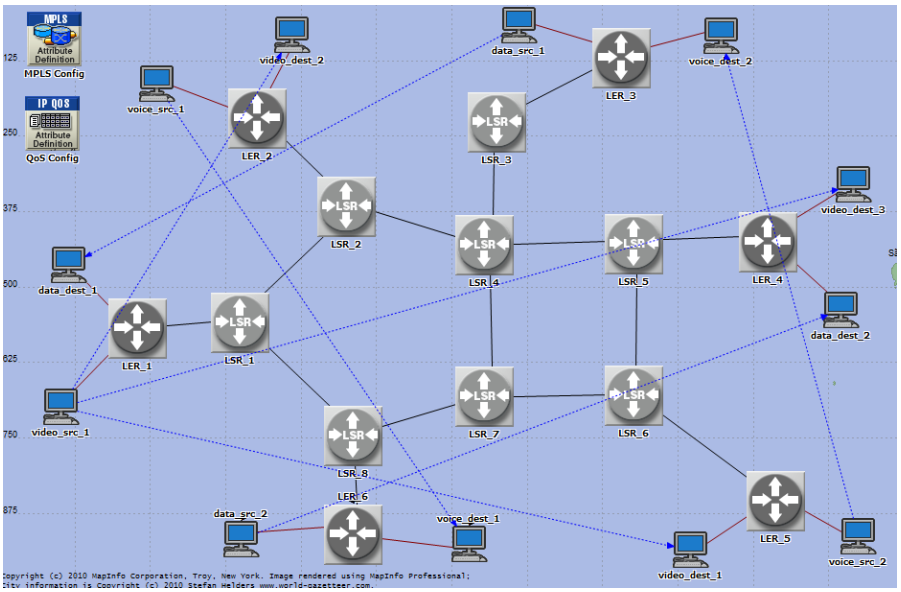
Teraz pristúpime k vkladaniu modelov uzlov predstavujúcich koncové zariadenia. Pre zdroje prevádzky v sieti použijeme ethernetové pracovné stanice,

ktoré sa nachádzajú v palete **ethernet** pod názvom **ethernet_wkstn**. Tieto modely použijeme taktiež ako klientov – ciele prevádzky.

Uzly chrbticových smerovačov pomenujeme ako LSR_1 až LSR_8. Uzly okrajových smerovačov pomenujeme ako LER_1 až LER_6. Koncové zariadenia pomenujeme podľa toho aký typ prevádzky budú podporovať. Napríklad zdroj video prevádzky pomenujeme **video_src** a cieľ pomenujeme **video_dest**. Analogicky nastavíme názvy aj pre ostatné koncové zariadenia.

Prepojenie vložených uzlov bude nasledovné. Chrbticové LSR smerovače prepojíme medzi sebou ako ukazuje obrázok 7. Okrajové LER smerovače pripojíme k smerovačom LSR. Koncové uzly pripojíme na LER smerovače. Výsledné zapojenie by malo vyzerať nasledovne, ako na obrázku 7.

Linky použité na prepojenie smerovačov nájdeme v rovnakej palete ako modely týchto smerovačov, teda v palete s názvom **MPLS**. Smerovače medzi sebou prepojíme linkami typu **PPP_DS_3**. Na pripojenie koncových zariadení k okrajovým smerovačom použijeme linku **100BaseT**, ktorú môžeme nájsť napríklad v palete s názvom **internet_toolbox**.



Obrázok 7. Návrh modelu siete.

9.2 Modelovanie tokov v sieti

Simulovanie prevádzky v sieti je možné realizovať viacerými spôsobmi. V sieti budeme simulovať 3 typy prevádzky. Dáta, hlas a video. V nasledujúcich častiach si popíšeme ako tieto prevádzky vytvoriť dvoma spôsobmi.

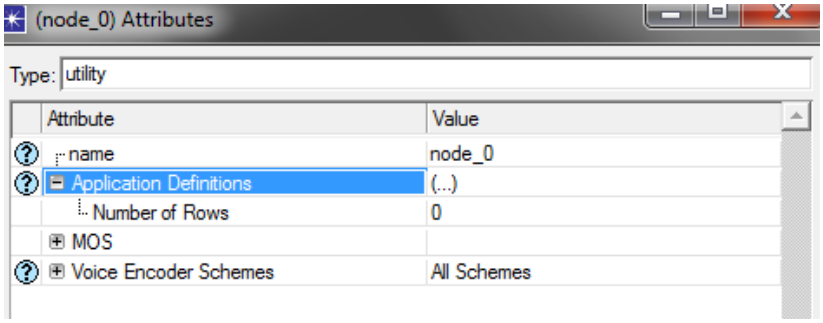
Prvý spôsob zahŕňa vytvorenie aplikácií a ich následné zaradenie do profilov, ktoré sú potom použité pre špecifikovanie tokov medzi konkrétnymi uzlami.

Druhý spôsob predstavuje vytváranie tokov medzi dvoma uzlami. Týmto tokom je možné určiť ich charakteristiku.

V našej simulácii sme použili druhý uvedený spôsob. Bližšie si o ňom povieme neskôr.

9.2.1 Definovanie aplikácií

Pre zadefinovanie aplikácií slúži objekt **Application Config**. Pravým kliknutím na tento objekt vyvoláme kontextové menu a vyberieme možnosť **Edit attributes**.



Obrázok 8. Definovanie aplikácie

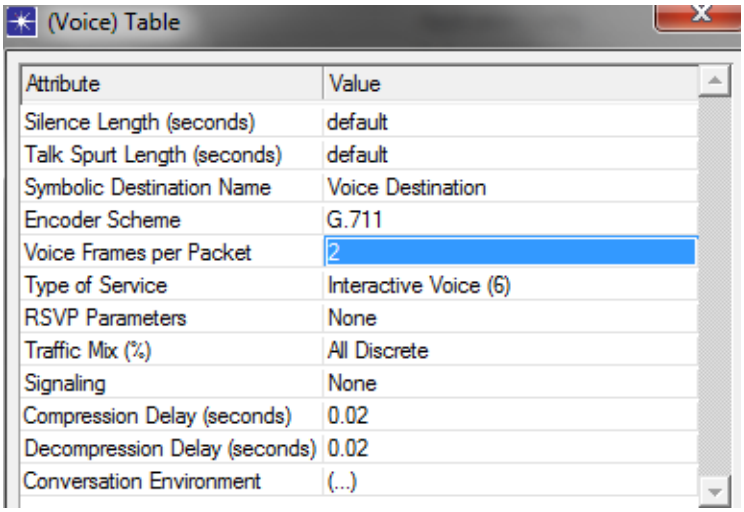
V položke **Application definitions** máme možnosť zadefinovať ľubovoľný počet aplikácii, ktoré budeme v projekte využívať. Na výber máme z niekoľkých preddefinovaných aplikácií:

- Database (Low/Medium/High load)
- Email (Low/Medium/High load)
- FTP (Low/Medium/High load)

- HTTP (Low/Medium/High load)
- Print (Low/Medium/High load)
- Remote Login (Low/Medium/High load)
- Video Conferencing (Low/High resolution, VCR quality)
- Voice (Low/PCM/GSM quality, IP telephoni quality – s možnosťou detekcie ticha)

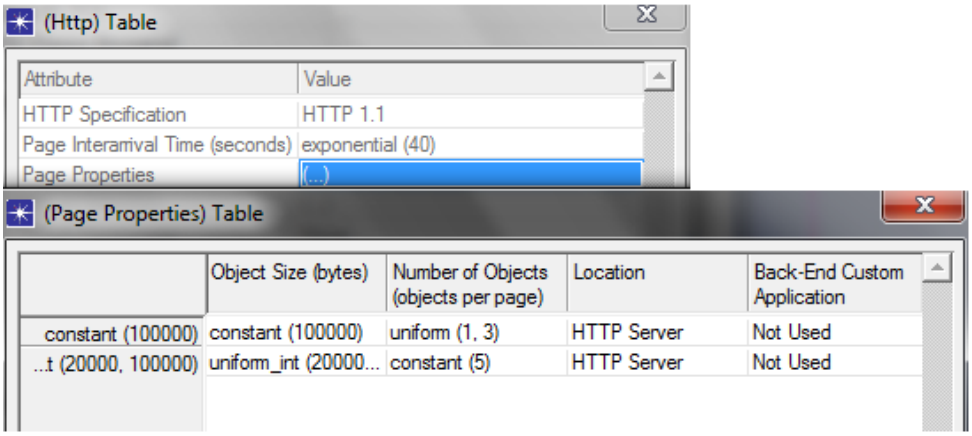
Každú z týchto preddefinovaných aplikácii je možné modifikovať. Taktiež môžeme vytvoriť vlastnú aplikáciu vybraním možnosti **Custom**.

Pre hlasovú prevádzku je možné použiť preddefinovanú aplikáciu **Voice** s nastavením na voľbu **PCM quality**. Toto nastavenie nám pomôže ľahšie nadefinovať potrebné parametre. Použitý je kódek G.711 s dátovým tokom 64 kbit/s. Pre editáciu zvoleného nastavenia vyberieme možnosť **Edit...**.



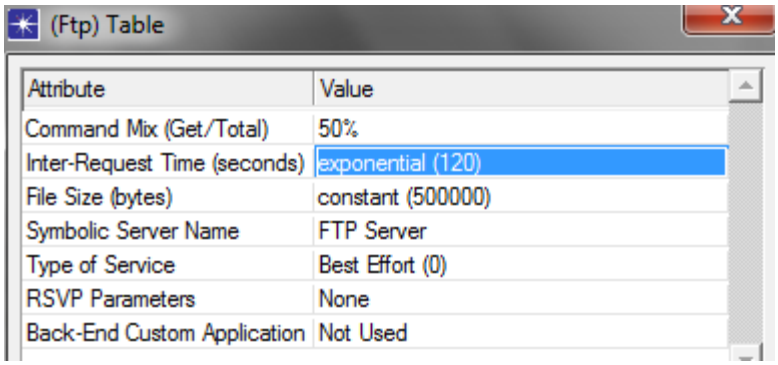
Obrázok 9. Nastavenie aplikácie pre hlasovú prevádzku.

Pre dátovú prevádzku je možné použiť hneď niekoľko aplikácií. Uvedieme príklady definície HTTP a FTP prevádzky. Pri definícii prvej aplikácie vyberieme pri atribúte **Http** voľbu s názvom **Heavy browsing**. Túto definíciu upravíme na požadované parametre. Atribút **Page Interarrival Time** predstavuje časový rozostup medzi dvoma požiadavkami na prístup k stránke v sekundách (exponenciúlne rozdelenie). Ďalej je možné zmeniť atribút **Page Properties**. Tento sa skladá z dvoch hodnôt, každá hodnota má dva parametre. Prvá hodnota môže reprezentovať počet načítavených html (alebo podobných) dokumentov. Druhou môžeme určiť napríklad počet načítavaných obrázkov.



Obrázok 10. Nastavenie vlastností stránky v http aplikácii.

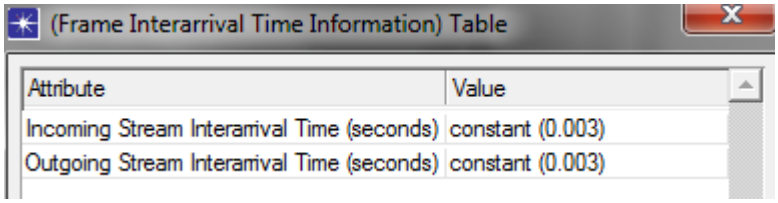
Druhá aplikácia bude s atribútom **FTP**. Hodnotu nastavíme napríklad na **Heavy Load**. Táto voľba predstavuje požiadavku na stiahnutie alebo odoslanie súboru. Pri tejto aplikácii je možné upraviť nasledovné atribúty. Časový interval medzi dvoma požiadavkami určuje atribút **Inter-request Time**. Položka **File size** určuje veľkosť súboru.



Obrázok 11. Príklad nastavenie aplikácie ftp.

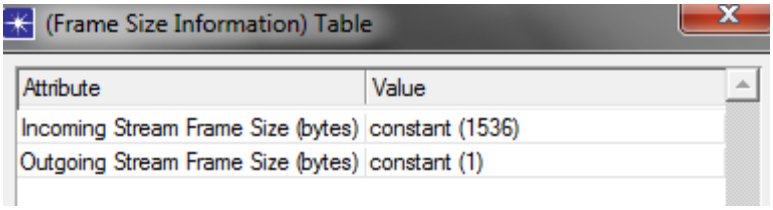
Pre video prevádzku vyberieme atribút **Video conferencing**. Hodnotu nastavíme na voľbu **High Resolution Video**. Pre modifikáciu parametrov opätovne zvolíme voľbu **Edit...**. Pri experimentovaní s nastaveniami tieto aplikácie sme zistili nasledovné skutočnosti, ktoré je potrebné zohľadniť pri modifikácii. Atribút **Frame Interval Time Information** predstavuje počet paketov vyslaných za sekundu. Atribút **Frame Size Information** určuje veľkosť paketu. Ak chceme modelovať dátový tok streamovaného televízneho kanálu s veľkosťou približne 4 Mbit/s. Ak zvolíme veľkosť paketu 1536 bajtov, je potrebný tok 333 paketov za sekundu. Tieto hodnoty nastavíme do aplikácie. Pri atribúte **Frame Interval Time Information** vyberieme voľbu **Edit...**. Tu nastavíme hodnoty ako je vidieť na

obrázku č. 12. Hodnota sa nastavuje pre prichádzajúci a odchádzajúci stream samostatne. Napriek tomu nastavíme obe hodnoty na rovnako. Pri rozdielnych hodnotách aplikácia nepracovala správne.



Obrázok 12. Nastavenie toku paketov vo video aplikácii.

Keďže je preddefinovaná aplikácia primárne určená na simulovanie video konferencie, kde dátový tok je v oboch smeroch rovnaký, je potrebné túto skutočnosť zobrať do úvahy. Rovnako ako pri nastavovaní počtu paketov je nutné nastaviť tieto hodnoty samostatne. Tu však nepožijeme rovnakú hodnotu. Pre prichádzajúci tok nastavíme hodnotu 1536, pre odchádzajúci tok nastavíme hodnotu 1. Tým sme vytvorili aplikáciu, ktorá generuje dátový tok smerom k užívateľovi.



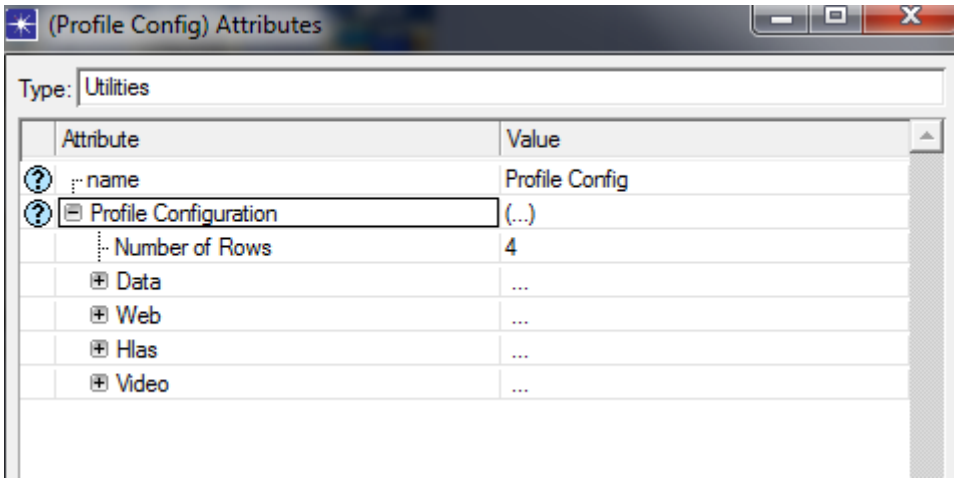
Obrázok 13. Nastavenie veľkosti paketu vo video aplikácii.

Na záver je ešte potrebné nastaviť v špecifikácii tejto aplikácie hodnotu atribútu **Type of Service**. Vyberieme voľbu **Streaming multimedia(4)**. Táto hodnota je dôležitá pre zabezpečie kvality služby pre daný tok.

9.2.1 Definovanie profilov

Pre definovanie profilov aplikácii slúži objekt **Profile Config**. Pravým kliknutím na tento objekt vyvoláme kontextovú ponuku a vyberieme voľbu **Edit attributes**. V strome vyhľadáme položku **Profile Configuration**. Tá nám zatiaľ ponúka iba definíciu počtu riadkov, kde každý riadok predstavuje jeden profil. Pristúpime k samotnej definícii profilu. Zadáme názov profilu, ktorý by mal

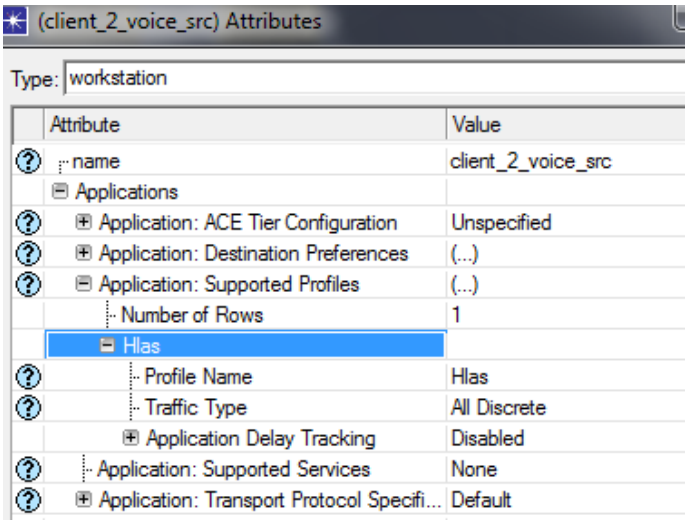
korešpondovať s jeho vlastnosťami. Do každého profilu je možné pridať jednu alebo viac aplikácií. Ďalej pri každej aplikácii je potrebné nastaviť parametre správania sa aplikácie. Atribút **Operation Mode** určuje spôsob spúšťania aplikácii v profile. Hodnota **Serial (Ordered)** znamená, že aplikácie sa budú spúšťať sériovo za sebou. Ďalším atribútom je **Star Time**, ktorý určuje, kedy od začiatku simulácie sa daný profil spustí. Atribút **Duration** určuje maximálny čas, do ktorého aplikácia skončí. Posledným atribútom je **Repeatability**. Hodnoty tohto atribútu je potrebné nastaviť iba v tom prípade, keď sme zvolili čas trvania profilu kratší ako je čas celej simulácie.



Obrázok 14. Ukážka zadefinovaných profilov aplikácii.

9.2.2 Nastavenie aplikácií na klientoch

Ak už máme vytvorené a nadefinované aplikácie a profily, môžeme pristúpiť k nastaveniu aplikácii na klientoch. Vyberieme model uzla klienta, pravým kliknutím vyvoláme kontextové menu a vyberieme **Edit attributes**. Tu rozklikneme položku **Applications** a potom položku **Application: Supported Profiles**. Zvolíme požadovaný počet riadkov. Pre každú aplikáciu jeden riadok. Potom v položke **Profile Name** vyberieme nami definovaný profil, ktorý podporuje aplikáciu potrebnú na danom klientovi. Toto opakujeme pre každý riadok.

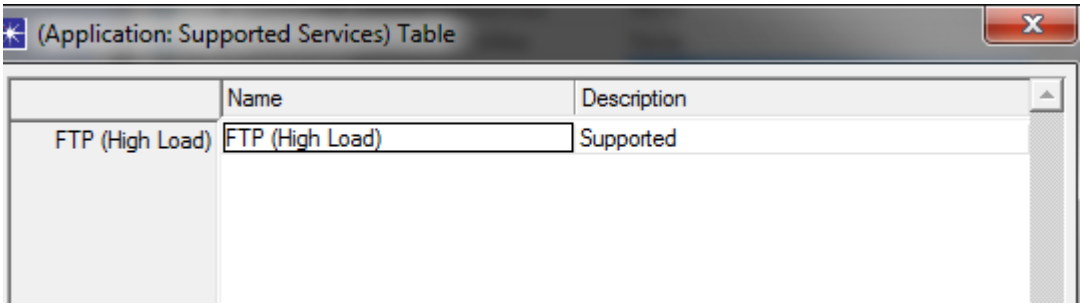


Obrázok 15. Priradenie klientovi profil hlasovej prevádzky.

9.2.3 Nastavenie serveru pre podporu aplikácií

Servery priamo neiniciujú aplikácie, iba odpovedajú na požiadavky klientov. Preto je potrebné nastaviť, ktoré aplikácie bude daný server podporovať.

Vyberieme požadovaný server a podobne ako pri klientoch, z kontextového menu vyberieme **Edit Attributes**. V položke **Applications** klikneme na editáciu atribútu **Application: Supported Services**. Tu vyberieme pre požadovaný počet podporovaných aplikácií počet riadkov. V stĺpci **Name** vyberieme z ponuky nami zadefinovaných aplikácií. V stĺpci **Description** nastavíme hodnotu Supported. Ak by sme napríklad chceli dočasne deaktivovať podporu danej aplikácie nemusíme ju mazať, stačí nastaviť v stĺpci Description hodnotu Not Supported.



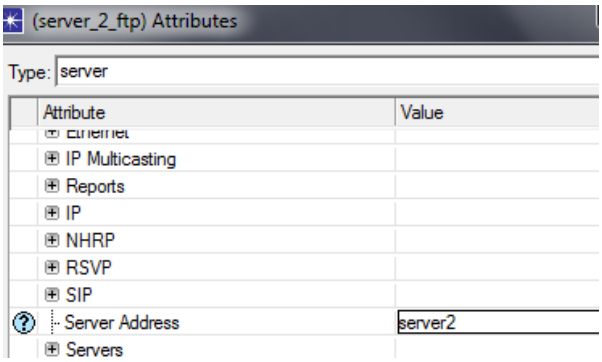
Obrázok 16. Nastavenie serveru pre podporu FTP aplikácie.

9.2.4 Nastavenie spojenia klient-server

Toto nastavenie je dôležite, keď chceme vytvoriť konkrétne spojenie medzi klientom a serverov, prípadne medzi dvoma klientmi. Ak takúto špecifikáciu medzi uzlami nevykonáme, OPNET vyberie pre daného klienta server náhodne, na základe podporovaných aplikácií.

Ak teda požadujeme aby konkrétny klient komunikoval s určitým serverom je potrebné špecifikovať nastavenia atribútu **Application: Destination Preferences**. Tento atribút umožňuje mapovať symbolické meno serveru na reálne meno serveru. Keďže každá z aplikácií používa symbolické meno serveru. K jednému symbolickému meno servera je možné priradiť viacero reálnych mien. Preto je pri výbere serveru položka **Selection Weight**. Hodnota tejto položky predstavuje váhu, podľa ktorej je z viacerých serverov vyberané. Čím väčšia váha tým je pravdepodobnosť výberu daného serveru väčšia. Hodnota **Server address** identifikuje server podľa mena. Toto meno musí byť unikátne. Takýto unikátny identifikátor je možné nastaviť aj pre klienta editovaním atribútu **Client address** a pre LAN sieť editovaním atribútu **LAN Server Name**.

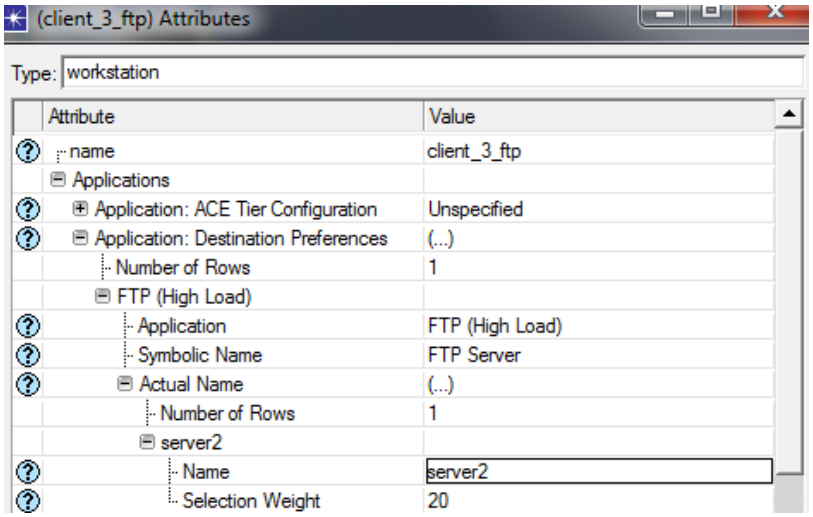
Pri vytváraní takéhoto spojenia je vždy vhodné najskôr nastaviť adresu servera (meno). Samozrejme je tiež potrebné aby daný server podporoval aj potrebnú aplikáciu.



Obrázok 17. Nastavenie unikátneho názvu pre server.

Ak už máme na servery nastavené meno, môžeme pristúpiť k nastaveniu klienta. V editácii atribútov prejdeme na položku **Applications >> Appication: Destination Preferences**. Pridáme riadok a vyberieme serverom podporovanú aplikáciu a symbolický názov serveru. Pod položku **Actual name** pridáme riadok a vyberieme reálnu adresu serveru, ktorú sme zadefinovali na servery. Ak by sme

týchto serverov pridávali viac môžeme určiť pre každý server váhu v položke **Selection Weight**.



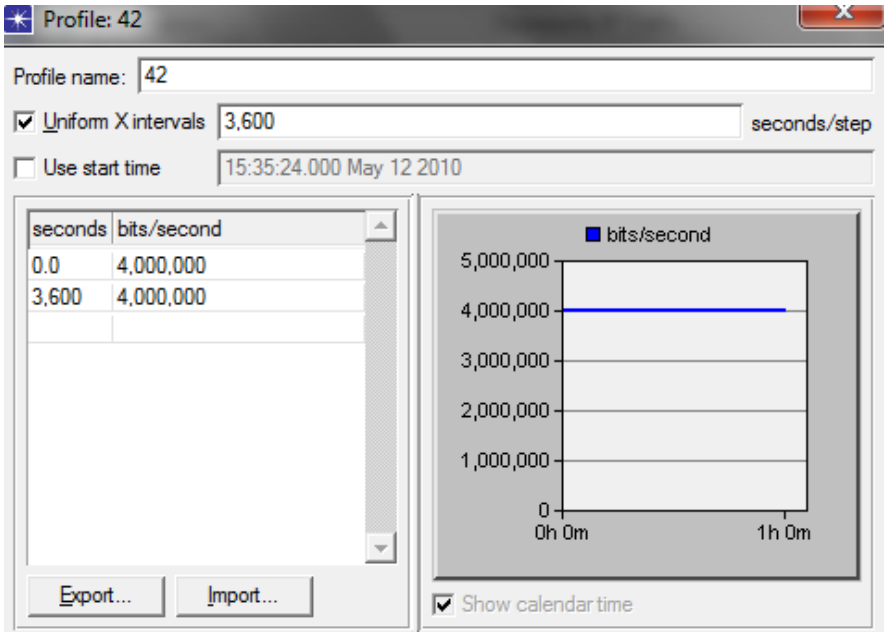
Obrázok 18. Nastavenie komunikácie klienta s vybraným serverom.

9.2.3 Definovanie tokov prevádzky

Ak sú v modelovanej sieti vyžadované toky s veľkým bitovým tokom a množstvom paketov je vhodnejšie použiť modely tokov prevádzky. Oproti aplikáciam je jednoduchšie určiť zdroj a cieľ toku. Taktiež špecifikovanie toku je jednoduchšie. Pre tok je možné špecifikovať tok v bitoch za sekundu a počte paketov za sekundu. Z týchto dvoch parametrov je potom automaticky vypočítaná veľkosť paketov.

Model pre vytvorenie prevádzkového toku nájdeme v Palete objektov pod **Demand Models >By Type >IP**. Vyberieme model s názvom **ip_traffic_flow**. Kliknutím na uzol určíme zdroj prevádzky. Tým sme začali definíciu smeru toku. Teraz určíme uzol, kam bude tok smerovať. Tým je smer toku zadefinovaný. Teraz máme možnosť definovať ďalšie toky, alebo definíciu ukončiť pravým kliknutím na pozadie a výberom voľby *Abort Demand Definition.* Teraz, keď máme vytvorené toky, pristúpime k špecifikácii vlastností toku. Vstúpime do dialógového okna editácie atribútov. Tu postupne nastavíme atribúty. Atribút **Traffic (bits/second)** určuje bit rate toku. Je možné vybrať z preddefinovaných profilov, alebo zadefinovať vlastný. Pri definícii je potrebné určiť celkovú dĺžku trvania toku a počiatočný čas. Ak sa bit rate toku v čase mení je možné zadefinovať časy, kedy

má k zmenám dôjsť. Obdobným spôsobom nastavíme aj druhý parameter **Traffic (packets/second)**, kde zadefinujeme koľko paketov za sekundu bude daný tok generovať. Opäť je možné zadefinovať počas trvania toku viacero hodnôt v špecifikovaných časoch. V podstrome atribútu **Traffic Characteristics** zvolíme typ služby zmenou atribútu **Type of Service**.



Obrázok 19. Definovanie bitového toku v prevádzke.

Pre hlasovú prevádzku sme zvolili bitový tok 80000 bitov za sekundu, počet paketov za sekundu 50. To reprezentuje 1 hlasový hovor.

Video prevádzku jedného prenášaného video streamu sme zadefinovali s bitovým tokom 4Mbit/s a s 735 paketmi za sekundu.

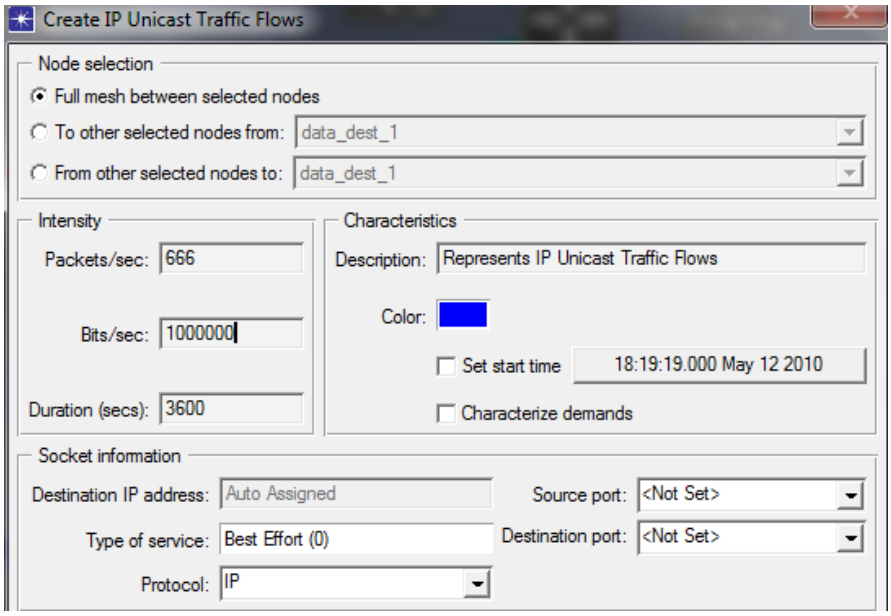
Jeden tok dátovej prevádzky je simulovaný s bitovým tokom 2Mbit/s a s 500 paketmi za sekundu.

9.2.4 Nastavenie prevádzky na pozadí

V modely siete je potrebné zadefinovať aj prevádzku na pozadí, ktorá nám bude modelovať záťaž v sieti. Prevádzku na pozadí sme definovali nasledovne. Medzi všetkými koncovými uzlami sme vytvorili toky na pozadí pre každý typ prevádzky. Prvá skupina tokov, predstavujúca tok videa na pozadí, bude mať bitový tok 900000 bitov za sekundu a 150 paketov za sekundu. Druhá skupina

s bitovým tokom 500000 bitov a 125 paketmi za sekundu predstavujúca dátový tok. Posledná skupina, predstavujúca hlasovú prevádzku na pozadí, bude mať bitový tok 128000 bitov a 100 paketov za sekundu. Keďže komunikuje každý s každým, prichádzajúce aj odchádzajúce toky od jedného koncového uzla sa pohybuje na úrovni 11 násobku súčtu týchto tokov.

Nastavenie tokov sa vykonáva hromadne. Označíme všetky uzly, medzi ktorými chceme tieto toky vytvoriť. Ak su tieto uzly jedného typy, stačí vyvolať nad jedným z nich kontextovú ponuku a vybrať voľbu *Select Similar Nodes*. Tým sa nám vyberú všetky uzly daného typu. Pokračujeme výberom z hlavného menu: *Traffic > Create Traffic Flows > IP Unicast...*. Otvorí sa nám okno, v ktorom môžeme špecifikovať vlastnosti tokov.

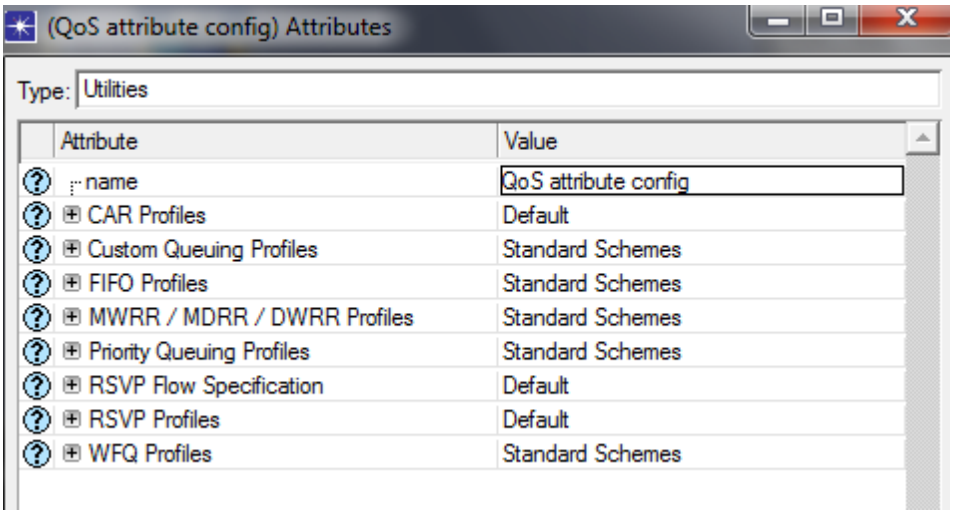


Obrázok 20. Nastavenie tokov na pozadí.

Vo voľbe *Node selection* vyberieme prvú možnosť, teda **Full mesh between selected nodes**, čo nám vytvorí toky medzi uzlami každý s každým. Ďalej zadefinujeme intenzitu toku. Počet 666 paketov za sekundu s bitovým tokom 1 mega bit za sekundu vytvoria tok s veľkosťou paketov približne 1500 bajtov. Typ služby zvolíme *Best Effort* a protokol *IP*. Tým máme špecifikovanú prevádzku na pozadí.

9.4 Definovanie QoS profilov

Na zadefinovanie a konfiguráciu profilov zabezpečujúcich kvalitu služby slúži objekt **QoS Attribute Config**, ktorý sme už na začiatku vložili do projektu. Pre konfiguráciu parametrov klikneme pravým na tento objekt a vyberieme voľbu **Edit Attributes**.



Obrázok 20. Preddefinované QoS profily.

Tu máme možnosť vybrať si z už preddefinovaných schém, ktoré je možné použiť v našej simulovanej siete. OPNET nám ponúka nasledovné:

- CAR Profiles (Committed Access Rate)
- Custom Queuing Profiles
- FIFO Profiles
- MWRR / MDRR / DWRR Profiles
- Priority Queuing Profiles
- RSVP Flow Specification
- RSVP Profiles
- WFQ Profiles

Ďalej si popíšeme niektoré zo schém, ktoré budeme používať.

9.4.1 FIFO

Vo FIFO profile je možné nastaviť dva základné parametre. Veľkosť fronty a funkciu RED (WRED). Veľkosť fronty sa udáva v počte paketov, ktoré môžu byť vo fronte uložené. Parametre pre RED sú nasledovné:

- **RED Status** – zapne alebo vypne funkciu RED/WRED
- **Exponential Weight Factor** – táto hodnota je použitá pre vypočítanie priemernej veľkosti fronty, pomocou predchádzajúceho priemeru a súčasnej veľkosti fronty
- **Minimum Threshold** – ak je front väčšia ako táto hodnota, pakety sú náhodne zahadzované.
- **Maximum Threshold** – ak je priemerná veľkosť fronty väčšia ako táto hodnota, všetky pakety sú zahadzované.
- **Mark Propability Denominator** – určuje pravdepodobnosť s akou su pakety zahadzované. Napr. pre hodnotu 10 znamená, že jeden z desiatich je zahodený.

9.4.2 Custom Queuing

Pri vytváraní Custom Queuing (CQ) profilu máme na výber z troch možností, podľa ktorých sa budú pakety radiť do jednotlivých frontov. Pakety je možné zaraďovať podľa typu služby, podľa typu protokolu a podľa portu. Bližšie si povieme o nastavení frontov podľa typu služby. Najskôr je potrebné zadefinovať počet frontov. Potom pre každý front sa nastavujú nasledovné parametre:

- **Byte Count** – počet bajtov, ktoré sú odoslané z daného frontu.
- **Maximum Queue Size** – veľkosť frontu v počte paketov
- **Classification Scheme** – definuje kritéria, podľa ktorých sú pakety zaraďované do frontu, v tomto prípade volíme typ služby (napr. Best Effort)
- **RED Parameters** – rovnaké nastavenie ako pri FIFO

9.4.3 Priority Queuing

Nastavenie Priority Queuing profilu je podobné ako pri CQ. Tu máme na výber zo štyroch rôznych spôsobov, podľa ktorých sa budú pakety zaraďovať do frontov. Pakety sa môžu radiť podľa typu služby, podľa typu protokolu, podľa portu a podľa diffserv DSCP (Differentiated Services Code Point). Po výbere spôsobu radenia paketov do front je potrebné určiť počet frontov. Pre každý front potom nastavujeme nasledovné parametre:

- **Priority Label** – určuje prioritu frontu. Čím väčšia hodnota tým väčšia priorita.
- **Maximum Queue Size** – veľkosť frontu v počte paketov.
- **Classification Scheme** – definuje kritéria, podľa ktorých sú pakety zaraďované do frontu, v tomto prípade volíme typ služby (napr. Best Effort)
- **RED Parameters** – rovnaké nastavenie ako pri FIFO

9.4.4 Weighted fair queueing (WFQ)

Počiatočné nastavenia podobné ako pri predchádzajúcich dvoch profiloch. Máme na výber zo štyroch spôsobov radenia paketov. Definujeme počet frontov. Tento krát sa však jednotlivým frontom prideľujú váhy. Väčšia hodnota znamená väčší podiel z danej šírky pásma pre daný front.Ostatné parametre ako Maximum Queue Size, Classification Scheme a RED Parameters sa nastavujú rovnako ako pri predchádzajúcich dvoch prípadoch.

9.5 Natavenie kvality služby na smerovači

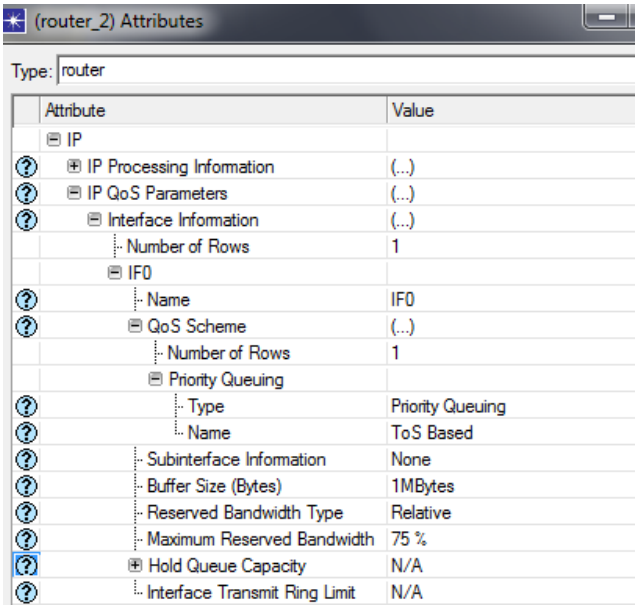
Aby smerovač podporoval funkciu kvality služby je potrebné mať nastavenú schému QoS. O tom ako sa QoS schéma nastavuje sme si už ukázali. Teraz si vysvetlíme ako nastaviť smerovač pre podporu kvality služby na základe už zadefinovanej schémy.

Podpora kvality služby sa definuje na smerovači pre jednotlivé komunikačné rozhrania. Na zvolenom smerovači vyvoláme pravým kliknutím kontextové menu. Vyberieme voľbu **Edit attributes**. Tu prejdeme cez položky: **IP**

>> IP QoS Parameters >> Interface Information. Podľa počtu rozhraní, na ktorých chceme aplikovať danú QoS schemu, zvolíme počet riadkov. Pre každý riadok potom definujeme nasledovné parametre:

- **Name** – názov rozhrania; môžeme sa stretnúť s označením IFxx, kde xx je poradové číslo rozhrania
- **QoS schema**:
  - **Type** – typ profilu QoS (napríklad: Priority Queuing)
  - **Name** – meno nami nastaveného profilu (napríklad: ToS Based)
- **Subinterface Information** – slúži na nastavenie QoS na sub-rozhraniach
- **Buffer Size** - veľkosť bufferu pre dané rozhranie
- **Reserved Bandwidth Type** – určuje typ maximálnej rezervovanej šírky pásma (možnosť nastaviť hodnoty: Relative a Absolute)
- **Maximum Reserved Bandwidth** – určuje maximálnu veľkosť rezervovanej šírky pásma (pre voľbu Relative sa zadáva v percentách, pre Absolute v bitoch za sekundu)
- **Hold Queue Capacity** – špeciálne nastavenie pre CISCO smerovače.

Tento postup opakujeme na všetkých smerovačoch na ktorých požadujeme podporu kvality služby.



Obrázok 21. Príklad nastavenia QoS schémy na rozhraní smerovača.

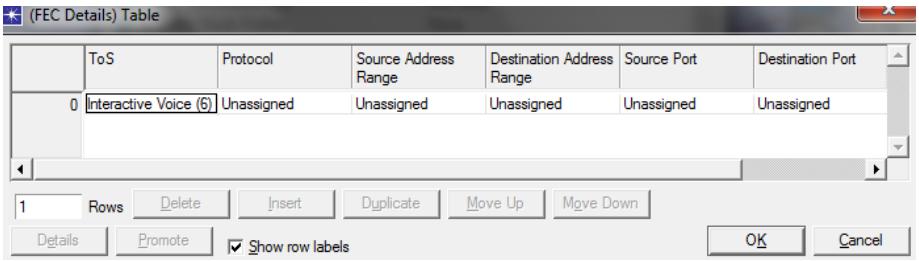
V simulácií sme na aktívnych rozhraniach smerovačov nastavili QoS profil *Weighted Fair Queuing* so špecifikáciou *DSCP Based*. Keďže presná špecifikácia pre nastavenia neboli k dispozícii, ponechali sme predvolené nastavenia OPNETom.

9.6 Nastavenie MPLS

9.6.1 Všeobecné vlastnosti

Objekt MPLS Config už máme na ploche, tak si povieme bližšie o niektorých dôležitých atribútoch, ktoré tento objekt obsahuje:

FEC Specification – tento atribút slúži na špecifikáciu Forwarding Equivalce Class (FEC) parametrov používaných v MPLS sieťach. FEC klasifikuje a zoskupuje pakety tak, aby vštky pakety v skupine boli preposlané rovnakým spôsobom. FEC využíva niektoré z polí IP hlavičky – ToS, protokol, adresa zdroja, cieľová adresa, port zdroja, port cieľa. Všetky tieto hodnoty môžu byť použité pri definícií FEC.



Obrázok 22. Špecifikácia podmienok vo FEC zázname.

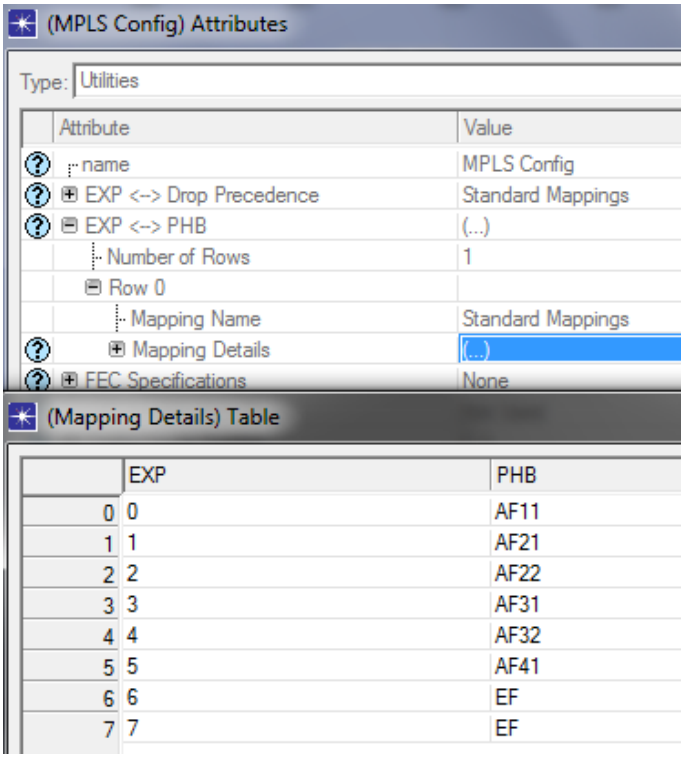
FEC tabuľka pomáha pri definícii prostredníctvom súboru pravidiel, ktoré sú kombináciou TCP, UDP a polí IP hlavičky. FEC sú určované použitím logickej operácie AND nad stĺpcami a použitím logickej operácie OR nad riadkami tabuľky. Pre to aby bol paket zaradený do konkrétnej FEC, polia IP hlavičky musia spĺňať všetky podmienky určené stĺpcami aspoň na jednom riadku definovanými vo FEC tabuľke.

LSP Specification File – tento atribút umožňuje načítať definíciu LSP (Label Switched Path) ciet z textového súboru. Ak sme urobili v nastaveniach LSP ciet nejaké zmeny, je možné tento súbor aktualizovať kliknutím na tlačidlo OK

v prehliadači LSP ciest. LSP Browser spustíme z hlvného menu: *Protocols >> MPLS >>Browse/Edit LSP Information...*. Aktualizácia obnoví v danom súbore informácie o uz existujúcich cestách a pridá aj informácie o cestach vytvorených ručne.

Traffic Trunk Profiles – tento atribút umožňuje definovať profily kanálov prevádzky (Traffic Trunk). V nich je špecifikované čo sa má stať s tokom, ktorý je mino parametrov určených v danom profile. Aplikovateľné akcie sú zahodie alebo preposlanie s preznačením alebo bez. Pre Traffic Trunk je možné nastaviť rôzne charakteristiky prevádzky, ako sú maximálna rýchlosť, priemerná rýchlosť, veľkosť špičky, maximálna veľkosť vlny.

EXP<-->Drop Precedence a **EXP<-->PHB** – tieto atribúty špecifikujú ako sa EXP bity v hlavičke MPLS prehladajú na DiffServ informácie v každom LSR smerovači. Pre cesty typu E-LSP určuje LSR smerovač spávanie pre ďalší skok (PHB – Per Hop Behavior). Zatiaľ čo pre cesty typu L-LSP určuje prioritu zahadzovania (Drop Precedence).



Obrázok 23. Mapovanie DiffServ informácií do EXP bitov MPLS hlavičky.

9.6.2 Vlastnosti smerovača

Teraz si popíšeme niektoré atribúty MPLS parametrov, ktoré je možné nastaviť na smerovači.

Traffic Mapping Configuration – tento atribút určuje väzby medzi FEC a LSP. Každý riadok v tabuľke **Traffic Mapping Configuration** špecifikuje rozdielne riadenie prevádzky (TE – traffic engineering). Každý TE záznam špecifikuje pomocou FEC, traffic trunk a LSP cesty hodnotu návestia, ktoré je aplikované na paket. Pre vytvorenie TE záznamu je potrebné, aby všetky tieto náležitosti existovali.

Keď na port vstupného LER smerovača príde neoznačený paket, nasledujúcou sekvenciou dôjde k jeho označeniu:

- na základe rozhrania, z ktorého daný paket prišiel a FEC je vybraný TE záznam,
- overí sa, či paket spĺňa charakteristiky prevádzky v súlade s vybraným TE záznamom,
- paket je na základe tohto vyhodnotenia označený a odoslaný primárnou LSP cestou definovanou v TE zázname

EXP<-->Drop Precedence a **EXP<-->PHB** – tieto atribúty určujú, ktoré mapovanie, definované v MPLS Config, bude použité pre daný smerovač

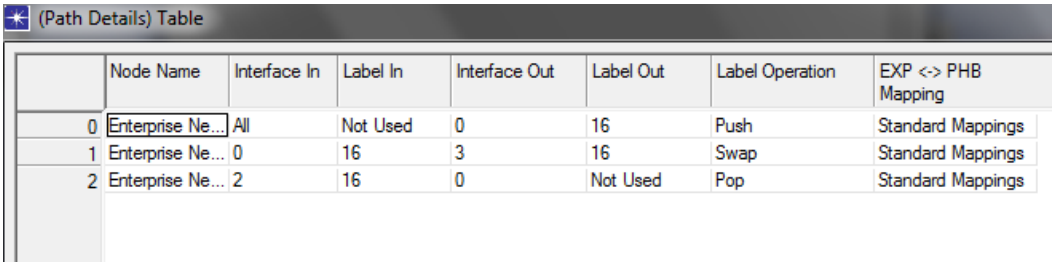
LDP Parameters – špecifikuje parametre protokolu distribuujúceho návestia (LDP – Label Distribution Protokol). Parametre tohto protokolu sa skladajú z nasledujúcich sub-atribútov:

- **Discovery Configuration** – určuje parametre *Hello* správy, ktorá sa rozposiela všetkým susedným smerovačom
- **Session Configuration** – špecifikuje parametre správy *Keep-alive*, použitej na vytvorenie LDP spojenia
- **Recovery Configuration** – špecifikuje ako sú detekované zlyhania uzlov a liniek

9.6.3 Vlastnosti LSP cesty

V tejto časti si popíšeme atribúty LSP cesty. Niektoré z týchto atribútov je možné konfigurovať cez *LSP Browser*.

- **Directionality** – špecifikuje, či je daná cesta jednosmerná alebo obojsmerná. Dynamické cesty sú vždy jednosmerné.
- **LSP Type** – určuje, či je cesta typu E-LSP alebo L-LSP. Pri E-LSP ceste je v troch experimentálnych bitoch obsiahnutá informácia o type služby. To umožňuje prenášať cez jednu LSP až osem typov služieb. Pri L-LSP ceste je síce informácia v hlavičke MPLS obsiahnutá ale ku všetkým paketom prechádzajúcim cez linku sa správa rovnako.
- **Path Details** – zobrazuje detaily trasy a definuje ako budú pakety preposlané po danej LSP ceste. Tieto atribúty sú na dynamických LSP configurované automaticky. Pre konfiguráciu týchto atribútov na statickej LSP veberieme v hlavnom menu *Protocols* > *MPLS* > *Update LSP Details* .


(Path Details) Table

	Node Name	Interface In	Label In	Interface Out	Label Out	Label Operation	EXP <-> PHB Mapping
0	Enterprise Ne...	All	Not Used	0	16	Push	Standard Mappings
1	Enterprise Ne...	0	16	3	16	Swap	Standard Mappings
2	Enterprise Ne...	2	16	0	Not Used	Pop	Standard Mappings

Obrázok 24. Ukážka definície LSP cesty.

9.6.4 Postup konfigurácie

Na konfiguráciu MPLS v sieti sú potrebné tri základné kroky:

- Vytvoriť LSP cesty v sieťovej topológii
- Vytvoriť FEC a Traffic Trunk v objekte MPLS Config
- Nakonfigurovať LER smerovač pre posielanie paketov danou LSP cestou

Vytvorenie LSP cesty

Pre vytvorenie LSP cesty vyberieme v objektovej palete s názvom **MPLS** model cesty s názvom **MPLS_E-LSP_STATIC**. Na pracovnej ploche projektu začneme vytvárať požadovanú LSP cestu. Cestu začneme kliknutím na okrajový

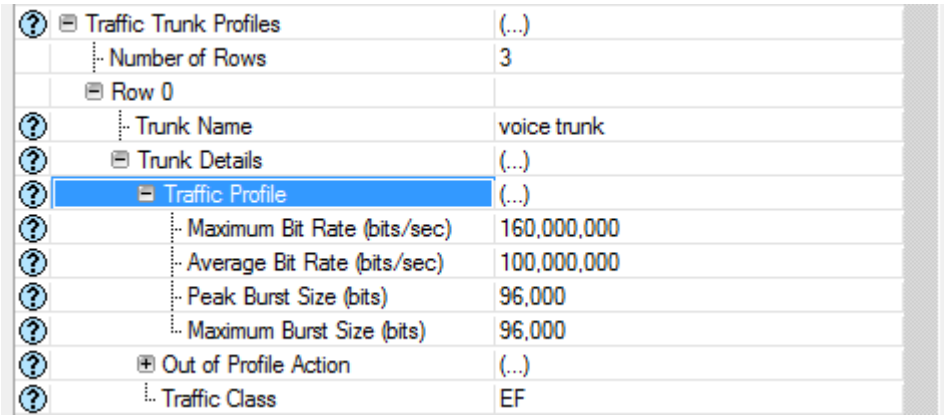
LER smerovač. Pokračujeme označovaním smerovačov po trase, až skončíme označením LER smerovača na okraji sieti. Ukončenie vytvárania cesty uskutočníme pravým kliknutím na pozadie a vybraním voľby **Finish path definition**. Teraz máme možnosť začať vytvárať ďalšiu LSP cestu rovnakým spôsobom alebo ukončiť opätovným kliknutím na pozadie vybrať voľbu **Abort path definition**. Po ukončení vytvárania LSP ciet vyberieme v hlavnom menu *Protocols > MPLS >Update LSP Details*, aby sa vytvorili definície práve špecifikovaných ciest.

Vytvorenie FEC

Na vytvorenie FEC záznamu otvoríme dialógové okno pre editáciu atribútov objektu MPLS Config. Dvojklikom na hodnotu pri zázname **FEC Specification**. Zvolíme počet riadkov, jeden riadok jeden FEC záznam. Pre každý záznam zvolíme názov. V špecifikácii detailov zvolíme podmienky, podľa ktorých sa budú posudzovať prichádzajúce pakety. V našom prípade to bude parameter ToS, teda typ služby a cieľová adresa. Pre dátovú prevádzku zvolíme *Best Effort*, pre hlasovú prevádzku *Interactive Voice* a pre video prevádzku *Streaming Multimedia*. Cieľovú adresu sme nastavovali iba pre video adátovú prevádzku. Pre video prevádzku z toho dôvodu, že sa z jedného zdroja vytvárajú viaceré toky s rôznymi cieľovými uzlami. Pri dátovej prevádzke, kvôli prevádzke na pozadí.

Vytvorenie Traffic Trunk

Na vytvorenie Traffic Trunk záznamu otvoríme dialógové okno pre editáciu atribútov objektu MPLS Config. Na hodnote pri položke **Traffic Trunk Profiles** vyberieme voľbu *Edit*. Otvorí sa nám tabuľka. Pre našu simuláciu vytvoríme tri záznamy, pre každý typ prevádzky jeden. Názvy zvolíme tak aby korešpondovali s typom prevádzky. Podstrom atribútu **Traffic Profile** nastavíme podľa obrázku 25. V atribúte **Out of Profile Action** nastavíme hodnotu *Transmit Unchanged*.

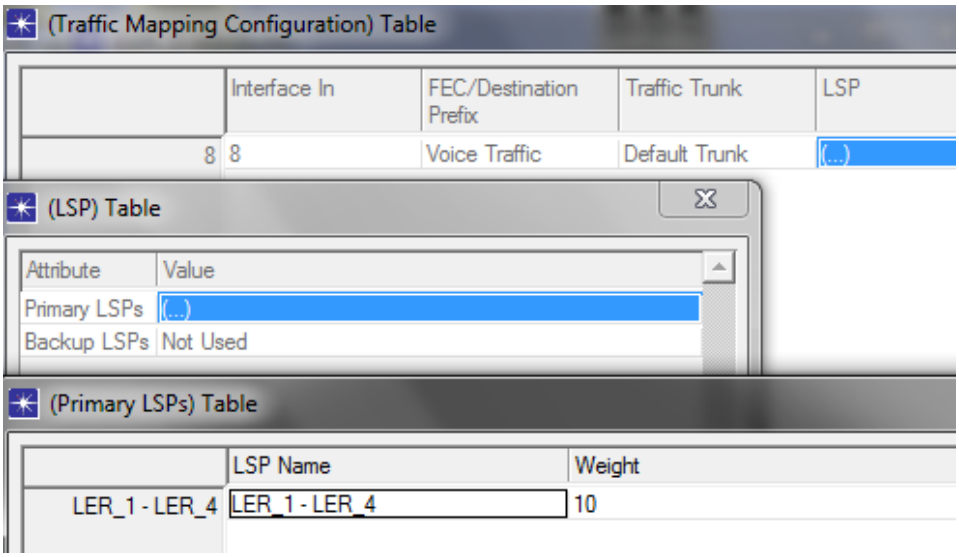


Obrázok 25. Nastavenie Trunk Details.

Vytvorenie TE na okrajovom smerovači

Po tom ako sme špecifikovali LSP cestu, FEC a Traffic Trunk môžeme vytvoriť TE záznamy, ktoré budú určovať, ktoré pakety budú posielané cez danú LSP. Urobíme tak v dialógovom okne editoru atribútov. Tu prejdeme na položku **MPLS>MPLS parameters>Traffic Mapping Configuration.** Pridáme riadok a nastavíme nasledovné atribúty:

- **Interface In** – zvolíme vstupné rozhranie, z ktorého pakety sa budú kontrolovať
- **FEC/Destination Prefix** – vyberieme jeden z našich konfigurovaných FEC záznamov
- **Traffic Truk** – vyberieme nami definovaný záznam pre Traffic Trunk
- **LSP** – pridáme LSP cestu definovanú na danom smerovači



Obrázok 26. Definovanie TE záznamu.

10 Simulácia siete

Táto kapitola bude venovaná nielen simulácii ale aj krokom potrebným pred vykonaním samotnej simulácie.

10.1 Nastavenie sledovania výsledkov

Na to aby sme po ukončení simulácie mali nejaké výsledné hodnoty je potrebné pred spustením simulácie vybrať, ktoré z parametrov sa budú zaznamenávať. Na výber máme z rôznorodých štatistík v závislosti od typu modelu.

Základné delenie štatistík je nasledovné:

- Global Statistics – štatistiky zaznamenávané z celej siete
- Node Statistics – štatistiky zaznamenávané len na uzloch
- Link Statistics – štatistiky zaznamenávané len na fyzických linkách
- Path statistics – štatistiky zaznamenávané na logických trasách

Jeden typ štatistiky môže byť zaznamenávaný pre všetky modely toho istého typu (uzly, linky, cesty) alebo môžeme danú štatistiku zaznamenávať iba na jednom konkrétnom modely.

Ak teda chceme napríklad niektorú zo štatistík zaznamenávať pre všetky uzly v sieti, klikneme na pozadie plochy pravým tlačidlom a vyberieme voľbu **Choose Individual DES Statistics**. Následne vstúpime do stromu **Node Statistic** a zaškrtneme požadovanú štatistiku. Ak štatistika obsahuje podstrom sú vybraté všetky štatistiky podstromu. Keď nemáme záujem zaznamenávať všetky štatistiky, je potrebné preklikať sa až na najnižšiu úroveň stromu a vybrať konkrétne štatistiky.

Pre záznam štatistiky iba na konkrétnom uzle, povedzme na smerovači, klikneme naň pravým tlačidlom a vyberieme voľbu **Choose Individual DES Statistics**. Ďalej pokračujeme ako pri výbere štatistík pre všetky uzly. Tu však budeme mať k dispozícii iba tie štatistiky, ktoré je na danom uzle možné zaznamenávať.

10.2 Nastavenie simulácie

Pred spustením simulácie je potrebné nastaviť niekoľko základných hodnôt. Z hlavného menu vyberieme voľbu *DES>Configure/Run Discrete Event Simulation...*.Okno, ktoré sa nám otvorilo umožňuje nastaviť nasledovné hodnoty:

- Duration – určuje dĺžku trvania simulácie. Na výber máme z nasledovných časových intervalov: týždne, dni, hodiny, minúty, sekundy.
- Seed – špecifikuje hodnotu násady pre generátory. Keďže sa jedná o generátory pseudonáhodných čísel, pri opätovnom spustení simulácie s rovnakou násadou budú generované rovnaké hodnoty. Vhodné pre porovnanie výsledkov pri vykovaní zmien na modely siete.
- Values per statistic – nastavuje maximálny počet hodnôt zaznamenaných pre každú štatistiku.
- Update Interval – jedná sa interval aktualizácie priebehu simulácie. Interval sa udáva v počte odsimulovaných udalostí.

Nastavenia týchto parametrov v našej simulácii je nasledovné: dĺžka trvania simulácie 1 hodina, hodnota násady pre generátor 27, počet hodnôt zaznamenaných na jednu štatistiku 600, interval aktualizácie priebehu simulácie 10000 udalostí.

Nastavené hodnoty potvrdíme a uložíme kliknutím na tlačidlo *Apply*.

10.3 Priebeh simulácie

Po nastavení potrebných parametrov môžeme pristúpiť k samotnej simulácii. Z hlavného menu vyberieme voľbu *DES>Configure/Run Discrete Event Simulation...*. Tu si môžeme opätovne skontrolovať nastavené hodnoty pre simuláciu. Pokračujeme kliknutím na tlačidlo *Run*. Tým sa spustí samotná simulácia.

Počas simulácie vidíme ubiehajúci simulačný čas. Taktiež môžeme pozorovať rýchlosť simulácie, či už v číselnom alebo grafickom prevedení.

Obrázok 27. Priebeh simulácie.

10.4 Výsledky simulácie

V tejto časti si zhrnieme výsledky získané simuláciou. Cieľom simulácie bolo vyhodnotiť oneskorenia a straty pre jednotlivé typy prevádzky a pre jednotlivé uzly v sieti.

Nasledujúca tabuľka 1 zobrazuje výsledné štatistiky pre jednotlivé relácie:

Relácia	Odoslané pakety	Prijaté pakety	Straty v %	Stredné oneskorenie (s)
Video 1	2289673	2286686	0,13%	0.048
Video 2	2287423	2286827	0,03%	0.107
Video 3	2292487	2283624	0,39%	0.418
Hlas 1	172245	171461	0,46%	0.216
Hlas 2	171583	171487	0,06%	0.021
Dáta 1	1716064	1711251	0,28%	1.256
Dáta 2	1719553	1718965	0,03%	1.805

Tabuľka 1. Výsledky simulácie pre jednotlivé relácie.

Z výsledkov môžeme vyčítať, že straty paketov a oneskorenie pre rovnaké typy relácií sa líšia. Je to spôsobené tým, že každá relácia prechádzala inou časťou siete s odlišným zaťažením.

Tabuľka 2 obsahuje informácie o prijatých dátach od koncových zariadení a množstve paketov zahodených pred odoslaním do siete.

Uzol	Priorita	Počet vstúpených paketov	Počet zahodených paketov	Straty (%)	Stredné oneskorenie (s)
1	1	N/A	N/A	N/A	N/A
	2	25789823	0	0,00%	0,026
	3	31528714	17238490	54,68%	0,193
2	1	14277909	0	0,00%	0,004
	2	N/A	N/A	N/A	N/A
	3	59488206	9221159	15,50%	0,118
3	1	N/A	N/A	N/A	N/A
	2	N/A	N/A	N/A	N/A
	3	31039718	1935699	6,24%	0,086
4	1	N/A	N/A	N/A	N/A
	2	N/A	N/A	N/A	N/A
	3	25996366	0	0,00%	0,0002
5	1	N/A	N/A	N/A	N/A
	2	11676320	0	0,00%	0
	3	23501870	0	0,00%	0
6	1	N/A	N/A	N/A	N/A
	2	N/A	N/A	N/A	N/A
	3	28339761	0	0,00%	0,007

Tabuľka 2. Výsledky simulácie pre okrajové smerovače.

Tu môžeme vidieť že kvôli uprednostňovaniu tokov s vyššou prioritou sú najviac zahadzované pakety dátovej prevádzky typu Best Effort.

Poznámka: Štatistiky boli zaznamenávané iba pre toky relácií. Toky na pozadí boli nastavené a parametrom *All Background*, čo znamená, že dané toky sa nesimulujú explicitne a nie sú pre ne zaznamenávané žiadne hodnoty. Toto nastavenie bolo realizované pre zrýchlenie simulácie. Preto sa v tabuľke 2 nachádzajú nevyplnené miesta (N/A). Keďže nie všetky typy meraných relácií vstupovali do siete v každom okrajovom smerovači, simulačný nástroj nezaznamenal pre daný typ toku žiadne štatistiky.

Záver

Cieľom diplomovej práce bolo vytvoriť simulačný model siete Metro Ethernet s MPLS chrbticovou sieťou v OPNET Modeler a vyhodnotiť štatistiky o oneskoreniach a stratách v sieti vzhľadom na Triple-play služby (hlas, video, internet).

Zamerali sme sa na popis jednotlivých mechanizmov, ktoré sú pri zabezpečovaní kvality služby v súčasnosti používané. Niektoré z mechanizmov sú implementované v simulačnom modeli siete.

Ďalej práca pojednáva o tom ako vytvoriť simulačný model v nástroji OPNET Modeler. Jednotlivé state obsahujú potrebné informácie o použitých modeloch uzlov, liniek a konfiguračných objektov. Postupne približujú spôsob tvorby modelu siete v krokoch tak, aby bolo možné pracovať s modelom aj v budúcnosti a meniť jeho konfiguráciu, prípadne nadviazať na súčasný model a rozširovať ho podľa potrieb.

V práci sme sa zamerali pri vytváraní simulačného modelu práve na technológiu MPLS., ktorá je v OPNET Modeler implementovaná. V modely je použitá hlavne ako nástroj pre správu tokov v chrbticovej sieti.

S konkrétnymi špecifikáciami by bolo možné vytvorený simulačný model siete použiť pri návrhoch reálnych sietí.

Bibliografia

GABAUER, Martin. 2007. *Príspevok k experimentálnemu vyhodnocovaniu parametrov QoS v MPLS sieťach.* Košice : Technická univerzita v Košiciach, 2007. Diplomová práca.

Jančo, Juraj. 2008. *Vytvoriť simulačný model siete Metro Ethernet.* Žilina : Žilinská univerzita v Žiline, 2008. Diplomová práca.

Kavička, Klima a Adamko. 2005. *Agentovo orientovaná simulácia dopravných uzlov.* s.l. : EDIS, 2005. ISBN 80-8070-477-5.

OPNET Technologies, Inc. 2008. *OPNET Modeler Documentation Set.* 2008.

Zeman, Otto. 2006. *Modelování chování páteřních směrovačů DiffServ domény.* Brno : Vysoké učení v Brne, 2006. Bakalárska práca.

Prílohy

CD príloha obsahujúca vytvorený simulačný model v OPNET Modeler.