Universitatea “Politehnica” din Bucureşti Facultatea de Electronică,
Telecomunicaţii şi Tehnologia Informaţiei

Optimizarea traficului de mare viteză între reţele de calculatoare

Proiect de diplomă

prezentat ca cerinţă parţială pentru obţinerea titlului de Inginer în domeniul Calculatoare şi Tehnologia Informaţiei programul de studii de licenţă Ingineria Informaţiei

Conducător ştiinţific
Conf. dr. ing. Ştefan STĂNCESCU

Absolvent
Gabriel GRĂDINARU-TAŞCĂU

2013

Universitatea "Politehnica" din Bucureşti
Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei
Departamentul Electronică Aplicată şi Ingineria Informaţiei

Aprobat Director de Departament :
Prof. Dr.Ing. Sever Paşca

TEMA PROIECTULUI DE DIPLOMĂ
a studentului
Grădinaru-Taşcău I. Gabriel, grupa 442A

1. Titlul temei: Optimizarea traficului de mare viteză între reţele de calculatoare

2. Contribuţia practică, originală a studentului va consta în: comparaţie între mijloace de gestiune a traficului folosind simulatoare de reţea: NS (Network Simulator), Cisco Packet Tracer, OPNET, etc.; analiza unui mecanism de dirijare de trafic ATM şi a unei arhitecturi ATM; optimizarea acestei arhitecturi şi simularea funcţionării acesteia; justificarea avantajelor soluţiei găsite; formularea de concluzii.

3. Proiectul se bazează pe cunoştinţe dobîndite în principal la următoarele discipline: Arhitecturi de Reţele şi Internet, Reţele de Calculatoare, Sisteme de Comunicaţii

4. Realizarea practică/ proiectul rămân în proprietatea: Grădinaru-Taşcău Gabriel şi *UPB, Faculatatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei*

5. Proprietatea intelectuală asupra proiectului aparţine: Grădinaru-Taşcău Gabriel şi *UPB, Faculatatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei*

6. Locul de desfăşurare a activităţii: *UPB, Faculatatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei*

7. Data eliberării temei: 15.11.2012

CONDUCĂTOR LUCRARE:

Conf. Dr. Ing. Ştefan Stăncescu

STUDENT:

Gabriel Grădinaru-Taşcău

Declaraţie de onestitate academică

Prin prezenta declar că lucrarea cu titlul “ Studiul şi compararea unor simulatoare de reţea diverse şi combaterea congestiei cu ajutorul unui algoritm cunoscut”, prezentată în cadrul Facultăţii de Electronică, Telecomunicaţii şi Tehnologia Informaţiei a Universităţii “Politehnica” din Bucureşti ca cerinţă parţială pentru obţinerea titlului de *Inginer* în domeniul *Calculatoare şi Tehnologia Informaţiei*, programul de studii *Ingineria Informaţiei* este scrisă de mine şi nu a mai fost prezentată niciodată la o facultate sau instituţie de învăţământ superior din ţară sau străinătate.

Declar că toate sursele utilizate, inclusiv cele de pe Internet, sunt indicate în lucrare, ca referinţe bibliografice. Fragmentele de text din alte surse, reproduse exact, chiar şi în traducere proprie din altă limbă, sunt scrise între ghilimele şi fac referinţă la sursă. Reformularea în cuvinte proprii a textelor scrise de către alţi autori face referinţă la sursă. Înţeleg că plagiatul constituie infracţiune şi se sancţionează conform legilor în vigoare.

Declar că toate rezultatele simulărilor, experimentelor şi măsurătorilor pe care le prezint ca fiind făcute de mine, precum şi metodele prin care au fost obţinute, sunt reale şi provin din respectivele simulări, experimente şi măsurători. Înţeleg că falsificarea datelor şi rezultatelor constituie fraudă şi se sancţionează conform regulamentelor în vigoare.

Bucureşti, *15.12.1012*

Absolvent Gabriel Grădinaru-Taşcău

Cuprins

Lista acronimelor

WAN	Wide Area Network
LAN	Local Area Network
TCP	Transmission Control Protocol
ISP	Internet Service Provider
QoS	Quality of Service
IP	Internet Protocol
SSH	Secure Shell
DNS	Domain Name System
ATM	Asynchronous Transfer Mode
VCI	Virtual Channel Identifier
VPI	Virtual Path Identifier
CRC	Cyclic Redunduncy Check
MPLS	Multiprotocol Label Switching
LSP	Label Switched Path
LER	Label Edge Router
LSR	Label Switching Router
FEC	Forward Equivalent Class
LDP	Label Distribution Protocol
VPN	Virtual Private Network
IGP	Interior Gateway Protocol

Lista figurilor

Capitolul 1

1. Introducere

1.1 WAN (Wide Area Network)

Când vorbim despre trafic de mare viteză între reţele de calculatoare vorbim despre trafic în reţele de tip WAN (Wide Area Network).

O reţea WAN acoperă o arie geografică întinsă, o ţară sau un întreg continent. Astfel de reţele comunică între ele prin Internet sau prin alte mijloace realizate de către companii de telefonie sau alţi furnizori de servicii. Reţeaua WAN conţine o colecţie de maşini utilizate pentru a executa programele utilizatorilor sau gazde conectate între ele printr-o subreţea de comunicaţie. De cele mai multe ori, subreţeaua este compusă din linii de transmisie şi elemente de comutare a mesajelor. Liniile de transmisie pot fi alcătuite din fire de cupru, fibră optică sau legături radio şi au rolul de a transporta biţii de date între gazde. Elementele de comutare sunt calculatoare specializate care conectează două sau mai multe linii de transmisie. Elementul de comutare are următorul rol: când primeşte date pe o anumită linie, trebuie să aleagă o nouă linie pentru a retransmite datele mai departe. De multe ori elementul de comutare se numeşte ruter (sau router).

În cele mai multe WAN-uri, reţeaua conţine un număr mare de linii de transmisie, fiecare dintre ele legând o pereche de rutere. Dacă două rutere nu sunt conectate direct prin acelaşi cablu, ele pot comunica, dar indirect, prin intermediul altor rutere. Când un pachet este transmis de la un ruter la altul prin intermediul unuia sau mai multor rutere, pachetul este primit de fiecare ruter intermediar, păstrat până când linia de ieşire dorită este liberă şi apoi este retransmis. O subreţea care funcţionează pe acest principiu se numeşte subreţea cu comutare de pachete. Aproape toate reţelele WAN au subreţele cu comutare de pachete. Atunci când pachetele sunt de dimensiune mică şi au toate aceeaşi mărime, ele sunt adesea numite celule (în cazul ATM).

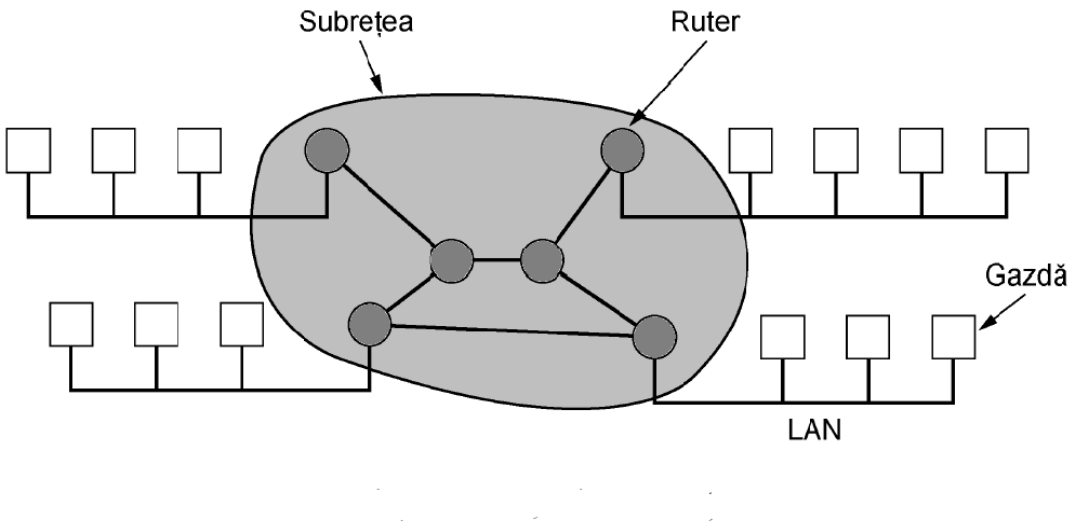

Figura 1.1: Relaţia dintre gazde şi subreţea[1]

În general, atunci când un proces al unei gazde vrea să transmită un mesaj către un proces de pe o altă gazdă, gazda care transmite va împărţi mesajul în pachete, fiecare dintre ele reţinându-şi numărul de ordine din secvenţă. Aceste pachete sunt apoi transmise prin reţea unul câte unul într-o succesiune rapidă şi când ajung la gazda receptoare, aceasta le depozitează după care sunt reasamblate în mesajul iniţial şi furnizate procesului receptor.

Există şi WAN-urile care nu sunt cu comutare de pachete. Poate exista un WAN format dintr-un sistem de sateliţi. Fiecare ruter are o antenă prin care poate trimite şi primi date. Toate ruterele pot asculta ieşirea altui satelit, iar în unele cazuri pot să asculte chiar şi transmisia celorlalte rutere către satelit. Câteodată, ruterele sunt conectate la o reţea punct-la-punct şi numai unele dintre ele deţin antene de satelit. De obicei, reţelele satelit sunt reţele cu difuzare şi sunt foarte utile când proprietatea de difuzare este importantă.

Capitolul 2

2. Tehnici de optimizare WAN

2.1 Introducere

Optimizarea WAN este o colecţie de tehnici destinate a creşte eficienţa transferurilor de date peste reţele WAN. Cele mai uzuale criterii de măsura a eficienţei transferurilor TCP sunt rata de transfer, lăţimea de bandă, latenţa, optimizarea protocoalelor şi congestia, care se manifestă prin pierderea de pachete. De asemenea, şi WAN-ul se poate clasifica dupa distanţa dintre nodurile terminale şi cantitatea de date transferate.

Companiile comerciale folosesc doua topologii de reţele, prima este de tipul sediu şi agenţie iar cea de-a doua este de tipul centru de date către centru de date. În general, legăturile agenţiilor cu WAN-ul sunt la distanţa mică, folosesc mai puţină lăţime de bandă, suportă mai multe conexiuni simultan, suportă conexiuni mai mici şi de mai scurtă durată şi sunt traversate de o mare varietate de protocoale. Sunt folosite pentru aplicaţii de afaceri cum ar fi poşta electronică, sisteme de gestionare a conţinutului, aplicaţii pentru baze de date şi acces către internet. În comparaţie, legăturile centrelor de date cu WAN au tendinţa să folosească mai multă lăţime de bandă, sunt la distanţe mai mari, şi au mai puţine conexiuni dar mai mari (flux-uri de 100Mb/s pana la 1Gb/s) şi de durata mare. Traficul între centre de date poate include replicare de baze de date, back up, migrări de date, virtualizare şi alte flux-uri specifice bazelor de date.

Optimizarea WAN a fost subiectul de cercetare academică intensă de când a apărut WAN-ul. La început s-au concentrat eforturile pe optimizarea conexiunilor WAN-ului cu agenţiile, dar mai recent, creşterea rapidă a datelor digitale şi nevoia de a le stoca şi proteja, au stat la baza nevoii de optimizare a legăturilor centrelor de date cu WAN-ul.

2.2 Scăderea lăţimii de bandă

Tratarea simptomelor şi nu a cauzelor este o soluţie comună, dar de cele mai multe ori greşită pentru multe probleme, inclusiv utilizarea lăţimii de bandă. Rezultatele pot părea pozitive la început dar de obicei, de scurtă durată, neprevizibile şi nu scalează bine. Scăderea lăţimii de bandă reflectă această abordare de tratare a simptomelor (transferuri încete şi timpi de răspuns excesivi) decât cauzele (puţină lăţime de bandă diponibilă,

aplicaţii neautorizate, legături către resurse necorespunzătoare etc.). Totuşi, această tehnică încă se foloseşte şi încă destul de mult.

Scăderea lăţimii de bandă asigură faptul că dispozitive care folosesc multă lăţime de bandă, precum routerele sau gateway-urile, limitează cantităţile de date transmise şi recepţionate pe anumite perioade de timp. Scăderea lăţimii de bandă limitează congestia reţelei şi reduce instabilitatea server-ului rezultată din saturaţia reţelei. Pentru furnizorul de internet (Internet Service Provider - ISP), scăderea lăţimii de bandă restricţioneaza viteza utilizatorului pentru anumite aplicaţii sau în timpul perioadelor de folosire la maxim. Dar fără a înţelege cauzele creşterii traficului care determină scăderea lăţimii de bandă, este doar o chestiune de timp până când această creştere va reveni şi iar se va scădea lăţimea de bandă.

Serverele care conţin date funcţionează pe un principiu simplu de cerere şi ofertă: clienţii fac cereri şi serverul le răspunde. Dar serverele care furnizează date în internet sunt predispuse supraîncărcării în timpul orelor de vârf şi trebuie să prelucreze foarte multe cereri. Aceste perioade de încărcare maximă produc congestie sau îngreunează conexiunea care poate cauza instabilitatea serverului şi o eventuală cădere a sistemului, rezultând un timp de inactivitate. Scăderea lăţimii de bandă este folosită ca o metodă preventivă de a controla nivelul de răspuns al serverului la creşterea cererilor din partea clienţilor în timpul orelor de vârf.

În februarie 2008, membrii Comisiei Federale de Comunicaţii (Federal Communications Commission) au anunţat că se consideră stabilirea unor regulamente pentru a descuraja furnizorii de internet să micşoreze lăţimea de bandă de la anumite site-uri şi servicii care sunt mari consumatoare de lăţime de bandă. Organizaţiile pot (şi ar trebui) să micşoreze lăţimea de bandă sau filtre firewall pentru a limita sau a bloca traficul care violează politica de uz acceptabil (Acceptable Use Policy). Dar în alte conditii, micşorarea lăţimii de bandă este aplicată cel mai bine sub forma clasei serviciului (Class of Service - CoS) sau calitatea serviciului (Quality of Service - QoS) aplicate anumitor tipuri de trafic. CoS şi QoS reprezintă scheme de clasificare a traficului reţelei care acordă prioritate traficului dependent de timp sau traficului critic pentru misiune (mission-critical) decât să limiteze un anumit tip de trafic. Mulţi experţi recomandă ca pentru protocoale neautorizate sau nedorite să le fie micşorata lăţimea de bandă la nivele foarte mici (sub 10Kb/s) decât să fie blocate complet, pentru a oferi administratorilor de reţea şansa de a le înlătura şi a rezolva cu utilizatorii sau programele implicate. De exemplu, limitând lăţimea de bandă disponibilă pentru protocoale peer-to-peer, precum BitTorrent(folosit pentru video şi alte descărcări media personale), la 5Kb/s sau 10Kb/s,

administratorii ar putea avea timp să identifice staţiile sau serverele care sunt puncte terminale pentru activităţi peer-to-peer şi să identifice indivizii care le utilizează. Pot să consilieze sau să disciplineze utilizatorii care sunt politicile de uz acceptabil.

2.3 Repararea protocoalelor LAN care sunt nepotrivite pentru WAN

Multe protocoale care menţin funcţii vitale interne şi externe ale unei afaceri funcţionează cu scop limitat sau scalabilitate limitata. Acestea sunt, de cele mai multe ori, protocoale LAN învechite care pot impune o mare incetinire asupra reţelei WAN. De exemplu, sistemul de fişiere comune pe internet (Common Internet Files System - CIFS) creşte exponenţial în timp ce este transportat printr-un WAN. De asemenea, fluxurile de date în timp real şi protocoalele de voce introduc nevoia pentru modelarea traficului. Pentru aceasta, traficul în reţea este controlat pentru a optimiza performanţa, a satisface garanţiile de performanţa, sau pentru a creşte lăţimea de bandă utilizabilă, de obicei prin întârzierea pachetelor care pot fi categorizate ca relativ insensibile la întârzieri sau care îndeplinesc anumite criterii de clasificare care marchează acest trafic cu prioritate scăzută. Modelarea traficului este folosită pentru a oferi utilizatorilor servicii voce şi video satisfăcătoare, în timp ce se menţine reţeaua în mod obişnuit pentru celelalte protocoale.

Urmărind aceeaşi idee, criptarea şi aplicaţii de accelerarea a protocoalelor de securitate devin mari consumatoare de resurse dar sunt poveri necesare, în special când traficul sensibil trebuie să traverseze prin internet. Până şi cel mai folosit protocol din internet, HTTP-ul, poate fi descris ca având comunicaţii frecvente şi constante şi în acelaşi timp având perioade în care sunt cereri de resurse excesive.

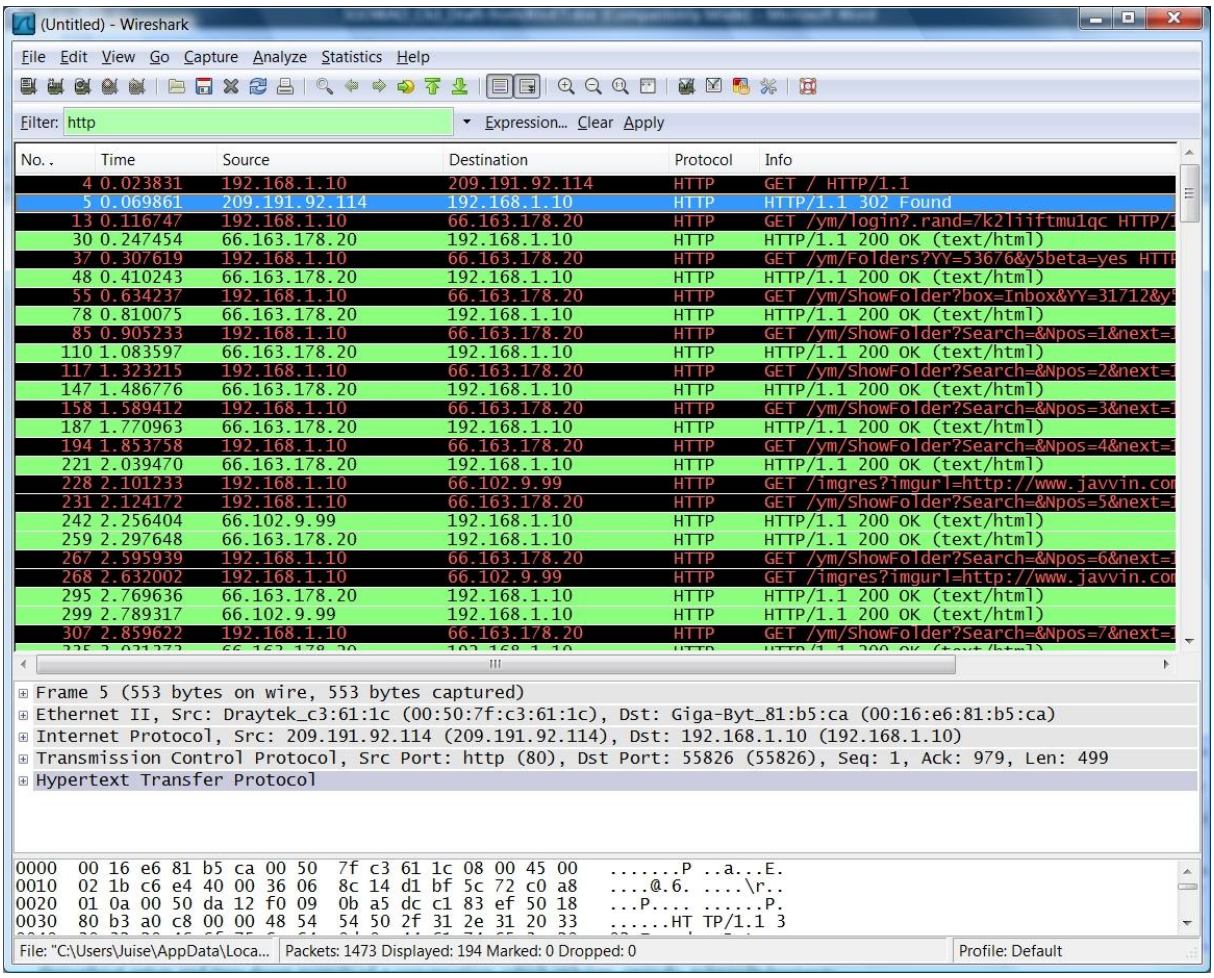

Figura 2.1: Captură de pachete pentru afişarea unei pagini web

Repararea acestor aspecte ale mediului reţelei devine o propunere de soluţionare a traficului care ia în considerare nu doar aplicaţiile în sine dar şi programarea aplicaţiilor în general. Ştiind cum funcţioneaza o aplicaţie, formatul şi parametrii protocoalelor sale şi comportamentul în timpul rulării este esenţial pentru înţelegerea integrării sale cu alte aplicaţii, servicii şi protocoale din reţea. Nu este doar o propunere de reparare care presupune rezolvarea componentelor individuale, dar care cere folosirea celor mai bune practici pentru comunicaţii WAN eficiente: transmite şi recepţioneaza rar, în cantităţi mari, şi sub forma unor tranzacţii complete pe cât posibil.

Formatul TCP a fost proiectat iniţial să funcţioneze in mod fiabil peste medii de transmisie nesigure, fără a depinde de ratele de transmisie, întârzieri, coruperea protocolului, date duplicate şi reordonarea segmentelor. Din cauza aceasta, TCP este întradevăr un mecanism robust şi fiabil pentru transportul aplicaţiilor, serviciilor şi protocoalelor. Dar acest avantaj de design scoate la iveală dezavantajul TCP când este folosit peste o reţea modernă, medii de transmisie de mare viteză care întrec cu mult condiţiile sub care TCP a fost proiectat să funcţioneze.

Protocolul IP trimite pachete fără să se asigure dacă au ajuns la destinaţie; TCP menţine conexiunile în desfăşurare de la un capăt la altul, de la începutul transmisiei până la terminarea acesteia, are nevoie şi de confirmări periodice că au fost transmise datele. Datele care nu au fost confirmate determină un algoritm de back-off exponenţial după care se reîncearcă transmisia până când este recepţionata şi confirmată, sau trece o anumită perioadă de timp şi semnalizează căderea conexiunii. Există un termen, sliding TCP window sizes, care semnifică numărul de pachete care poate fi trimis până când se recepţionează confirmarea primirii pachetelor şi care influenţeaza în mod direct performanţa când valori mai mari determină rate de transfer mai mari(dar şi potenţiale întârzieri mai mari). TCP foloseşte un algoritm bine definit de început lent care iniţiază comunicaţia cu window size-ul foarte mic, după care scalează acest număr la proporţii optime în timp ce conexiuni sunt stabilite şi menţinute active. Fiecare dintre acestea şi alte proceduri asemănătoare ale stivei TCP/IP introduc întârzieri în reţea adresate în soluţii de optimizare WAN prin tehnici de optimizare a conexiunii.

2.4 Folosirea compresiei pentru a reduce lăţimea de bandă

Ca rezultat al abilităţii lor de a reduce volumul de trafic, mai multe scheme de compresie destinate îmbunătăţirii performanţei au fost o parte integrala din comunicaţiile între reţele încă de pe vremea protocolului de transfer de fişiere ZMODEM introdus in 1986. Chiar şi după 20 şi de ani compresia apare în mod regulat în reţele moderne cu protocoale moderne precum Secure Shell (SSH) şi compresia octeţilor din fluxurile de date în produse de optimizare WAN contemporane.

În termeni de telecomunicaţii, compresia lăţimii de bandă înseamnă: o reducere a nevoii de lăţime de bandă necesară pentru a transmite o anumită cantitate de date într-un anumit timp; sau o reducere în timpul necesar transmiterii unei anumite cantităţi de date cu o anumita lăţime de bandă disponibilă. Aceasta implică o reducere în lăţimea de bandă normală (sau timpul) pentru semnalele purtătoare de informaţii fără a reduce conţinutul informaţiei, datorită folosirii adecvate a tehnicilor de compresie de date. Acestea sunt bune şi frumoase într-un mediu WAN, dar pot fi ineficiente fără acces către hardware de accelerare a compresiei pentru a obţine compresie şi decompresie maximă cu întârzieri minime (cu toate că software-ul este rapid, hardware-ul este, de obicei, cu câteva ordine de mărime mai rapid). În general, dispozitivele de accelerare WAN includ simboluri şi dicţionare pentru a reduce volumul datelor chiar mai mult decat poate compresia.

Redundanţa repetitivă, care descrie modurile în care şabloanele şi elementele tind să se repete în fluxurile de trafic regulat între perechi de transmiţători şi receptori, rămâne motivul esenţial pentru care schemele de compresie sunt valabile şi eficiente. Dintr-o perspectivă analitică corectă, multe date care traversează reţeaua includ repetiţie inutilă, care face risipă de biţi, timp şi lăţime de bandă necesară pentru transportul lor. Compresia datelor prin toate mijloacele posibile ajută la restaurarea echilibrului reţelei şi este doar una dintre mai multe metode împotriva întârzierii nedorite.

Mai multe tehnici de compresie sunt potrivite pentru mediul reţelelor, dar toate se străduiesc să reducă lăţimea de bandă consumată în timp ce traversează WAN-ul. Tehnici de compresie ale antetului şi încărcăturii folosesc algoritmi de asemănare a şabloanelor pentru a identifica serii de octeţi scurte, frecvente şi repetitive din reţea care sunt înlocuite cu segmente mai scurte de cod pentru a reduce dimensiunea finală de transmisie. Algoritmi simplificaţi identifică şabloane de octeţi din pachetele individuale pe când formele sofisticate de compresie pot analiza şabloane pe mai multe pachete şi fluxuri de date.

Orice câştig oferit de strategiile de compresie trebuie să varieze conform tipului de trafic din reţeaua WAN. Arhivele comprimate de date (fişiere ZIP sau tar) nu pot fi reduse prea mult folosind scheme de compresie specifice reţelelor, dar aplicarea compresiei asupra diverselor fluxuri de date poate sa mărească lăţimea de bandă efectivă din WAN. Protocoalele de voce au parte de îmbunătăţiri semnificative de pe urma compresiei antetului UDP în combinaţie cu alte tehnici precum combinarea pachetelor, descrisă în secţiunea următoare.

2.5 Suprapunerile redundante aduc fişierele mai aproape de utilizatori

Redundanţa nu este întotdeauna un lucru rău pentru performanţa reţelelor sau pentru ratele de transfer. Există multe aplicaţii unde multiplicitatea poate îmbunătăţi performanţa. De fapt mediile WAN au mari beneficii ca urmare a suprapunerilor redundante de servicii de fişiere care aduc datele şi fişierele mai aproape de utilizatori.

Aşa cum ne-a aratat sistemul DNS, o reţea poate funcţiona eficient în moduri consolidate şi descentralizate în acelaşi timp. Un bloc uniform de informaţie poate fi consolidat de la mai multe surse şi distribuit printr-un sistem complet descentralizat pentru transmitere oriunde există o reţea. Consolidarea datelor agregate într-un singur centru de date creează beneficii pentru acei utilizatori care sunt apropiaţi de centrul de date, dar poate crea dificultăţi de accesare pentru alţi utilizatori care sunt mai departaţi, în special cei care trebuie să folosească conexiuni WAN înguste sau scumpe pentru a se conecta cu acel centru de date.

Apelând la o autoritate centrală sau la o sursă de informaţii pentru o companie dispersată pe tot globul pot apărea probleme pentru utilizatorii la distanţă sau care se conectează din alte reţele, deci este mai bine să reproducă informaţia în locuri de unde pot fi accesate rapid de către utilizatori, oriunde s-ar afla aceştia. Bazele de date DNS, cu versiunile sale cu autoritate mai mare sau mai mică, cu principiile sale master şi slave, împreună cu folosirea depozitelor cache a activităţii recente creează un model pentru distribuirea şi dispersarea informaţiei care funcţioneaza destul de bine încât menţin internetul global. În mod similar, suprapunerile redundante caută să păstreze fişierele pe care utilizatorii le accesează nu mai departe de una sau doua hop-uri WAN distantă de maşinile lor, indiferent unde s-ar afla şi la orice moment.

Astăzi, multe companii se străduiesc cu consolidarea şi multiplicarea serverelor. Numeroase servere locale trebuie să găzduiască o varietate de servicii şi protocoale şi în ciuda numarului lor ridicat, acestea nu sunt întotdeauna în locul potrivit şi la momentul potrivit. În schimb, multe companii preferă să instaleze câteva servere cheie în locaţii strategice geografice şi teritoriale pentru a acomoda mai uşor utilizatori şi echipe „în deplasare". Această abordare permite unui grup de servere sincronizate cu grijă să furnizeze accesibilitate, lăţime de bandă şi securitate comparabilă oricui şi oriunde. Reproducerea pe multiple servere fizice face această abordare posibilă, în timp ce virtualizare face posibilă rularea serviciilor în spaţii virtuale separate şi face intregul proces mai practic uşureaza întreţinerea şi monitorizarea.

Resursele reţelei şi utilizatorii sunt de multe ori dispersaţi pe mai multe agenţii, în locaţiile clienţilor şi pe mai mulţi furnizori de internet care se întind pe regiuni multiple. O strategie corectă de consolidare a serverelor centralizează, în mod necesar, infrastructura şi reduce numarul de servere pentru a reduce costul şi a îmbunătăţi organizarea. Din păcate, acest lucru creează nevoia de a asigura conectivitate neîntreruptă între locaţii la distanţă pe legaturi cu WAN-ul şi livrarea datelor eşuează atunci cand aceste legaturi cedează. Înseamnă că trebuie să existe un mecanism sau mai multe care să prelucreze traficul atunci când numărul utilizatorilor conectaţi la distanţă creşte.

În timpul orelor de deschidere, multe afaceri suferă de o creştere de trafic iniţial care este format, în mare parte, din utilizatorii care se înregistrează simultan pe unul sau mai multe servere. Autentificarea şi accesarea serviciilor DNS este problematică în timpul acestei rutine în care toată lumea apare şi se înregistrează în acelaşi timp, deci trebuie să existe un mod de a optimiza si prioritiza traficul astfel încât timpii de înregistrare să fie minimi.

O soluţie consolidată de optimizare WAN ajută la obţinerea celei mai bune utilizări a arhitecturii reţelei deoarece oferă o posibilitate de a profita la maxim de lăţimea de bandă pe care o are o legatură WAN, în timp ce se optimizează traficul cu prioritate astfel încât, chiar şi atunci când are loc o congestie, acest trafic să ajungă la destinaţie. Prin urmare, utilizatorul de dimineata ar putea astepta chiar şi un minut să descarce un e-mail, dar tot o să obţină un răspuns rapid atunci cand o să furnizeze un cont de utilizator şi o parola pentru a se înregistra.

Capitolul 3

3. Comparaţie simulatoare de reţea

3.1 Introducere

În domeniul reţelelor, stabilirea unui scenariu de utilizare în timp real este foarte dificil. Pentru a reproduce fizic o reţea cu echipamente reale necesită un cost foarte mare, timp şi bani. Aşadar implementarea unei reţele întregi nu este un lucru uşor de făcut. Un simulator de reţea ajută administratorul să verifice dacă reţeaua este capabilă să lucreze în timp real. Prin urmare, timpul şi costul testării funcţionalităţii reţelei este redus şi implementarea este facilitată. În următoarele pagini vom prezenta principalele caracteristici ale unor simulatoare de reţea şi vom analiza avantajele şi dezavantajele lor. Această analiză este utilă pentru a ajuta un proiectant sau administrator de reţea în alegerea tipului de retea potrivit nevoilor sale.

Simularea este una dintre cele mai importante tehnologii din era modernă. Simulările computerizate pot modela obiecte reale sau ipotetice pentru a fi studiate. De asemenea, o reţea de calculatoare poate fi simulată pe un computer.

Un simulator de reţea este o tehnică de implementare a reţelei pe un computer care permite cercetătorilor să testeze scenarii care sunt dificile sau costisitoare de implementat în lumea reală. Este util în special pentru a testa protocoale noi sau pentru a modifica protocoalele existente într-un mediu de control şi reproductibil. Se pot crea diferite tipologii folosind tipuri variate de noduri (staţii de lucru, switch-uri, routere, unităţi mobile, etc). Simulatoarele de reţea sunt folosite de persoane din diverse arii, cum ar fi cercetări academice, software development şi controlul calităţii pentru a proiecta, verifica şi analiza performanţa diverselor protocoale de reţea. În general, un simulator de reţea este format dintr –o varietate de tehnologii şi protocoale şi ajută utilizatorii să construiască reţele complexe din blocuri de bază cum ar fi clustere de noduri şi legături. De menţionat că simulatoarele de reţea nu sunt perfecte. Ele nu pot modela perfect toate detaliile reţelelor. Totuşi, daca sunt corect modelate pot oferi cercetătorului o perspectivă folositoare asupra reţelei testate şi cum modifică;rile pot afecta funcţionalitatea acesteia.

3.2 Simulare şi emulare

În domeniul cercetării reţelelor de calculatoare şi comunicaţii, simularea este o tehnică utilă deoarece comportamentul unei reţele poate fi modelat prin calcularea interacţiunii dintre componentele reţelei folosind formule matematice. O reţea mai poate fi modelată prin captarea şi reproducerea de observaţii experimentale din reţele reale aflate în producţie. Dupa ce obţinem datele dintr-un experiment de simulare, comportamentul reţelei şi protocoalele suportate pot fi observate şi analizate într-o serie de teste fără a mai avea nevoie de reţeaua reală. Pot fi modificate diverse atribute ale mediului într-un mod controlat pentru a vedea cum se comportă reţeaua în condiţii diferite sau cu diverse combinaţii de parametrii.

Emularea unei reţele înseamnă că o anumita reţea este simulată pentru a evalua performanţa ei sau pentru a prezice impactul unor posibile modificări sau optimizări. Diferenţa majoră între simulare şi emulare sunt sistemele terminale, precum computerele, pot fi ataşate emulatorului şi se vor comporta ca şi cum s-ar afla într-o reţea reală. Scopul unui emulator de reţea este să emuleze o reţea care interconectează staţii terminale, dar nu emulează şi aceste staţii terminale. Anumite aplicaţii de emulare includ NS2, care este în sine un simulator dar poate fi utilizat şi ca un emulator cu funcţionalitate limitată. În contrast, un emulator de reţea veritabil este WANsim, care emulează o reţea WAN şi care utilizează o parte din funcţionalitatea Linux.

3.3 Clasificarea simulatoarelor de reţea

Există diverse simulatoare de reţea cu diverse caracteristici cum ar fi: NS2, NS3, OPNET, NetSim, OMNet++, J-Sim, QualNet, Cisco Packet Tracer. Acestea se pot clasifica pe baza unor criterii precum preţul şi complexitatea.

Există simulatoare comerciale care nu pot fi folosite gratuit, utilizatorii trebuie să plătească pentru a achiziţiona o licenţa pentru întreg software-ul sau doar pentru anumite pachete specifice. Exemple: OPNET, QualNet. Avantajul acestor simulatoare este documentaţia completă şi actualizată care este menţinută de catre personal specializat dintr-o companie. În schimb, avantajul simulatoarelor gratuite şi open source este faptul că orice persoana sau organizaţie poate contribui la dezvoltare şi la găsirea bug-urilor. De asemenea interfaţa poate fi îmbunătăţită şi poate reflecta dezvoltări tehnologice recente mai rapid decât un simulator comercial. Exemple: NS2,NS3, OMNET++, J-Sim.

Simulatoarele de reţea existente sunt foarte variate începând de la unele foarte simple pana la cele foarte complexe. În mod minimal, un simulator ar trebui să permită

utilizatorilor să reprezinte o topologie de reţea, să definească scenariile, să specifice nodurile reţelei, legăturile dintre noduri şi traficul dintre noduri. Sistemele mai complicate pot permite utilizatorului să specifice toate aspectele protocoalelor folosite pentru a procesa traficul reţelei. De asemenea, aplicaţiile grafice permit vizualizarea funcţionalităţii mediului simulat. Unele pot fi bazate pe text şi furnizează o interfaţa mai puţin vizuală sau intuitivă, dar pot permite forme mai avansate de personalizare. Altele pot fi orientate pe programare şi pot oferi un framework care permite utilizatorilor crearea unei aplicaţii care sa simuleze mediul de testare.

Cel mai utilizat şi popular simulator de reţea este NS2. Acesta şi-a stabilizat poziţia ca fiind standardul simulatoarelor de reţea deoarece are foarte multe librării şi foarte mulţi au participat la dezvoltarea sa. Organizaţii non-profit au contribuit cu componente în librăriile sale şi s-a demonstrat că modul de dezvoltare al NS2 a fost un succes. Cu toate acestea, din cauza unor limitări de design, a fost nevoie să se dezvolte succesorul său, NS3, care pune accentul pe documentaţie şi nişte persoane foarte specializate s-au oferit voluntare pentru a se ocupa de diferite componente. Totuşi, NS3 nu este doar o versiune actualizată a NS2. Au fost reimplementate mai multe mecanisme bazate pe experienţe reuşite sau mai puţin reuşite ale NS2.

3.4 Analiza unor simulatoare de reţea

3.4.1 NS2(Network Simulator version 2)

NS2 este un simulator bazat pe evenimente discrete orientat către cercetare. Oferă suport pentru TCP, rutare şi protocoale multicast peste toate tipurile de reţele. A fost dezvoltat in 1995 în proiectul VINT (Virtual Inter Network Testbed).

Simulările NS2 sunt compuse din cod C++ care modelează comportamentul nodurilor şi scripturi oTcl care controlează simularea şi alte aspecte cum ar fi topologia retelei. Această alegere de design a fost facută pentru a evita recompilări inutile dacă se produc schimbări în configuraţia simulării. In anul 1996, cand a fost lansată prima versiune NS2, această alegere era rezonabilă deoarece recompilările frecvente C++ erau mari consumatoare de timp dar pentru standardele actuale, acest design al NS2 îl impiedica sa fie eficient în simularea reţelelor la o scară mare. Pentru a anima simulările este folosit Nam (Network Animator), un tool scris în Tcl/TK, care suportă animarea la

nivel de pachet, topologia reţelei si alte tool-uri de inspecţie a pachetelor. De obicei, datele de intrare pentru Nam sunt provenite din simulări NS2 sau pot proveni din măsurători dintr-o reţea reală, de exemplu din tcdump sau wireshark.

Cea mai recentă versiune de NS2 este NS-2.35 şi a fost lansată în Noiembrie 2011. Această versiune are peste 300.000 linii de cod şi foarte probabil încă pe atât cod contribuit de utilizatori care nu a fost integrat în versiunea principală. NS2 poate fi rulat pe următoarele platforme: Linux, FreeBSD, Solaris, Mac OS X şi Windows.

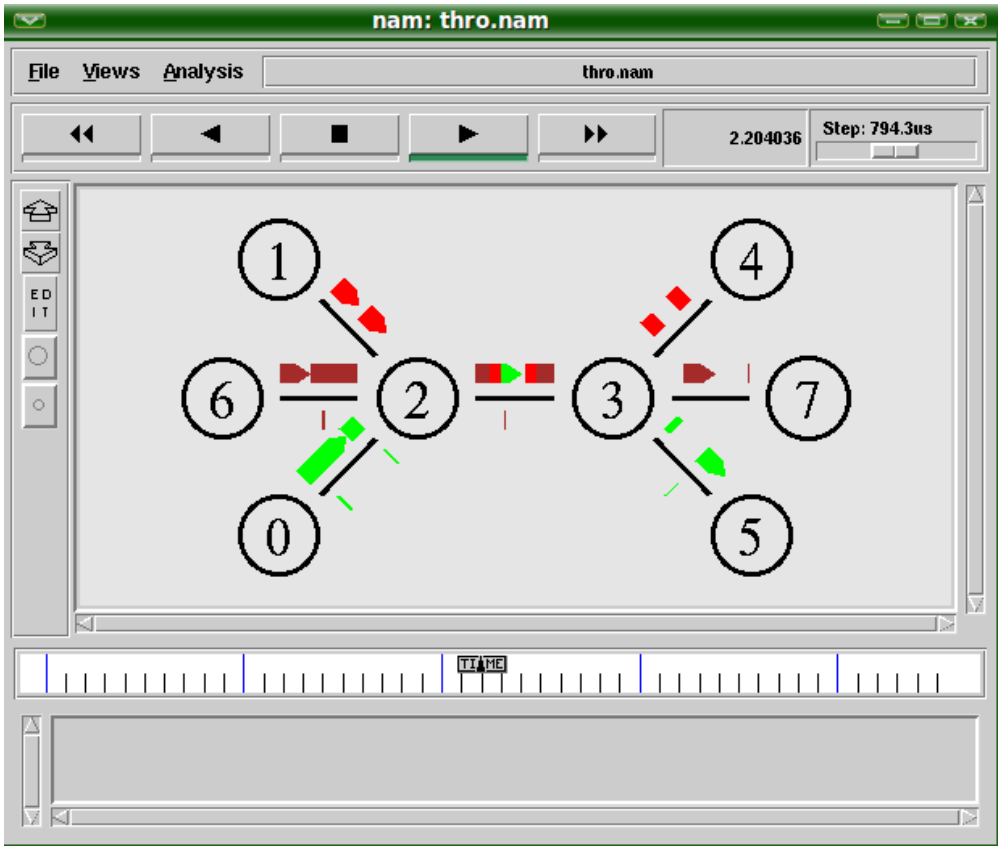

Figura 3.1: Simulatorul NS2

3.4.2 NS3 (Network Simulator version 3)

La fel ca predecesorul său, NS3 se bazează pe C++ pentru implementarea nodurilor simulării. Totusi, NS3 nu mai foloseşte scripturi oTcl pentru a controla simularea, astfel abandonând probleme introduse de combinaţia celor două limbaje, C++ şi oTcl din NS2. În schimb, simulările din NS3 pot fi implementate în limbaj pur C++, în timp ce unele părţi ale simulării pot fi realizate opţional folosind limbajul Python. În plus, NS3 integrează concepte arhitecturale şi cod din GTNetS, un simulator cu capabilităţi bune de scalabilitate. Aceste decizii de design au fost făcute cu preţul compatibilităţii, adică modelările din NS2 trebuie sa fie portate manual.

Pe lângă îmbunătăţirea performanţei, au mai fost adăugate caracteristici noi precum socket-uri Berkley sau thread-uri POSIX. Proiectul NS3 a început în anul 2006, este gratuit, open-source şi destinat pentru cercetare şi dezvoltare.

Comparat cu NS2, avem următoarele diferenţe majore între cele două simulatoare:

-Atenţie sporită către realism: entităţile de protocol sunt proiectate astfel încât să fie mai apropiate de computerele reale;

-Integrare Software: suport ţncorporat pentru mai mult sofware de reţelistică open source şi astfel se reduce nevoia de a rescrie modele pentru simulare;

-Suport pentru virtualizare: sunt folosite maşini virtuale mai puţin performante pentru a crea medii de testare potrivite.

-Arhitectura de analiza a datelor: NS3 construieşte un framework care să gestioneze datele şi statisticile astfel încât să poată fi prelucrate datele simulării.

Există totuşi câteva limitări a simulatorului de reţea NS3, la baza acestora stă nevoia de participare a comunităţii de cercetare. O limitare generală a simulărilor este credibilitatea, care ar putea fi îmbunătăţită şi la NS3, lucru realizabil prin folosirea NS3 în publicaţii de specialitate, documentaţie completă şi actualizată, mijloace flexibile de configurare şi de a inregistra date.

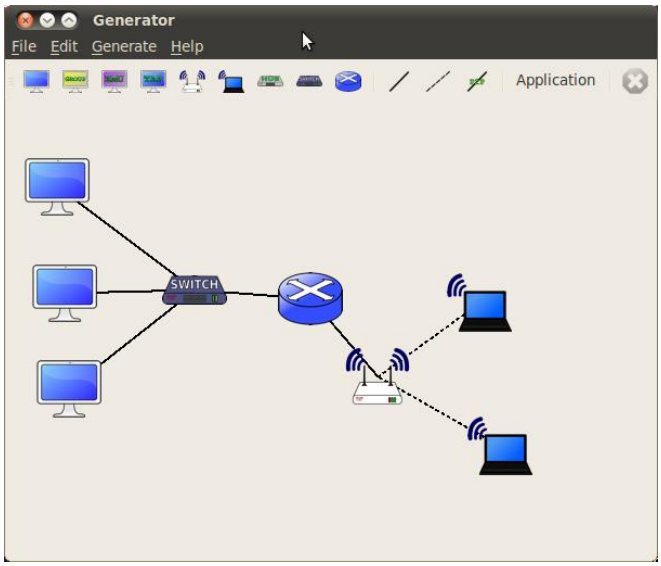

Figura 3.2: Simulatorul NS3

3.4.3 OPNET(Optimized Network Engineering Tool)

Este un software de simulare puternic şi scump cu o mare varietate de posibilităţi de simulare de reţele heterogene cu protocoale diverse. A fost dezvoltat iniţial la Massachusetts Institute of Technology (MIT) şi din anul 1987 a devenit software comercial dezvoltat de către OPNET Technologies. Suporta modelarea reţelelor de comunicaţii cât şi a sistemelor distribuite. Atât comportamentul cât şi performanţa sistemelor modelate pot fi analizate folosind simulări ale evenimentelor discrete.

Deoarece este un software comercial, OPNET oferă suport grafic şi vizual foarte bun pentru utilizatori. Editorul grafic poate fi folosit pentru a construi topologii de reţea şi entităţi începând cu nivelul aplicaţie până la nivelul fizic. Tehnici de programare orientate pe obiecte sunt folosite pentru a crea o mapare între designul grafic şi implementarea sistemelor reale. De asemenea, parametrii pot fi ajustaţi şi experimentele pot fi repetate cu usurinţă prin intermediul interfeţei grafice.

OPNET are trei funcţii principale: modelarea, simularea şi analiza. Pentru modelare, acesta oferă un mediu grafic intuitiv pentru a crea diverse modele de protocoale. Pentru simulare sunt folosite 3 tehnologii avansate şi poate fi folosit pentru o mare varietate de studii. Pentru analiză, rezultatele simulării şi datele pot fi analizate şi afişate foarte uşor. Grafice, diagrame, statistici, chiar şi animaţii pot fi generate cu usurinţă de către OPNET pentru a fi convenabile utilizatorului. Incepand cu 2008, OPNET Technologies a anunţat introducerea a doua aplicaţii capabile de managementul performanţei. Aceste doua noutăţi includ capabilitatea de a vizualiza performanţa aplicaţiilor pentru organizaţii ce folosesc soluţii de optimizare WAN precum şi capabilitatea de a captura şi vizualiza date NetFlow. Datorită efortului consistent, OPNET a devenit un simulator matur şi este recunoscut in industrie. În plus, dezvoltarea OPNET a ţinut cont de cerinţele utilizatorulor şi produsul este îmbunătăţit constant, ceea ce îl face competitiv în comparaţie cu alte simulatoare de reţea comerciale.

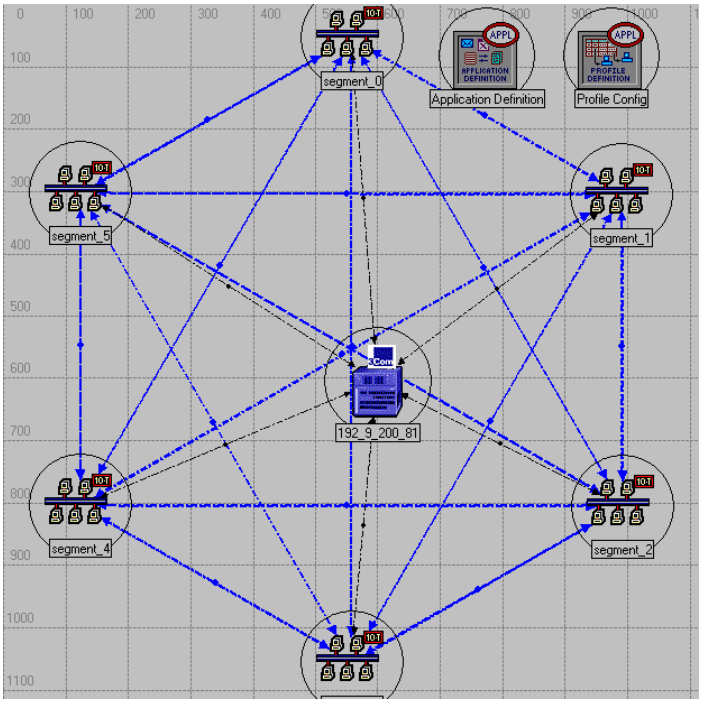

Figura 3.3: Simulatorul OPNET

3.4.4 OMNET++(Optical Micro-Networks Plus Plus)

Este un framework pentru a dezvolta simulatoare de reţea, care conţine o librărie extensibilă, modulară, bazată pe componente C++. Poate fi folosit pentru simularea reţelelor într-un sens mai larg precum reţele clasice, reţele wireless, reţele on-chip. Nodurile sunt programate în C++ şi apoi asamblate într-un limbaj de nivel înalt numit NED(Network Description). Acest limbaj suportă specificarea de parametrii variabili în descrierea reţelei, de exemplu numărul de noduri într-o reţea poate fi dinamic şi poate fi configurat mai târziu în timpul execuţiei simulării. În acest caz, modulele care reprezintă nodurile sunt înstanţiate dinamic de catre simulator in timpul execuţiei. Această caracteristică este o consecinţă directă a designului orientat pe obiecte. OMNET++ formează o baza pentru a construi simulatoare orientate pe aplicaţii specifice.

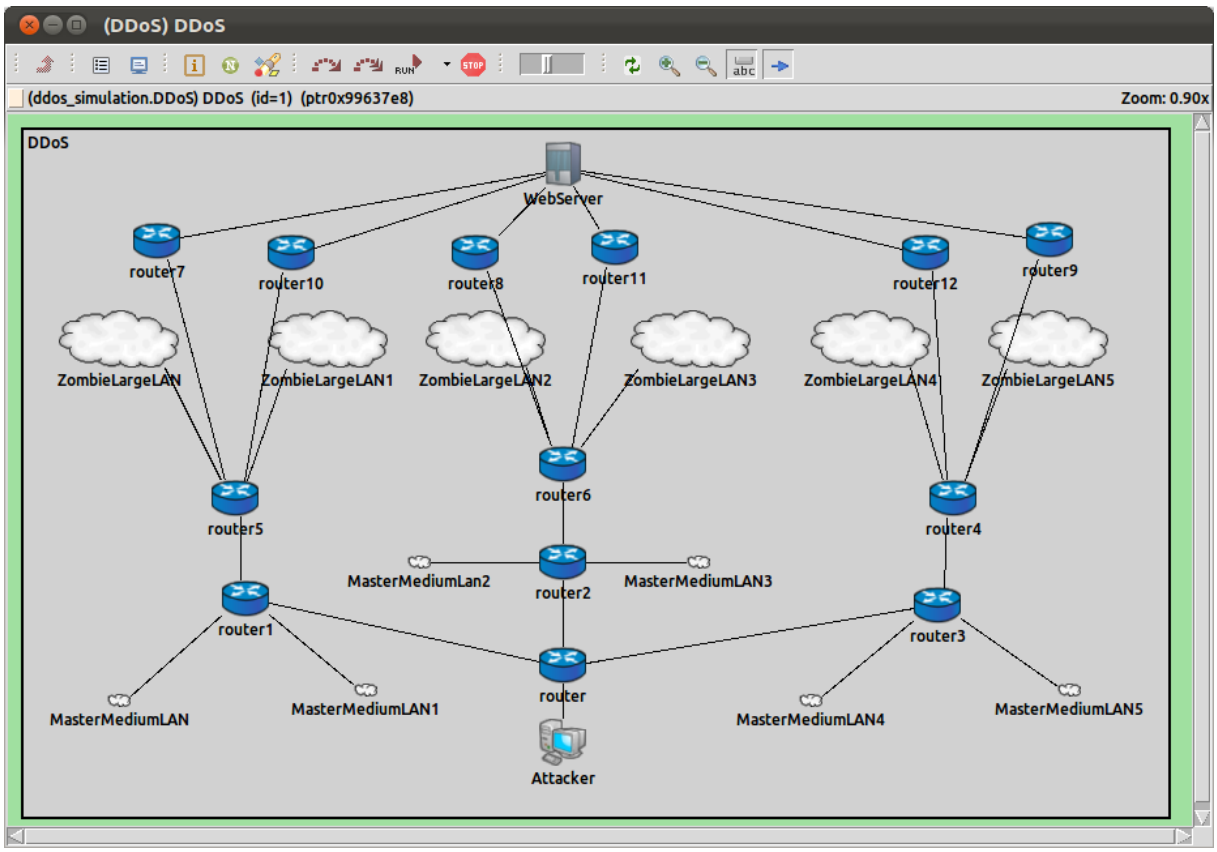

Figura 3.4: Simulatorul OMNET++

3.4.5 Cisco Packet Tracer

Packet Tracer este un software care poate fi utilizat în formare şi educaţie, dar şi în domeniul cercetării pentru simulări simple a reţelelor de calculatoare. Instrumentul este creat de Cisco Systems şi este prevăzut pentru distribuţie gratuită în facultate, pentru studenţi şi absolvenţii care sunt sau au participat la programul Cisco Academy. Scopul Packet Tracer este de a oferi elevilor şi profesorilor un instrument pentru a afla principiile de reţea precum şi să dezvolte abilităţi specifice tehnologiei Cisco. Avantajul acestui simulator este interfaţa sa grafică uşor de folosit şi faptul că este actualizat în mod constant, ultimele versiuni suportând Ipv6 (Internet Protocol version 6), BGP (Border Gateway Protocol) şi HSRP(Hot Standby Router Protocol). Ca dezavantaj ar fi posibilităţile limitate de simulare, scopul acestuia fiind în principal didactic.

Acest software reproduce o mică parte din capabilităţile hardware disponibile pe echipamentele Cisco cu propriul sistem de operare IOS. Totuşi, există alte doua aplicaţii software, GNS3 şi Dynagen, care folosesc imagini reale de routere Cisco IOS. Diferenţa dintre cele două este că GNS3 are interfaţă grafică pe când Dynagen are doar linie de

comandă. Aceste doua aplicaţii nu pot simula switch-uri şi necesită multe resurse hardware, în special memorie RAM şi putere de procesare.

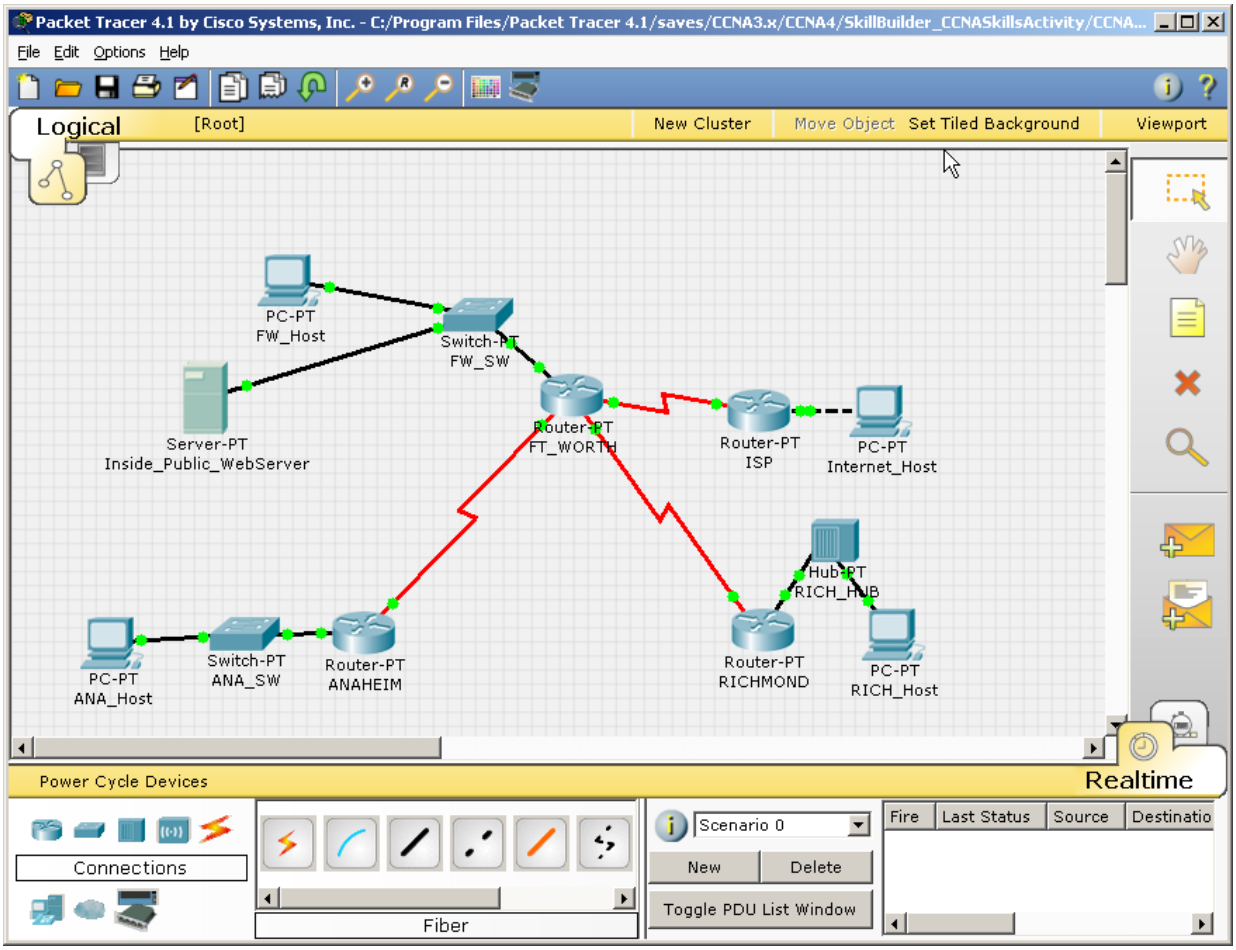

Figura 3.5: Cisco Packet Tracer

3.4.6 JiST(Java in Simulation Time)

JiST este un simulator de reţea care permite implentarea simulării in limbajul Java. Este, folosind în majoritatea cazurilor împreună cu SWANS, un simulator pentru reţele mobile şi reţele ad hoc construit pe baza JiST. Simulările sunt compuse din entităţi care reprezintă elementele reţelei, de exemplu nodurile, iar evenimentele simulării sunt apelări ale metodelor asociate entităţilor. Entităţile sunt independente între ele, comunicând prin intermediul simulării de bază. În timp ce codul unei entităţi este un program Java obişnuit, doar interacţiunile dintre entităţi sunt relevante pentru timpul simulării. Aşadar aceste interacţiuni între entităţi corespund momentelor de sincronizare şi facilitează execuţia în paralel a codului diferitelor entităţi de unde rezultă un potenţial spor de performanţa.

Pentru a executa implementarea în timpi specifici simulării, JiST foloseşte un încărcător de clase Java dinamic care rescrie codul aplicaţiei în mod automat.

Din nefericire, dezvoltarea oficială JiST a fost oprită deoarece nu mai este întreţinută de către autorul original, Rimon Barr. Cu toate acestea, o serie de îmbunătăţiri şi actualizari au fost lansate de către Universitatea Ulm.

Capitolul 4

4. Tehnologii WAN

4.1 ATM (Asynchronous Transfer Mode)

Utilizatorii LAN-ului sunt obişnuiţi cu viteza şi fiabilitatea propriilor reţele locale, şi cum nevoile forţeaza LAN-urile să se extindă peste limitele lor originale, servicii precum liniile închiriate, ISDN, X.25, SMDS şi Frame Relay au fost folosite pentru a interconecta LAN-uri. Dar utilizatorii se aşteaptă la aceeaşi viteză şi fiabilitate a traficului transmis peste reţele intermediare metropolitane şi WAN. Aceste aşteptări pun o substanţială presiune asupra furnizorilor de servicii şi echipamente pentru a crea dispozitive care leagă diverse tehnologii de comunicaţie într-o reţea fără întreruperi. Această nevoie pentru viteză mare atât în LAN cât şi în WAN este principala motivaţie pentru care au apărut primele aplicaţii ATM. Despre ATM se poate spune ca este o tehnologie de bază care încorporează alte tehnologii de LAN şi WAN într-o singură arhitectură.

Numele tehnologiei ATM se justifică prin faptul că, în timp ce în reţelele telefonice majoritatea transmisiilor sunt sincrone (au legatură cu un semnal de ceas), în reţelele ATM transmisiile nu sunt sincrone. ATM a fost proiectat la începutul anilor 1990 şi lansat la mijlocul acestei perioade incredibile [11].

ATM este un serviciu de comutare de celule (pachete de dimensiune fixă de 53 octeţi). Un antet de 5 octeţi indică destinaţia unei anumite celule şi restul de 48 octeţi sunt încărcătura efectivă. Câmpul de adresă este împărţit în două sub-câmpuri numite VCI (Virtual Channel Identifier) şi VPI (Virtual Path Identifier), care formează o conexiune virtuală şi au semnificaţie locală. Alte câmpuri importante din antet sunt CLP (Cell Loss Priority), care indică prioritatea unei celule în cazul în care traficul trebuie să fie aruncat. PTI (Payload Type Indicator) este un câmp de 3 biţi dintre care primul bit indică faptul că celula conţine date utile sau date pentru management şi operaţii, al doilea bit este EFCI (Explicit Forward Congestion Indication), care indică dacă celula a întâlnit vreo congestie în drumul ei de la sursă la destinaţie şi ultimul bit indică ultima celulă dintr-un bloc de celule. Câmpul HEC (Header Error Check) are rolul de a corecta un bit sau a detecta mai mulţi biţi eronaţi din primii 4 octeţi ai antetului. Verificarea datelor utile este responsabilitatea unor protocoale de nivel superior.

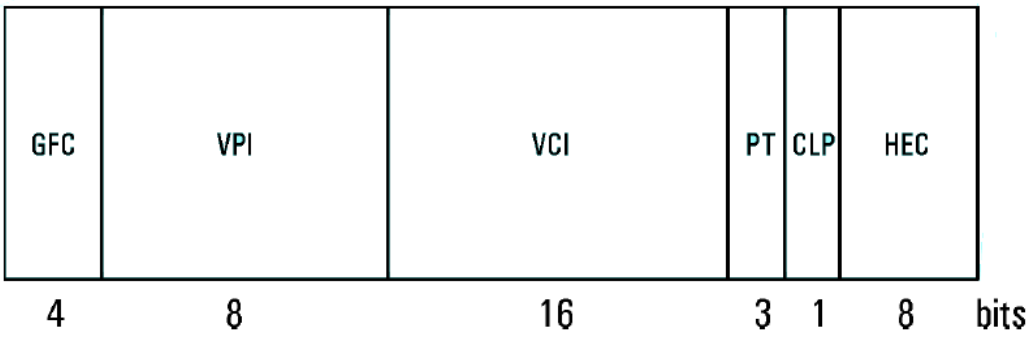

Figura 4.1: Antetul celulei ATM

ATM oferă posibilitatea creării unor conexiuni virtuale permanente cât şi conexiuni virtuale temporare. Primele implementări ATM puteau crea doar conexiuni virtuale permanente dar ulterior au fost standardizate şi conexiuni virtuale temporare. Traficul transportat prin reţele ATM trebuie divizat în celule mici cu încărcătura de 48 octeţi. Acest proces se numeşte segmentare. Diverse metode de segmentare sunt utilizate pentru transmisii de voce, video sau date. Datorită vitezei ATM, celulele sunt comutate în hardware către linii cu lăţimea de bandă mare. Celulele pot fi transmise cu orice viteză dar există unele standard: 45 Mb/s, 155 Mb/s, 622 Mb/s, 2.4Gb/s dar se poate atinge chiar şi 10 Gb/s. Transmisia de celule de dimensiune mică şi constantă în locul cadrelor de dimensiune variabilă a permis întrepătrunderea traficului sensibil la timp (ex: traficul telefonic) cu cadre de dimensiune mare (ex: traficul IP) cu un control mare al întârzierii transmiterii. De aici rezultă şi latenţe mult mai mici faţă de alte tehnologii. Un alt avantaj este că celulele ATM pot satisface cerinţe de lăţime de bandă mult mai mari decât cele existente la ora actuală.

Deoarece reţelele ATM sunt orientate pe conexiune, transmisia datelor necesită mai întâi transmisia unui pachet pentru iniţializarea conexiunii. Pe măsură ce pachetul de iniţializare este transportat prin subreţea, toate ruterele de pe drumul pe care îl parcurge îşi creează câte o înregistrare în tabelele de dirijare în care înregistrează existenţa conexiunii şi rezervă resursele necesare pentru ea. Conexiunile sunt de obicei denumite circuite virtuale, în analogie cu circuitele fizice utilizate în sistemele de telefonie. Majoritatea reţelelor ATM suportă şi circuite virtuale permanente, care sunt conexiuni permanente între două gazde aflate la distanţă.[*] Acestea sunt similare cu liniile închiriate din lumea telefoniei. Fiecare conexiune, indiferent dacă ea este temporară sau permanentă, are un identificator de conexiune unic.

Din momentul în care o conexiune a fost stabilită, se pot transmite date de oricare dintre cele doua părţi. O parte din antet reprezintă identificatorul de conexiune, astfel încât atât transmiţătorul cât şi receptorul, precum şi toate ruterele intermediare pot şti corespondenţa dintre celule şi conexiuni (care celule aparţin cărei conexiuni). Acest indentificator de conexiune permite fiecărui ruter să dirijeze fiecare celulă pe care o primeşte. Dirijarea celulelor este implementată direct în partea hardware a ruterelor şi este

o operaţie rapidă. Justificarea alegerii de celule de dimensiune fixă este aceea că este mai uşor de construit partea hardware pentru dirijare dacă ea are de a face cu pachete scurte şi egale ca dimensiune. Pachetele IP de lungime variabilă trebuie dirijate de programe (software), proces care este mai lent şi acesta constituie cel mai mare avantaj al tehnologiei ATM. Un alt avantaj al reţelelor ATM este acela că partea hardware poate fi configurată cu uşurinţă să multiplice o celulă pe care o primeşte la intrare pe mai multe linii de ieşire, o proprietate obligatorie în cazul în care trebuie abordată transmisia unui program de televiziune difuzat către mai mulţi receptori. Un lucru important este că celulele mici nu blochează nici o linie pentru prea mult timp, ceea ce face garantarea calităţii serviciilor (QOS – Quality Of Service) mai uşoară. Pentru că ATM realizează o conexiune, toate celulele urmează aceeaşi cale către destinaţie de unde rezultă faptul că este garantată ordinea lor, dacă se transmit celule într-o ordine, ele vor ajunge exact în aceeaşi ordine. Ceea ce nu se garantează însă este livrarea celulelor, nu se poate ştii dacă odată trimise nişte celule, ele ajung la destinaţie, acest lucru fiind de datoria protocoalelor de nivel superior, care repară eroarea cauzată de celulele pierdute.

4.1.1 Mecanismul de dirijare a traficului într-o reţea ATM

Există două tipuri de conexiuni ATM: căi virtuale (virtual paths - VP), care sunt identificate de către câmpul VPI şi canale virtuale (virtual channels - VC), care sunt identificate de combinaţia dintre VPI şi VCI. O cale virtuală este formată din mai multe canale virtuale, dintre care toate sunt comutate transparent în reţeaua ATM pe baza VPI-ului comun. O cale de transmisie este formată din mai multe VP-uri. Asemănător cu o linie telefonică, cu ajutorul căilor virtuale se pot direcţiona mai eficient canalele virtuale. De-a lungul unei căi, dacă un switch cedează, circuitele pot fi re-direcţionate modificând calea virtuală, în loc de a re-direcţiona fiecare canal individual. Figura 1.3 ilustrează modul în care canalele virtuale se concatenează pentru a crea căi virtuale, care, la rândul lor se concatenează pentru a crea o cale de transmisiune.

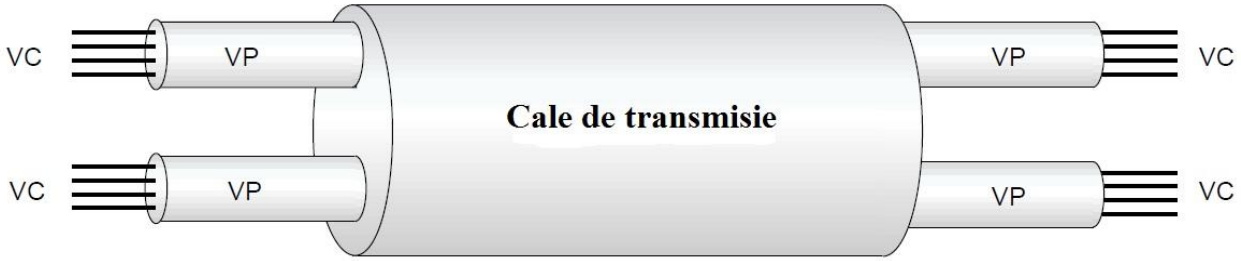

Figura 4.2: Căile virtuale sunt formate din canale virtuale

Funcţionarea unui switch ATM este destul de simplă: o celulă este recepţionată pe o interfaţă cu o valoare VCI sau VPI cunoscută. Switch-ul caută valoarea conexiunii într-un tabel local de translaţii pentru a determina portul (sau porturile) de ieşire ale conexiunii şi noua valoare VPI/VCI a conexiunii de pe portul respectiv. După aceasta, switch-ul retransmite celula pe portul (sau porturile) de ieşire cu identificatorii de conexiune potriviţi. Deoarece toate valorile VCI şi VPI au semnificaţie locală pe o anumită conexiune, aceste valori sunt modificate la fiecare switch, dupa cum este nevoie.

4.1.2 Segmentarea şi reasamblarea

Pachetele folosite de către stiva de protocoale TCP/IP sunt aproape întotdeauna mai mari decât celulele ATM. Pentru ca dispozitivele ATM să comunice cu aceste reţele existente, pachetele trebuie segmentate pentru a fi de dimensiunea celulelor înainte de a intra în reţea ATM şi apoi reasamblate în pachete mai mari când părăsesc reţeaua ATM. În continuare sunt prezentaţi paşii necesari pentru segmentare şi reasamblare:

- Un nod ATM primeşte un cadru de date dintr-un nivel superior.
- Se înlătura CRC-ul (Cyclic Redunduncy Check) pachetului. Aceasta este porţiunea din pachet care se ocupă cu detectarea erorilor.
- Se adaugă un antet şi subsol pachetului care conţine informaţii despre nodul terminal ATM.
- Se adaugă date pentru ca dimensiunea pachetului să fie un multiplu de 32 biţi.
- Se adaugă un nou CRC pachetului.
- Se segmentează pachetul în segmente de 48 biţi (care includ antetul şi subsolul)
- Se plasează fiecare segment de 48 biţi într-o celulă ATM şi se adaugă un antet care include VCI-ul.
- Switch-urile ATM de-a lungul canalului se uită doar la VCI şi la portul pe care a sosit celula, se consultă cu un tabel de direcţionare care conţine porturile de intrare şi porturile de ieşire şi trimit celula mai departe.
- La nodul ATM terminal, celulele sunt reasamblate în pachete folosind paşii anteriori în ordine inversă.
- Dupa ce un pachet este reasamblat, este verificat de erori de un protocol de nivel superior.

4.2 MPLS (Multiprotocol Label Switching)

Comutarea Multiprotocol cu Etichete (Multi Protocol Label Switching - MPLS) reprezintă o arhitectură în care nodurile terminale adaugă o etichetă unui pachet IP ce identifică drumul spre destinaţie iar pachetele sunt direcţionate pe baza etichetei fără inspectarea antetului iniţial.

MPLS foloseşte o soluţie ce integrează atât controlul rutării IP (nivelul 3 din modelul OSI), cât şi comutarea de la nivelul legăturii de date (nivelul 2 din modelul OSI), de aceea se spune că MPLS este un „protocol de nivel 2,5". Această tehnologie oferă bazele unor servicii de rutare avansate, rezolvând o serie de probleme:

- Se adresează problemelor privind scalabilitatea, legate de modelul IP-over-ATM;
- Reduce compexitatea operaţiilor din reţea;
- Facilitează apariţia de noi posibilităţi de rutare, ce îmbunătăţesc tehnicile de rutare IP existente;
- Oferă o soluţie standardizată, ce are avantajul interoperabilităţii între diverşi furnizori de produse şi servicii.

Într-o reţea tradiţională IP fiecare router efectuează o căutare a adresei de destinaţie, determină next-hop-ul pe baza tabelei de rutare şi trimite pachetul către acel next-hop. Aceste operaţii au loc în fiecare router, fiecare având propriile sale decizii independente de rutare, până când este atinsă destinaţia. Într-o reţea MPLS doar primul echipament efectuează o căutare a adresei destinaţie dar, în loc să găsească un next-hop, găseşte router-ul destinaţie şi o cale predefinită către acel router destinaţie. Următorul pas este aplicarea unui etichete pe baza informaţiilor găsite şi următoarele routere folosesc această etichetă pentru a direcţiona traficul fără a mai fi nevoie de a efectua alte căutări specifice IP. Când pachetul ajunge la router-ul destinaţie această etichetă este înlăturată şi pachetul este direcţionat în continuare folosind IP tradiţional.

4.2.1 Concepte de bază MPLS

Cale comutată după eticheta (Label Switched Path - LSP).

Este unul dintre cele mai importante concepte folosite în practică. În esenţă, LSP-ul, este un tunel unidirecţional între o pereche de routere, care este dirijat în interiorul unei reţele MPLS. Un LSP este necesar pentru a dirija orice trafic.Fără un LSP configurat nu putem avea trafic.

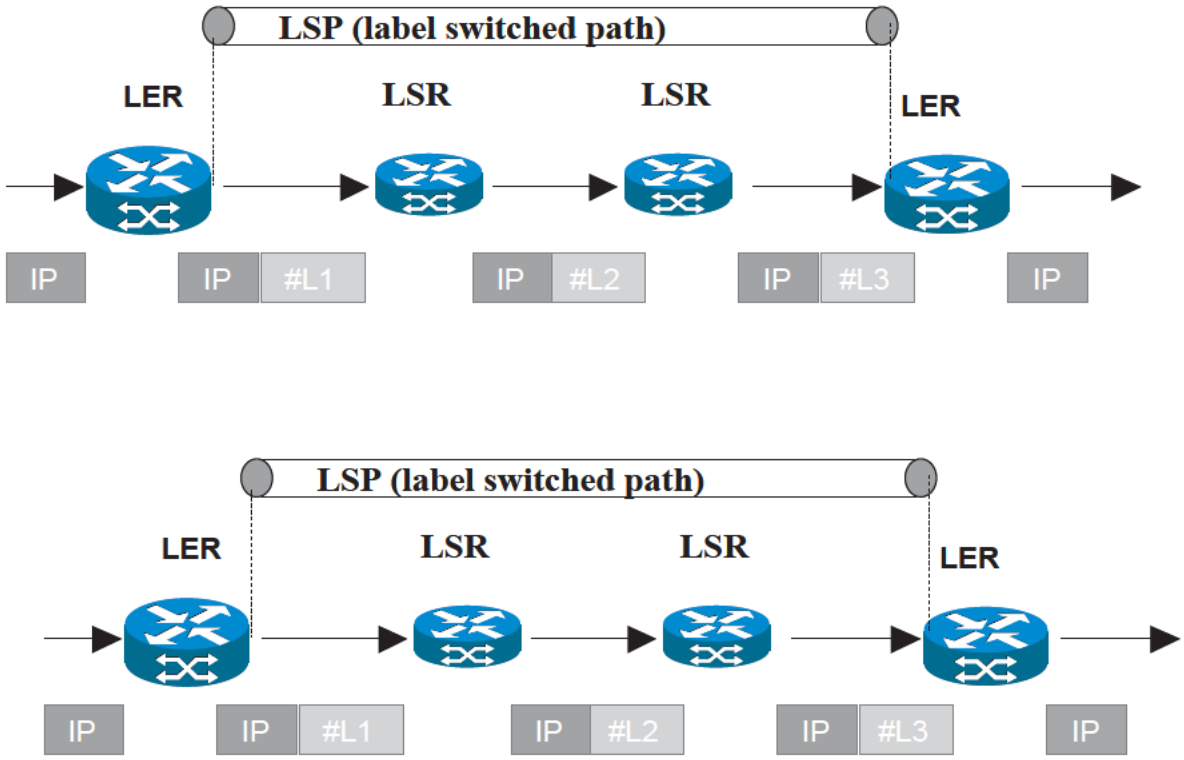

Figura 4.3: Cale comutată după etichetă (Label Switched Path - LSP)

Router de etichetare marginal (Label Edge Router - LER). Se mai numeşte nod de intrare (ingress node), deoarece este router-ul care încapsulează prima dată un pachet într-un LSP MPLS. De asemenea, este router-ul care face alegerea iniţială a LSP-ului.

Router de comutare a etichetelor (Label Switching Router - LSR). Este un router care efectuează doar comutare MPLS pe baza etichetei de-a lungul unui LSP.

Nod de ieşire (egress node). Este router-ul situat la capătul unui LSP şi este cel care înlătură eticheta.

Legătura protejată (Protected Link). Este o legatură fizică cu o formă de redundanţă integrată astfel încât transferurile de date nu sunt întrerupte de căderea unei compenente a legăturii. O legătură protejată apare în nivelul de control MPLS ca un singur punct de conexiune în interiorul reţelei. Există multe scheme de leagături protejate, dar una populară foloseşte protocoale SONET/SDH pe o buclă de fibră optică.

Legături paralele (Parallel or Bundled Links). Există mai mult de o legătură între o pereche de noduri într-o reţea. Spre deosebire de o legătură protejată, aceste legături individuale apar ca puncte de conexiune separate în reţea. Pot fi organizate ca

entităţi distincte suportând rute diferinte (dar paralele) în interiorul reţelei sau pot fi organizate ca o singură legătură unde alegerea componentei individuale este făcută doar de către nodurile conectate direct la legatură.

Cale alternativă (Alternate Path). Este o rută diferită în interiorul reţelei care leagă aceleaşi puncte terminale. Legăturile paralele asigură cele mai simple căi alternative. Căi alternative mai complicate trebuie să traverseze legături distincte şi noduri diferite.

Clasa de dirijare echivalentă (Forward Equivalent Class - FEC). Reprezintă un set de pachete care sunt tratate identic de către LSR. Astfel, un FEC este un grup de pachete IP care sunt dirijate peste acelasi LSP, sunt tratate la fel si pot fi asociate unei singure etichete de către un LSR chiar dacă pachetele sunt diferite în privinţa informaţiei din antetul de nivel reţea. Eticheta minimizează informaţiile esenţiale despre pachet. Acestea pot include destinaţia, precedenta, informaţii legate de QoS chiar şi întreaga rută a pachetului, după cum decide LSR-ul de intrare bazat pe politici administrative.

Eticheta (Label sau shim label). Reprezintă un identificator scurt, de lungime fixă şi cu valoare locală al unui FEC. Deşi informaţiile de nivel reţea sunt folosite pentru asocierea etichetei, eticheta nu conţine în mod direct informaţii în nivel reţea precum adresa sursa si destinatia. Etichetele au doar valoare locală, ce înseamnă ca o etichetă este utilă şi relevantă pe o singură legatură între LSR-uri adiacente.

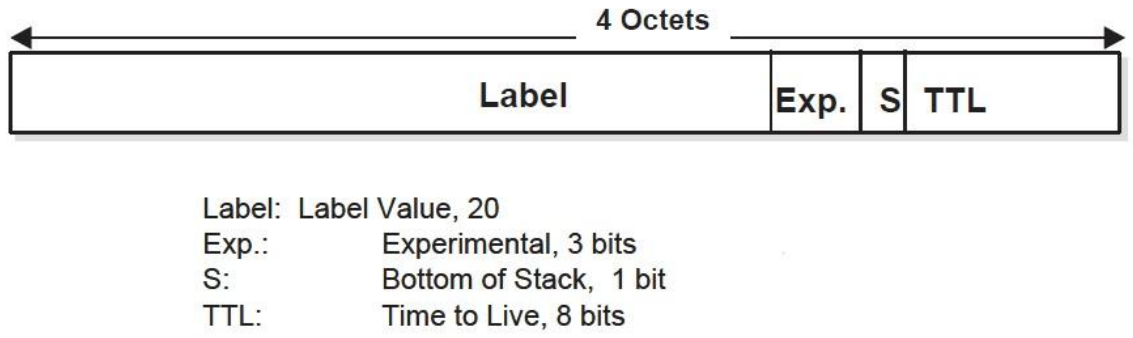

Figura 4.4: Structura antetului MPLS

Pentru a folosi un LSP, acesta trebuie sa fie anuntat catre celelate routere. Un LSP este un tunel intins pe o intreaga retea, dar o eticheta are valoare doar locala de-a lungul unei legaturi. Un protocol de semnalizare MPLS asociaza LSP-uri cu anumite valori ale etichetelor. Sunt folosite doua protocoale principale pentru semnalizare MPLS:

- Protocol de distribuţie a etichetelor (Label Distribution Protocol - LDP). Este un protocol simplu, fără constrângeri(nu suportă ingineria traficului)
- Protocol de rezervare a resurselor cu ingineria traficului (Resource Reservation Protocol with Traffic Engineering – RSVP-TE). Este un protocol mai complex,

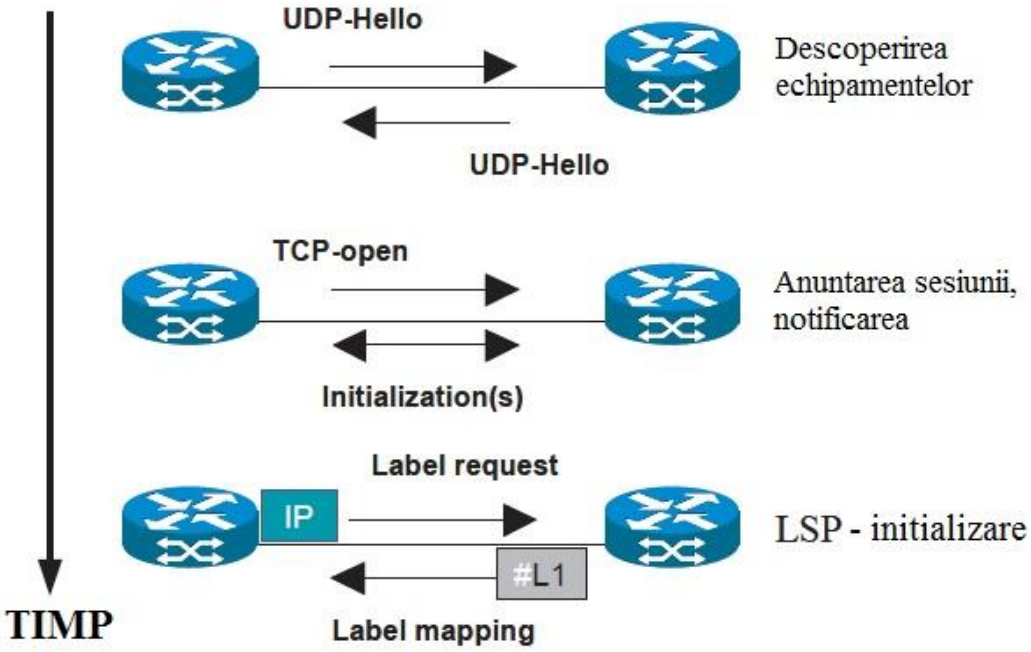

Figura 4.5: Funcţionarea LDP (Label Discovery Protocol)

Anumite reţele complexe folosesc ambele protocoale în acelaşi timp. LDP se foloseşte de obicei pentru servicii VPN (Virtual Private Network) pentru transport de date. Multe reţele MPLS configureaza tunele ce folosesc LDP in interiorul protocolului RSVP. Etichetele pot fi suprapuse una peste alta şi eticheta de deasupra este folosită pentru a controla livrarea pachetului. Cand este atinsă destinaţia, eticheta de deasupra este înlăturată şi urmatoarea etichetă preia sarcina de a controla livrarea pachetului mai departe.

Există două feluri de a termina un LSP, Null implicit şi Null explicit. Null implicit este o tehnică de optimizare şi înseamnă că eticheta este înlăturată de către penultimul router, nu de către ultimul router ca în mod obişnuit. Acest lucru reprezintă o optimizare deoarece ultimul router nu mai este nevoie să caute calea de dirijare şi să înlăture eticheta,

fiind necesaă doar o dirijare pe baza informaţiei continute. Null explicit înseamnă că eticheta este păstrată până la ultimul router.

IGP-urile(Interior Gateway Protocol) clasice folosesc rutare fără ingineria traficului, de exemplu o metrică(cost) pe o legătură şi un algoritm de cea mai scurtă cale (Shortest Path First - SPF) pentru a găsi cea mai scurtă cale. Ingineria traficului adaugă la acestea anumite constrângeri, de exemplu găsirea drumului cel mai scurt şi care are lăţime de bandă disponibilă. Acest lucru se mai numeşte rutare cu constrangeri, folosind un algoritm de cea mai scurtă cale cu constrangeri (Constraint Shortest Path First - CSPF). Principiul inginieriei traficului este simplu: este mai bine să alegem o cale necongestionată chiar dacă întârzierea poate fi mai mare, decât să congestionăm cea mai scurtă cale în timp ce pe alta legatura avem laţime de bandă disponibilă şi rămâne nefolosită.

Ingineria traficului se poate realiza şi manual, modificând costurile IGP-ului, dar acest lucru este scalabil doar până la un anumit punct. Pe măsură ce reţelele devin din ce în ce mai complexe, acest lucru devine mai greu de întreţinut şi modificarea unui cost IGP chiar şi cu o unitate poate afecta dirijarea traficului la o distanta de zeci de hop-uri. De aceea nu este recomandată ingineria traficului modificând manual costurile IGP-ului.

4.2.2 Potenţiale eşecuri ale resurselor

Orice resursă a reţelei poate eşua. Pentru a asigura o reţea cu high availability, furnizorul reţelei trebuie să prezică aceste căderi şi să planifice din avans.

Eşecul tradiţional este căderea unei legături cauzată de tăierea unui cablu sau scoaterea din soclu. Deasemenea, eşecul este posibil şi în reţele optice rezultând o lipsă a luminii, ca urmare a defectării laserului. Alte probleme obişnuite includ defectarea routerelor când un întreg LSR eşuează. Acest lucru poate fi cauzat de o întrerupere de curent sau o defectare a unei componente esenţiale a LSR-ului. Când un router eşuează, toate legaturile direct conectate eşuează.

Dacă are loc un eşec hardware al unei componente, LSR-ul îşi poate reveni. Eşecul unei plăci de reţea (sau port) este echivalent cu eşecul unei legături, dar poate fi recuperată intern dacă există o placă de reţea de rezervă pentru acea legătura. În acest mod, o placă de reţea de rezervă poate fi privită ca un element al unei legături protejate. Ambele plăci de reţea sunt conectate cu legătura şi doar cea principală trimite date, iar când aceasta eşuează, placa de rezerva este deja conectată cu legătura.

4.2.3 Obiective pentru supravieţuirea în caz de eşec

Obiectivul de bază pentru supravieţuirea în caz de eşec este minimizarea întreruperii traficului de date a oricărui tip de eşec.

Pe cât posibil, LSP-urile prestabilite să nu fie întrerupte deloc în timp ce reţeaua se reface după eşec. Acest lucru înseamnă că legăturile ar trebui menţinute şi pachetele nu trebuie aruncate. În practică, multe eşecuri întrerup traficul existent în timp ce se face o tranziţie de la vechile resurse la cele noi, dar aceste întreruperi ar trebui să fie minimizate pe cât posibil. O cifră de 60ms este folosită în lumea telecomunicaţiilor ca fiind cea mai mare întârziere a traficului de voce care poate fi acceptată de creierul uman până când devine ca o pauză care opreşte înţelegerea fluxului de voce. Acest lucru înseamnă că pentru traficul de voce, în mod ideal, orice eşec trebuie detectat, raportat şi reparat în timp de 60ms.

Chiar dacă repararea LSP-ului durează mai mult de 60ms este important să se restabilească conexiunea în mod automat. Să considerăm un utilizator telefonic, ideal ar fi să nu observe eşecul deloc, dar este mai bine să audă câteva clic-uri decât să piardă de tot conexiunea şi să fie nevoit să sune din nou. Dacă este întrerupt traficul de date, trebuie făcute anumite consideraţii dacă s-au pierdut date şi câte s-au pierdut. Nici reţelele IP, nici alte reţele precum ATM sau Frame Relay nu încearcă să asigure livrarea datelor, decât prin folosirea altor protocoale de nivel superior precum TCP. Totuşi, dacă s-au pierdut prea multe date, astfel de protocoale pot declara conexiunea eşuată şi să încerce reconectarea.

Un alt obiectiv cu prioritate ceva mai mică este acela că, serviciul de semnalizare ar trebui să rămână funcţional. Altfel spus, ar trebui să fie posibilă stabilirea de noi conexiuni după ce eşecul a fost reparat. Deşi nu este de dorit, în general, este acceptat ca un utilizator să reîncerce iniţializarea unei conexiuni dacă conexiunea nu s-a stabilit de prima dată. Având în vedere date statistice cu privire la posibilitatea iniţierii unei conexiuni în timp ce un eşec este reparat, de multe ori este considerat acceptabil ca semnalizarea să fie suspendată temporar.

Procesul reparării într-o anumită parte a reţelei ar trebui să întrerupă cât mai puţin alte părţi ale reţelei. Distribuind informaţii despre eşec în toată reţeaua ar putea întrerupe alte semnalizări şi trafic de date. O soluţie de reparare nu are nici o valoare dacă este proiectată să supravieţuiască unui singur eşec. Mulţi furnizori de reţele şi echipamente nu încearcă să asigure soluţii care să supravieţuiască mai multor eşecuri simultane. Chiar dacă acest lucru reduce complexitatea, implică faptul că timpul de recuperare al unui

echipament trebuie să fie scurt pentru a asigura că perioada de vulnerabilitate este scurtată pe cât posibil.

Toate soluţiile acestor cerinţe implică forme de redundanţa precum legături paralele, legături suplimentare în reţea sau prin adăugarea de componente hardware în interiorul swith-ului. Costul acestor soluţii impune o cerinţa adiţionala conform căreia resursele redundante ar trebui să fie minime şi preferabil să fie partajate de utilizatori multipli.

4.2.4 Avantaje MPLS

Deoarece MPLS foloseşte dirijarea pe baza etichetei de dimensiune fixă, comutarea fiecarui pachet este determinată de o singură cautare indexată într-un tabel de comutare. Acest lucru simplifică funcţionarea routerului în comparaţie cu algoritmul de găsire al celui mai lung prefix folosit pentru dirijarea datagramelor normale.

Unul din principalele avantaje MPLS este abilitatea sa de a realiza ingineria traficului în reţele IP. Ingineria traficului este necesară pentru a se asigura că traficul este dirijat printr-o anumită reţea în cel mai eficient şi sigur mod posibil. Cu ajutorul ingineriei traficului se poate distribui traficul în toată infrastructura reţelei şi permite furnizorilor de internet să dirijeze traficul într-un mod în care pot oferi cele mai bune servicii utilizatorilor în termeni de rate de transfer şi întârzieri.

Dirijarea QoS bazată pe sursă este un mecanism de dirijare în care LSR-urile pe care le va străbate o cale sunt determinate în nodul sursă pe baza resurselor disponibile în reţea şi pe baza cerinţelor QoS. Cu alte cuvinte, este un protocol de rutare care şi-a extins criteriile de selecţie a căilor să includă parametrii QoS precum lăţimea de bandă disponibilă, gradul de utilizare a căii şi a legăturii, resursele consumate în nod, întârziere şi latentă.

O reţea privată virtuală (Virtual Private Network - VPN) bazată pe internet foloseşte infrastructura deschisă şi distribuită a internetului pentru a transmite date între locaţii, menţinand securitatea prin folosirea unui protocol de încapsulare pentru a stabili tuneluri. O reţea privata virtuală poate fi asemanată cu un sistem de linii deţinute sau închiriate care pot fi folosite de o singură companie. Scopul principal al unui VPN este să ofere companiilor aceleaşi capabilităţi ca liniile închiriate private cu un cost mult mai mic prin folosirea infrastructurii publice. Arhitectura MPLS satisface toate cerinţele necesare pentru suportul VPN prin stabilirea de tuneluri LSP folosind rutare explicită. Aşadar, tehnologia MPLS folosind tuneluri LSP permite furnizorilor de servicii să livreze aceleaşi capabilităţi utilizând aceeaşi infrastructură cu care livrează şi servicii de internet.

Comutarea ierarhică se face pe baza suprapunerii etichetelor (sau controlul pachetelor pe mai multe nivele) şi astfel se permite încapsularea unui LSP în interiorul

altui LSP. Acest lucru nu este o noutate în tehnologia reţelelor, ATM furnizează comutare ierarhică pe două nivele cu noţiunea de cale virtuală şi circuit virtual. Totuşi, MPLS permite suprapunerea arbitrară a LSP-urilor, furnizând astfel mai multe nivele de control a pachetelor.

Comutarea etichetelor permite o separaţie mai completă între rutarea în interiorul domeniului şi rutarea între domenii, fapt care ajută la îmbunătăţirii scalabilităţii procesului de rutare. Mai mult, scalabilitatea MPLS este îmbunătăţită de FEC deoarece se agregă fluxurile şi suprapunerea etichetelor pentru combinarea LSP-urilor. Asocierea unei etichete pentru fluxuri de date individuale nu este ideea dorită pentru scalabilitate deoarece creşte numărul de etichete, care în consecinţa determină creşterea tabelei LIB. Pentru că FEC permite agregarea fluxurilor se îmbunătăţeşte scalabilitatea MPLS. În plus, LSP-uri multiple, ce aparţin de FEC-uri diferite pot fi unite într-un singur LSP.

Capitolul 5

5. Soluţii de recuperare MPLS

În ultimii ani, noi servicii şi aplicaţii au fost dezvoltate cu caracteristici de timp real şi orientate pe conexiune, servicii precum Voice-over-IP sau protocolul RTSP (Real Time Streaming Protocol).

Căderea unei legături majore sau a unui router important poate avea efecte severe asupra acestor servicii şi protocoale. După terminarea redirijării, aceste servicii ar putea suferi de o degradare a serviciului calităţii, deoarece ruta alternativă ar putea fi mai lungă sau mai aglomerată. De luat în considerare că traficul care nu este afectat în mod direct de eşec dar redirecţionat pe o cale alternativă este şi el afectat de această degradare.

Pe de altă parte, durata întreruperii datorită căderii unei legături sau căderii unui nod este, de obicei, prea mare pentru servicii în timp real şi aplicaţii multimedia ca sa îşi păstreze sesiunile. În acelaşi timp, fluxurile QoS pot suferi de o reducere a calităţii pe rutele alternative şi prin urmare nu mai pot fi restabilite.

Deoarece MPLS este o arhitectură orientată pe conexiune, în caz de eşec, trebuie să se stabilească un nou LSP şi după aceea să comute pachetele de la punctul de eşec sau de la alt nod către LSP-ul nou format. Din această cauză MPLS are un răspuns de restaurare încet ca urmare a unui eşesc al unei legături sau al unui nod dintr-un LSP.

Una din provocările unui protocol orientat, precum MPLS este garantarea serviciului în timpul eşecului. Abilitatea de a redirija traficul rapid în jurul unui punct de eşec sau în jurul unui punct de congestie într-un LSP poate fi important pentru reţele MPLS astfel încât să asigure calitatea serviciului chiar şi în situaţii de eşec.

Mai multe metode au fost propuse pentru a redirija traficul în MPLS. Există două scheme luate în considerare în cadrul IETF oferind abordări diferite a problemelor de restaurare a LSP-ului în reţele bazate pe MPLS. Primul este cel mai rapid mecanism de redirijare disponibil care a fost propus de către Haskin şi Krishnan [9], iar cel de-al doilea este un mecanism mai lent dar mai puţin complex propus de Makam et al. [8] cunoscut ca tunel de back-up bazat pe RSVP (RSVP-based Backup Tunnel).

5.1 Redirijarea

Redirijarea este o tehnică care poate fi folosită şi în reţele cu comutare de etichete şi reţele cu comutare de pachete. Redirijarea este definită ca crearea unui nou traseu sau a unui segment al traseului la cerere pentru restaurarea traficului dupa apariţia unui eşec. Aşadar este un mecanism de recuperare în care traseul de revenire sau segmentul traseului este creat dinamic după detecţia eşecului pe traseul în funcţiune. Pentru acest scop, un traseu alternativ sau un traseu de rezervă sunt necesare în afară de calea principală folosită de trafic. Traseul principal şi traseul de rezervă ar trebui să fie total disjuncte. Componentele unui reţele sunt formate, în mare parte, din legături şi noduri. Deoarece eşecul unui nod cauzează căderea legăturilor adiacente nodului, vom folosi căderea legăturilor ca un eşec al reţelei.

Atunci când are loc o cădere a unei legături pe traseul principal, procesul de restaurare începe în mod automat. Paşii principali care trebuie efectuaţi de către metoda de redirijare sunt detectarea eşecului, notificarea eşecului, calcularea traseului alternativ şi redirijarea traficului de pe traseul principal pe traseul alternativ.

5.2 Modelul lui Makam

Una din primele modele propuse de recuperare MPLS a fost prezentată de Makam în draftul [OSMH01]. Modelul propus este referenţiat ca modelul lui Makam şi furnizează propecţie capăt la capăt pentru un LSP prin instalarea unui traseu de recuperare globală între LSR-ul de intrare şi LSR-ul de ieşire. Acest traseu de recuperare este total disjunct faţă de traseul principal. Atunci când este detectat un eşec oriunde de-a lungul căii principale, un semnal de indicare a eşecului (fault indication signal - FIS) este folosit pentru a transporta informaţii despre eşec către nodul responsabil cu comutarea traseelor (Path Switch LSR - PSL). Când acest semnal este primit, PSL-ul comută traficul către traseul de recuperare.

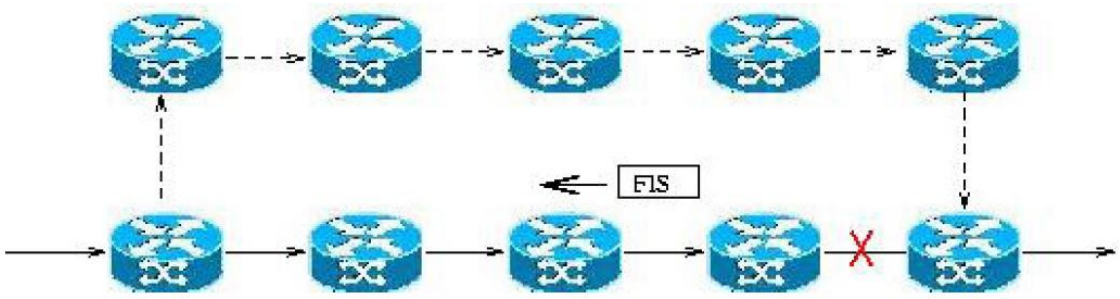

Figura 5.1: Modelul lui Makam

Figura 5.1 arată cum semnalul FIS este trimis în direcţia opusă traseului principal atunci când un eşec este detectat pe acest traseu.

Acest model are propuneri atât pentru un un traseu de recuperare prestabilit cât şi pentru un traseu de recuperare stabilit dinamic (redirijat). Deoarece traseul de recuperare stabilit dinamic va adauga mai mult timp operaţiei de restaurare, traseul de recuperare prestabilit este folosit de cele mai multe ori cu modelul lui Makam[8].

5.3 Modelul lui Haskin (Reverse backup)

Când este folosită recuperarea globală ca în modelul lui Makam, PSL-ul trebuie informat despre eşecul în traseul principal înainte ca traficul să fie comutat pe calea de recuperare. Când acest lucru este realizat cu ajutorul semnalului FIS, traficul va fi trimis în continuare pe traseul principal până când semnalul va fi recepţionat de către PSL. Înseamnă că vor fi pierdute pachete la LSR-ul care a detectat eşecul deoarece acest nod nu are nici o informaţie despre cum să dirijeze pachetele. Dacă eşecul este situat la distanţă mare de punctul de reparare şi rata de transmisie este mare, numărul de pachete pierdute poate fi foarte mare.

Idea din spatele recuperării inversate este redirijarea traficului în direcţia pe care a sosit la punctul de eşec de pe traseul principal către PSL. Din moment ce un LSR detectează un eşec pe traseul principal, acesta redirijează traficul pe LSP-ul alternativ care a fost creat în sens opus traseului principal. Atunci când traficul inversat este recepţionat de PSL, acesta Îl comută pe o cale de protecţie globală. Atât calea inversată cât şi traseul de protecţie globală sunt prestabilite. Această schemă a fost introdusă de către Haskin în draftul [HK00] şi este referenţiat ca modelul lui Haskin.

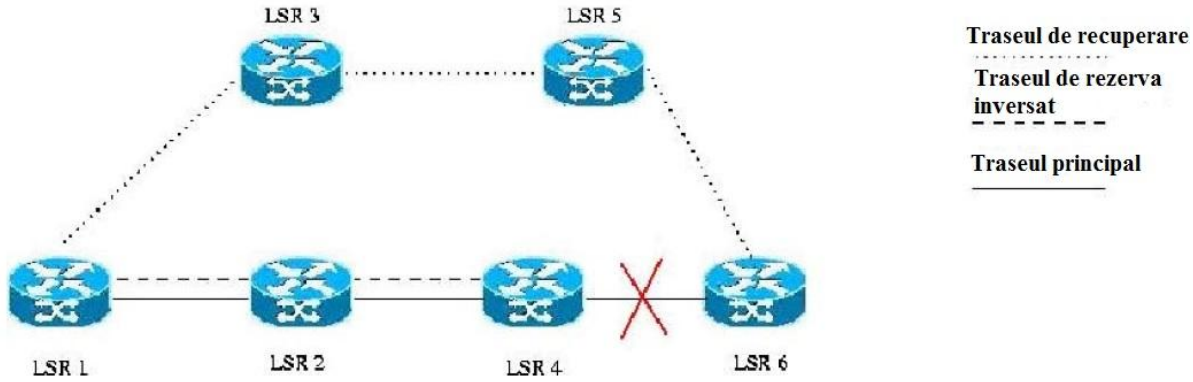

Figura 5.2: Modelul lui Haskin

Figura 5.2 arată un exemplu de recuperare inversată. Traseul principal şi traseul de recuperare sunt stabilite la fel ca şi în protecţia globală, dar în plus există o cale inversată de la LSR4 la nodul de intrare LSR1. Atunci când un eşec este detectat de LSR4, traficul este redirijat înapoi către LSR1 prin LSP-ul de rezervă inversat şi apoi comutat pe calea globală de recuperare. Dupa cum se observă, acest model protejează împotriva eşecurilor situate oriunde pe calea de la LSR1 la LSR6.

Există şi optiunea de a face PSL-ul să comute traficul direct pe traseul global de recuperare, din moment ce se va observa că este folosit traseul de rezervă inversat. În acest fel, traficul inversat este folosit ca un semnal FIS, când primul pachet este recepţionat de pe traseul de rezervă inversat, PSL-ul începe să comute traficul direct pe calea de recuperare globală. Astfel, traseul de recuperare se scurtează, deoarece nu mai este nevoie să comutăm traficul pe traseul principal care este eşuat.

Această optimizare are un dezavantaj, până cand PSL-ul recepţionează traficul inversat, pachetele vor fi trimise în continuare pe traseul principal care a eşuat. Atunci când PSL-ul primeşte trafic de pe traseul de rezervă inversat, acesta începe să comute traficul pe calea globală de restaurare. Pentru o perioadă scurtă de timp, traficul intrat va fi combinat cu traficul inversat pe traseul de recuperare. Această reordonare de pachete poate cauza probleme protocoalelor de nivel superior care au nevoie ca pachetele să ajungă în ordine.

O altă problemă cu modelul lui Haskin este utilizarea mai puţin eficientă a resurselor, deoarece traseul de recuperare este mai lung decât traseul principal. Partea pozitivă este numărul scăzut de pachete pierdute când are loc un eşec[9].

5.4 Redirijarea rapidă (Fast Rerouting)

Mecanismul de restaurare Fast Rerouting prestabileşte calea alternativă la calea protejată înainte să aibă loc orice eşec. Criteriul creării căii alternative prestabilite este bazat pe politici de rutare, cerinţe de restaurare a traficului protejat şi consideraţii administrative. Când are loc un eşec, LSR-ul responsabil cu redirijarea traficului comută traficul de pe calea protejată pe calea alternativa prestabilită. Deoarece modelul restaurării prestabileşte calea de recuperare înainte de apariţia unui eşec, timpul de recuperare este mai mic decât modelul redirijării.

Deoarece întregul traseu este setat într-un LSP, nu pot apărea bucle de rutare în timpul convergenţei, chiar dacă traseul este suboptimal.

Avantajul major al modelului de redirijare rapidă este timpul redus de recuperare, deoarece nu este nevoie de calculul unui nou traseu în caz de eşec. Rerutarea rapidă este folosită ca un mecanism de recuperare locală care foloseşte protecţia legăturii sau a nodului prin stabilirea unui traseu alternativ care utilizează comutarea pentru a ocoli legătura căzută sau nodul eşuat. Dacă se foloseşte redirijarea rapidă de la un capăt la altul, atunci traseele de recuperare trebuie să fie prestabilite pentru fiecare legătura sau fiecare nod de pe traseul principal. În [PSA00] sunt definite extensii ale protocolului RSVP-TE pentru stabilirea unui LSP cu tunele de rezervă de la un capăt la altul. Sunt descrise două tehnici în draft, modelul de rezervă unu la unu(one-to-one backup) şi modelul de rezervă cu facilitate(facility backup)[10].

5.4.1 Redirijarea rapidă cu rezervă unu la unu

Folosind tehnica de redirijare rapidă cu rezervă unu la unu, se calculează un LSP de rezervă pentru fiecare LSR din calea protejată. Aceste LSP-uri de rezervă sunt configurate să folosească restaurarea nodurilor şi dacă nu este posibil, să folosească restaurarea legăturilor. Pentru a proteja în întregime un LSP care traversează N noduri pot fi până la N-1 LSP-uri de rezervă.

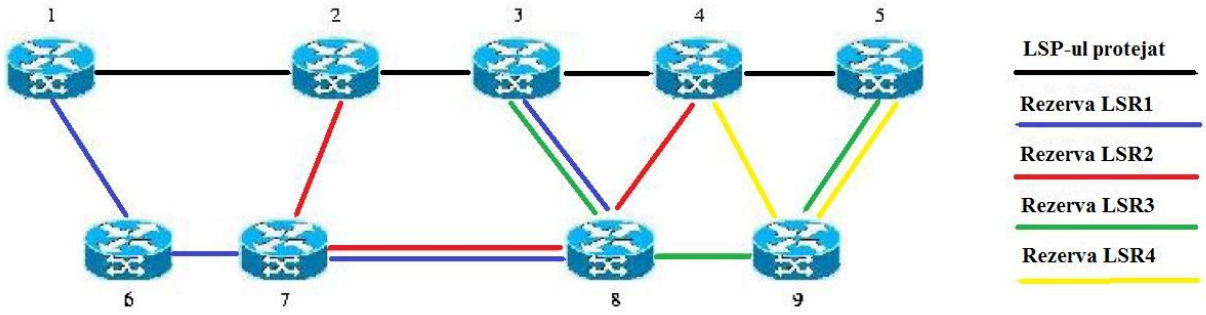

Figura 5.3: Redirijarea rapidă cu rezervă unu la unu

În figura de mai sus, LSP-ul protejat are 4 LSP de rezervă. Alternativele 1-3 folosesc protecţia nodului, în timp ce alternativa 4 folosesc protecţia legăturii. Dacă are loc un eşec oriunde pe calea protejată, LSR-ul care a detectat căderea poate oricând să comute traficul pe o cale alternativă fara a mai fi nevoie să trimită un semnal FIS pentru ca restaurarea să înceapă. Prin urmare, operaţiunea de recuperare devine o decizie locală pentru LSR-ul care a detectat eşecul. Folosind acest model, există câteva extensii sugerate pentru protocolul RSVP-TE care fac posibilă găsirea şi crearea căilor alternative în mod automat la configurarea traseului principal.

După cum se observă în figura 5.3, folosirea protecţiei locale pentru fiecare LSR din traseul principal pastrează multe resurse pentru scopuri de rezervă. Deşi există doar un singur traseu principal în reţeaua din exemplu, există unele legături pe care se fac doua rezervări pentru backup. Acest lucru nu este necesar deoarece nu există decât un singur traseu principal şi acele rezervări nu pot fi folosite în acelaşi timp dacă are loc un singur eşec pe traseul principal. Acest lucru este legat de rezervările LSR7 – LSR8 şi LSR9 – LSR5, şi nu de rezervarea dintre LSR3 şi LSR8 deoarece avem doua direcţii opuse. Pentru a preveni această rezervare dublă, se pot unifica acele rezervări şi pot fi partajate. Unificarea se poate realiza dacă cele două sau mai multe rezervări sunt rezervări de backup pentru acelaşi traseu principal. Sunt folosite următoarele reguli pentru unificarea rezervărilor:

Pentru toate rezervările care partajează aceeaşi interfaţa de ieşire şi acelaşi LSR next-hop, rezervarea se poate efectua doar pe traseul cel mai scurt către nodul de iesire al LSP-ului protejat. Dacă traseele au acelasi număr de hop-uri, traseul către străbate noduri pe care altele le evită, atunci acel traseu ar trebui eliminat. Dacă încă mai există trasee alternative multiple, atunci este o decizie locală care se va alege.

În exemplul de mai sus, acest lucru înseamnă că pe legătura dintre LSR7 şi LSR8, rezerva LSR1 va fi unită cu rezerva LSR2 în LSR7. Acest lucru se întamplă deoarece cea de-a doua rezervă are un traseu mai scurt către nodul de ieşire. LSR9 va unifica rezervele LSR3 şi LSR4 pe legătura dintre LSR8 şi LSR9. Alternativa de la LSR3 este proiectată să ocolească LSR4, aşa că alternativa de la LSR3 va face ultima rezervare.

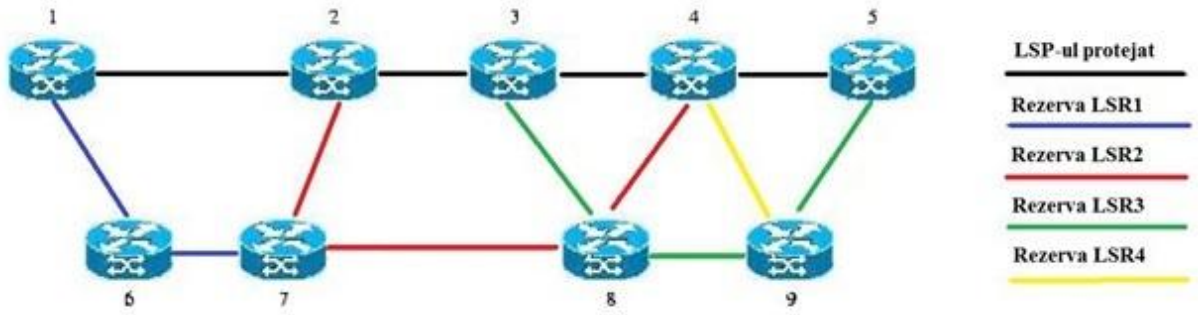

Figura 5.4: Redirijarea rapidă cu rezervă unu la unu şi unificare

Traseul de la LSR7 la LSR4 este folosit acum de rezerva lui LSR1 şi rezerva LSR2 iar traseul de la LSR9 la LSR5 este folosit de reverza lui LSR3 şi rezerva lui LSR4. Folosind această tehnică de unificare, putem observa că de la 12 rezervări iniţiale am ajuns la 9 rezervări iniţiale.

5.4.2 Redirijarea rapidă cu rezervă de facilitate (Fast reroute Facility Backup)

Folosind acest model, este creat un singur LSP care este folosit ca alternativă la multiple LSP-uri în loc să folosească o rezervă separată pentru fiecare LSP protejat. Acest lucru constrânge setul de LSP-uri protejate să traverseze un LSR comun pentru a putea fi redirijate pe calea de rezervă. Toate LSP-urile care traversează un anumit punct de reparare şi acest LSR comun pot fi protejate de tunelul ocolitor. Rezultă că modelul cu rezervă de facilitate foloseşte asocieri ale căilor unu la unu.

Tunelul foloseşte suprapunerea etichetelor, deoarece, atunci când mai multe LSP-uri sunt protejate de un singur tunel de ocolire, trebuie să existe o cale să separe LSP-urile diferite când traficul de pe tunelul de rezervă ajunge la nodul de ieşire.

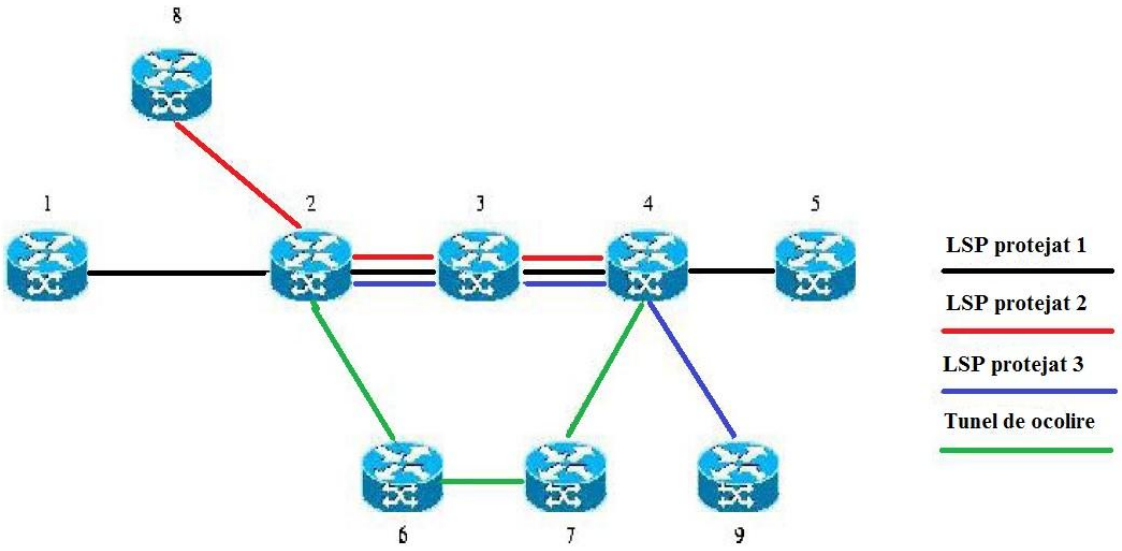

Figura 5.5: Modelul de redirijare rapidă cu rezervă de facilitate

În figura de mai sus avem 3 LSP uri care sunt protejate de un singur tunel de ocolire. După cum se observă, tunelul de rezervă cu facilitate foloseşte restaurarea locală cu protecţia nodului. În acest caz, LSR3 este nodul protejat pentru cele trei LSP-uri. Dacă s-ar fi folosit tehnica unu la unu, s-ar fi utilizat trei legături de rezervă pentru a proteja LSR3. Chiar dacă mai multe LSP-uri pot partaja acelaşi tunel de ocolire, resursele nu sunt partajate. Fiecare LSP protejat trebuie să facă o rezervare adiţională dacă vrea să folosească tunelul.

5.5 Soluţia propusă de restabilire a traficului

Având în vedere modelele anterioare de restabilire a traficului, am propus o soluţie care combină comutarea traficului, specifică redirijării rapide, cu stabilirea unui traseu disjunct total faţă de traseul principal, specific metodelor Makam şi Haskin. Această soluţie caută să îmbunătăţească numărul de resurse folosite în modelul redirijării rapide.

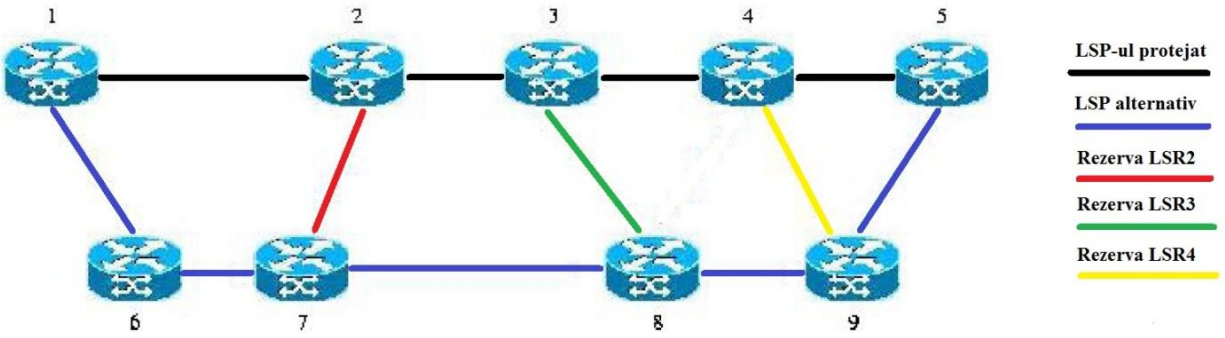

Figura 5.6: Soluţia propusă de stabilire a LSP-urilor de rezervă

Această schemă de restabilire presupune formarea LSP-urilor de rezervă înaintea eşecului. Primul LSP creat trebuie să fie un traseu disjunct faţă de calea protejată care va uni nodul de intrare în reţeaua MPLS cu nodul de ieşire. Acesta va fi şi traseul folosit în cazul în care LSR-ul de intrare va fi cel care detectează eşecul în LSP-ul protejat.

Urmatoarele LSP-uri de rezervă vor fi create astfel încat pentru fiecare LSR din traseul principal se va calcula drumul cel mai scurt către LSP-ul alternativ şi disjunct. Pentru a găsi cel mai scurt drum se va folosi nivelul de reţea deoarece etichetele MPLS nu sunt suficiente, ele oferind acces doar la LSR-urile vecine. Deşi calcularea acestor traseuri va implica schimburi de mesaje IP între LSR-uri, acest lucru se face doar o singură dată, când se creează toate LSP-urile de rezervă.

Funcţionarea acestui model este următoarea: atunci când un LSR din calea protejată detectează un eşec, acesta va comuta traficul pe traseul său de rezervă. Atunci când acest trafic ajunge la un LSR prin care trece primul LSP de rezervă creat (LSP-ul alternativ), va fi comutat pe acest LSP către nodul de ieşire.

Capitolul 6

6. Simulări

6.1 Simulatorul NS2

NS2 este un simulator în care un nod este format din clasificatori şi agenţi. Un agent este obiectul care primeşte sau recepţionează, în timp ce clasificatorul este un obiect responsabil cu clasificarea pachetelor şi cu dirijarea lor către nodurile convenabile sau cu livrarea către agentul local dacă nodul receptor este destinaţia pachetului. Astfel, pentru a construi un nod MPLS, un clasificator nou numit „Clasificator MPLS" ar trebui creat pentru a putea clasifica pachetele recepţionate, a determina dacă sunt etichetate sau nu şi tratate în mod corespunzator. De asemenea, un nou agent, „Agent LDP" trebuie inserat in nodul IP pentru a distribui etichetele între noduri MPLS şi pentru a construi LSP-urile.

Arhitectura unui nod MPLS incluzând clasificatorul MPLS şi agentul LDP:

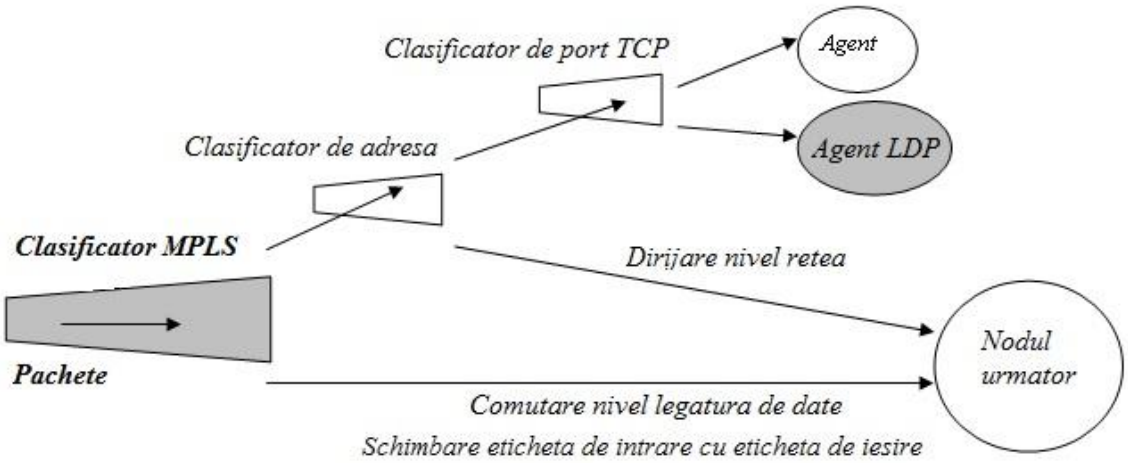

Figura 6.1: Arhitecura unui nod MPLS în NS2

Atunci când un nod recepţionează un pachet, clasificatorul MPLS determină dacă este etichetat sau nu. Dacă pachetul este etichetat, clasificatorul va executa o comutare la nivel de legatură de date. Va înlocui eticheta prezentă în antetul MPLS al pachetului cu eticheta corespunzatoare destinaţiei pachetului (FEC) după care îl va trimite către nodul următor. Dacă pachetul recepţionat nu este etichetat şi există un LSP creat pentru destinaţia pachetului(LSP) atunci clasificatorul va crea un antet MPLS, îi va asocia o etichetă şi îl va trimite către next-hop-ul corespunzător.

În cazul in care pachetul recepţionat nu este etichetat şi nu există LSP, nodul va livra pachetul clasificatorului de adresă care va executa dirijare obişnuită de nivel reţea prin examinarea adresei IP destinaţie.

Pachetul va fi livrat clasificatorului de port când nodul receptor este şi destinaţia pachetului. Acest clasificator de port va furniza pachetul agentului convenabil. Agentul LDP este folosit pentru a distribui etichetele şi pentru a iniţializa LSP-uri pe baza protocolului LDP.

Un nod MPLS are trei tabele pentru a organiza informaţia legată de LSP-uri şi de distribuţia tabelelor. Cele trei tabele sunt:

- Tabel cu informaţii despre etichete(Label Information Base – LIB): Acest tabel este construit şi folosit pentru a asocia eticheta şi interfaţa de intrare cu eticheta şi interfaţa de ieşire. Este folosit atunci când se efectuează comutarea la nivel de legatură de date.
- Tabel cu comutări parţiale(Partial Forwarding Table - PFT): Acest tabel este folosit atunci când soseşte un pachet neetichetat. Nodul MPLS va cauta în acest tabel pentru pentru a găsi FEC-ul reprezentat de adresa desţinatie a pachetului. Instanţa gasită ar putea indica o intrare in tabelul LIB pentru a eticheta pachetul sau ar putea indica NULL şi ca rezultat se va recurge la dirijare obişnuită la nivel de reţea.
- Tabel cu informaţii de rutare explicite(Explicit Routing information Base - ERB): Acest tabel este folosit pentru a stoca informaţii cu rute explicite. Dacă se doreşte o rută explicită, singurul lucru de făcut este inserarea unei instanţe care va conţine un indicator către acelaşi pointer LIB din tabelul PFT.

Network Animator(Nam)

Nam este o unealtă pentru animații, bazată pe Tcl/TK , folosită pentru vizualizarea datelor simulărilor de rețele. Teoria de implementare din spatele Nam a fost crearea unui animator care să poată citi seturi foarte mari de date pentru animații și care să poată fi îndeajuns de extensibil pentru a putea vizualiza o viariatate foarte mare de situații de rețele. Având această constrângere, nam a fost creat să poată citi comenzi simple de animație din interiorul unor fișiere foarte mari de urmărire. Comenzile sunt reținute în fișier și sunt recitite ori de câte ori este necesar.

Primul pas pentru a putea folosi nam este crearea unui fișier de urmărire. Acest fișier conține informații referitoare la topologii, noduri, legături dar și pachete. De obicei, acest fișier este creat de către Network Simulator.

După ce a fost generat fișierul pentru urmărire, acesta este pregătit pentru a putea fi animat de către nam. La pornire, nam citește fișierul, crează topologia, creează o fereastră pop-up, realizează layout-ul necesar și se oprește la momentul de timp 0. Network animator are o interfață pentru utilizator însă permite controlul asupra multor aspecte ale animației.

6.2 Scenariul simulat

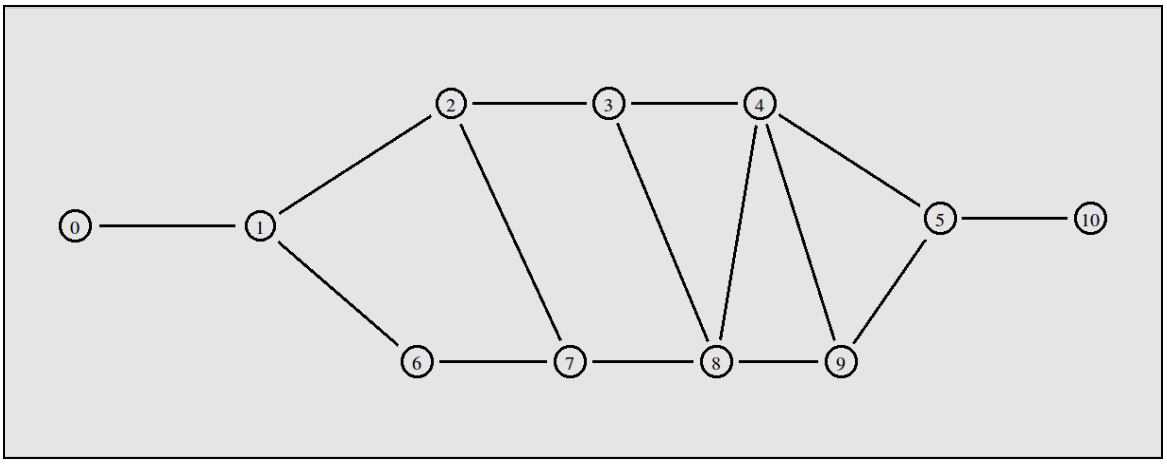

Figura 6.2: Topologia utilizată

Topologia utilizată este alcatuită din 9 noduri MPLS şi 2 noduri IP(0 si 10).

Considerăm că nodul 0 este sursa traficului şi nodul 10 este destinaţia. Iniţial, traficul se va trimite folosind traseul protejat(1-2-3-4-5) şi se vor folosi alte trasee atunci când reţeaua va suferi un eşec.

În simulare am folosit eşecul unei singure legături de pe traseul principal, legătura dintre LSR4 şi LSR5. Timpul de detecţie al eşecului este 0 pentru a nu avea diferenţe între modelele folosite.

Toate legăturile între noduri sunt identice, legături de 100Mb şi întârziere de 10ms. Pentru a varia rata de transfer, am folosit diferite dimensiuni ale pachetelor, începând cu 100 octeţi până la 10000 octeţi.

Numărul de pachetele pierdute se calculează ca diferenţa între numărul de pachete trimise de nodul sursa(0) şi numărul de pachete recepţionate de nodul desţinatie(10).

Tabelul 6.1 Date legate de scenariul simulat

Tipul cozii de aşteptare	**DropTail**
Numărul de noduri IP	2
Numărul de noduri MPLS	9
Durata timpului simulat	1s
Timpul de detecţie al eşecului	0s
Lărgimea de bandă a legăturilor	100 MB/s
Tipul de trafic	UDP
Dimensiunea pachetelor	100/500/1000/5000/10000 bytes
Intervalul de transmitere al **Pachetelor**	0,001s
Protocolul de distribuţie al etichetelor	LDP

Pentru realizarea scenariilor am creat scripturi în limbajul TCL pentru fiecare model de recuperare de trafic. Acestea se găsesc în anexa.

Pentru a extrage datele din fişierul rezultat în urma simalării(trace file) am folosit scripturi realizate în limbajul AWK. Sunt folosite două scripturi, unul care calculează timpul de întrerupere al serviciului şi unul care calculează numărul de pachete pierdute.

Capitolul 7

7. Prezentarea rezultatelor

Tabel 7.1 – Rezultatele obţinute pentru diverse metode de recuperare

Metoda de recuperare	Timp de întrerupere al serviciului (secunde)	Pachete pierdute	Resurse rezervate
Makam	0,0858	74	5
Haskin	0,0858	11	8
Redirijare	0,0648	53	0
Redirijare rapidă	0,0228	11	12
Propunere	0,0228	11	8

Tabel 7.2 – Numărul de pachete pierdute în funcţie de rata de transfer

Metoda de recuperare	Numărul de pachete pierdute				
	Rata de transfer				
	0,1 MB/s	0,5 MB/s	1MB/s	5 MB/s	10 MB/s
Makam	74	70	70	350	702
Haskin	11	10	10	50	504
Redirijare	53	51	51	251	530
Redirijare rapidă	11	10	10	50	101
Propunere	11	10	10	50	101

Prin varierea ratei de transfer nu se afecteaza numărul de resurse rezervate şi nici timpul de întrerupere al serviciului nu are decât variaţii foarte mici şi de aceea nu ar fi avut sens prezentarea acestor date. Pentru rate de transfer de 0,1 MB/s si 0,5 MB/s nu avem diferenţe mari între numărul de pachete pierdute deoarece nu se variază decât dimensiunea acestora şi nu numărul lor.

Pentru verificarea rezultatelor am folosit şi o reţea cu doar 7 noduri MPLS, dar rezultatele obţinute au fost asemănătoare şi prezentarea lor nu ar fi avut sens.

7.1 Timpul de întrerupere al serviciului folosind diverse metode de recuperare

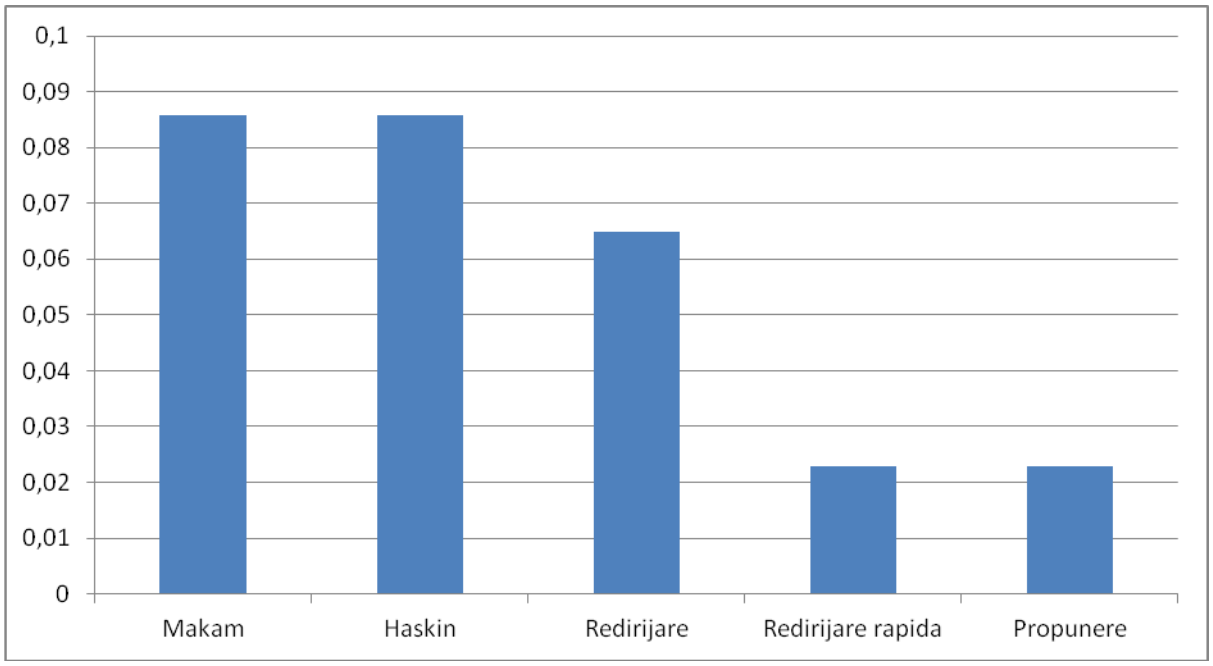

Figura 7.1: Timpul de întrerupere al serviciului metodelor de recuperare

Timpul de întrerupere al serviciului se măsoară ca timpul trecut de la recepţionarea ultimului pachet în nodul 10, ca rezultat al căderii legăturii, până când soseşte primul pachet dupa ce a fost restabilit traficul.

După cum se observă din figura de mai sus, metodele Makam, Haskin şi metoda redirijării nu sunt potrivite pentru transferul de voce deoarece au timpul de întrerupere al serviciului mai mare de 60ms. Orice întarziere mai mare de 60ms în semnalul de voce este percepută ca o pauză în vorbire şi este întrerupt fluxul vorbirii.

Pentru metoda redirijării rapide şi pentru soluţia propusă, timpii de întrerupere al serviciului sunt minimi deoare comutarea se face local pe traseele de rezervă stabilite anterior. Oriunde ar avea loc un eşec de-a lungul LSP-ul protejat, traficul este comutat pe rezerva corespunzătoare şi astfel timpul de întrerupere al serviciului va fi constant.

Metodele Makam şi Haskin au performanţe asemanatoare deoarece pentru a redirija traficul trebuie notificat nodul de intrare despre eşecul reţelei. Cu cât eşecul se produce mai departe de nodul de intrare, cu atât va fi mai mare timpul de întrerupere al serviciului.

7.2 Numărul de pachete pierdute folosind diverse metode de recuperare

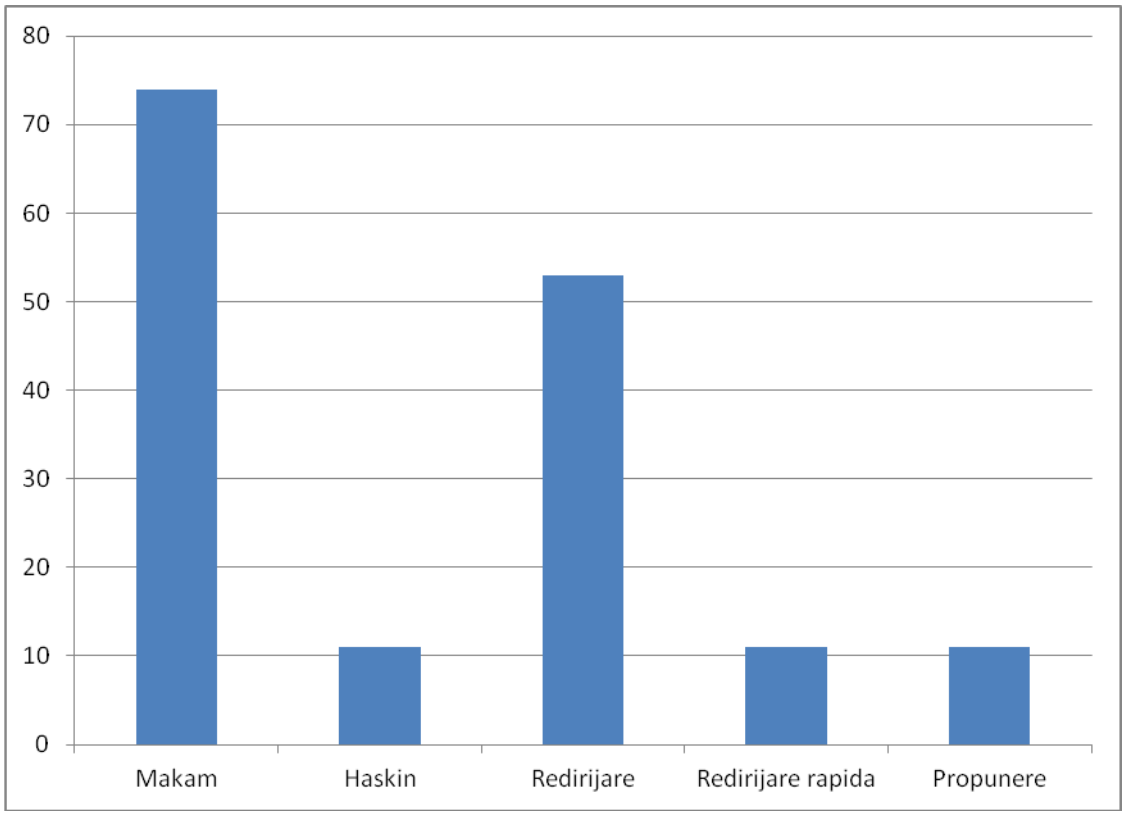

Figura 7.2: Numărul de pachete pierdute folosind diverse metode de recuperare

Numărul de pachetele pierdute se calculează ca diferenţa între numărul de pachete trimise de nodul sursă şi numărul de pachete recepţionate de nodul destinaţie.

Se observă în figura de mai sus că se pierd multe pachete folosind metoda Makam deoarece pachetele sunt trimise pe un traseu eşuat până când nu este notificat nodul de intrare să comute acest trafic pe calea de rezervă. Numărul de pachete pierdute va scădea cu cât esecul se produce mai aproape de nodul de intrare.

Metoda redirijării trebuie sa gasescă un traseu alternativ după ce a fost detectat un eşec şi astfel pachetele se vor pierde până când nu se va găsi un traseu potrivit.

Numărul de pachete pierdute este minim în cazul metodelor Haskin, redirijării rapide şi soluţiei propuse deoarece traficul este comutat pe căile de rezervă în momentul în care a fost detectat un eşec. Folosind metoda Haskin, ar putea fi probleme cu protocoale de nivel superior deoarece pachetele nu vor sosi in ordinea în care au fost trimise.

7.3 Numărul de resurse rezervate folosind diverse metode de recuperare

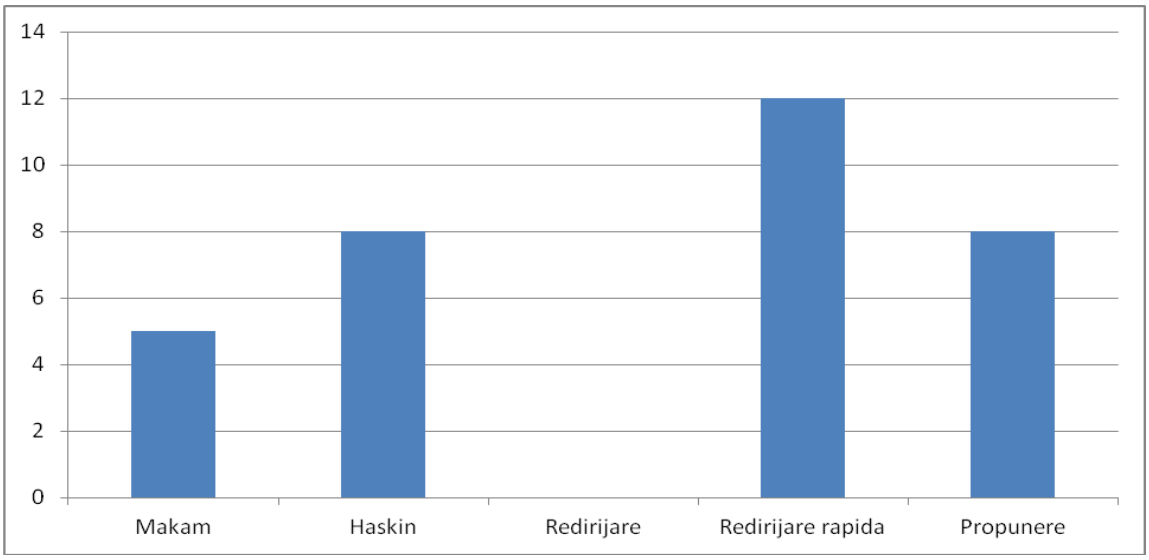

Figura 7.3: Numărul de resurse rezervate folosind diverse metode de recuperare

Numărul de resurse rezervate se determină ca fiind numărul total de legături din toate LSP-urile create şi nu numărul de LSP-uri create.

În exempul folosit, modelul Makam foloseşte 5 rezervări de resurse. Modelul Haskin va folosi întotdeauna mai multe deoarece, în plus faţă de traseul global de rezervă, va avea nevoie de un traseu pentru traficul inversat. Modelul redirijării nu foloseşte deloc resurse rezervate anterior, deoarece calea de rezervă se va crea când va fi nevoie, la producerea unui eşec.

Numărul de LSP-uri folosit de metodele redirijării rapide şi soluţiei propuse este identic şi egal cu N-1, N fiind numărul de noduri din LSP-ul protejat. Deoarece metoda redirijării rapide caută să găsească un LSP de rezervă pentru fiecare nod din traseul principal şi apoi să revină la traseul principal se folosesc mai multe resurse faţă de soluţia propusă.

În acest exemplu, numărul de resurse rezervate este acelaşi pentru modelul Haskin şi pentru soluţia propusă deoarece avem o distanţa de un singur hop între nodurile de pe traseul principal şi nodurile de pe LSP-ul alternativ. Într-o reţea de mari dimensiuni, numărul de resurse rezervate va fi mai mare pentru soluţia propusă decât metoda Haskin şi mult mai mare pentru metoda redirijării rapide.

Numărul de resurse rezervate depinde de topologia reţelei. Deşi metoda redirijării rapide cu rezervă unu la unu şi unificare foloseşte mai puţine resurse, acesta porneşte de la premisa ca nu putem avea mai mult de un singur eşec la un moment dat.

7.4 Numărul de pachete pierdute în funcţie de rata de transfer

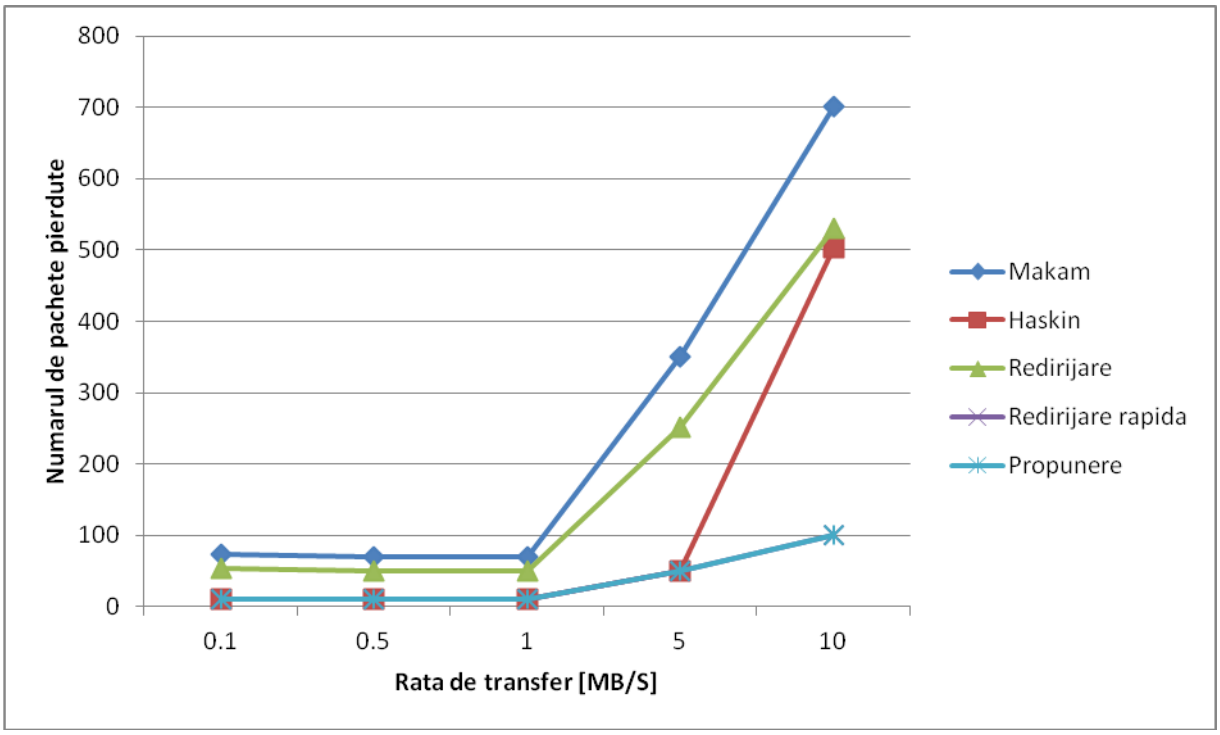

Figura 7.4: Numărul de pachete pierdute în funcţie de rata de transfer

În figura de mai sus, pe axa orizontală avem dimensiunea pachetelor şi pe axa verticală avem numărul de pachete pierdute. După cum se observă, cu cât creşte rata de transfer, cu atât vor fi mai multe pachete pierdute.

Pentru metodele redirijarii rapide şi soluţiei propuse avem aceleaşi valori de pachete pierdute şi graficele se suprapun. Defapt singurele pachete care se pierd la aceste două metode sunt pachetele care se află pe legătură în momentul când aceasta eşuează deoarece timpul de deţectie al eşecului este 0.

Metoda Makam are pierderi foarte mari de pachete şi nu este potrivită pentru reţele în care rata de transfer este foarte mare. Cam acelaşi lucru se poate spune şi despre metoda redirijării, deşi pierderile sunt mai mici.

Metoda Haskin trebuie să inverseze traficul spre nodul de intrare iar la sosire este amestecat cu traficul de pe LSP-ul alternativ şi se depaşeşte lăţimea de bandă pentru acest LSP alternativ. Astfel apare congestia traficului şi, în exemplul meu, pachetele au fost aruncate deoarece nu am folosit nici un mecanism de tratare a congestiei. Din această cauză se observă în figură că pentru rate mari de transfer, se pierd mai multe pachete cu metoda Haskin decât cu metoda redirijării rapide sau metoda propusă.

Concluzii

Tehnologia MPLS a combinat capabilităţile de comutare a etichetei şi noţiunea de circuit virtual specifice ATM cu orientarea pachetelor, specifică tehnologiei IP. Principalul avantaj MPLS îl reprezintă flexibilitatea în sensul că a fost proiectat să lucreze într-un mediu cu protocoale multiple.

Într-o reţea WAN modernă, trebuie ca utilizatorii să nu aibă de suferit în cazul în care are loc o cădere a unui nod sau a unei legături. În mod ideal ar trebui ca o reţea să aibă redundanţa totală, dar practic acest acest lucru nu este întotdeauna posibil din cauza costului. Pentru a studia o reţea fără a fi nevoie de implementarea ei fizică, am studiat comparativ diverse simulatoare de reţea şi pot afirma că sunt nişte instrumente foarte utile în domeniul reţelisticii.

În această lucrare am analizat diverse metode de recuperare a traficului în reţele MPLS şi am propus o soluţie pentru îmbunătaţirea numărului de resurse folosite în metoda redirijării rapide fără a sacrifica capabilitatea de a trata mai multe eşecuri la un moment dat. În topologia utilizată, s-a realizat o îmbunataţire de 33,3% dar în alte topologii poate varia foarte mult.

În final, pot afirma că nu există o metodă de recuperare a traficului care să fie cea mai bună în mod exclusiv, fiecare metodă are avantajele şi dezavantajele sale şi acestă lucrare ar putea ajuta administratorii de reţele să facă o alegere bună în această privinţă.

Bibliografie şi referinţe

[1]- Andrew S. Tanenbaum - Computer networks, Prentice Hall Professional Technical Reference, 4th edition, 2002, pag. 37-39, 75-78, 392-394;

[2]- Johan Martin Olof Petersson - MPLS Based Recovery Mechanics, University of Oslo, May 2005;

[3]- Lemma Hundessa Gonfa - Enhanced Fast Rerouting Mechanisms for Protected Traffic in MPLS Networks, University of Catalunya, February 2003;

[4]- Dr. Yogesh P Kosta, Minesh P Thaker,Bhaumik Nagar - Performance Comparison of Rerouting Schemes of Multi Protocol Label Switching Network, ACEEE Int. J. on Control System and Instrumentation, Vol. 02, No. 01, Feb 2011;

[5]- Gaeil Ahn, Jongsoo Jang, Woojik Chun - An Efficient Rerouting Scheme for MPLS-Based Recovery and Its Performance Evaluation, Telecommunication Systems 19:3,4, 481–495, 2002;

[6]- Ravi Tomar, Anuj Kumar Yadav, Deep Kumar, Prashant Chaudhary - MPLS-A Fail Safe Approach to Congestion Control, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 7, July 2012;

[7]- Ed Harrison, Ben Miller, Adrian Farrel - Protection And Restoration In MPLS Networks, An examination of the methods for protecting MPLS LSPs against failures of network resources, First issued October 2001;

[8]- K. Owens, V. Sharma, S. Makam, and C. Huang - A Path Protection Restoration Mechanism for MPLS Networks. Work in progress, Internet draft <draft-chang-mpls-protection-03.txt>, July 2001;

[9]- D. Haskin and R. Krishnan - A Method for Setting an Alternative Label Switched Paths to Handle Fast Reroute. Work in progress, Internet draft <draft-haskin-mpls-fast-reroute-05.txt>, November 2000;

[10]- P. Pan, G. Swallow, A.Atlas - "Fast Reroute Extensions to RSVP-TE for LSP Tunnels" Internet Draft draft-ietf-mpls-rsvp-lsp-fastreroute-06.txt, June 2004;

[11]- Ginsburg, 1996; Goralski, 1995; Ibe, 1997; Kim et al., 1994; at Stallings, 2000;

[12]- Cemal Kocak, Ismail Erturk, Huseyin Ekiz - Comparative Performance Analysis of MPLS over ATM and IP over ATM Methods for Multimedia Transfer Applications;

[13]- Gaeil Ahn, Woojik Chun – Overview of MPLS Network Simulator: Design and Implementation, Department of Computer Engineering, Chungnam National University, Korea;

[14]- John Cleary, Murray Pearson,Ian Graham, Brian Unger - High Performance Simulation for ATM Network Development, Department of Computer Science, University of Waikato, June 1996;

[15]- Mrs. Saba Siraj, Mr. Ajay Kumar Gupta, Mrs Rinku-Badgujar - Network Simulation Tools Survey, International Journal of Advanced Research in Computer and Communication Engineering, Vol. 1, Issue 4, June 2012;

[16]- Paul Meeneghan and Declan Delaney - An Introduction to NS, Nam and OTcl

Scripting, National University of Ireland, Maynooth,Maynooth, Co. Kildare, Ireland, Department of Computer Science, 2004;

[17]- Elias Weing¨artner, Hendrik vom Lehn and Klaus Wehrle - A performance comparison of recent network simulators, Distributed Systems Group, RWTH Aachen University,Aachen, Germany;

[18]- Jianli Pan - A Survey of Network Simulation Tools: Current Status and Future Developments, http://www.cse.wustl.edu/~jain/cse567-08/index.html;

[19]- http://en.wikipedia.org/wiki/WAN_optimization;

[20]- http://searchenterprisewan.techtarget.com/feature/WAN-optimization-techniques;

[21]- http://www.bcs.org/content/conwebdoc/6849;

[22]- http://en.verysource.com/code/2773401_1/test-reroute.tcl.html;

Anexe

Makam.tcl

```
set ns [new Simulator]

set nf [open MPLSexample.nam w]
$ns namtrace-all $nf
set f0 [open MPLSexample.tr w]
$ns trace-all $f0

proc finish {} {
      global ns nf f0
      $ns flush-trace
      close $nf
      close $f0
        exec nam MPLSexample.nam &
      exit 0
}

#Agent/LDP set trace_ldp_ 1
#Classifier/Addr/MPLS set trace_mpls_ 1

#$ns rtproto DV

set node0  [$ns node]
$ns node-config -MPLS ON
set LSR1   [$ns node]
set LSR2   [$ns node]
set LSR3   [$ns node]
set LSR4   [$ns node]
set LSR5   [$ns node]
set LSR6   [$ns node]
set LSR7   [$ns node]
set LSR8   [$ns node]
set LSR9   [$ns node]
$ns node-config -MPLS OFF
set node10  [$ns node]

$ns duplex-link $node0 $LSR1  100Mb  10ms DropTail
$ns duplex-link $LSR1  $LSR2  100Mb  10ms DropTail
$ns duplex-link $LSR2  $LSR3  100Mb  10ms DropTail
$ns duplex-link $LSR3  $LSR4  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR5  100Mb  10ms DropTail
$ns duplex-link $LSR5  $node10  100Mb  10ms DropTail
$ns duplex-link $LSR1  $LSR6  100Mb  10ms DropTail
$ns duplex-link $LSR6  $LSR7  100Mb  10ms DropTail
$ns duplex-link $LSR7  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR8  $LSR9  100Mb  10ms DropTail
$ns duplex-link $LSR9  $LSR5  100Mb  10ms DropTail
$ns duplex-link $LSR2  $LSR7  100Mb  10ms DropTail
$ns duplex-link $LSR3  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR9  100Mb  10ms DropTail

#
# configure ldp agents on all mpls nodes
#
for {set i 1} {$i < 10} {incr i} {
```

```
      set a LSR$i
      for {set j [expr $i+1]} {$j < 10} {incr j} {
            set b LSR$j
            eval $ns LDP-peer $$a $$b
      }
      set m [eval $$a get-module "MPLS"]
      $m enable-reroute "new"
}

#configuram culori pentru diferitele pachete LDP
$ns ldp-request-color       blue
$ns ldp-mapping-color       red
$ns ldp-withdraw-color      magenta
$ns ldp-release-color       orange
$ns ldp-notification-color  yellow

Classifier/Addr/MPLS enable-on-demand
Classifier/Addr/MPLS enable-ordered-control

set src0 [new Application/Traffic/CBR]
set udp0 [new Agent/UDP]
$src0 attach-agent $udp0
$ns attach-agent $node0 $udp0
$src0 set packetSize_ 10000
$src0 set interval_ 0.001

set Dst0 [new Agent/LossMonitor]
$ns attach-agent $node10 $Dst0

$ns connect $udp0 $Dst0

$ns at 0.1  "$src0 start"

#protected LSP
$ns at 0.1  "[$LSR1 get-module MPLS] make-explicit-route 5 2_3_4_5 1000 -1"
$ns at 0.1  "[$LSR1 get-module MPLS] flow-erlsp-install   10 -1   1000"

#alternate LSP
$ns at 0.1  "[$LSR1 get-module MPLS] make-explicit-route 5 6_7_8_9_5 1001 -1"

#link failure
$ns rtmodel-at 0.3 down $LSR4 $LSR5
$ns at 0.3  "[$LSR4 get-module MPLS] make-explicit-route 1 3_2_1 1002 -1"
$ns at 0.3  "[$LSR4 get-module MPLS] ldp-trigger-by-withdraw 10 -1"

#change flow to backup LSP
$ns at 0.33  "[$LSR1 get-module MPLS] flow-erlsp-install   10 -1   1001"

#restore link
$ns rtmodel-at 0.5 up   $LSR4 $LSR5

$ns at 0.6 "$src0 stop"

$ns at 1.0 "finish"
$ns run
```

Haskin.tcl

```
set ns [new Simulator]

set nf [open MPLSexample.nam w]
$ns namtrace-all $nf
set f0 [open MPLSexample.tr w]
$ns trace-all $f0

proc finish {} {
      global ns nf f0
      $ns flush-trace
      close $nf
      close $f0
        exec nam MPLSexample.nam &
      exit 0
}

#Agent/LDP set trace_ldp_ 1
#Classifier/Addr/MPLS set trace_mpls_ 1

$ns rtproto DV

set node0  [$ns node]
$ns node-config -MPLS ON
set LSR1   [$ns node]
set LSR2   [$ns node]
set LSR3   [$ns node]
set LSR4   [$ns node]
set LSR5   [$ns node]
set LSR6   [$ns node]
set LSR7   [$ns node]
set LSR8   [$ns node]
set LSR9   [$ns node]
$ns node-config -MPLS OFF
set node10  [$ns node]

$ns duplex-link $node0 $LSR1  100Mb  10ms DropTail
$ns duplex-link $LSR1  $LSR2  100Mb  10ms DropTail
$ns duplex-link $LSR2  $LSR3  100Mb  10ms DropTail
$ns duplex-link $LSR3  $LSR4  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR5  100Mb  10ms DropTail
$ns duplex-link $LSR5  $node10  100Mb  10ms DropTail
$ns duplex-link $LSR1  $LSR6  100Mb  10ms DropTail
$ns duplex-link $LSR6  $LSR7  100Mb  10ms DropTail
$ns duplex-link $LSR7  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR8  $LSR9  100Mb  10ms DropTail
$ns duplex-link $LSR9  $LSR5  100Mb  10ms DropTail
$ns duplex-link $LSR2  $LSR7  100Mb  10ms DropTail
$ns duplex-link $LSR3  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR9  100Mb  10ms DropTail

#
# configure ldp agents on all mpls nodes
#
for {set i 1} {$i < 10} {incr i} {
      set a LSR$i
      for {set j [expr $i+1]} {$j < 10} {incr j} {
            set b LSR$j
            eval $ns LDP-peer $$a $$b
```

```
      }
      set m [eval $$a get-module "MPLS"]
      $m enable-reroute "new"
}

#configuram culori pentru diferitele pachete LDP
$ns ldp-request-color       blue
$ns ldp-mapping-color       red
$ns ldp-withdraw-color      magenta
$ns ldp-release-color       orange
$ns ldp-notification-color  yellow

Classifier/Addr/MPLS enable-on-demand
Classifier/Addr/MPLS enable-ordered-control

set src0 [new Application/Traffic/CBR]
set udp0 [new Agent/UDP]
$src0 attach-agent $udp0
$ns attach-agent $node0 $udp0
$src0 set packetSize_ 10000
$src0 set interval_ 0.001

set Dst0 [new Agent/LossMonitor]
$ns attach-agent $node10 $Dst0

$ns connect $udp0 $Dst0

$ns at 0.1  "$src0 start"

#alternative LSP
$ns at 0.1  "[$LSR1 get-module MPLS] make-explicit-route 5 6_7_8_9_5 1000 -1"

#backward LSP
$ns at 0.2  "[$LSR5 get-module MPLS] make-explicit-route 5 4_3_2_1_1000 1005 -1"

#link failure
$ns rtmodel-at 0.3 down $LSR4 $LSR5

#change the working LSP to backward LSP
$ns at 0.3  "[$LSR4 get-module MPLS] flow-erlsp-install 10 -1 1005"
$ns at 0.33  "[$LSR1 get-module MPLS] flow-erlsp-install 10 -1 1000"

#restore link
$ns rtmodel-at 0.5 up   $LSR4 $LSR5

$ns at 0.6 "$src0 stop"

$ns at 1.0 "finish"
$ns run
```

Reroute.tcl

```
set ns [new Simulator]

set nf [open MPLSexample.nam w]
$ns namtrace-all $nf
set f0 [open MPLSexample.tr w]
$ns trace-all $f0

proc finish {} {
      global ns nf f0
      $ns flush-trace
      close $nf
      close $f0
        exec nam MPLSexample.nam &
      exit 0
}

#Agent/LDP set trace_ldp_ 1
#Classifier/Addr/MPLS set trace_mpls_ 1

$ns rtproto DV

set node0  [$ns node]
$ns node-config -MPLS ON
set LSR1   [$ns node]
set LSR2   [$ns node]
set LSR3   [$ns node]
set LSR4   [$ns node]
set LSR5   [$ns node]
set LSR6   [$ns node]
set LSR7   [$ns node]
set LSR8   [$ns node]
set LSR9   [$ns node]
$ns node-config -MPLS OFF
set node10  [$ns node]

$ns duplex-link $node0 $LSR1  100Mb  10ms DropTail
$ns duplex-link $LSR1  $LSR2  100Mb  10ms DropTail
$ns duplex-link $LSR2  $LSR3  100Mb  10ms DropTail
$ns duplex-link $LSR3  $LSR4  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR5  100Mb  10ms DropTail
$ns duplex-link $LSR5  $node10  100Mb  10ms DropTail
$ns duplex-link $LSR1  $LSR6  100Mb  10ms DropTail
$ns duplex-link $LSR6  $LSR7  100Mb  10ms DropTail
$ns duplex-link $LSR7  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR8  $LSR9  100Mb  10ms DropTail
$ns duplex-link $LSR9  $LSR5  100Mb  10ms DropTail
$ns duplex-link $LSR2  $LSR7  100Mb  10ms DropTail
$ns duplex-link $LSR3  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR9  100Mb  10ms DropTail

#
# configure ldp agents on all mpls nodes
#
for {set i 1} {$i < 10} {incr i} {
      set a LSR$i
      for {set j [expr $i+1]} {$j < 10} {incr j} {
            set b LSR$j
```

```
            eval $ns LDP-peer $$a $$b
      }
      set m [eval $$a get-module "MPLS"]
      $m enable-reroute "new"
}

#configuram culori pentru diferitele pachete LDP
$ns ldp-request-color       blue
$ns ldp-mapping-color       red
$ns ldp-withdraw-color      magenta
$ns ldp-release-color       orange
$ns ldp-notification-color  yellow

Classifier/Addr/MPLS enable-on-demand
Classifier/Addr/MPLS enable-ordered-control
[$LSR1 get-module "MPLS"] enable-data-driven

set src0 [new Application/Traffic/CBR]
set udp0 [new Agent/UDP]
$src0 attach-agent $udp0
$ns attach-agent $node0 $udp0
$src0 set packetSize_ 10000
$src0 set interval_ 0.001

set Dst0 [new Agent/LossMonitor]
$ns attach-agent $node10 $Dst0

$ns connect $udp0 $Dst0

$ns at 0.1  "$src0 start"

#protected LSP
$ns at 0.1  "[$LSR1 get-module MPLS] make-explicit-route 5 2_3_4_5 1000 -1"
$ns at 0.2  "[$LSR1 get-module MPLS] flow-erlsp-install   10 -1   1000"

#link failure
$ns rtmodel-at 0.3 down $LSR4 $LSR5

#restore link
$ns rtmodel-at 0.5 up   $LSR4 $LSR5

$ns at 0.6 "$src0 stop"

$ns at 1.0 "finish"
$ns run
```

Fast_reroute.tcl

```
set ns [new Simulator]

set nf [open MPLSexample.nam w]
$ns namtrace-all $nf
set f0 [open MPLSexample.tr w]
$ns trace-all $f0

proc finish {} {
      global ns nf f0
      $ns flush-trace
      close $nf
      close $f0
        exec nam MPLSexample.nam &
      exit 0
}

#Agent/LDP set trace_ldp_ 1
#Classifier/Addr/MPLS set trace_mpls_ 1

$ns rtproto DV

set node0  [$ns node]
$ns node-config -MPLS ON
set LSR1   [$ns node]
set LSR2   [$ns node]
set LSR3   [$ns node]
set LSR4   [$ns node]
set LSR5   [$ns node]
set LSR6   [$ns node]
set LSR7   [$ns node]
set LSR8   [$ns node]
set LSR9   [$ns node]
$ns node-config -MPLS OFF
set node10  [$ns node]

$ns duplex-link $node0 $LSR1  100Mb  10ms DropTail
$ns duplex-link $LSR1  $LSR2  100Mb  10ms DropTail
$ns duplex-link $LSR2  $LSR3  100Mb  10ms DropTail
$ns duplex-link $LSR3  $LSR4  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR5  100Mb  10ms DropTail
$ns duplex-link $LSR5  $node10  100Mb  10ms DropTail
$ns duplex-link $LSR1  $LSR6  100Mb  10ms DropTail
$ns duplex-link $LSR6  $LSR7  100Mb  10ms DropTail
$ns duplex-link $LSR7  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR8  $LSR9  100Mb  10ms DropTail
$ns duplex-link $LSR9  $LSR5  100Mb  10ms DropTail
$ns duplex-link $LSR2  $LSR7  100Mb  10ms DropTail
$ns duplex-link $LSR3  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR9  100Mb  10ms DropTail

#
# configure ldp agents on all mpls nodes
#
for {set i 1} {$i < 10} {incr i} {
      set a LSR$i
      for {set j [expr $i+1]} {$j < 10} {incr j} {
            set b LSR$j
            eval $ns LDP-peer $$a $$b
```

```
      }
      set m [eval $$a get-module "MPLS"]
      $m enable-reroute "new"
}

#configuram culori pentru diferitele pachete LDP
$ns ldp-request-color       blue
$ns ldp-mapping-color       red
$ns ldp-withdraw-color      magenta
$ns ldp-release-color       orange
$ns ldp-notification-color  yellow

Classifier/Addr/MPLS enable-on-demand
Classifier/Addr/MPLS enable-ordered-control
[$LSR1 get-module "MPLS"] enable-data-driven
[$LSR3 get-module "MPLS"] enable-data-driven

set src0 [new Application/Traffic/CBR]
set udp0 [new Agent/UDP]
$src0 attach-agent $udp0
$ns attach-agent $node0 $udp0
$src0 set packetSize_ 10000
$src0 set interval_ 0.001

set Dst0 [new Agent/LossMonitor]
$ns attach-agent $node10 $Dst0

$ns connect $udp0 $Dst0

$ns at 0.1  "$src0 start"

#protected LSP
$ns at 0.1  "[$LSR1 get-module MPLS] make-explicit-route 5 2_3_4_5 1000 -1"
$ns at 0.1  "[$LSR1 get-module MPLS] flow-erlsp-install   10 -1   1000"

#LSR1's backup
$ns at 0.1  "[$LSR1 get-module MPLS] make-explicit-route 3 6_7_8_3 1001 -1"

#LSR2's backup
$ns at 0.1  "[$LSR2 get-module MPLS] make-explicit-route 4 7_8_4 1002 -1"

#LSR3's backup
$ns at 0.1  "[$LSR3 get-module MPLS] make-explicit-route 5 8_9_5 1003 -1"

#LSR4's backup
$ns at 0.1  "[$LSR4 get-module MPLS] make-explicit-route 5 9_5 1004 -1"

#link failure
$ns rtmodel-at 0.3 down $LSR4 $LSR5

#change flow to backup LSP
$ns at 0.3  "[$LSR4 get-module MPLS] flow-erlsp-install 10 -1   1004"

#restore link
$ns rtmodel-at 0.5 up   $LSR4 $LSR5

$ns at 0.6 "$src0 stop"

$ns at 1.0 "finish"
$ns run
```

Propunere.tcl

```
set ns [new Simulator]

set nf [open MPLSexample.nam w]
$ns namtrace-all $nf
set f0 [open MPLSexample.tr w]
$ns trace-all $f0

proc finish {} {
      global ns nf f0
      $ns flush-trace
      close $nf
      close $f0
        exec nam MPLSexample.nam &
      exit 0
}

#Agent/LDP set trace_ldp_ 1
#Classifier/Addr/MPLS set trace_mpls_ 1

$ns rtproto DV

set node0  [$ns node]
$ns node-config -MPLS ON
set LSR1   [$ns node]
set LSR2   [$ns node]
set LSR3   [$ns node]
set LSR4   [$ns node]
set LSR5   [$ns node]
set LSR6   [$ns node]
set LSR7   [$ns node]
set LSR8   [$ns node]
set LSR9   [$ns node]
$ns node-config -MPLS OFF
set node10  [$ns node]

$ns duplex-link $node0 $LSR1  100Mb  10ms DropTail
$ns duplex-link $LSR1  $LSR2  100Mb  10ms DropTail
$ns duplex-link $LSR2  $LSR3  100Mb  10ms DropTail
$ns duplex-link $LSR3  $LSR4  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR5  100Mb  10ms DropTail
$ns duplex-link $LSR5  $node10  100Mb  10ms DropTail
$ns duplex-link $LSR1  $LSR6  100Mb  10ms DropTail
$ns duplex-link $LSR6  $LSR7  100Mb  10ms DropTail
$ns duplex-link $LSR7  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR8  $LSR9  100Mb  10ms DropTail
$ns duplex-link $LSR9  $LSR5  100Mb  10ms DropTail
$ns duplex-link $LSR2  $LSR7  100Mb  10ms DropTail
$ns duplex-link $LSR3  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR8  100Mb  10ms DropTail
$ns duplex-link $LSR4  $LSR9  100Mb  10ms DropTail

#
# configure ldp agents on all mpls nodes
#
for {set i 1} {$i < 10} {incr i} {
      set a LSR$i
      for {set j [expr $i+1]} {$j < 10} {incr j} {
            set b LSR$j
            eval $ns LDP-peer $$a $$b
```

```
      }
      set m [eval $$a get-module "MPLS"]
      $m enable-reroute "new"
}

#configuram culori pentru diferitele pachete LDP
$ns ldp-request-color       blue
$ns ldp-mapping-color       red
$ns ldp-withdraw-color      magenta
$ns ldp-release-color       orange
$ns ldp-notification-color  yellow

Classifier/Addr/MPLS enable-on-demand
Classifier/Addr/MPLS enable-ordered-control
[$LSR1 get-module "MPLS"] enable-data-driven

set src0 [new Application/Traffic/CBR]
set udp0 [new Agent/UDP]
$src0 attach-agent $udp0
$ns attach-agent $node0 $udp0
$src0 set packetSize_ 10000
$src0 set interval_ 0.001

set Dst0 [new Agent/LossMonitor]
$ns attach-agent $node10 $Dst0

$ns connect $udp0 $Dst0

$ns at 0.1  "$src0 start"

#protected LSP
$ns at 0.1  "[$LSR1 get-module MPLS] make-explicit-route 5 2_3_4_5 1000 -1"
$ns at 0.1  "[$LSR1 get-module MPLS] flow-erlsp-install   10 -1   1000"

#alternate LSP
$ns at 0.1  "[$LSR1 get-module MPLS] make-explicit-route 5 6_7_8_9_5 1001 -1"

#LSR2's backup
$ns at 0.1  "[$LSR2 get-module MPLS] make-explicit-route 7 7 1002 -1"

#LSR3's backup
$ns at 0.1  "[$LSR3 get-module MPLS] make-explicit-route 8 8 1003 -1"

#LSR4's backup
$ns at 0.1  "[$LSR4 get-module MPLS] make-explicit-route 9 9 1004 -1"

#link failure
$ns rtmodel-at 0.3 down $LSR4 $LSR5

#change flow to backup LSP
$ns at 0.3  "[$LSR4 get-module MPLS] flow-erlsp-install   10 -1   1004"

#restore link
$ns rtmodel-at 0.5 up   $LSR4 $LSR5

$ns at 0.6 "$src0 stop"

$ns at 1.0 "finish"
$ns run
```