

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2682

**ISPITIVANJE PROTOKOLA CIP U
BEŽIČNIM MREŽAMA**

Vedran Mikac

Zagreb, svibanj 2006.

Mentor rada:

Prof.dr.sc. Alen Bažant

Voditelj rada:

Dr. Sc. Željko Ilić

Sadržaj

Uvod	1
1. Protokol CIP	2
1.1. Poboljšanja na protokolu	6
1.1.1. Potvrde kontrolnih poruka	6
1.1.2. Detekcija kvara linka	8
2. Implementacija protokola CIP u programskom alatu OpNet 8.0.C	12
2.1. Hjerarhijska struktura programskog alata OpNet	12
2.2. Osnovni preduvjeti za implementaciju protokola CIP	14
2.3. Omogućavanje prebacivanja veze (engl. <i>handoff</i>)	14
2.4. Umetanje procesa cip u protokolni složaj mobilne stanice	17
2.4.1. Objasnjenje koncepta stanja na konkretnom primjeru	19
2.4.2. Vrste stanja i objasnjenje prijelaza	19
2.4.3. Specifičnosti simulacije temeljene na diskretnim događajima u programskom paketu OpNet	21
2.4.4. Opisi osnovnih stanja	22
2.5. Komunikacija procesa cip i procesa mac	23
2.6. Prilagodba procesa wlan_mac	24
3. Opis implementiranih procesa i čvorova	26
3.1. Čvor vm_cip_switch2	26
3.1.1. Proces vm_switch_cip	27
3.2. Čvor wlan_ethernet_router_adv_vm6	30
3.2.1. Proces vm_ap_cip	31
3.3. Čvor wlan_wkstn_adv_vm4	33
3.3.1. Proces ip_arp_v4_vm4	34
3.3.2. Proces vm_cip	36
3.4. Čvor cip_failure	38

3.4.1. Proces cip_fail_recover	38
4. Rezultati simulacije	40
4.1. Polumeko i tvrdo prebacivanje veze	40
4.2. Detekcija nedostupnosti linka prema višem čvoru.....	46
5. Zaključak	50
6. Literatura	51

Uvod

Bežične lokalne mreže (engl. Wireless Local Area Networks) pojavile su se prije nekoliko godina i to s donošenjem standarda IEEE 802.11 (1997. godina). Novi tip lokalne mreže popularno je nazvan Wi-Fi (engl. Wireless Fidelity). Općenito gledano, Wi-Fi je naziv za bilo koji tip 802.11 mreže. U počecima korištenja bežičnih lokalnih mreža mobilnost korisnika je bila jako mala. Smanjenjem cijena dlanovnika, mobilnost korisnika koji koriste Wi-Fi mrežu sve se više povećavala. Pojavom pružatelja VoIP (engl. Voice over IP) usluga te pojmom Internet radio stanica i ostalih usluga koje zahtijevaju kvalitetnu mrežu, problem prebacivanja s trenutne pristupne točke na povoljniju pristupnu točku postao je sve izraženiji. Kako su svi dotadašnji protokoli bili predviđeni za korištenje u žičanim mrežama gdje takvi problemi ne postoje, korisnici su bili nezadovoljni ponuđenom uslugom. Primjer takvog protokola je TCP (engl. Transmission Control Protocol), koji u slučaju gubljenja paketa prepostavlja da je do gubitka došlo zbog zagušenja mreže. Ta pretpostavka na bežičnom mediju nije ispravna, pa se nepotrebno usporava slanje TCP paketa. Uzastopnim gubljenjem nekoliko TCP paketa veza se može usporiti do te mjere da postane gotovo pa neupotrebljiva. Prvi način rješavanja tog problema je preinaka protokola TCP. Neka od takvih rješenja su TCP–Snoop [13] i M-TCP (engl. Mobile TCP). Drugi način je rješavanje problema na nižim slojevima. Takva rješenja nude protokoli za mikro-mobilnost. Jedan od takvih protokola je i protokol CIP (engl. Cellular IP) koji će biti razmatran u ovom radu. CIP je zamišljen kao jednostavan protokol, koji smanjuje gubitak paketa za vrijeme prebacivanja veze korištenjem naprednije vrste prebacivanja veze nazvane polumeko prebacivanje (engl. semi-soft handover). U ovom radu bit će prikazana usporedba tradicionalnog tvrdog prebacivanja (engl. hard handover) i polumekog prebacivanja veze. Zbog ograničenja u standardu IEEE 802.11, te mreže ne mogu koristiti meko prebacivanje veze prisutno u UMTS mreži. CIP je protokol koji se može koristiti za LAN (engl. Local Area Network) i MAN (engl. Metropolitan Area Network) mreže. Budući da je za MAN mreže potrebno mnogo opreme, vjerojatnost kvara će se povećati do ne zanemarivih veličina, pa je u CIP uveden dodatak, koji donekle omogućava oporavak od kvara u vrlo kratkom vremenskom intervalu. U radu se također razmatra problem pouzdanosti zračnog medija, zbog čega je uveden drugi dodatak: potvrde kontrolnih paketa. Te potvrde omogućavaju pouzdanije prebacivanje i osiguravaju ispravan radi polumekog prebacivanja veze.

1. Protokol CIP

Protokol CIP (engl. *Cellular IP*) primjenjuje se u sustavima čiji se rad temelji na standardu IEEE 802.11. Osnovna ideja protokola je smanjiti gubitke paketa za vrijeme promjene pristupne točke (engl. *Access Point*, skraćeno AP) od strane mobilne stanice (engl. *Mobile station*, skraćeno MS). Ta minimizacija gubitaka paketa nužna je zbog sljedeće činjenice. Prilikom osmišljavanja protokola TCP (engl. *Transmission Control Protocol*), prepostavljalno se je da će se on koristiti u klasičnim Ethernet mrežama. U tom okruženju algoritmi koji se koriste u svrhu izbjegavanja zagušenja dobro funkcijiraju jer se prepostavlja da je do gubitka paketa došlo zbog zagušenja u mreži. Ta prepostavka nije zadovoljena u bežičnim mrežama gdje gubici paketa mogu nastati zbog drugih razloga od kojih su najvažniji:

1. smanjenje odnosa signal-šum (engl. *Signal to Noise Ratio*, skraćeno SNR),
2. vrlo složenih pojava u bežičnom mediju poput fedinga i višestaznog širenja signala
3. promjena pristupne točke – prebacivanje veze.

Protokol CIP namijenjen je za LAN (engl. *Local Area Networks*) i MAN (engl. *Metropolitan Area Networks*) mreže i zajedno s Mobile IP-om može se koristiti na širem geografskom području [6].

Svaka pristupna točka u pojedinim vremenskim intervalima emitira posebne, tzv. *beacon* pakete. Mobilna stanica osluškuje nailazak *beacon* paketa, te pomoću njih određuje koja pristupna točka ima za nju najpovoljniji odnos signal-šum. Ukoliko mobilna stanica šalje pakete u mrežu, prvo ih mora poslati svojoj pristupnoj točki, koja ih tada proslijeđuje do vršnog čvora u CIP mreži (engl. *gateway*). U tom čvoru odlučuje se da li će se paket poslati natrag prema nekoj drugoj mobilnoj stanici, ili će se poslati izvan CIP mreže. Druga uloga vršnog čvora je odjeljivanje CIP prometa od vanjske mreže. Iako se IP paketi ne tuneliraju niti enkapsuliraju u druge pakete unutar CIP mreže, čvorovi generiraju kontrolni promet (ICMP paketi) koji izvan CIP mreže nema smisla. Vršni čvor može služiti kao čvor u kojem se vodi sva statistika o tome koliko je prometa pojedina mobilna stanica slala i primala, te se na taj način može provoditi naplata korištenja mreže. Ako se koristi Mobile IP, tada u vršnom čvoru može završavati tunel između domaćeg agenta (engl. *Home Agent*, skraćeno HA) i stranog agenta (engl. *Foreign Agent*, skraćeno FA).

Paketi na putu između pristupne točke i vršnog čvora prolaze kroz jedan ili više CIP čvorova. CIP čvor može služiti za usmjeravanje prometa kroz CIP mrežu ili za komunikaciju s mobilnom stanicom. Budući da je već navedeno da pristupna točka komunicira sa mobilnom stanicom, u ovom tekstu će se za CIP čvor koristiti naziv CIP komutator, te će jedina njegova uloga biti usmjeravanje prometa unutar CIP mreže. Osnovna karakteristika CIP komutatora u odnosu na klasične komutatore ili usmjerivače je ta što on umjesto jedne ima dvije tablice usmjeravanja. Budući da svaka od tih dviju tablica ima posebnu namjenu, pretraživanje tih dviju tablica je brže nego pretraživanje samo jedne velike tablice. Prva tablica nazvana je aktivna tablica usmjeravanja (engl. *route cache*) i u njoj su zapisani podaci aktivnih mobilnih stanica, te mobilnih stanica koji često mijenjaju položaj. U tablici prozivanja (engl. *paging cache*) zapisani su podaci o neaktivnim ili rijetko aktivnim mobilnim stanicama. Razlog za podjelu na dvije tablice je slijedeći. Mnogi korisnici se za vrijeme pristupanja bežičnoj mreži ne kreću. Taj slučaj može se zamijetiti kada ljudi za pristup Internetu preko bežične mreže koriste prijenosna računala. Ako korisnik želi provjeriti svoje e-poštu, nakon spajanja na Internet i prikupljanja novih poruka, korisnik čita poštu te više nije aktivan. Prema specifikaciji protokola CIP, mobilna stanica prestaje biti aktivna 5 sekundi nakon što je poslala ili primila zadnji paket. Zapis se u aktivnoj tablici usmjeravanja briše 3 sekunde nakon zadnjeg osvježavanja. Naime, svaki puta kada mobilna stanica šalje paket prema mreži, zapisi vezani uz taj čvor se osvježavaju u svakom CIP čvoru na putu do vršnog čvora. Zbog toga nije potrebno često slati pakete za osvježavanje zapisa u dvjema tablicama, te se na taj način smanjuje opterećenje mreže. Ako mobilna stanica nakon dužeg perioda neaktivnosti ponovo prelazi u aktivno stanje, tada je prije slanja bilo kakvih paketa nužno poslati paket za ažuriranje aktivne tablice usmjeravanja (engl. *route update*). Taj paket ima ulogu stvaranja zapisa o mobilnoj stanicu u svim čvorovima na putu do vršnog čvora. Također postoji i paket za ažuriranje zapisa u tablici prozivanja (engl. *paging update*), koji se šalje prilikom prijavljivanja mobilnoj stanicu, promjene područja prozivanja (engl. *paging area*) ili prije nego što istekne zapis u tablici prozivanja. Područje prozivanja je područje koje obuhvaća područje nekoliko susjednih pristupnih točaka. Uobičajeno je da je za jedno područje prozivanja koristi jedan vršni čvor, gdje su prijavljene sve mobilne stanice koje se nalaze u tom području. Zapisi u tablici prozivanja ističu nakon tri minute.

Kretanjem mobilne stanice mijenja se odnos signal-šum. Kada taj odnos padne ispod određene razine, tada mobilna stanica prebací vezu na drugu pristupnu točku. Postoje tri vrste prebacivanja veze:

1. tvrdo prebacivanje (engl. *hard handover*)
2. polumeko prebacivanje (engl. *semi-soft handover*)
3. meko prebacivanje (engl. *soft handover*)

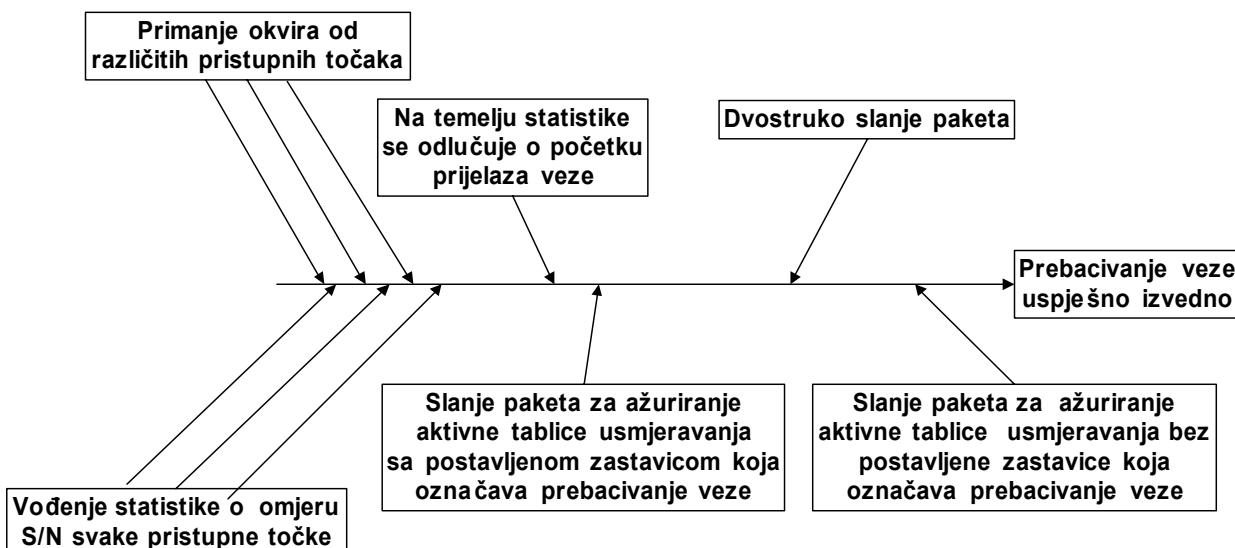
Tvrdo prebacivanje je vrsta prebacivanja veze prilikom kojeg mobilna stanica trenutačno prekida svoju vezu s jednom pristupnom točkom te se povezuje sa drugom pristupnom točkom. Prilikom takvog prebacivanja veze dolazi do velikog gubitka paketa. Taj gubitak uzrokuje pokretanje mehanizama izbjegavanja zagušenja u mreži te se TCP promet znatno usporava [8]. Budući da se u ovom radu polazi od prepostavke da se TCP neće mijenjati, ovakav način prebacivanja veze je najlošiji. Dobro je napomenuti da postoji implementacija TCP-a, pod nazivom M-TCP (engl. Mobile TCP), koji tolerira takve gubitke paketa [7].

Također, postoji veliki problem pri određivanju vremena prebacivanja veze. Obzirom da razina signala, kojeg mobilna stanica mjeri, podliježe Rayleighovoj razdiobi [10], odluka o prebacivanju veze ne može se donijeti na temelju kratkog mjerjenja. Donošenje odluke na temelju kratkog mjerjenja za posljedicu ima tzv. *ping-pong efekt*. Taj efekt očituje se u uzastopnim prebacivanjima veze s jedne pristupne točke na drugu i obratno. Spomenuti efekt može se izbjjeći ako se razmatra prosjek snage signala u nekom relativno kratkom vremenskom intervalu. Taj pristup također ima svoje nedostatke, koji se očituju u efektu kuta (engl. *corner effect*). Taj efekt pojavljuje se kada mobilna stanice u svom kretanju skrene iza kuta nekog većeg objekta. Tada se, zbog naglog pada snage signala, hitno mora provesti prebacivanje veze, kako se veza između mobilne stanice i pristupne točke ne bi prekinula zbog velike degradacije signala. Navedeni problem moguće je u potpunosti eliminirati dobrim planiranjem mreže. Tvrdo prebacivanje je nužno u mrežama u kojima mobilna stanica ne može istovremeno primati podatke od više pristupnih točaka. Ako to nije slučaj, koristi se meko prebacivanje.

Meko prebacivanje veze [2] koristi se u CDMA (engl. *Code Division Multiple Access*) sustavima. U tim sustavima, mobilna stanica istovremeno prima signal od više baznih stanica. Iako mobilna stanica prima signale od više različitih baznih stanica, samo ona stanica koja ima relativno najjači signal ima kontrolu nad pozivom. Nadalje, više baznih stanica može primati podatke od mobilne stanice i mobilna stanica može istodobno primati podatke od više baznih stanica. Prilikom primanja signala od različitih baznih stanica, za signale se pretpostavlja da su nastali zbog višestaznog širenja te se kombiniraju sa signalom trenutno najbolje bazne stanice. Kada se mobilna stanica kreće, signal do sada najbolje bazne stanice opada, a jačina signala bazne stanice čiji je signal bio smatrana jednim od signala višestaznog

širenja raste. U jednom trenutku, signal druge bazne stanice postaje glavni signal, a ostali se smatraju signalima nastalim zbog višestaznog širenja. Takva vrsta prebacivanja veze ima očitu prednost, jer mobilna stanica niti u jednom trenutku nije prekinula komunikaciju s mrežom. Takva vrsta prebacivanja veze koristi se u UMTS mrežama i nije ga moguće implementirati u IEEE 802.11 mrežama zbog određenih ograničenja u standardu. Postoji vrsta prebacivanja veze, koja se može implementirati u mrežama gdje meko prebacivanje nije moguće, a koje ima bolja svojstva od običnog tvrdog prebacivanja veze. Takvo prebacivanje veze zove se polumeko prebacivanje.

Polumeko prebacivanje veze [4] je svojevrsno proširenje metode tvrdog prebacivanja veze jer prije nego se provede tvrdo prebacivanje veze, mobilna stanica šalje paket u mrežu, kojim se najavljuje prebacivanje veze. Ovakav način prebacivanja veze koristi protokol CIP. Prije prebacivanja veze šalje se paket za ažuriranje aktivne tablice usmjeravanja s postavljenom zastavicom za prebacivanje veze. Taj paket putuje prema vršnom čvoru te se u prvom čvoru koji je zajednički novoj i staroj ruti stvara drugi zapis u tablici usmjeravanja. Svaki paket koji stigne u dotični čvor, a namijenjen je mobilnoj stanici koji je u procesu polumekog prebacivanja veze, šalje se po staroj ruti i s malim zakašnjnjem po novoj ruti. Trajanje tog zakašnjnjenja nije propisano protokolom, te će se u ovom radu prepostavljati da ono iznosi jednako koliko i prebacivanje veze sa jedne na drugu pristupnu točku. Na slici (Slika 1) prikazan je dijagram koji načelno opisuje *polumeko* prebacivanje veze.



Slika 1 : Polumeko prebacivanje veze

1.1. Poboljšanja na protokolu

Protokolu CIP dodana su dva poboljšanja: potvrde kontrolnih poruka i detekcija kvara linka između cip čvorova.

1.1.1. Potvrde kontrolnih poruka

Idejni začetnici protokola CIP nisu razmišljali o puzdanosti protokola. Česta pojava u bežičnom mediju je gubitak okvira. Iako MAC sloj u sebi ima ugrađenu funkcionalnost retransmisije, on nakon nekoliko uzastopnih neuspjelih pokušaja slanja može odbaciti okvir. Uobičajene vrijednosti za broj retransmisija su:

- 7 retransmisija za paket manji od 2347 okteta
- 4 retransmisije za paket veći od 2346 okteta

Duljina od 2347 okteta definirana je standardom i zove se RTS prag (engl. *RTS threshold*). Moglo bi se brzopletno zaključiti da bi broj retransmisija trebalo postaviti na čim veći broj. Učinak takve promijene imao bi dva negativna efekta. Zbog omogućenog većeg broja retransmisija, spremnici u mobilnim uređajima morali bi biti veći. Drugi problem odnosi se na TCP, koji nakon izgubljenog paketa izvodi algoritme za izbjegavanje zagruženja. Mnogo retransmisija može odgoditi proglašavanje paketa izgubljenim i može usporiti oporavak.

Vrijeme potrebno za 7 retransmisija manje je od 100 ms. Ako je važan kontrolni paket poslan baš u to vrijeme, tada bi moglo doći do ozbiljnijih problema pri radu protokola. Taj problem mogao bi biti uzrokovani zasjenjenjem ili destruktivnom interferencijom zbog višestaznog širenja.

Zbog opasnosti od gubitka kontrolnog paketa uveden je mehanizam potvrde na relaciji mobilna stanica – pristupna točka. Svaki kontrolni paket koji je poslan prema pristupnoj točki mora biti potvrđen u određenom vremenskom intervalu. U slučaju da paket nije potvrđen, kontrolni paket se ponovno šalje. Budući da kontrolnih paketa na zračnom sučelju ima malo, slanje potvrde za svaki kontrolni paket zanemarivo opterećuje vezu.

Kako bi se naglasila važnost potvrđivanja kontrolnih paketa, razmotriti će se slijedeći slučaj. Neka korisnik u na nekom mjestu gdje postoji bežična mreža pregledava svoju elektroničku poštu. Za vrijeme čitanja dužih poruka isteknu svi zapisi o korisniku u tablicama usmjeravanja. Nakon propisanog vremena, mobilna stanica šalje paket za osvježavanje tablica

prozivanja. Ako se taj paket izgubi, zapisi u tablici prozivanja neće biti osvježeni. Nakon brisanja zapisa iz obje tablice prozivanja, mreža više ne zna gdje se nalazi mobilni korisnik. Zbog toga će svi paketi namijenjeni za korisnika biti odbačeni u prvom čvoru koji posjeduje tablicu usmjeravanja i tablicu prozivanja (to je vjerojatno vršni čvor). Kada tablica prozivanja ne bi postojala, svi paketi bi se poslali prema svim susjednim čvorovima, pa bi korisnik u tom slučaju dobio paket.

Drugi slučaj pri kojem je još značajniji problem gubitka kontrolnog paketa je prebacivanje veze. Kao što je već spomenuto, ako se koristi polumeko prebacivanje veze, najprije je potrebno poslati jedan kontrolni paket na novu pristupnu točku. Taj kontrolni paket služi za udvostručavanje paketa koji su namijenjeni mobilnoj stanici. Druga kopija paketa se šalje sa malim zakašnjnjem, kako bi broj izgubljenih paketa bio što manji. Ako se taj kontrolni paket izgubi, umjesto polumekog prebacivanja izvršiti će se uobičajeno tvrdo prebacivanje.

Za vrijeme prebacivanja korisnik može primati veliku količinu podataka pa spremnici u pristupnoj točki mogu sadržavati više paketa odjednom. Ako bi se kontrolni paket tretirao kao i ostali podatkovni paketi, tada bi se on, budući da se radi o FIFO spremnicima, dodao na kraj spremnika. Ako je u repu previše podatkovnih paketa, vrijeme potrebno da bi se poslala potvrda znatno je veće od vremena koje mobilna stanica provede čekajući na potvrdu. Zbog toga nepotrebno ponovno šalje kontrolni paket, na koji se očekuje još jedna potvrda. Na taj način može doći do nepotrebnog slanja više od desetak kontrolnih poruka. Simulacijski rezultati pokazuju da, zbog čekanja u repu, mobilna stanica primi potvrdu tek nakon 14ms.

Kako bi se riješio navedeni problem, potrebno je uvesti prioritetno slanje kontrolnih poruka. Sve kontrolne poruke stavljuju se, umjesto na kraj, na početak repa. Zbog toga mobilna stanica primi potvrdu o uspješno posланом kontrolnom paketu za manje od 3ms. Tako kratko vrijeme odziva omogućava postavljanje vremenske kontrole na malenu vrijednost. U ovoj implementaciji protokola vremenska kontrola ističe nakon 10ms. Tako kratko vrijeme temelji se na pretpostavci da se istovremeno u repu ne nalazi više od jednog kontrolnog paketa. Ta tvrdnja bi u stvarnosti gotovo uvijek bila zadovoljena, jer se bežična mreža planira tako da se na jednoj pristupnoj točki nalazi maksimalno dvadeset do trideset korisnika [1].

Iako slanje potvrđnih paketa stvara dodatni kontrolni promet, potvrde u krajnjoj liniji omogućavaju smanjenje ukupnog kontrolnog prometa na zračnom mediju. Prema prijedlogu standarda [6], paketi za ažuriranje tablice usmjeravanja i paketi za ažuriranje tablice prozivanja šalju se u vremenskim intervalima tri puta kraćim od vremena potrebnog za

brisanje zapisa o usmjerenju odnosno zapisa o prozivanju u CIP komutatorima. Uvođenjem mehanizma potvrda, paketi se ne moraju slati tri puta češće, nego tik pred isticanje zapisa. Prema tome, paket za osvježavanje tablice prozivanja se neće slati svakih 60 sekundi, nego svakih 179, budući da je jedna sekunda i više nego dovoljna za mnogo retransmisija kontrolnog paketa.

Uvođenje prioritetnog slanja kontrolnih poruka unosi dodatno kašnjenje u slanje podatkovnih paketa. Maksimalno dodatno kašnjenje jednak je vremenu slanja paketa potvrde. CIP kontrolni paketi su u biti ICMP paketi duljine 20 okteta. nakon omatanja u IP paket koji ima minimalnu duljinu od 20 okteta, dodavanjem MAC zaglavlja, ukupna duljina okvira koji je potrebno poslati je 74 okteta. Vrijeme potrebno za slanje okvira te veličine u IEEE 802.11b mreži iznosi:

$$t_{\text{delay}} = \frac{74 \cdot 8}{11 \cdot 10^6} \text{ s} = 53,82 \mu\text{s}$$

Za IEEE 802.11g to vrijeme je $10,96 \mu\text{s}$. Iz ovog računa jasno je da prioritetno slanje kontrolnih paketa, koji su ionako vrlo rijetki u odnosu na podatkovne pakete, unosi zanemarivo kašnjenje podatkovnih paketa.

1.1.2. Detekcija kvara linka

Ako se postavlja vrlo velika bežična mreža, sa mnogo područja prozivanja i mnogim vršnim čvorovima, tada vjerojatnost kvara linka ili komutatora više nije zanemariva. Primjer vrlo velike mreže je Korejska *hotspot* mreža u Seoul-u sa oko 130000 pristupnih točaka [12]. Iako Koreja prednjači u tom pogledu, veći gradovi u Europi također imaju vrlo visok stupanj pokrivenosti Wi-Fi mrežom. Jasno je da je većina tih pristupnih točaka povezana na komutatore ili usmjerivače, koji su dalje povezani na temeljnu mrežu. Ako bi se za mikromobilnost koristio CIP, tada bi u protokol trebalo implementirati neki oblik sigurnosnog mehanizma u slučaju da zakaže neki link ili čvor. Budući bi CIP komutatori uz funkciju komutiranja paketa trebali i obnašati funkciju pristupne točke, vrlo je izvjestan scenarij u kojem bi CIP komutatori bili postavljeni na otvorenom. Zbog toga je osmišljen dodatak protokolu CIP, koji bi u slučaju prekida veze sa višim čvorom sav promet prebacio na zamjenski viši čvor. Zamjenski čvor ne mora biti bliže vršnom čvoru od sadašnjeg višeg čvora, ali mora imati djelomično odvojen put prema višem čvoru. Cilj ovog dodatka nije prebacivanje komunikacije na zamjenski čvor u rekordno kratkom vremenskom roku, nego

omogućavanje solidne usluge u slučaju kvara. Način rada dodatka protokolu opisan je u slijedećim odlomcima.

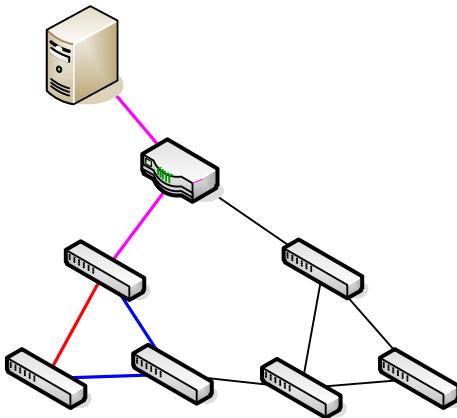
Prije opisa dodatka protokolu, potrebno je objasniti nove funkcijeske elemente protokola. Glavni link je link koji se koristi za slanje i primanje prometa prema i od prvog višeg čvora. Zamjenski link je link koji se koristi za slanje prometa u slučaju da glavni link nije raspoloživ. Zamjenski link ne bi smio biti spojen na isti čvor kao i glavni link, jer u slučaju kvara čvora oba linka bi bila nedostupna. Zamjenski čvor je čvor koji preuzima ulogu višeg čvora u slučaju kvara.

Kontrolni paketi su ICMP paketi koji imaju smisao jedino unutar CIP mreže do vršnog čvora. Prilikom inicijalizacije svakog komutatora odredi se vrijeme kada će se poslati prvi kontrolni paket¹. Vrijeme se određuje slučajnim odabirom u intervalu od 0,2 do 0,7 s. Kontrolni paketi šalju se samo po linkovima koji vode prema čvorovima koji su bliže vršnom čvoru ili po zamjenskim linkovima. Ako su dva susjedna čvora jedan drugom međusobno zamjenski čvorovi, tada oba šalju kontrolne pakete. Gledajući sa strane iskoristivosti linka, bilo bolje da samo jedan čvor šalje pakete. Sadašnji princip odabran je zbog slijedećih razloga:

1. brža detekcija kvara linka,
2. paketi su vrlo kratki, pa dvostruko slanje paketa značajnije ne opterećuje link,
3. CIP u osnovi treba biti jednostavan protokol.

Nakon slanja prvog kontrolnog paketa, komutator na istom portu očekuje odgovor. Po primanju kontrolnog paketa komutator odmah šalje odgovor. U slučaju da komutator nije primio odgovor na poslani kontrolni paket, nakon isticanja vremenske kontrolne, link se proglašava pokvarenim. Ako čvor ne primi odgovor na poslani paket započinje procedura prebacivanja komunikacije na zamjenski link koja je prikazana na slici (Slika 2). Crveni link je link koji se pokvari. Plavi linkovi su put kojim prolaze paketi prema vršnom čvoru nakon kvara linka. Ljubičasti linkovi su linkovi koji se koriste prije i poslije kvara linka.

¹ U ovom poglavlju će se radi jednostavnosti pod kontrolni paket podrazumjevati kontrolni paket za provjeru ispravnosti linka.



Slika 2 : Prebacivanje veze na zamjenski link

Nakon što je utvđeno da je glavni link nedostupan, za svaki zapis u aktivnoj tablici usmjeravanja i aktivnoj tablici prozivanja stvara se paket po sadržaju jednak paketu koji je stvorio taj zapis. Budući da je zamjenski link u potpunosti neopterećen, ti paketi se bez zadržavanja u repu mogu poslati prema zamjenskom CIP komutatoru. Odmah nakon poslanih paketa za ažuriranje tablica na putu do vršnog čvora smiju se slati podatkovni paketi. Za cijelo vrijeme korištenja zamjenskog linka, svakih pola sekunde provjerava se da li je glavni link proradio slanjem kontrolnih paketa. Ako CIP komutator po glavnom linku dobije odgovor na poslani paket, tada je glavni link proradio. Radi rasterećivanja zamjenskog komutatora se sva komunikacija prebacuje na glavni link. Procedura prebacivanja na glavni link jednaka je prebacivanju na zamjenski. Ponovo se stvore paketi za stvaranje zapisa u aktivnoj tablici usmjeravanja i tablici prozivanja te se pošalju prema vršnom čvoru. Nakon ažuriranja tablica u čvorovima na putu, komunikacija se odvija isto kao i prije kvara.

Kao i za potvrde, za kontrolne pakete je potrebno uvesti prioritetno slanje. Budući da je vrijeme u kojem se očekuje odgovor na kontrolni paket vrlo kratko, čekanje u repu moglo bi dovesti do nepotrebnog prebacivanja na zamjenski link. Zbog toga se odgovor uvijek postavlja na prvo mjesto u repu na određenom portu.

Kontrolni paketi šalju se svakih 10ms, dok se odgovor očekuje unutar 5ms. Budući da je okvir u kojem se nalazi kontrolni paket vrlo kratak, on na razini linka ima minimalnu duljinu od 512 okteta. Dodatno kašnjenje podatkovnih okvira, koje se uvodi zbog prioritetnog slanja kontrolnih okvira jednako je vremenu potrebnom za slanje okvira minimalne duljine:

$$t_{send} = \frac{512 \cdot 8}{10^9} = 4,096\mu\text{s}$$

Budući da se kontrolni okviri šalju svakih 10ms, u najgorem slučaju, kada su susjedni čvorovi međusobno zamjenski čvorovi, šalje se 400 kontrolnih okvira u sekundi. Ako se radi o gigabitnim linkovima, iskoristivost linka se neznatno smanji.

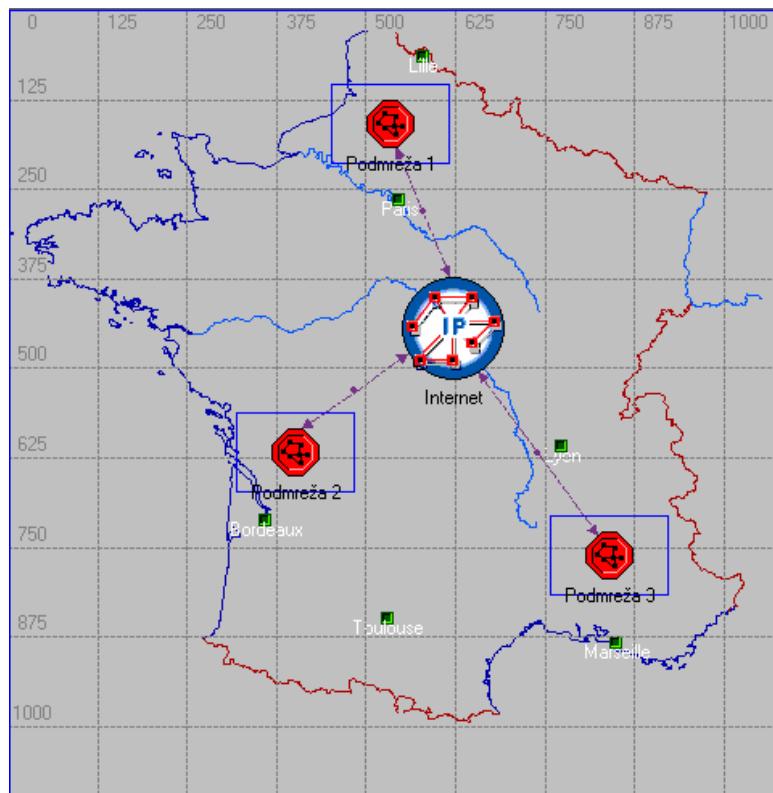
$$U = 1 - \frac{512 \cdot 400 \cdot 8}{10^9} = 0,99836$$

Budući da se unutar 5ms očekuje odgovor na kontrolni paket, vrijeme do detekcije kvara linka iznosi između 5 i 15 ms. Vrijeme oporavka sustava od kvara biti će prikazano u poglavljiju gdje su prikazani rezultati simulacije.

2. Implementacija protokola CIP u programskom alatu OpNet 8.0.C

2.1. Higerarhijska struktura programskog alata OpNet

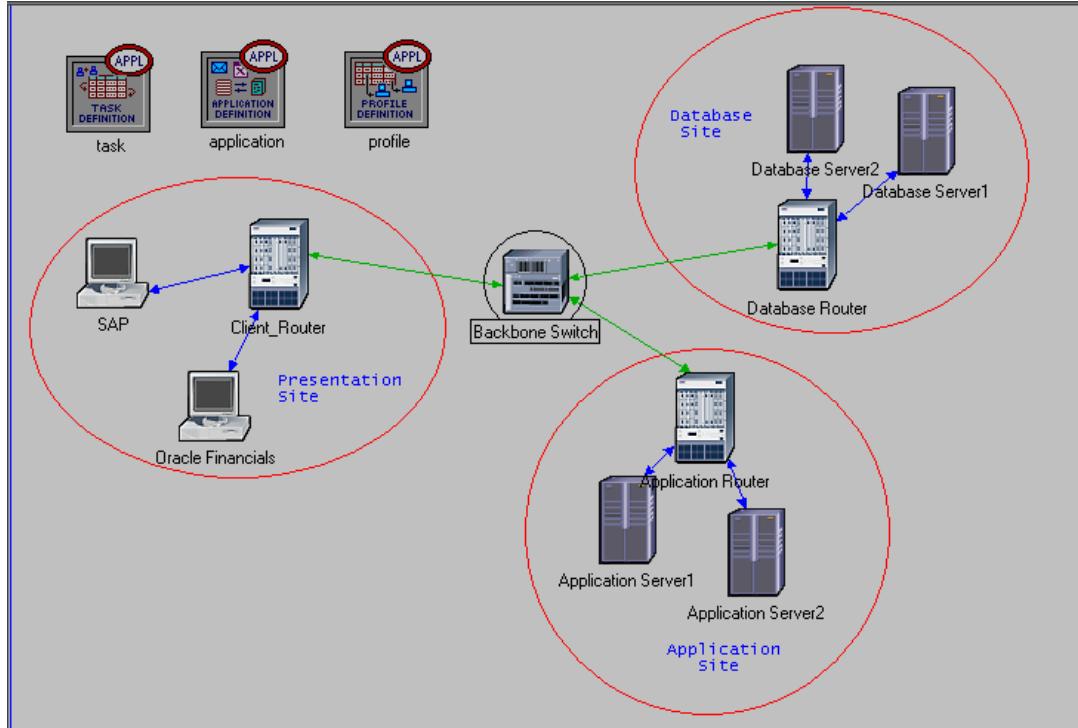
OpNet je programski paket koji služi za simuliranje telekomunikacijskih mreža. Jedna od osnovnih prednosti tog paketa je vrlo intuitivno grafičko sučelje. Grafičko sučelje sastoji se od nekoliko osnovnih sučelja, koja dozvoljavaju izmjene na različitim razinama. Postoje tzv. Network Model Editor, Node Model Editor, Process Model Editor te uređivač izvornog koda unutar svakog stanja nekog procesa[3].



Slika 3: Tri mreže povezane preko Interneta

Uređivač na najvišoj razini omogućava definiranje mreža na nekom geografskom području. Slika (Slika 3) opisuje tu razinu uređivanja. Na slici su prikazane tri mreže u Francuskoj, koje su međusobno povezane preko Interneta. Svaka mreža spojena je na Internet prijenosnim linkom proizvoljne brzine (strelica između mreže i Interneta). Prilikom spajanja mreže i čvora koji simbolizira Internet, moguće je odrediti na koji je čvor u mreži taj link

spojen. Ovakav pristup omogućava simuliranje mreža velikih korporacija koje imaju razasute urede po cijelom svijetu.



Slika 4: Konfiguriranje mreže

Network Model Editor omogućava stvaranje proizvoljne mreže. U tom uređivaču, moguće je odrediti topologiju mreže, kao i svojstva svakog navedenog čvora. Na slici (Slika 4) prikazana je mreža koja bi mogla biti jedna od mreža sa slike (Slika 3). Na slici se mogu zamijetiti čvorovi *task*, *application* i *profile*, koji nisu povezani s ostatkom mreže. Ti čvorovi služe za detaljno upravljanje generiranjem prometa. Kao i u stvarnom slučaju, većinu prometa generiraju klijenti i poslužitelji. Na koji način će se generirati promet u tim čvorovima opisano je u čvorovima *task* i *application*. U čvoru *task* opisuju se pojedinačni zadaci koji čine jednu aplikaciju. Jedan zadatak opisuje se s nekoliko parametara, od kojih su najvažniji: vrijeme kada počinje njegovo izvršavanje, izvorište i odredište generiranog prometa i razdioba po kojoj se generira promet. Ovaj čvor koristi se kod složenijih komuniciranja među poslužiteljima. Na primjer, ako je potrebno simulirati pristup mreži preko proxy poslužitelja, tada je to potrebno definirati u ovom čvoru. Čvor *application* služi za definiranje aplikacije koja generira promet. Aplikacija može biti proizvoljna ili predefinirana. Ako se radi o proizvoljnoj aplikaciji, tada se ona sastoji od nekoliko zadataka definiranih u čvoru *task*. Neke od predefiniranih aplikacija su: FTP promet, HTTP promet, ispis datoteka na mrežni pisač, video konferencija itd. Svaka predefinirana aplikacija može se do neke mjere izmjeniti. Na

primjer, ako se radi o FTP prometu, moguće je odrediti po kojoj će se razdiobi određivati veličina prenesene datoteke. Čvor *profile* određuje koja će se aplikacija u kojoj mjeri koristiti. Također je moguće stvoriti vlastite profile ili koristiti predefinirane. Za primjer se može uzeti predefinirani profil Multimedia User, koji koristi predefinirane aplikacije "Voice over IP" i "Video Conferencing". Pomoću navedena tri čvora moguće je lako vrlo vjerno generirati promet u jako velikim mrežama. Također, na slici (Slika 4) vidljivi su klijenti, poslužitelji, usmjerivači i komutator. Svaki od tih čvorova djeluje na drugačiji način. Ovaj uređivač dozvoljava minimalne izmjene u ponašanju čvora. Primjerice, moguće je podesiti koji profil koristi pojedini čvor. Također, moguće je odrediti koju implementaciju protokola TCP koristi neki od čvorova. Za drastično mijenjanje ponašanja čvora nužno se spustiti na još nižu razinu.

Node Model Editor, Process Model Editor bit će opisani u narednim poglavljima i potkrijepljeni stvarnim primjerima.

2.2. Osnovni preduvjeti za implementaciju protokola CIP

Simulacijski paket OpNet nema ugrađenu podršku za simuliranje protokola CIP. Kako bi se omogućilo simuliranje protokola CIP nužno je provesti određene izmjene izvornog koda pojedinih OpNet komponenti i stvoriti nove vrste procesa i čvorova. Osnovni preduvjeti koje je potrebno osigurati kako bi bilo moguće simuliranje protokola CIP su:

1. Omogućavanje prebacivanja veze (engl. *handoff*);
2. Umetanje procesa cip u protokolni složaj mobilne stanice;
3. Komunikacija procesa cip i procesa mac;
4. Prilagođavanje procesa wlan_mac;
5. Stvoriti nove vrste čvorova i po potrebi nove procese za:
 1. CIP komutator;
 2. CIP pristupna točka;
 3. CIP mobilna stanica;
6. Stvoriti datoteku sa eksternim izvornim kodom koja obuhvaća neke standardne funkcije.

2.3. Omogućavanje prebacivanja veze (engl. *handoff*)

OpNet 8.0.C ne podržava prebacivanje veze u IEEE 802.11 bežičnoj mreži. Pristup problemu je slijedeći. Mobilne stanice čiji BSS ID (engl. *Basic Service Set Identifier*) se razlikuje od BSS ID-a bazne stanice ne mogu primiti paket s te bazne stanice. U tom slučaju, potrebno je

omogućiti postojanje više baznih stanica u istom BSS području. To je grubo kršenje protokola IEEE 802.11, ali je u ovom slučaju nužno. Naime, okviri koji stižu s pristupne točke koja ima različit BSS ID od mobilne stanice, odbacuju se na razini prijemnika, što je komplikirano zaobići.

Osnovni problem jest u tome što simulator prilikom inicijalizacije simulacije izračuna koji bežični čvorovi mogu komunicirati sa drugim čvorovima. Oni čvorovi koji su na različitim kanalima ili koji su predaleko ne mogu komunicirati. Nadalje, pretpostavlja se da je svaki BSS ID na posebnom kanalu, pa zbog toga čvorovi padaju na testu da li se nalaze na istom kanalu. Ta optimizacija je vrlo korisna u slučaju da se radi o statičnim čvorovima i većoj mreži. Prilikom slanja paketa na bežični medij, potrebno je provjeriti koji čvorovi mogu primiti paket. Ako su te provjere već prije odradene, paket se dostavlja samo predodređenim čvorovima, pa se time štedi na vremenu, jer se provjera ne mora izvoditi za svaki paket posebno.

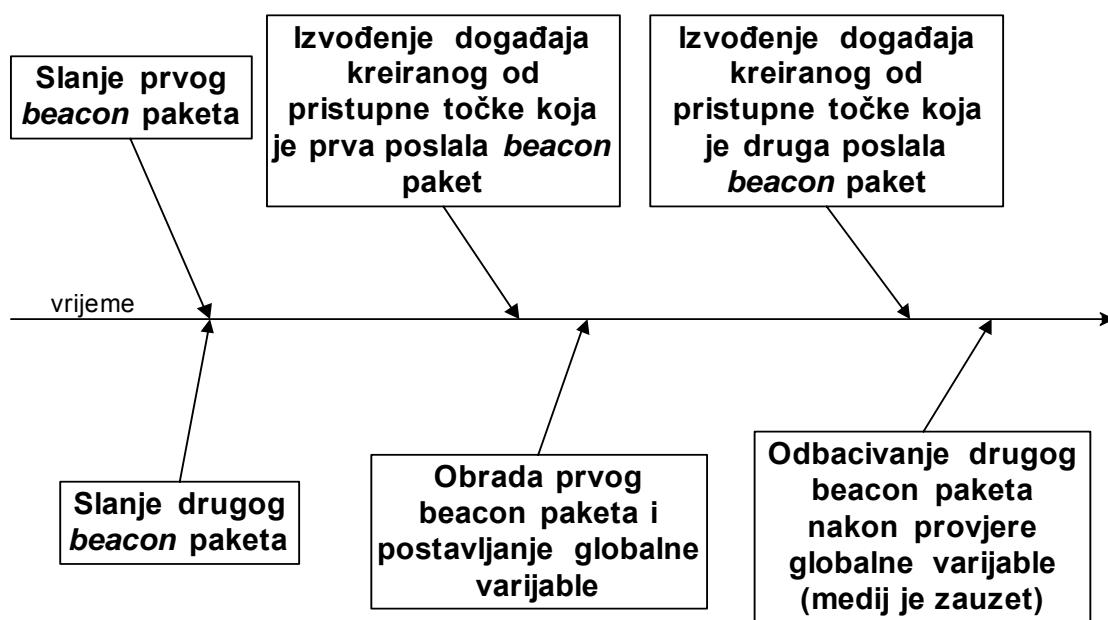
U OpNet-u je onemogućeno postavljanje dvije pristupne točke u istom BSS-u. Provjera da li postoji više od jedne pristupne točke u BSS-u izvodi se u procesu `wireless_lan_mac`, stanju `BSS_INIT` (Exit Execs).

Bežične stanice mogu raditi u dva načina rada. Prvi način rada je DCF (engl. *Distributed Coordiantion Function*). U tom načinu rada prije slanja podataka na medij, bežična stanica provjeri da li je medij slobodan. Drugi način rada je PCF (engl. *Point Coordination Function*), kod kojeg pristupna točka određuje kada određena mobilna stanica smije slati pakete na medij [1]

Komentiranjem navedenog dijela koda, PCF način rada nije moguće koristiti. Naime, pri pokretanju simulacije, simulator pokušava pristupiti memorijskoj adresi na koju pokazuje ne inicijaliziran ili pogrešno inicijaliziran pokazivač, što dovodi do trenutnog prekida simuliranja. Kako bi pristupne točke slale *beacon* okvire, OpNet zahtjeva da stanice rade u PCF načinu rada. Detaljnijim pretraživanjem izvornog koda procesa `wireless_lan_mac` ustanovaljeno je da ne postoji razlog zbog kojeg se *beacon* okviri ne bi slali u DCF načinu rada. Štoviše, iako se u DCF-u ne šalju *beacon* okviri, dio koda koji obrađuje primljeni *beacon* okvir pisan je tako da može funkcionirati u DCF ili PCF čvoru.

U realnom slučaju bežične stanice (pristupna točka ili mobilna stanica) određuju dostupnost medija na dva načina. Fizički (engl. *physical carrier-sensing*) ili virtualno (engl. *virtual carrier-sensing*). Fizičko određivanje dostupnosti medija implementirano je a razini fizičkog

sloja. Takav način određivanja dostupnosti ima dva velika nedostatka. Prvi nedostatak odnosi se na skupoću elektronike koja omogućava istovremeno slanje i primanje signala. Drugi nedostatak proizlazi iz elementarnog problema postojanja skrivene stanice. Virtualno određivanje dostupnosti medija temelji se na parametru NAV (engl. *Network Alocation Vector*). Prilikom svakog slanja paketa, stanice u NAV vektor zapisuju na koliko dugo rezerviraju medij. Tako dugo dok je NAV veći od nule druge stanice ne pokušavaju slati pakete. U OpNet-u se koristi drugi način određivanja zagušenosti medija. Kao primjer može se razmotriti što se događa kada dvije pristupne točke šalju *beacon* signal u isto vrijeme. Takav slučaj opisan je na slici (Slika 5).



Slika 5: Istovremeno slanje i primanje dva *beacon* okvira

OpNet simulator pokreće diskretni dogadaji te signal od pristupne točke do mobilne stanice u biti ne putuje. Predajnik na strani pristupne točke generira događaj koji nakon određenog vremena pokrene proces prijemnika na strani mobilne stanice. Ta dva procesa analitički računaju odnos signal-šum, vrijeme propagacije signala, dobitak antene i ostale parametre vezane za fizičko slanje paketa. Kada dvije pristupne točke istovremeno šalju paket, obje šalju paket bez kolizije. Na prijemnoj strani oba paketa stignu na MAC sloj bez greške i oba se obrađuju. Odbacivanje paketa zbog kolizije izvodi se na slijedeći način. Kada prvi paket stigne, on u globalnu varijablu (na razini čvora, ne na razini simulacije) zapisiše koliko dugo ta stanica ne može primiti niti jedan drugi paket. Dolaskom drugog paketa, u isto vrijeme, ali nekoliko događaja kasnije, provjerom globalne varijable ustanavljuje se da je ovaj paket potrebno odbaciti.

Kada se omogućuje postavljanje više pristupnih točaka u isti BSS, pojavljuje se sljedeći problem. Nema načina kako bi se dvije ili više pristupnih točaka uskladilo. U ovom radu to i nije značajan problem jer je glavna nakana ovog rada pokazati kako protokol CIP utječe na protokole višeg sloja prilikom prebacivanja veze. Osnovna ideja slanja *beacon* signala je slijedeća. Potrebno je organizirati pristupne točke, tako da ne šalju *beacon* okvire u isto vrijeme. Vrijeme između slanja dva *beacon* okvira mora u svim stanicama biti jednako, dok se prvi *beacon* okviri moraju slati u različita vremena. Cijeli postupak određivanja vremena kada bazna stanica šalje prvi i svaki naredni paket opisana je u poglavljima gdje se opisuju uvedene izmjene u proces *wireless_lan_mac*.

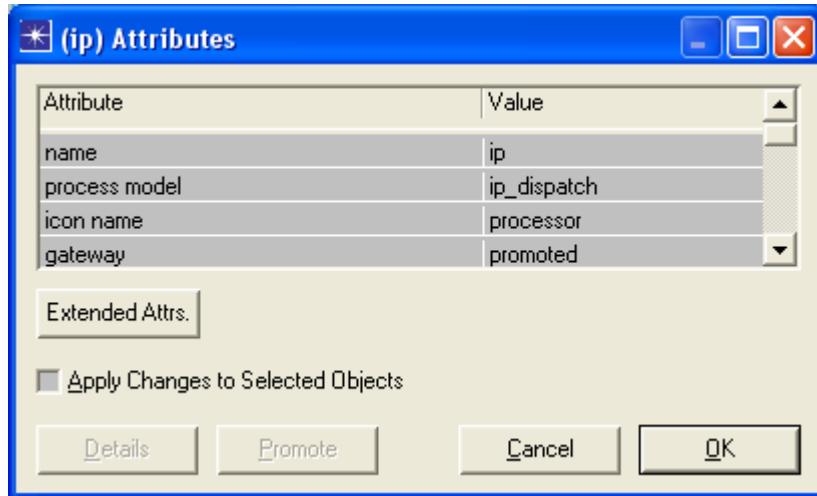
2.4. Umetanje procesa cip u protokolni složaj mobilne stanice

Prije stvaranja vlastitog CIP procesa, potrebno je odrediti kamo će se taj proces umetnuti. Rješenje tog problema posuđeno je iz [5]. U tom radu proces koji implementira protokol CIP nalazi se između procesa *ip_dispatch* i procesa *ip_arp_v4*. Razlozi za takav odabir su slijedeći:

1. CIP generira vlastite ICMP poruke. Budući da procesi viših slojeva nisu prilagođeni protokolu CIP, vjerojatno bi prijavili grešku kada bi primili takav paket. Zbog toga je potrebno sve kontrolne pakete presresti na nižoj razini te ih obraditi i uništiti ili, ako je potrebno, proslijediti ih dalje.
2. Proces *ip_dispatch* poziva nekoliko podprocesa u kojima se ovisno o tipu mreže koja se simulira odlučuje o obradi i usmjeravanju paketa. Budući da su ti procesi vrlo složeni, teško je odrediti kakav bi oni utjecaj imali na kontrolne CIP pakete (vjerojatno bi prijavili grešku koju bi bilo komplikirano otkloniti).
3. Kao što je već navedeno, proces *ip_dispatch* je vrlo složen. Bilo bi vrlo nepraktično mijenjati taj proces, te je puno jednostavnije napraviti zaseban proces koji prosljeđuje pakete sa višeg sloja na niži u nepromijenjenom obliku i, po potrebi, generira i obrađuje CIP kontrolne pakete. Također, poseban proces je mnogo lakše testirati i iz njega otklanjati greške.

Svaki čvor sastoji se od više modula. Nadalje, za svaki modul potrebno je definirati ime i proces koji taj modul izvršava. Budući da je ime "arp" uobičajeno za modul u kojem se izvršava proces *ip_arp_v4*, ta dva imena, kao i pojmovi modul i proces dalje u tekstu će se

koristiti ravnopravno. Proces ip_dispatch izvršava se u modulu "ip". Slika (Slika 6) prikazuje svojstva modula "ip". Svojstvo "process model" je ime procesa koji se mora izvoditi u ovom modulu.



Slika 6: Svojstva modula "ip"

Osnovni problem proizlazi iz toga što su procesi *arp* i *ip* pisani pod prepostavkom da su susjedni procesi i da mogu direktno komunicirati. Umetanje vlastitog modula između njih uzrokuje nemogućnost detektiranja višeg procesa od strane nižeg i obratno. Zbog toga proces koji se ubacuje između navedena dva procesa mora biti potpuno transparentan i ne smije smetati navedenim procesima u njihovom radu. Kako bi lakše bilo razumjeti na koji način je taj problem riješen, nužno je najprije opisati kako je riješena komunikacija među procesima u simulatoru.

Procesi su u nekom čvoru povezani tokovima paketa (engl. *packet stream*). Tok paketa prikazan je kao strelica od jednog do drugog procesa. Za komunikaciju među procesima postoje još "*statistic wire*" i "*logical rx/tx association*" veze.

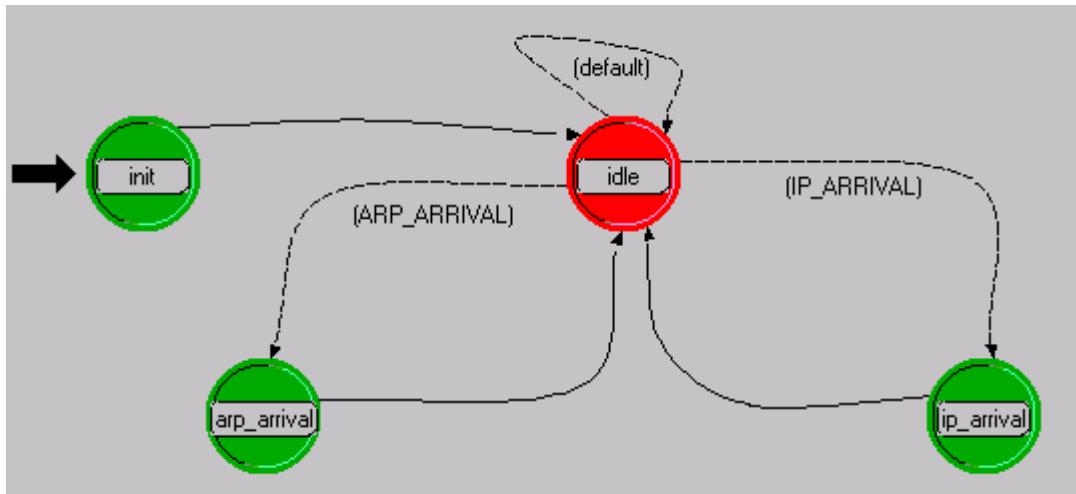
Kada se ubacuje novi proces između dva postojeća procesa, važno je da se detaljno promotre sva svojstva koja posjeduju tokovi paketa između ta dva procesa. Konkretno, uzlazni i silazni tok paketa između procesa ip i arp posjeduju sljedeće dodatno svojstvo (engl. *extended attribute*).

Tablica 1: Dodatno svojstvo toka podataka između procesa arp i ip

Ime dodatnog svojstva (engl. Attribute name)	Tip (engl. Type)	Prepostavljena vrijednost (engl. Default value)
ip adr index	Integer	0

Budući da se u arp i ip procesu čitaju svojstva tokova podataka, kako bi prihvatili novi proces, potrebno je prilagoditi izvorni kôd oba procesa. Promjene koje je potrebno napraviti odnose se uglavnom na detekciju ulaznih i izlaznih tokova paketa. Prema tome proces ip zna da pakete namijenjene nižem sloju mora slati putem toka koji ima navedeno dodatno svojstvo. Na isti način proces arp određuje kojim tokom mora slati pakete za namijenjene višem sloju.

2.4.1. Objasnjenje koncepta stanja na konkretnom primjeru



Slika 7: Najjednostavniji proces umetnut između procesa ip i arp

Slika 7 prikazuje dijagram stanja najjednostavnijeg mogućeg procesa koji je umetnut između procesa arp i ip. Taj proces nema nikakvu funkcionalnost osim što proslijeđuje pakete s višeg na niži sloj i obratno. Na ovom procesu bit će objašnjene oznake koje se koriste. Također, biti će objašnjeno na koji način funkcioniraju i kako se koriste stanja.

2.4.2. Vrste stanja i objašnjenje prijelaza

Na slici (Slika 7) može se uočiti da postoje dvije vrste stanja. Nadalje, svako stanje je podjeljeno na dva dijela. Dvostukim klikom na gornji dio zelenog ili crvenog stanja, može se pisati ulazni (engl. *Enter Execs*) ili izlazni (engl. *Exit Execs*) izvorni kod stanja. Logično je da će se ulazni kod izvršavati odmah nakon ulaska u stanje, dok će se izlazni izvorni kod izvršavati neposredno prije prijelaza iz stanja u neko drugo stanje.

Zelena stanja su *forced* stanja, te se nakon izvođenja tog stanja odmah prijelazi u slijedeće stanje. Na primjer, ako stigne paket od procesa arp, zadovoljiti će se uvjet ARP_ARRIVAL. Zbog toga proces prelazi u stanje arp_arrival. Budući da je to stanje *forced*, u jednom

događaju se izvede ulazni i izlazni izvorni kod tog stanja, te se izvrši, u ovom primjeru bezuvjetan, prijelaz natrag u stanje idle.

Crvena stanja su *unforced* stanja. Nakon izvođenja ulaznog izvornog koda stanja, proces ostaje u tom stanju. Kada bi stanje arp_arrival bilo *unforced*, nakon zadovoljavanja uvjeta ARP_ARRIVAL, proces bi se prebacio u stanje apr_arrival. Nakon izvođenja ulaznog koda, kontrola bi se predala jezgri simulacije (engl. *simulation kernel*). Nakon što neki događaj ponovno prozove taj proces, tada se izvršava izlazni izvorni kod i prijelazi se u slijedeće stanje. Kada bi sva stanja bila *unforced*, tada bi se prijelazi odvijali po slijedećem pravilu:

1. Pozvanje događaja;
2. Izvođenje exit execs prvog stanja;
3. Prijelaz u drugo stanje;
4. Izvođenje enter exect drugog stanja;
5. Vraćanje kontrole simulacijskoj jezgri.

Svaki puta kada se neki događaj obradi i kontrola se preda jezgri, tada se proces nalazi u nekom od *unforced* stanja. Forced stanja uvedena su samo zato da olakšaju implementaciju nekog procesa. Ako između dva *unforced* stanja postoji nekoliko *forced* stanja, za vrijeme jednog događaja će se izvesti sva *forced* stanja između dva unforced stanja (prvo i drugo *unforced* stanje mogu biti isto stanje, kao na slici (Slika 7).

Uvjet pod kojim se izvršava neki prijelaz napisan je iznad strelice koja označava taj prijelaz. Ti uvjeti definirani su u *header* datoteci tog procesa. Prijelaz “default” izvršava se kada nije zadovoljen uvjet izvršavanja niti jednog drugog prijelaza. Taj prijelaz je obavezan, jer se prilikom svakog događaja mora izvršiti neki prijelaz, makar on bio povratak u isto stanje. U slučaju da se iz nekog stanja ne može izaći ili da postoji mogućnost prijelaza iz stanja u više od jednog stanja, tada simulacija završava uz ispis pripadajuće pogreške. Za vrijeme prijelaza može se izvršiti i neka funkcija definirana u funkcijском bloku (engl. *Function Block*, skraćeno FB). Taj dio služi za definiranje svih pomoćnih funkcija i funkcija koje bi bilo nespretno implementirati u ulaznim ili izlaznim dijelovima stanja. “Pomoćne” funkcije obično zauzimaju većinu izvornog koda nekog procesa. Za primjer se može uzeti proces wireless_lan_mac, koji u funkcijском bloku ima nekoliko tisuća linija izvornog koda.

2.4.3. Specifičnosti simulacije temeljene na diskretnim događajima u programskom paketu OpNet

OpNet simulator je u simulator pogonjen diskretnim događajima, pri čemu su objekti događaja vrlo složeni i imaju mnoga svojstva koja se ne podrazumijevaju kada se govori o toj vrsti simulatora. Svaki događaj, kao i svi ostali objekti, ima određena svojstva. Neka od tih svojstava služe za olakšano implementiranje procesa, dok su neka nužna za ispravan rad simulatora. U dalnjem izlaganju, ravnopravno će se koristiti pojmovi prekid (engl. *interrupt*) i događaj (engl. *event*). Razlika između ta dva pojma je za ovo razmatranje neznatna. Po konvenciji navedenoj u [3] prekid je događaj koji se upravo izvršava odnosno prekid je prvi događaj u listi svih zakazanih događaja..

Radi jednostavnosti implementiranja procesa, u simulacijskoj jezgri postoje dva skupa funkcija. Jedan skup funkcija služi za rukovanje događajima, a drugi za rukovanje prekidima. Jasno je da skupina za rukovanje događajima ima funkcionalnost druge skupine funkcija, koja je uvedena radi lakšeg implementiranja nekog procesa.

Dva, najznačajnija za korisnika, svojstva svakog događaja su tip (engl. *type*) događaja i kôd (engl. *code*) događaja. Ovisno o tipu događaja, kôd može imati različito značenje. Iako postoji trinaest različitih tipova događaja, ovdje će biti opisana samo dva najčešće korištena.

U stanju idle obrađuje se tip koji je vezan na tok podataka, tj. radi se o događaju čiji tip odgovara konstanti OPC_INTRPT_STRM. Taj događaj označava da se u toku koji spaja dva procesa nalazi paket koji čeka na obradu. Još uvijek nije poznato u kojem toku se nalazi paket. Taj podatak zapisan je u kôdu događaja, koji je moguće očitati naredbom `op_intrpt_code ()` ili `op_intrpt_strm ()`. Obje naredbe daju isti rezultat, samo što druga naredba izvodi provjeru da li je tip prekida ispravan. Važno je napomenuti da kôd prekida ima značenje ovisno o tipu prekida.

Drugi tip prekida označava se konstantom OPC_INTRPT_SELF. Taj tip prekida nastaje pozivom funkcije `op_intrpt_schedule_self ()`. Kôd tog prekida određuje korisnik. Obično su u *header* datoteci procesa definirane konstante koje se proslijeduju funkciji prilikom generiranja ove vrste prekida. Ovaj tip prekida često se koristi za uvođenje čekanja i brojača vremena (engl. *timer*). Npr. ako proces za upravljanje protokolom TCP ne dobije ACK paket od odredišta on mora nakon određenog vremena ponovno poslati isti paket. Prilikom prvog slanja proces generira događaj za ponovno slanje paketa. Ako je u međuvremenu stigao ACK paket, tada se događaj za retransmisiju poništava.

Uz navedena dva svojstva, uz svaki događaj dolaze svojstva koja korisniku nisu interesantna (osim ako ne otklanja greške u vlastitom kodu). Neka od tih svojstava imaju funkciju: određivanja u kojem modulu će se događaj aktivirati, koji objekt je generirao događaj, redni broj izvođenja i stvaranja događaja itd.

Korisnik uz svaki događaj može vezati proizvoljne informacije. U strukturi događaja postoje dva polja koja to omogućuju: state_ptr i ICI. Prvo polje u ovom radu nije niti jednom korišteno. U ovo polje zapisuje se pokazivač na vlastite podatke. Važno je napomenuti da je korisnik dužan sam osloboditi zauzetu memoriju. ICI polje je puno praktičnije jer se za nju veže struktura kojom se lako upravlja pomoću funkcija implementiranih u jezgri. ICI strukture detaljnije su objašnjene niže u tekstu.

2.4.4. Opisi osnovnih stanja

Prema imenima stanja može se zaključiti njihova osnovna funkcionalnost. Stanje init služi za početnu inicijalizaciju procesa. U tom stanju moraju se izvršiti slijedeće akcije:

1. Inicijalizacija osnovnih varijabli stanja (engl. *state variables*, skraćeno SV).
2. Registracija procesa u globalni registar procesa (engl. *process registry*).
3. Otkrivanje tokova paketa prema nižem i višem sloju.

Proces se, kada ne obrađuje dolazak paketa s višeg ili nižeg sloja, nalazi u stanju idle. Dolaskom paketa s višeg ili nižeg sloja izvršava se prijelaz u stanje ip_arrival odnosno stanje arp_arrival. Povratak u stanje idle događa se u istom događaju kada je i prijelaz u to stanje izvršen, jer se radi o *forced* stanju. U stanju idle, potrebno je pomoći funkcije op_intrpt_strm() postaviti varijablu *intrpt_strm*. Ta varijabla označava sa koje veze je primljen paket. Svaki proces može biti povezan sa drugim procesima sa više veza. Npr. model komutatora ima jedan centralni proces, koji je povezan sa mnogo procesa koji opisuju ponašanje portova. Centralni proces može imati nekoliko desetaka ulaznih i izlaznih veza. U slučaju da se proces umeće između procesa ip i arp moraju postojati četiri veza (po jedna veza za slanje i prijem prema svakom procesu).

Stanja arp_arrival i ip_arrival vrlo su slična i ovdje će se detaljnije opisati samo stanje ip_arrival. Priložen je izvorni kôd tog stanja, kako bi se na njemu mogli objasniti neki važni koncepti komunikacije među procesima.

1. tmp_packet = op_pk_get (intrpt_strm);
4. iciptr = op_intrpt_ici();

```
5. // iciptr = op_pk_ici_get (tmp_packet);  
6. op_pk_ici_set (tmp_packet, iciptr);  
7. op_pk_send ( tmp_packet, low_out_strm);
```

U prvoj liniji koda u varijablu *tmp_packet* sprema se pokazivač na paket koji je stigao po toku koji je izazvao ovaj događaj.

Prethodno navedena tri načina komuniciranja nisu jedini načini komunicirajna među procesima. Ta tri načina vidljiva su u *Node Model Editor*-u. Postoji još nekoliko vrsta komuniciranje među procesima od kojih je najzastupljeniji komunikacija ICI (engl. *Inter Control Information*) porukama[3]. ICI poruka se može vezati na paket ili na događaj i prije korištenja potrebno ju je definirati. Do izbornika za definiranje nove vrste ICI poruke dolazi se tako da se umjesto novog projekta ili novog procesa u istom izborniku odabere opcija ICI Editor. Prilikom definiranja, određuje se ime svojstva, tip svojstva (cijeli broj, realni broj ili proizvoljna struktura) i prepostavljena vrijednost (engl. *default value*).

U drugoj i trećoj liniji koda prikazano je kako se izlučuje poruka iz događaja odnosno paketa. Detaljnija kontrola podsloja MAC od strane protokola CIP obavlja se pomoću ICI poruka. Korišteni format ICI poruke za kontrolu podsloja MAC detaljnije će biti objašnjen u slijedećem poglavlju. U četvrtoj i petoj liniji koda ICI poruka se veže uz paket koji se šalje na izlazni tok. Dijeljenje zajedničke memorije je čest način komuniciranja između procesa roditelja i procesa dijeteta, ali takav se način komuniciranja nije koristio, pa se ovdje neće opisivati.

2.5. Komunikacija procesa cip i procesa mac

Prema zamisli iznesenoj u [4], protokol CIP odlučuje kada će se izvršiti prebacivanje veze na drugu pristupnu točku. Prednosti takvog odlučivanja su slijedeće:

1. Budući da se o prebacivanju veze odlučuje unutar istog procesa koji je zadužen za slanje kontrolnih paketa, nije potrebno koordinirati rad dva procesa. Prije nego što prebacivanje veze počne, a isto tako i nakon prebacivanja veze, proces cip može po potrebi slati kontrolne pakete i izvršavati “kućanske poslove” (postavljanje brojača na vrijednosti propisane protokolom, zapisivanje statistika itd.).
8. Otklanjanje pogrešaka u vlastitom izvornom kodu mnogo je lakše nego u kodu koji je dopisan na postojeći, tuđi, kôd.

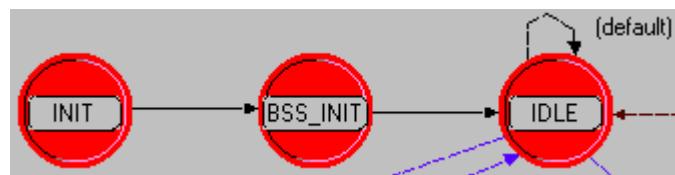
9. Mogućnost implementiranja naprednijeg odlučivanja o prebacivanju veze(u ovoj verziji odluka o prebacivanju veze na bolju pristupnu točku donosi se kada nova pristupna točka određeno vrijeme ima bolji S/N od sadašnjeg)

Komuniciranje između procesa cip i podsloja MAC izvodi se ICI porukama vezanima uz događaje. Tablica (Tablica 2) prikazuje strukturu ICI poruke. U desnom stupcu nalazi se objašnjenje svakog parametra.

Tablica 2: Svojstva i opisi svojstva ICI poruke wmac_cip_comm

Ime svojstva	Tip	Pretpostavljena vrijednost	Objašnjenje
snr_value	double	0	Odnos signal-šum izmjerjen na zadnjem pristiglom <i>beacon</i> okviru
new_ap	integer	0	Parametar se koristi prilikom izvršavanja prebacivanja veze
ap_address	integer	0	MAC adresa pristupne točke na koji se odnosi snr_value
Code	integer	0	Kôd ICI poruke
src_mod_objid	integer	0	Identifikator objekta modula koji je poslao ovaj ICI
ap_ip_address	integer	0	IP adresa nove bazne stanice

2.6. Prilagodba procesa wlan_mac



Slika 8: Inicijalizacijska stanja procesa wireless_lan_mac

Na slici (Slika 8) prikazana su inicijalizacijska stanja svakog wireless_lan_mac procesa. Prilikom izvršavanja prvog stanja (INIT) svaki navedeni proces se registrira u globalni registar procesa. Globalni registar procesa poznat je svim procesima bez obzira na to u kojem se čvoru nalazili i sadrži podatke o svim procesima koji su trenutno registrirani. Svaki proces može se registrirati i istovremeno može pretraživati podatke o svakom registriranom procesu. U svakoj simulaciji postoji samo jedan globalni registar procesa. Uz registrirane procese, taj registar može sadržati proizvoljan broj dodatnih svojstava. Uobičajena svojstva koja se

definiraju sa svakim procesom su “node objid”, “protocol” i “module objid”. Ta svojstva, uz još desetak drugih, definirana su i kod svake pristupne točke.

U drugom stanju (BSS_INIT), između ostalog, određuje se kada će se poslati *beacon* okvir. Problem sinkronizacije pristupnih točaka riješen je na sljedeći način. U svim pristupnim točkama parametar “*Beacon Interval (secs)*” potrebno je postaviti na istu vrijednost. Nadalje, potrebno je pretražiti globalni registar procesa. Rezultat pretraživanja sprema se u sortiranu listu sa svim procesima koji su zadovoljili kriterij pretraživanja. Budući da je rezultat pretraživanja sortirana lista, ta lista ima isti oblik u svakoj pristupnoj točki koja pretražuje registar. Broj članova te liste jednak je broju pristupnih točaka u mreži. Nadalje, svaka pristupna točka određuje koji je redni broj zapisa u listi procesa (indeks) koji se odnosi na nju, te se vrijeme kada će se događaj izvršiti određuje po slijedećoj formuli:

$$vrijeme = \frac{(Indeks)}{(BrojAPa)} \cdot BeaconInterval$$

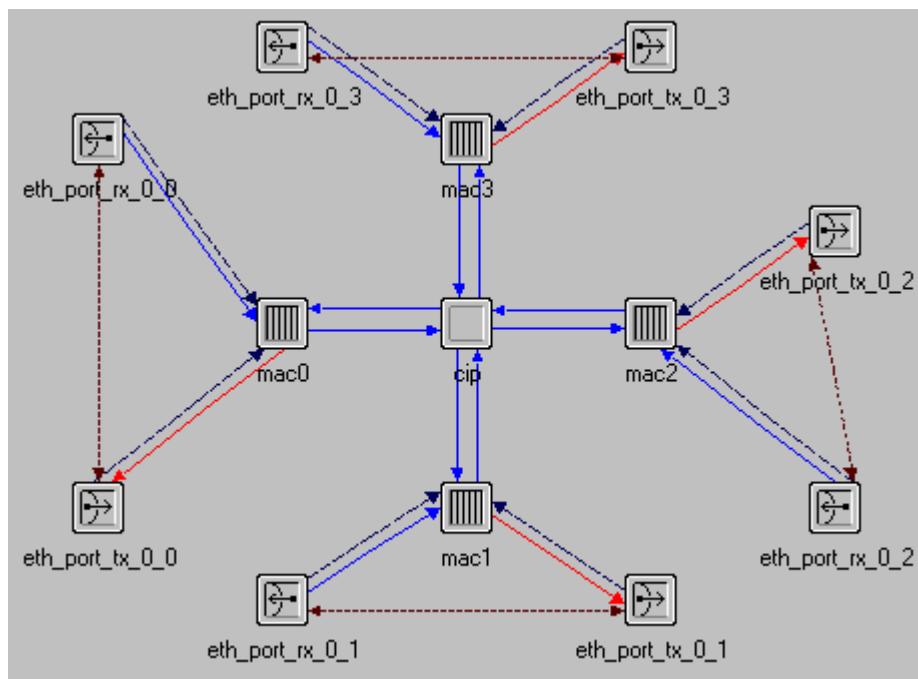
Nakon slanja prvog *beacon* okvira događaj slijedeći događaj za pokretanje slanja *beacon* intervala na redu je za *BeaconInterval* sekundi. Važno je napomenuti da je s većim brojem pristupnih točaka potrebno povećati i *BeaconInterval*.

Proces za kontrolu MAC podsloja može čekati u pet različitih stanja: idle, defer, backoff, transmit i wait_for. Proces je u stanju idle kada ne mora slati ili primati pakete. Proces u stanju defer ili backoff kada čeka da se medij oslobodi. U stanju transmit čeka se da se završi slanje paketa, dok se u stanju wait_for čeka na potvrdu o uspješnom slanju okvira. Proces prebacivanja veze može započeti u svakom od tih stanja, jer se odluka o prebacivanju donosi u procesu cip. Zbog toga, je uz prijelaz default, u svakom od pet navedenih stanja, dodan prijelaz stanja u samo sebe, koji se odvija u slučaju da je potrebno obraditi događaj koji označava početak prebacivanja veze. Obrada tog događaja vrlo je jednostavna. Svodi se na izlučivanje ICI poruke iz događaja. Nakon toga, u ICI poruci je potrebno očitati MAC adresu nove pristupne točke, te se ta vrijednost ažurira u ovom procesu. Bez ažuriranja MAC adrese svaki odlazni paket bi se do nove pristupne točke slao preko stare pristupne točke, što bi znatno usporilo komunikaciju.

3. Opis implementiranih procesa i čvorova

Prilikom implementiranja protokola CIP, potrebno je stvoriti tri nove vrste čvora, koji imaju ulogu usmjeritelja, pristupne točke i mobilne stanice. Novi čvorovi i procesi opisani su u nastavku rada.

3.1. Čvor vm_cip_switch2

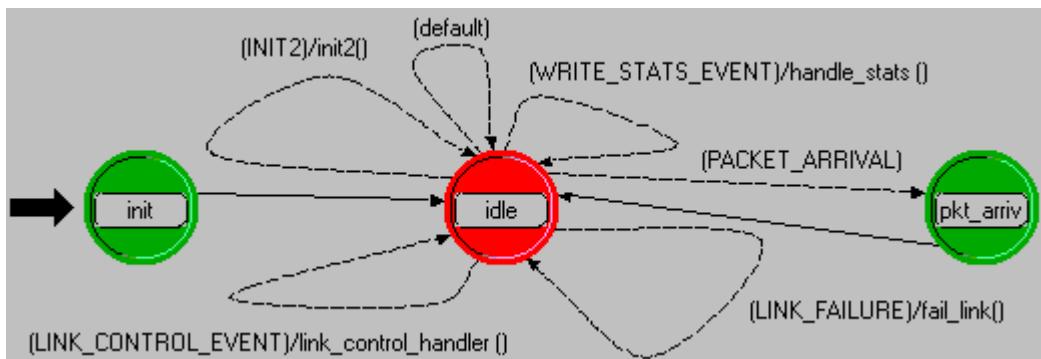


Slika 9: CIP komutator

Organizacija procesa u čvoru prikazana je na slici (Slika 9). Odmah se može uočiti da je svaki sloj OSI referentnog modela zastupljen jednom vrstom procesa. Za fizički sloj zaduženi su moduli eth_port_rx ili eth_port_tx. Sloj podatkovnog linka upravljan je modulima s imenom mac. Modul cip, kao i protokol CIP, nalazi se na trećem sloju. Kombinacija modula eth_port_tx, eth_port_rx i mac nalazi se u svakom Ethernet čvoru. Modul cip je u potpunosti nov, te niti jedan dio tog modula nije nastao od nekog drugog modula. Važno je napomenuti da su crvene i plave veze tokovi paketa.

3.1.1. Proces vm_switch_cip

Ime ovog procesa nije u skladu sa pravilom o imenovanju procesa u OpNet-u. Pravilan naziv bio bi komponenta_inicijali-autora, ako se radi o vlastitoj komponenti, dakle switch_cip_vm. Ovakvo ime odabранo je zato što su prilikom otvaranja svi procesi sortirani po abecedi, pa se zbog toga nalazi na kraju liste. Osnovna verzija ovog procesa nije složena, jer mu je jedina uloga prebacivanje paketa s jednog ulaza na drugi izlaz. Proces također zadržava pakete za vrijeme prebacivanja veze. Dijagram stanja tog procesa prikazan je na slici (Slika 10).



Slika 10 : Proces vm_switch_cip

Inicijalizacija procesa odvija se u dva koraka. Prvi korak odvija se u stanju init. U tom stanju postavljaju se standardne varijable stanja (my_id, my_node_id, own_prochandle), inicijalizira se aktivna tablica usmjeravanja i ,ako je ta opcija odabrana, tablica prozivanja, te se izvodi registracija procesa u registar procesa. Dodatna svojstva koja se zapisuju dana su u tablici (Tablica 3).

Tablica 3 : Svojstva procesa vm_switch_cip

Ime svojstva	Tip svojstva	Sadržaj svojstva
"protocol"	OMSC_PR_STRING	"cip_sw"
"node objid"	OMSC_PR_OBJID	my_node_id

Drugi dio inicijalizacije treba provesti nakon što se ostali čvorovi inicijaliziraju. Budući da se glavni dio inicijalizacije čvorova događa u nultoj sekundi, u prvom dijelu inicijalizacije postavlja se događaj koji će započeti drugi dio inicijalizacije u trenutku malo poslije nulte sekunde. Na taj način se osigurava da će se čvor inicijalizirati nakon svih ostalih čvorova, bez obzira na poredak u listi događaja. Taj korak obuhvaća pronalaženje prvog višeg čvora na putu od ovog čvora do vršnog čvora, određivanje njegove MAC adrese i porta pomoću kojeg je ovaj čvor spojen sa susjednim. Prelazak iz stanja idle u stanje pkt_arriv događa se samo ako je zadovoljen uvjet PACKET_ARRIVAL. Taj uvjet je vrlo jednostavan i zadovoljen je svaki

puta kada je na redu izvršavanje događaja koji je stvoren od strane dolznog toka paketa. Prvo što je potrebno napraviti s pristiglim paketom je odrediti koja je MAC adresa pošiljaoca paketa. Taj podatak je važan, jer je nužan prilikom svakog ažuriranja aktivne tablice usmjeravanja ili tablice prozivanja. Nakon što je određena navedena adresa, IP datagram se dekapsulira iz pristiglog paketa te se pristigli paket "uništava". Nakon toga potrebno je odrediti da li je pristigli paket ICMP poruka ili uobičajeni podatkovni paket.

Ako se radi o ICMP poruci, određuje se da li je to paket za ažuriranje aktivne tablice usmjeravanja, paket za ažuriranje tablice prozivanja ili paket za kontrolu linka među čvorovima. Prve dvije vrste paketa se obrađuju prema standardu i u nepromijenjenom obliku prosljeđuju se prema višim čvorovima sve do vršnog čvora, gdje bivaju uništeni. U (češćem) slučaju radi se o paketu za kontrolu linka i on se obrađuje kao što je već opisano.

Vrlo važan korak koji se obavezno mora obaviti odnosi se na ažuriranje NATO tablice (*Numerical Address Table Optimization*). NATO tablica se koristi ako se simulacija izvodi s uključenom opcijom ARP Sim Efficiency. Ako je ta opcija uključena, tada se ARP paketi ne šalju, nego se umjesto ARP tablice za slanje paketa koristi NATO tablica. Obzirom da je usmjeravanje između CIP čvorova implementirano, potrebno je ažurirati NATO tablicu kako bi čvor koji ne podržava CIP mogao poslati podatke pravim putem. Jedan od zapisa u NATO tablici je indeks izlaznog toka podataka od čvora. S perspektive mobilne stanice i vanjske mreže, između normalne IP mreže i mobilne stanice ne postoji veza jer se između ne koriste standardne OpNet komponente. Zbog toga je potrebno ažurirati tablicu kako bi podaci iz vanjske mreže mogli doći do mobilne stanice.

Tablica 4 : Pravila usmjeravanja IP datagrama u CIP komutatoru

Ulazni port	Način usmjeravanja	Tip čvora
Veza sa višim čvorom (engl. <i>uplink neighbor</i>) ili veza s vanjskom mrežom u slučaju gateway čvora	Obavezno je usmjeravanje prema nižem čvoru (engl. <i>downlink neighbor</i>)	Gateway ili CIP komutator
Veza s nižim čvorom	Obavezno usmjeravanje prema gatewayu	CIP komutator
Veza s nižim čvorom	Ako ne postoje podaci o odredišnoj IP adresi niti u jednoj tablici, tada se usmjerava prema vanjskoj mreži. U protivnom paket se prosljeđuje prema jednom od nižih čvorova.	Gateway

Podatkovni IP paket prosljeđuje se ovisno o tome na koji je port stigao. U tablici (Tablica 4) dana su pravila po kojima se usmjeravaju podatkovni paketi.

Prilikom dolaska paketa s nižeg čvora nužno je obnoviti vrijeme istjecanja zapisa u tablici prozivanja i aktivnoj tablici usmjeravanja, kako ne bi bilo potrebno često slati kontrolne pakete. Usmjeravanje protokola CIP nije u potpunosti implementirano. Trenutna implementacija protokola pretpostavlja da svaki CIP komutator ima tablicu prozivanja. Ta pretpostavka mora biti zadovoljena jer nije implementirana funkcija razašiljanja paketa prema svim nižim susjedima. Navedena funkcija je nužna, budući da po standardu čvorovi koji nemaju zapis o određenoj mobilnoj stanici u aktivnoj tablici usmjeravanja, a ne posjeduju tablicu prozivanja, pakete koji dolaze sa višeg čvora moraju razaslati nižim čvorovima. Trenutna implementacija u takvom slučaju uništava paket. Mreža koja služi za simuliranje protokola podešena je tako da je navedena pretpostavka uvijek zadovoljena. Prilikom planiranja mreže uobičajeno je omogućiti korištenje tablice prozivanja u čvorovima kroz koje prolazi veliki promet, kako bi se izbjegla česta razašiljanja. U slabo korištenim čvorovima povremena razašiljanja ne predstavljaju veliki problem. Prema tome, ako je pretpostavka da mobilne stanice međusobno vrlo malo komuniciraju točna, čvorovi bliže vršnom čvoru bi trebali imati tablice prozivanja, jer se u vršnom čvoru koncentrira velika većina prometa .

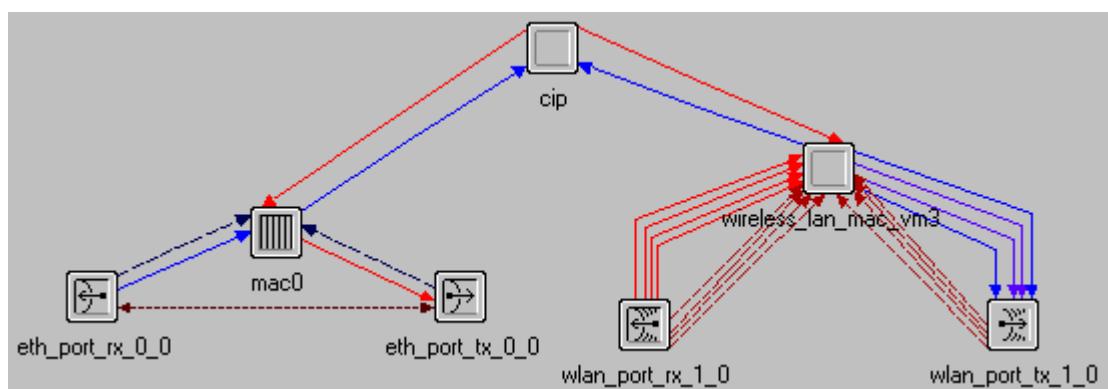
Tablica 5 : Svojstva koja je moguće mijenjati u Network Model Editor-u

Ime svojstva	Tip	Prepostavljena vrijednost	Objašnjenje
IP Address	string		IP adresa označenog čvora
Uplink Neighbor IP Address	string		IP adresa prvog višeg čvora na putu do gatewaya.
Gateway Node	toggle	disabled	Određuje da li označeni čvor obavlja funkciju gatewaya
Paging Cache	toggle	enabled	Određuje da li označeni čvor posjeduje tablicu prozivanja. U trenutnoj implementaciji ovo svojstvo mora imati vrijednost enabled
Link failure control	toggle	enabled	Određuje da li se prilikom simulacije koristi dodatno svojstvo za kontrolu ispravnosti linka prema višem čvoru
Alt-neighbour IP address	string		IP adresa zamjenskog čvora (Alt je kratica od alternative)

Prilikom implementiranja procesa, dodana su nova svojstva koja su promovirana na razinu čvora. U tablici (Tablica 5) su zajedno sa kratkim objašnjenjem navedena dodana svojstva.

IP adresa obavezno se upisuje kao *string*. Nakon toga se pomoću funkcija implementiranih u datoteci s vanjskim izvornim kodom `ip_addr_v4`, taj *string* pretvara u važeću IP adresu. Dobro je napomenuti da je tip `IpT_Address` u biti tip `unsigned int`. Budući da je tip `int` 32 bitni, on je idealan za pohranjivanje IP adrese. Tip `toggle` može poprimiti vrijednosti `enabled` ili `disabled`. Prilikom implementacije nekog procesa, vrijednost `enabled` je definirana konstantom `OPC_BOOLINT_ENABLED`. `Disabled` je definiran konstantom `OPC_BOOLINT_DISABLED`.

3.2. Čvor `wlan_ethernet_router_adv_vm6`



Slika 11 : CIP pristupna točka

Kao što je vidljivo iz imena, čvor je nastao preinakom čvora `wlan_ethernet_router_adv`. Originalni čvor sastoje se od osamnaest procesa, te su u njemu implementirani složeni protokoli usmjeravanja. Također je implementiran protokol TCP. Sva ponuđena funkcionalnost je u ovom radu nepotrebna, te je jedina uloga ovog čvora veza žičanog i bežičnog medija. Pojednostavljena, pristupna točka prikazana je na slici (Slika 11). Ovaj čvor bi po standardu trebao, kao i svaki drugi CIP čvor, imati tablicu usmjeravanja i po želji tablicu prozivanja. Budući da je ovo krajnji čvor žičane mreže i više nema potrebe za usmjeravanjem, u ovom čvoru nisu implementirane navedene tablice. Tokovi paketa koji povezuju čvorove imaju slijedeća dodatna svojstva. Svaki tok ima dodatno svojstvo «`ip addr indeks`», kako bi prilagodba modula `mac` i `wireless_lan_mac` s modulom `cip` bila što lakša. Nadalje, svaki tok, ovisno o svojem smjeru i izvorišnom i odredišnom modulu ima jedno od sljedećih svojstava: «`wlan_in_strm`», «`wlan_out_strm`», «`ethernet_in_strm`», «`ethernet_out_strm`». Svrha ovih tokova biti će detaljnije opisana u slijedećem poglavlju. Tablica prikazuje najvažnija svojstva ovog čvora.

Tablica 6 : Najvažnija svojstva čvora wlan_etherent_router_adv_vm6

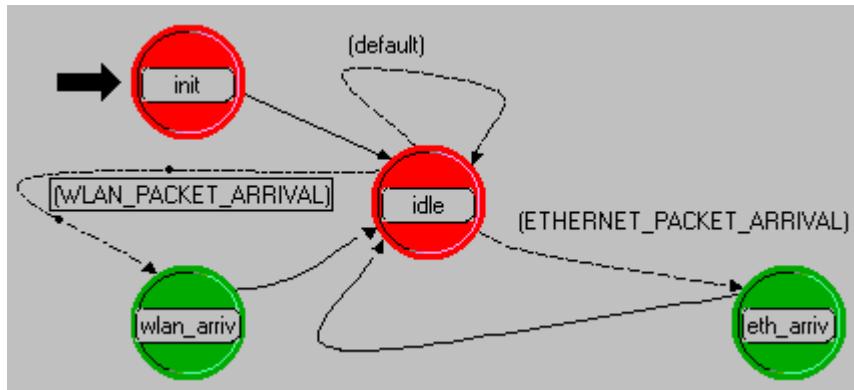
Ime svojstva	Tip	Prepostavljena vrijednost	Objašnjenje
IP Address	string		IP adresa označenog čvora
Uplink Neighbor IP Address	string		IP adresa prvog višeg čvora na putu do gatewaya.
Station address	int		MAC adresa bežičnog sučelja
condition	toggle	Enabled	Svojstvo koje određuje da li čvor radi ili je u kvaru
ACK	toggle	Enabled	Određuje da li se šalju potvrde na kontrolne pakete

3.2.1. Proces vm_ap_cip

Ime ovog procesa također nije po standardu OpNet-a, te je tako odabранo iz već objašnjeno razloga. Uloga ovog procesa slična je ulozi procesa vm_switch_cip, samo što u ovom procesu nije implementirana nikakva logika o usmjeravanju paketa.

Slika (Slika 12) prikazuje dijagram stanja procesa vm_ap_cip. U stanju init izvodi se standardna inicijalizacija varijabli stanja te se obavlja registracija procesa u globalni registar procesa. Svojstva koja se zapisuju u registar procesa su ista kao i u procesu vm_switch_cip, pri čemu je vrijednost svojstva «protocol» jednaka «cip_ap». Kako bi se odredio koji je ulazni, a koji izlazni tok paketa prema nekom od modula nižeg sloja koriste se prije navedena dodatna svojstva svakog toka podataka. Takav način određivanja indeksa toka podataka za određeni ulaz ili izlaz iz modula nije uobičajen. Obično tokovi podataka nemaju nikakva dodatna svojstva. Tada je potrebno odrediti identifikator svakog dolaznog ili odlaznog toka podataka. Nakon toga potrebno je odrediti identifikator modula na koji je taj tok spojen. Iz identifikatora modula, dobije se tip modula, te se tada u jednu od varijabli stanja upisuje indeks tog čvora.

Implementirana metoda znatno je jednostavnija. Najprije je potrebno prikupiti identifikatore svih ulaznih i izlaznih tokova cip modula. Nakon toga se pregledavaju dodatna svojstva svakog prikupljenog toka podataka. Ako neko svojstvo postoji, tada se indeks tog toka zapisuje u pridruženu mu varijablu stanja. Taj način pribavljanja možda simulacijski nije brži, ali je puno jednostavniji za implementiranje. Sporost tog postupka nije niti toliko važna, jer se taj postupak izvodi samo jednom prilikom inicijalizacije procesa.



Slika 12 : Proces vm_ap_cip

Stanja wlan_arriv i eth_arriv krajnje su jednostavna. Zadaća prvog stanja je da preuzme paket s ulaznog toka paketa, stvori ICI poruku za Ethernet (ime poruke je «ip_mac_req») te taj paket pošalje na izlazni tok podataka. Tablica (Tablica 7) prikazuje vrijednosti svojstava u ICI poruci uz objašnjenja zašto se ista koriste.

Tablica 7 : Postavljena svojstva u ICI poruci ip_mac_req

Ime svojstva	Vrijednost svojstva	Objašnjenje
dest_addr	Uplink_phy_addr	Ovdje se upisuje fizička adresa prvog višeg čvora.
type_of_service	-1	Prilikom stvaranja ip paketa u modulu ip_encap, u TOS polje u ip datagramu upisuje se ova vrijednost.
Protocol_type	NET_PROT_IP	Tip protokola koji se koristi u ovoj simulaciji (IP)

Stanje eth_arriv vrlo je slično stanju wlan_arriv. Jedina razlika je što se umjesto «ip_mac_req» ICI poruke koristi poruka «wlan_mac_ind». Vrijednosti svojstava ove poruke nalaze se u tablici (Tablica 8).

Tablica 8: Postavljena svojstva u ICI poruci wlan_mac_ind

Ime svojstva	Vrijednost svojstva	Objašnjenje
dest_addr	downlink_mac_addr	MAC adresa mobilne stanice na koju se šalje paket
src_addr	My_mac_addr	MAC adresa ove bazne stanice
Protocol_type	NET_PROT_IP	IP je protokol koji se koristi u ovoj simulaciji

Nužno je napomenuti da se u oba stanja ICI poruke vežu uz događaj, a ne uz paket.

U ovom čvoru je potrebno implementirati slanje potvrda na dolazeće kontrolne pakete. Implementacija dodatka nalazi se u stanju wlan_arriv. Izbor stanja je logičan, jer kada paket stigne sa zračnog medija, odmah se mora poslati potvrda. Kao što je već navedeno, potvrda se šalje prioritetno. Kako mobilna stanica može slati pakete za ažuriranje aktivne tablice

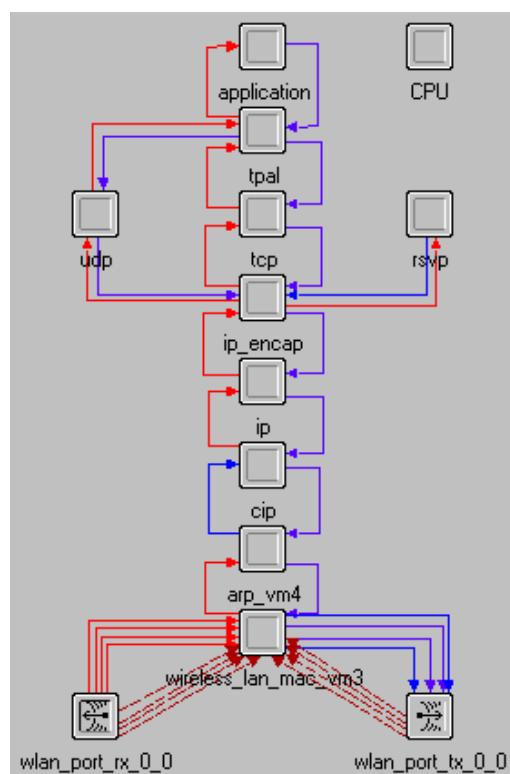
usmjerenja odnosno tablice prozivanja, potrebno je u potvrdu upisati koja vrsta paketa se potvrđuje. Zbog toga se u code polje ICMP paketa mogu zapisati slijedeće konstante:

- a. PU_ACK, ako se radi o potvrdi paketa za ažuriranje tablice prozivanja,
 - b. RU_ACK, ako se radi o potvrdi paketa za ažuriranje aktivne tablice usmjeravanja.

Paket potvrde stvara se kao i svaki drugi ICMP paket, a šalje se pozivom pomoćne funkcije `download_packet()`. Ta funkcija koristi se i za slanje podatkovnih paketa.

Kako ovaj proces nema implementiran nikakav spremnik paketa, nego je spremnik implementiran u procesu za upravljanje MAC slojem, nema načina kako bi se u ovom čvoru omogućilo prioritetno slanje kontrolnih poruka. Zbog toga je potrebno u procesu wireless_lan_mac_vm4 u pomoćnoj funkciji wlan_hlpk_enqueue () izvršiti provjeru da li se radi o kontrolnom paketu ili o podatkovnom paketu. Tip paketa može se doznati pozivom funkcije op_pk_format (). Ako se radi o kontrolnom paketu, tada se kao parametar funkcije za punjenje liste op_prg_list_insert () prosljeđuje konstanta OPC_LISTPOS_HEAD (paket se stavlja na početak liste). Za ostale pakete koristi se konstanta OPC_LISTPOS_TAIL (paket se stavlja na kraj liste).

3.3. Čvor wlan_wkstn_adv_vm4



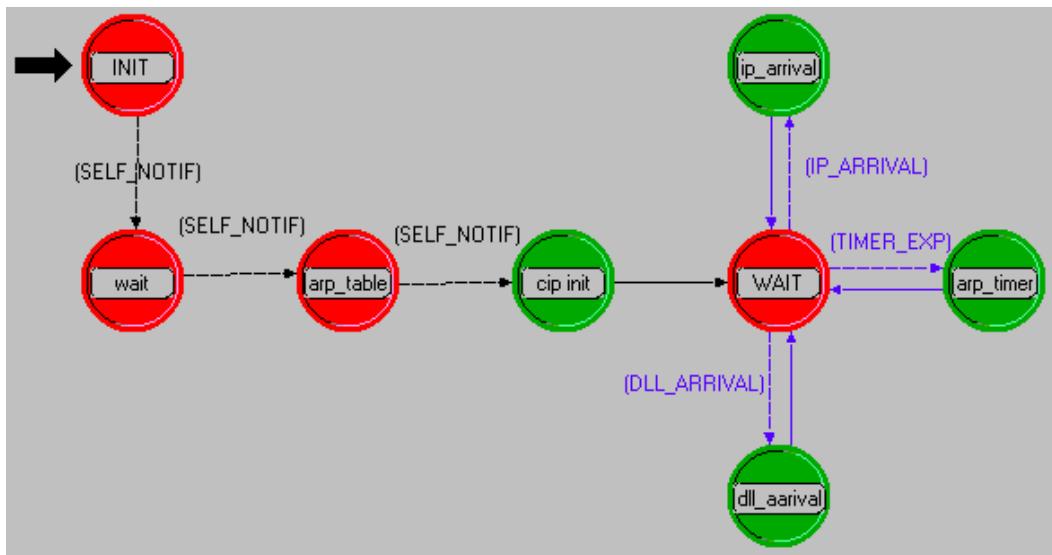
Slika 13: Model čvora wlan wkstn adv vm4

Slika (Slika 13) prikazuje model čvora wlan_wkstn_adv_vm4. Čvor je izведен iz čvora wlan_wkstn_adv. Očita promjena koja je uvedena odnosi se na umetanje modula cip između modula arp_vm4 i modula ip. Modul arp_vm4 znatno je prerađen, kako bi mogao funkcionirati sa modulom cip. Tablica (Tablica 9) prikazuje najznačajnija dodana svojstva čvora.

Tablica 9 : Najznačajnija dodana svojstva čvoru wlan_wkstn_adv_vm4

Ime svojstva	Tip	Prepostavljena vrijednost	Objašnjenje
Uplink Neighbor IP Address	string		IP adresa pristupne točke
Semi-soft handoff	toggle	enabled	Određuje da li se kao metoda prebacivanja koristi polumeko ili tvrdo prebacivanje
ACK	toggle	enabled	Određuje da li se očekuju potvrde na kontrolne pakete

3.3.1. Proces ip_arp_v4_vm4



Slika 14: Proces ip_arp_v4_vm4

Kao što je objašnjeno u jednom od prethodnih poglavlja, taj proces je pisan pod pretpostavkom da se iznad njega nalazi modul ip. Korisno je zamijetiti da veze procesa nižih slojeva s procesima viših slojeva nisu veze tipa "statistic wire" nego su tokovi paketa, ali je tim strelicama naknadno promijenjena boja. Promjene u modulu wireless_lan_mac opisane su u jednom od prethodnih poglavlja, te se ovdje neće dodatno opisivati.

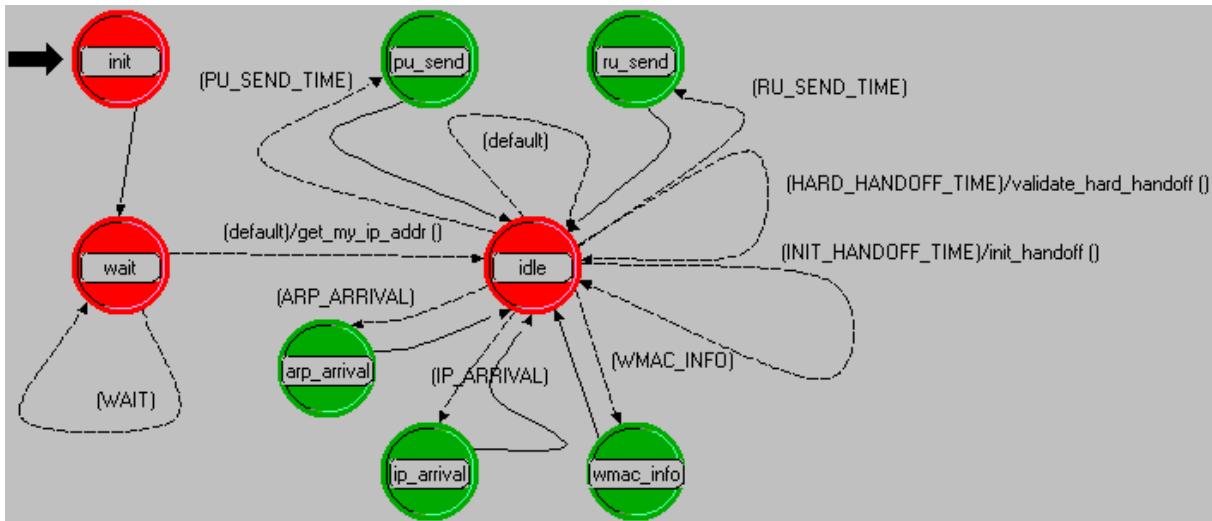
Slika (Slika 14) prikazuje dijagram stanja procesa ip_arp_v4_vm4. U proces je dodano stanje cip init koje se izvršava nakon svih ostalih inicijalizacijskih stanja. Jedina promjena u stanju INIT je inicijalizacija vlastite statistike koja broji broj paketa koji su prošli kroz ovaj proces. Ta statistika služi za lakše otklanjanje grešaka (utvrđuje se da li se neki paketi zagube i da li okolni procesi uopće šalju pakete). Ona se ažurira u stanju ip_arrival. Budući da ovaj proces originalno pisan pod pretpostavkom da se iznad njega nalazi proces ip_dispatch, a ispod njega modul za upravljanje MAC podslojem, u stanju arp_table bile su nužne velike promjene. Prvo je potrebno provjeriti da li se modul nalazi u čvoru u kojem je prisutan cip modul. Ako taj uvjet nije zadovoljen, tada se stanje izvršava isto kao i originalno, nepromijenjeno, stanje. U protivnom, najprije se detektiraju tokovi podataka prema jednom višem i jednom nižem sloju. Nadalje, detektiraju se tokovi podataka od višeg sloja prema još jednom višem sloju i od nižeg sloja prema još jednom nižem sloju. Ako je modul cip okružen modulima kao na slici (Slika 13) tada su detektirani tokovi podataka prema modulima cip, ip, wireless_lan_mac_vm3 i wlan_port_tx_0_0. Detektiranje svih tih tokova nije potrebno, ali ta metoda je odabrana zbog jednostavnosti i čitljivosti izvornog koda. Budući da se ova detekcija izvršava samo jednom u fazi inicijalizacije procesa, detektiranje nepotrebnih tokova podataka ne uzrokuje smanjenje brzine simuliranja. Nadalje, među svim tim tokovima potrebno je odrediti koji tok ima dodatno svojstvo «ip addr index», jer je to svojstvo potrebno za daljnju inicijalizaciju. Zadnja promjena potrebna u ovom stanju odnosi se na određivanje tokova koji povezuju arp modul sa modulom višeg sloja. Očito je da je potrebno odabratи tokove koji počinju i završavaju u modulu cip, a ne, kao u originalnom modulu, u procesu ip.

Stanja ip_arrival i dll_arrival također su promijenjena. Najznačajnija promjena je brisanje velikog dijela koda koji upravlja procesom ako opcija ARP Sim Efficiency nije uključena. Na početku tih stanja izvodi se provjera da li je opcija uključena. Ako nije, simulacija se trenutno prekida. Stanje dll_arrival tada postaje krajnje jednostavno i samo proslijeđuje pakete sa nižeg sloja na viši. Stanje ip_arrival nije tako jednostavno i ima dva načina rada. Ako se arp modul nalazi u mobilnoj stanci, tada se svi paketi usmjeravaju na pristupnu točku. U protivnom, paketi se usmjeravaju po pravilima zapisanima u NATO tablici. Ta tablica, kao što je već spomenuto, koristi se za usmjeravanje ako se ne šalju arp kontrolni paketi.

Stanje cip_init služi za inicijalizaciju varijabli stanja (state variables) vezanih uz proces cip. Najprije je potrebno odrediti da li se ovaj proces nalazi u pristupnoj točki ili mobilnoj stanci. Nadalje, potrebno je odrediti identifikator (Objid) modula cip. Nakon što je određen identifikator modula cip, šalje se ICI poruka čiji parametar «code» ima sadržaj

`CREATE_AP_LIST`. Primitkom te poruke, modul cip stvara popis svih pristupnih točaka u mreži. Popis se ne stvara, kao u stvarnom slučaju, na temelju primljenih *beacon* okvira. Filtriranjem sadržaja registra svih procesa, dobiva se popis svih pristupnih točaka. Nadalje, potrebno je odrediti MAC i IP adresu čvora kojemu se prosljeđuju svi paketi. Inicijalizacija tih varijabli nužna je za ispravan rad stanja `ip_arrival` i `dll_arrival`.

3.3.2. Proces `vm_cip`



Slika 15 : Dijagram stanja procesa `vm_cip`

Proces `vm_cip` je najsloženiji od svih implementiranih procesa. Kao što je na slici (Slika 15) prikazano i ovaj proces počinje stanjem `init`. Najprije se izvršava registracija procesa u globalni registar procesa. Nakon toga se određuju tokovi podataka prema višem i nižem sloju. Nakon inicijaliziranja varijabli stanja, potrebno je registrirati događaje koji će izazvati slanje paketa za ažuriranje tablice prozivanja i aktivne tablice usmjeravanja. Ti paketi se ne šalju odmah, nego nekoliko μ s nakon što počne simulacija. Taj vremenski pomak nužan je kako bi se svi procesi i čvorovi inicijalizirali. Problem je u tome što se događaj za slanje paketa generira u prvom stanju procesa, a procesi kao što su `arp` i `ip` inicijaliziraju se kroz nekoliko stanja. Kada bi se paketi slali u trenutku kada je simulacijsko vrijeme nula sekundi, tada bi se ti paketi poslali prije nego što bi bilo dovršena inicijalizacija drugih procesa i simulator bi prijavio pogrešku. Zbog istog razloga postoji i stanje `wait`². Proces se u tom stanju može nalaziti proizvoljan broj događaja. Testiranjem procesa utvrđeno da se prijelaz `WAIT` nikad

² Obratiti pozornost na to da se stanje zove `wait`, a prijelaz `WAIT`. Također ulaskom u *unforced* (crveno) stanje predaje se kontrola simulacijskoj jezgri pa se tada mogu izvršavati prekidi u drugim procesima.

ne mora izvoditi i da se funkcija `get_my_ip_addr()` ispravno odvija. Bez stanja `wait`, funkcija `get_my_ip_addr()` ne radi ispravno, jer modul `ip` još uvijek nije inicijaliziran. Nakon inicijalizacije proces dolazi u stanje `idle`. Iz tog stanja, ovisno o pristiglom događaju, prelazi u ostala stanja. Sva stanja u koja proces može preći iz stanja `init` su tipa *forced* tako da se ona izvode unutar jednog događaja.

Najjednostavnije stanje je stanje `arp_arrival`. Osnovna funkcija tog stanja je prosljeđivanje paketa na višu razinu. Proces `ip_arrival`, osim prosljeđivanja paketa na niži sloj, izvodi pretvaranje ICI poruka sa višeg sloja (`ip_arp_req_v4`) u vlastite poruke (`ip_arp_req_v4_cip`). Pretvaranje poruka nužno je kako bi modul `cip` mogao kontrolirati na koju će se pristupnu točku slati paketi. Stanja `pu_send` i `ru_send` izvode se kada je potrebno poslati paket za ažuriranje tablice prozivanja odnosno aktivne tablice usmjeravanja. Osnovna zadaća tih stanja je stvaranje ICMP paketa koji će se poslati prema pristupnoj točki. Ovisno o stanju koje se izvodi, ICMP paket ažurira tablicu prozivanja ili aktivnu tablicu usmjeravanja, te je, kao što je već prije opisano, propagiran do vršnog čvora. Proces prijelazi u stanje `wmac_info` kada od procesa `wireless_lan_mac_vm4` dobije ICI poruku vezanu uz specifični događaj. To stanje se izvodi kada je potrebno odrediti sve pristupne točke u mreži (jednom, prilikom inicijalizacije sustava) i svaki puta kada se šalje nova vrijednost odnosa signal-šum. U ovom se stanju također odlučuje kada će se i na koji način izvoditi prebacivanje veze. Vrsta prebacivanja veze odabire se prije simulacije mijenjanjem svojstva `Semi-soft handoff`.

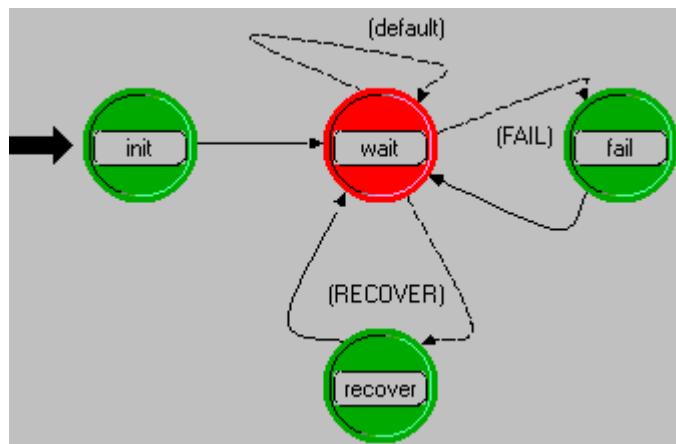
U slučaju da se koristi tvrdo prebacivanje veze, prijelaz `INIT_HANDOFF_TIME` neće se nikad izvesti. Taj prijelaz izvodi se samo u slučaju da na kontrolni paket koji se šalje za vrijeme nije stigla potvrda o uspješnom primitku. Tada se ponovno inicijalizira prebacivanje slanjem još jednog kontrolnog paketa. Za vrijeme inicijalizacije prebacivanja, generira se događaj, koji će potaknuti prijelaz `HARD_HANDOFF`. Taj prijelaz poziva funkciju `validate_hard_handoff()`. Svrha te funkcije biti će objašnjena na slijedećem primjeru. Neka za vrijeme polumekog prebacivanja mobilna stanica ne dobije potvrdu o uspješnom slanju kontrolnog paketa. Tada će se za vrijeme druge inicijalizacije generirati drugi događaj za pokretanje tvrdog prebacivanja. Ova funkcija osigurava da se tvrdo prebacivanje odradi samo jednom. U slučaju višestrukog prebacivanja, simulacija ne bi davala točan rezultat, jer bi se više puta prekinula komunikacija mobilne stanice sa pristupnom točkom. Iako je vrlo važna, sve što ova funkcija u biti radi je odgađanje *tvrdog* prebacivanja za malo kasnije, u slučaju da svi kontrolni paketi nisu potvrđeni. *Tvrdo* prebacivanje izvodi se kroz dvije faze. U prvoj fazi ažuriraju se neke globalne varijable, tako da se od sada paketi šalju preko bolje pristupne

točke. Nakon toga potrebno je generirati događaj za izvođenje druge faze prebacivanja. U ovoj fazi se isključuje proces koji kontrolira bežični medij postavljanjem svojstva "condition" na "disabled". Sada proces za kontrolu bežičnog medija odbacuje sve primljene prekide. Prvobitno su se svi paketi pristigli za vrijeme prebacivanja odbacivali u arp_arrival stanju. Takav način rada nije ispravan, jer je drugi sloj normalno funkcionirao. Proces wireless_lan_mac_vm4 normalno je potvrdio pakete koje je primio, i poslao ih na viši sloj. Mijenjanjom svojstva "condition", to je izbjegnuto i time je postignuto da čvor zaista ne može primiti nikakav paket, pa se paketi koje pristupna točka šalje nakon nekoliko neuspješnih pokušaja slanja brišu.

3.4. Čvor cip_failure

Svrha ovog čvora je simuliranje prekida i obnavljanje linkova, te kvarenja i popravljanja CIP komutatora. Ovaj čvor je nužan za testiranje i simuliranje dodatka protokola koji je zadužen za kontrolu ispravnosti linka. Čvor se sastoji samo od procesa cip_fail_recover.

3.4.1. Proces cip_fail_recover



Slika 16: Čvor cip_fail_recover

U stanju init svi se retci datoteke cip_fail_rec.gdf pomoću funkcije op_prg_gdf_read () prebacuju u listu. Ova funkcija je vrlo korisna, jer se u listu ne stavljaju komentirani i prazni retci. Analiziranjem svakog retka posebno, generiraju se događaji koji će uzrokovati prijelaze u stanje fail ili u stanje recover. Datoteka ima slijedeći oblik:

```
#node/link_name;fail_time;recover_time
cip_switch2;4;5
```

Prvi parametar je ime linka ili CIP komutatora. Drugi i treći parametar određuju vrijeme u sekundama kada će se objekt pokvariti odnosno popraviti. Prije analiziranja liste potrebno je utvrditi koliko u podmreži postoji linkova, a koliko fiksnih čvorova. Identifikator podmreže moguće je dobiti tako da se zatraži identifikator roditelja od roditelja od ovog procesa ili:

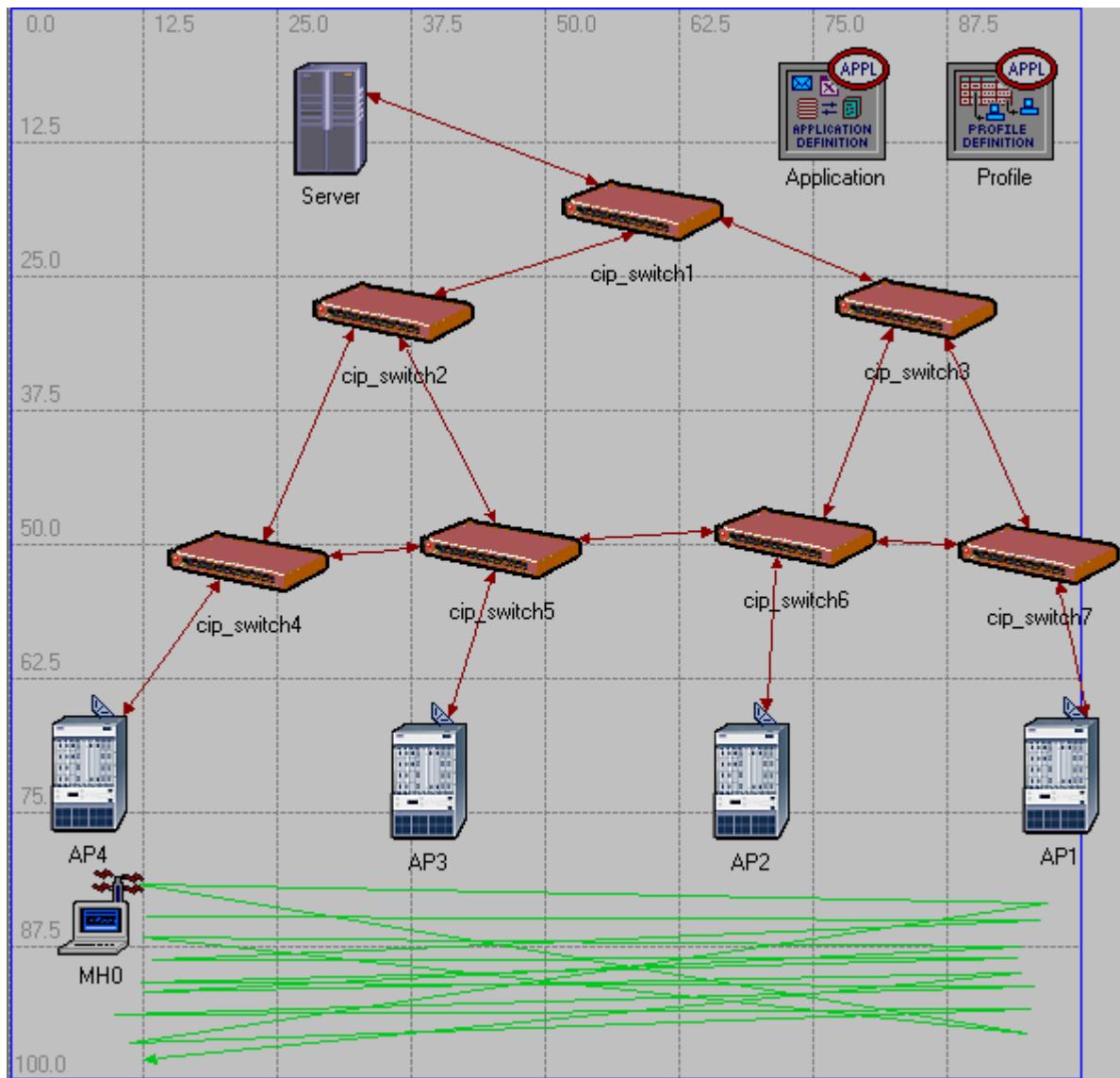
```
subnet_objid = op_topo_parent(op_topo_parent (op_id_self()));
```

Roditelj od procesa je čvor, dok je roditelj čvora podmreža. Funkcija op_topo_child_count() pribavlja tražene iznose. Jedan od parametra koji funkcija zahtjeva je vrsta objekata koji će se brojati. U slučaju linkova proslijeduje se konstanta OPC_OBJTYPE_LKDUP, dok se u slučaju fiksnih čvorova proslijeduje konstanta OPC_OBJTYPE_NDFIX. Simulacijska jezgra također ima implementiranu korisnu funkciju op_prg_str_decomp (). Pomoću nje se iz retka dobije lista sa tri stringa, jer joj se proslijeduje znak koji odvaja zasebne podatke, što je u ovom slučaju točka-zarez.

Stanja "fail" i "recover" gotovo su identična. Jedina razlika između ta dva stanja je ta što se u prvom slučaju svojstvo "condition" odabranog objekta postavlja na vrijednost OPC_BOOLINT_DISABLED, dok se u stanju "recover" to svojstvo postavlja na vrijednost OPC_BOOLINT_ENABLED. Uz događaje vezane za kvarenje ili popravljanje objekata veže se i ICI poruka. Jedini zapis u toj poruci je identifikator objekta kojeg je potrebno pokavariti odnosno popraviti.

4. Rezultati simulacije

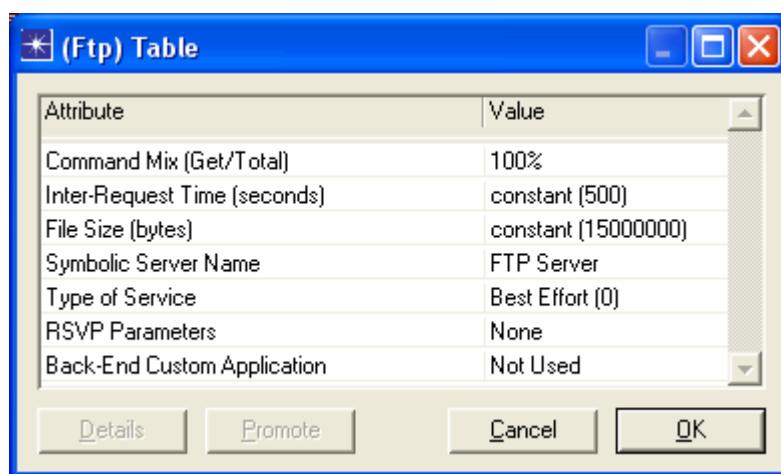
4.1. Polumeko i tvrdo prebacivanje veze



Slika 17: Mreža korištena za testiranje obje vrste prebacivanja

Korištena mreža prikazana je na slici (Slika 17). Svi parametri su jednaki u oba scenarija. U scenarijima se ne koristi potvrđivanje kontrolnih paketa i ne koristi se utvrđivanje neraspoloživosti linka. Čvor cip_switch1 ima ulogu vršnog čvora. Zelene crte na dnu slike predstavljaju ručno definirani put koji mobilna stanica prelazi. Stanica pređe s jednog kraja mreže na drugi u 30 sekundi, što znači da se prebacivanje veze u prosjeku događa svakih 10 sekundi. To vrijeme je dovoljno da se svako prebacivanje u potpunosti provede i da mobilna

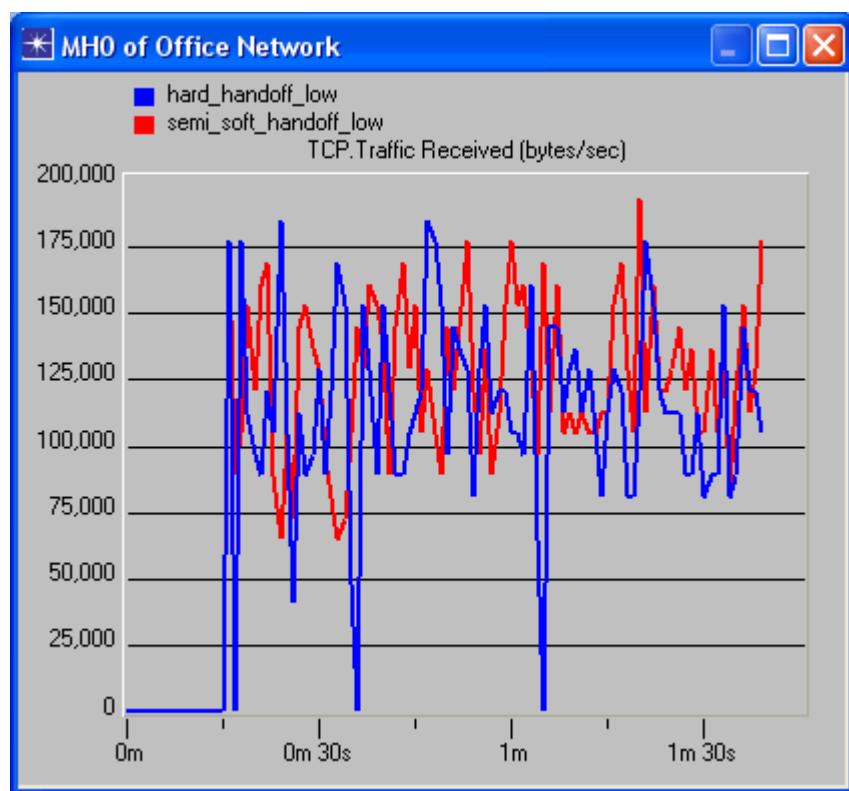
stanica određeno vrijeme prima pakete s neke pristupne točke. Brzina kretanja mobilne stanice iznosi oko 25 km/h. Za prikaz dobrih i loših stana tvrdog i polumekog prebacivanja izvedene su četiri simulacije. Dvije simulacije prikazuju rezultate polumekog i tvrdog prebacivanja, kada su pristupne točke slabije opterećene. U druge dvije simulacije poveća se količina prometa koji mobilna stanica prima. Između ta dva slučaja morala bi postojati razlika, koja proizlazi iz činjenice da za vrijeme polumekog prebacivanja veze stara pristupna točka šalje pakete kako joj stignu, dok nova pristupna točka šalje pakete sa zadrškom. Zbog toga se na kratko vrijeme medij dva puta zauzima za slanje paketa istoj mobilnoj stanici. Slučaj sa slabijim prometnim intenzitetom generiran je tako da je za link koji povezuje vršni čvor sa poslužiteljem odabran 10BaseT link. Svaki link ima svojstvo koje određuje koliki postotak kapaciteta se koristi za pozadinski promet. Ta opcija koristi se kada se simulira postojeća mreža sa određenim opterećenjima linkova, pa se testira kako bi neka nova usluga utjecala na mrežu. Pozadinsko opterećenje postavljeno je na 90%, tako da je za prijenos datoteke u prosjeku korišteno samo 10% kapaciteta linka, što u ovom slučaju iznosi 1Mbit/s. Slika (Slika 18) prikazuje generirani promet u simulaciji s manjim opterećenjem mobilne stanice. U simulacijama s većim prometnim intenzitetom, veličina prenesene datoteke iznosi 45MB. Kao što se može vidjeti i na grafovima, simulator u slučaju kad se prenosi veća datoteka, iz sada nepoznatih razloga, datoteku počinje slati s većom zadrškom.



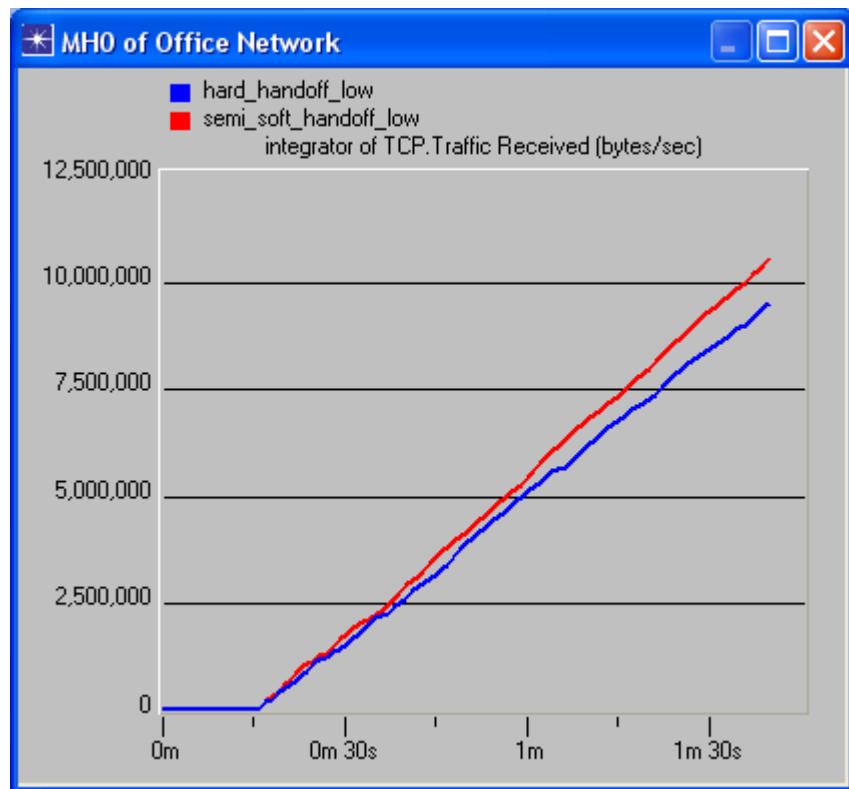
Slika 18 : Parametri generiranog prometa

Kao što je vidljivo na slici (Slika 19), tvrdo prebacivanje označeno je plavom bojom, a polumeko prebacivanje crvenom. Na slici se mogu uočiti duboki propadi u grafu koji prikazuje tvrdo prebacivanje. Ti duboki propadi nastaju zbog gubitka velikog broja paketa za vrijeme tvrdog prebacivanja. Iako je iz grafa teško razabrati koja vrsta prebacivanja je bolja, polumeko prebacivanje se čini konzistentnije. U idealnom slučaju generirani promet bio bi

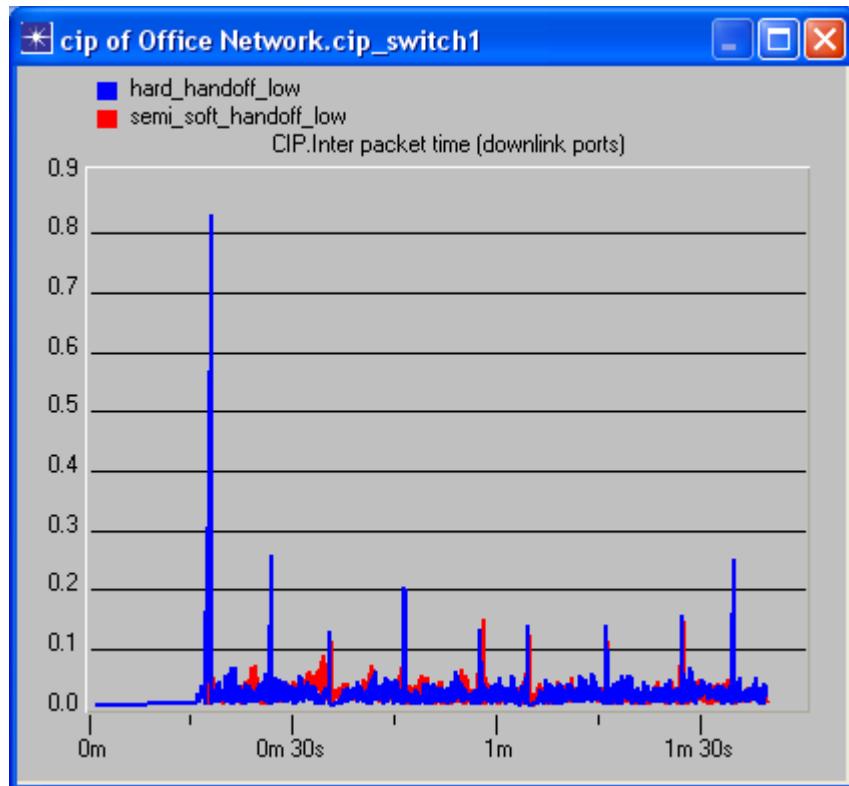
konstantan, pa bi graf zornije prikazivao slabosti i vrline prebacivanja. Budući da se pozadinsko opterećenje generira po nekoj razdiobi, raspoloživ kapacitet linka između vršnog čvora i poslužitelja varira. Slika (Slika 20) prikazuje ukupnu količinu podataka primljenu u mobilnoj stanici. Odmah je vidljivo da je polumeko prebacivanje daleko bolje od tvrdog prebacivanja. Detaljnijim pregledavanjem grafa može se uočiti da je u 124 sekundi (od početka prijenosa, do kraja simulacije), preneseno oko 1,06 MB više podataka koristeći polumeko prebacivanje veze. Slika (Slika 21) prikazuje vrijeme između dolaska dva paketa po nekom od linkova koji spajaju vršni čvor i njegove niže čvorove. Na ovom grafu također je vidljivo da polumeko prebacivanje veze ima znatno manje odstupanje od prosječnog kašnjenja.



Slika 19 : Primljeni TCP promet na mobilnoj stanici

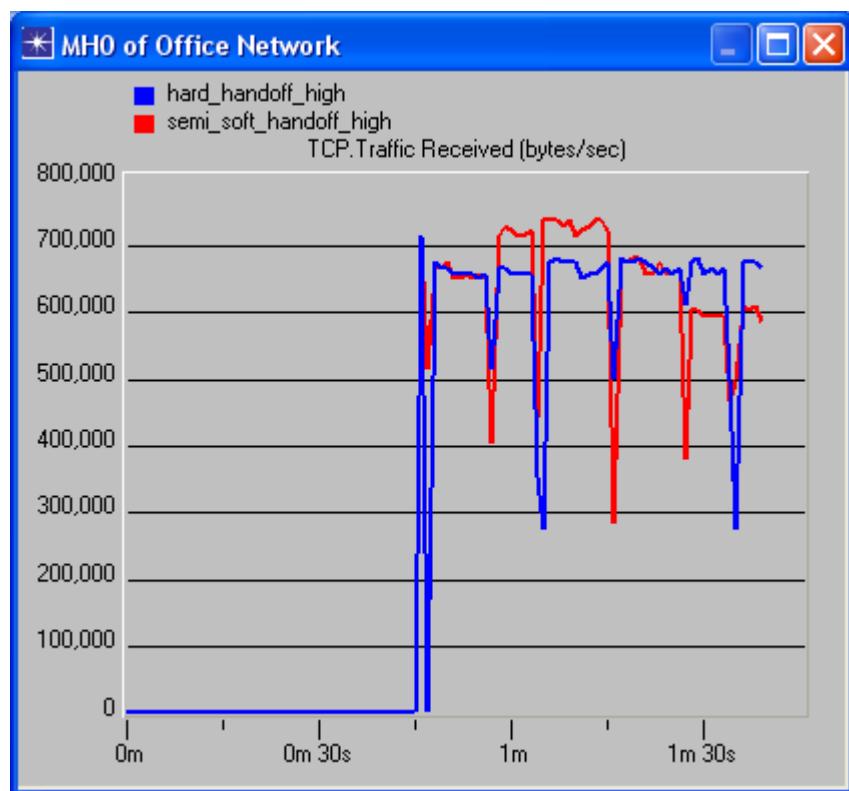


Slika 20 : Ukupan primljen promet

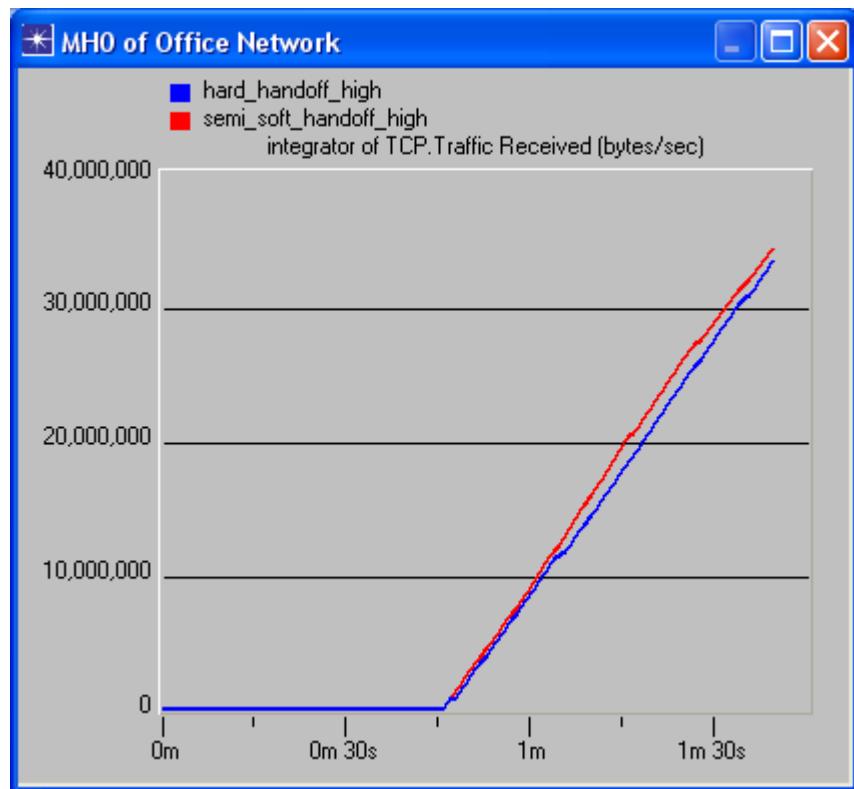


Slika 21 : Kašnjenje medu dolazećim paketima na vršnom čvoru

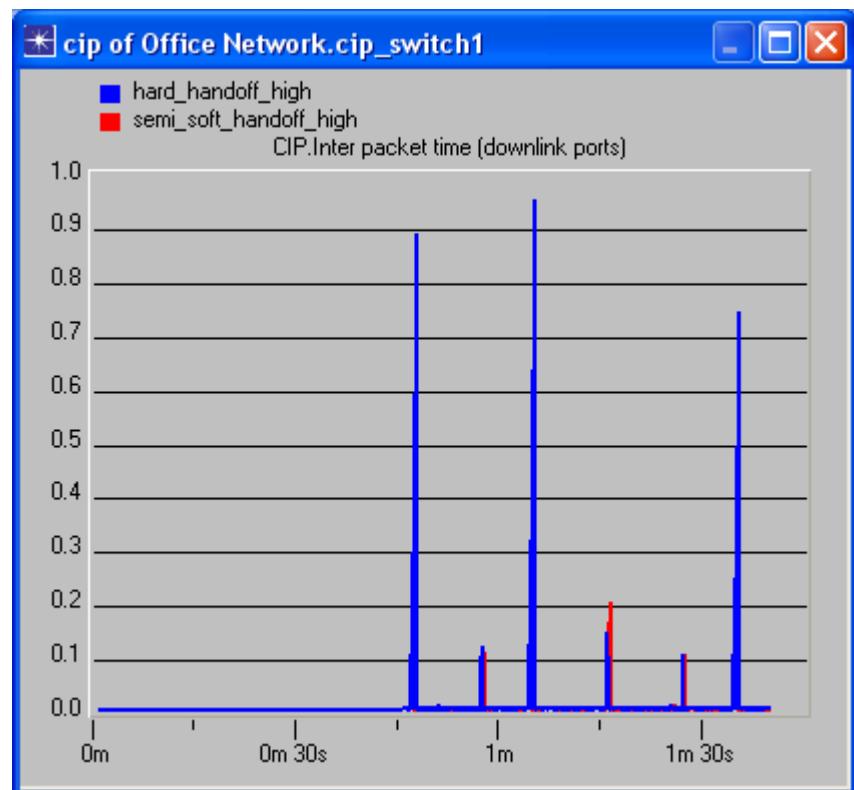
Rezultati simulacije s većim opterećenjem mobilne stanice predviđeni su na slikama: (Slika 22), (Slika 23) i (Slika 24). Iako takvi rezultati nisu bili očekivani, simulacija pokazuje da je i pod većim prometnim opterećenjem polumekro prebacivanje bolje. Na slici (Slika 22) mogu se zamijetiti propadi usporedivi sa propadima prilikom korištenja tvrdog prebacivanja. Slika (Slika 23) prikazuje da je polumekro prebacivanje bolje od tvrdog prebacivanja, iako se promatrajući prvi graf čini obratno, jer je brzina prijenosa nakon tvrdog prebacivanja uglavnom konstantna. Graf na slici (Slika 24) prikazuje da je kašnjenje između dva paketa uglavnom veće prilikom tvrdog prebacivanja.



Slika 22 : TCP promet primljen na mobilnoj stanici u slučaju većeg prometnog intenziteta



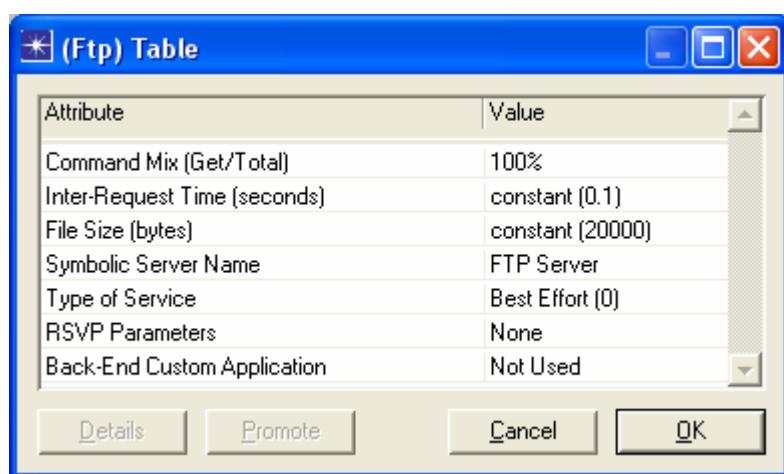
Slika 23 : Ukupan primljen promet u slučaju većeg prometnog intenziteta



Slika 24 : Kašnjenje među dolazećim paketima na vršnom čvoru u slučaju većeg prometnog intenziteta

4.2. Detekcija nedostupnosti linka prema višem čvoru

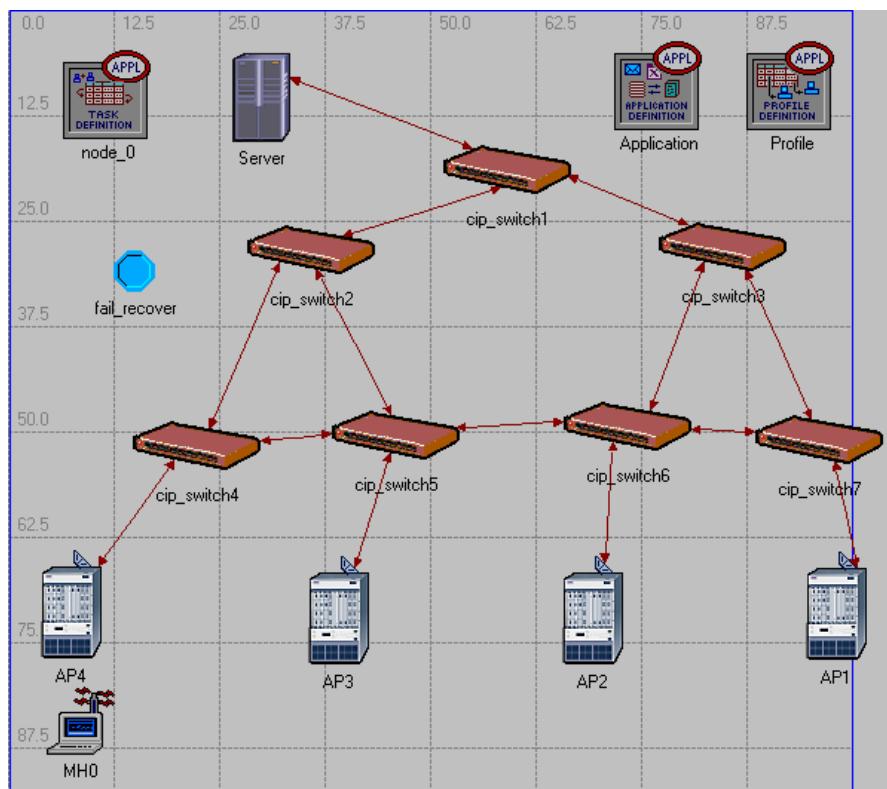
Simuliranje detekcije kvara linka izvedeno je na dvije različite mreže. Jednostavnija mreža prikazana na slici (Slika 26) korištena je za simuliranje slučaja u kojem dolazi do prekida linka. Složenija mreža prikazana na slici (Slika 27) koristi se za simuliranje slučaja u kojem se kvari CIP komutator cip_switch1. Važno je napomenuti da je u ovoj simulaciji vršni čvor cip_switch5. Tablica (Tablica 10) prikazuje koji je čvor kojem pomoćni, a koji glavni. Za razliku od simulacija probacivanja, simulacije na čvorovima su vremenski mnogo kraće, jer je osnovni cilj simulacije prikazati način na koji se provede prebacivanje na zamjenski link. Parametri generiranog prometa u te dvije simulacije prikazane su na slici (Slika 25):



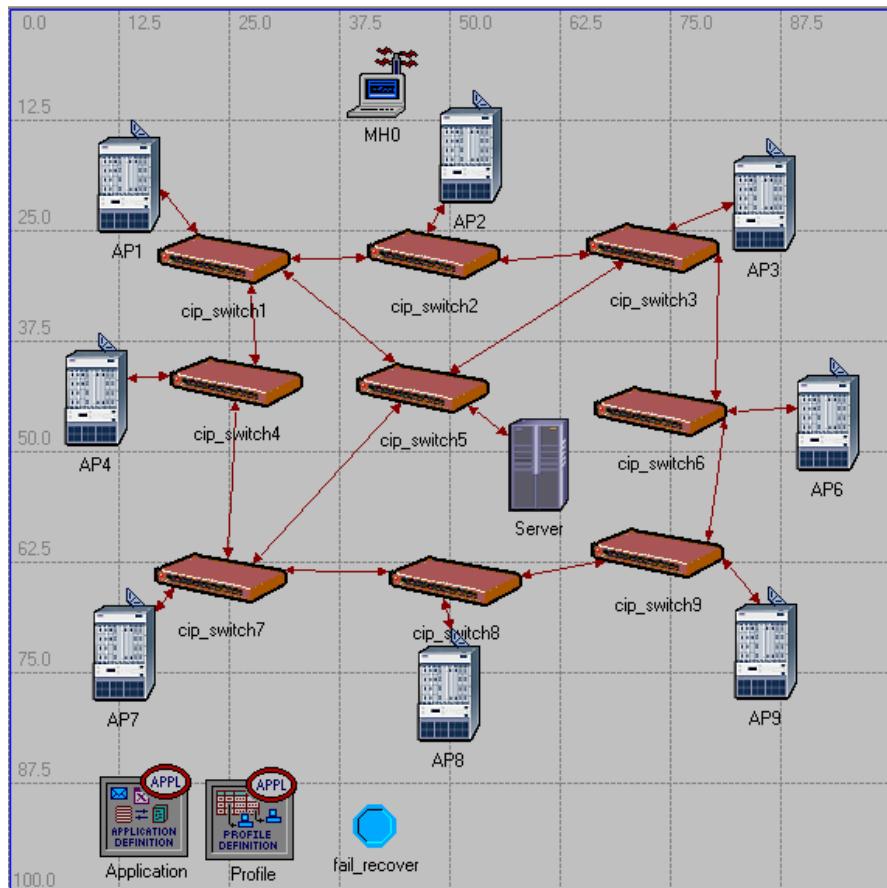
Slika 25 : Parametri generiranog prometa za simuliranje detekcije kvara linka

Tablica 10 : Veze među čvorovima

Čvor	Glavni viši čvor	Zamjenski viši čvor	Čvor	Glavni viši čvor	Zamjenski viši čvor
cip_switch1	cip_switch 5	cip_switch 4	cip_switch 2	cip_switch 1	cip_switch 3
cip_switch 3	cip_switch 5	cip_switch 2	cip_switch 4	cip_switch 1	cip_switch 7
cip_switch 5			cip_switch 6	cip_switch 3	cip_switch 9
cip_switch 6	cip_switch 3	cip_switch 9	cip_switch 7	cip_switch 6	cip_switch 4
cip_switch 8	cip_switch 7	cip_switch 9	cip_switch 9	cip_switch 8	cip_switch 6

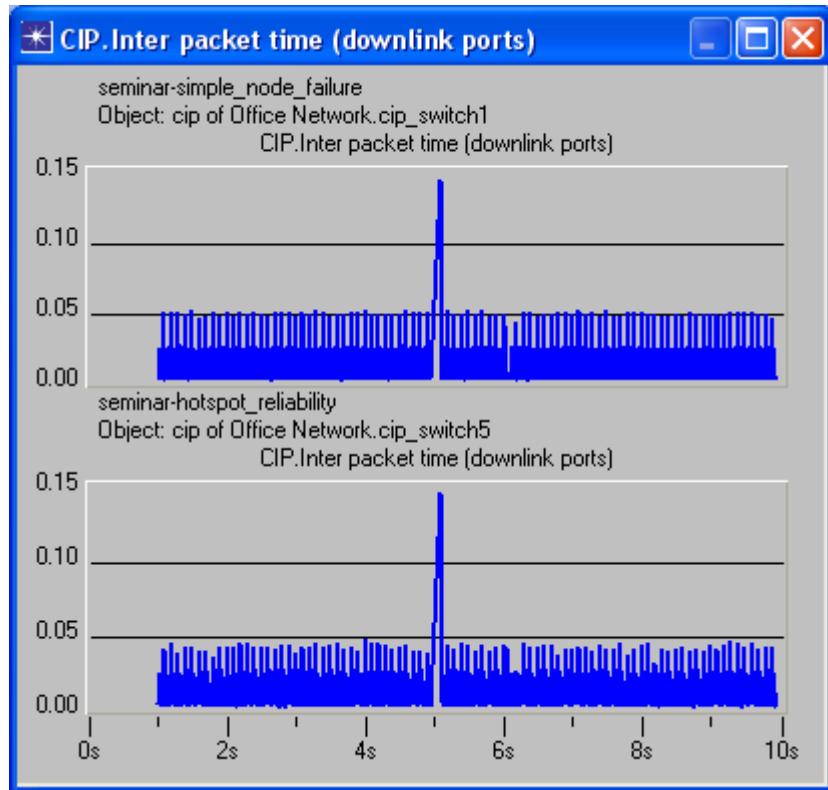


Slika 26 : Mreža za simuliranje jednostavnijeg kvara



Slika 27 : Mreža za simuliranje kvara čvora cip_switch1

Važno je zamijetiti da se u oba slučaja promet preusmjerava preko dva linka i zamjenskog CIP komutatora. Zbog toga je vrijeme između dva dolaska paketa na dolazne portove vršnog čvora jednako u obje simulacije. Važno je napomenuti da se u složenijem slučaju radi o dva različita porta, dok u jednostavnijem slučaju paketi po istom linku dolaze do vršnog čvora.

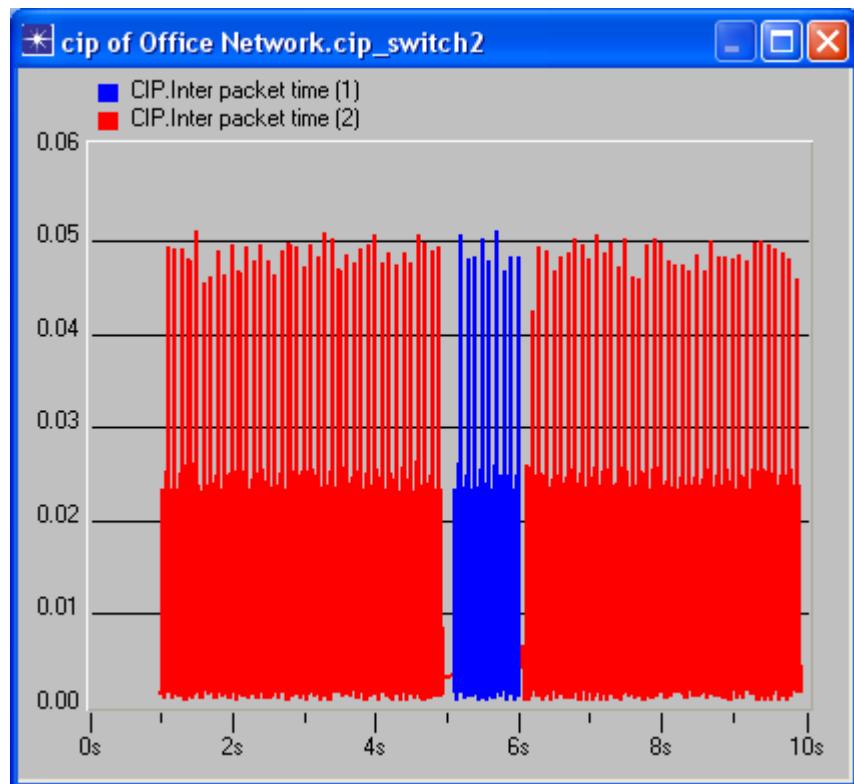


Slika 28 : Usporedba međudolaznih vremena paketa u vršni čvor

Iako slika (Slika 29) prikazuje vrijeme između dolaska dva paketa na čvoru cip_switch2 u jednostavnijem slučaju, ova slika je dana za prikaz načina prebacivanja ili promjene linka. Vidljivo je da se nakon kraće pauze potrebne za prebacivanje linka umjesto porta 1 koristi port 2. Nakon određenog vremena port 1 je ponovno raspoloživ, pa sav promet ponovno prebaci sa zamjenskog linka na glavni. Datoteka u kojoj su zadana vremena kvara i oporavka ima oblik:

```
#node/link_name;fail_time;recover_time
cip_switch4 <-> cip_switch2;5;6
#cip_switch1;5;6
#cip_switch2;5;6
```

Ovisno o simulaciji potrebno je staviti komentare na dvije linije datoteke.



Slika 29 : Međudolazno vrijeme paketa, zbog kvara linka, na različite portove komutatora

5. Zaključak

Implementacijom protokola CIP (engl. Cellular IP) u simulacijskoj paketu OpNet 8.0.C dobiveni su rezultati prikazani u poglavlju (4). Pregledom rezulata vidljivo je da je polumeko prebacivanje veze puno bolje od tvrdog prebacivanja, bilo u slučaju većeg ili manjeg prometnog opterećenja. Uvođenjem mehanizma potvrda kontrolnih paketa i mehanizma za kontrolu raspoloživosti linka, CIP postaje malo složeniji, ali također i puno pogodniji za primjenu u velikim mrežama gdje vrijeme do kvara neke komponente u sustavu nije podatak koji se zanemaruje, nego je vrlo važan pri projektiranju sustava. Ovim poboljšanjem, prilikom projektiranja velike mreže, kvaliteta komponenata ne mora biti vrhunska, jer je kvarom u nekom dijelu mreže zahvaćen vrlo mali broj korisnika. Točnije, ako je žičani dio bežične mreže dobro projektiran, u slučaju kvara nekog CIP komutatora, samo korisnici tog komutatora osjete kvar.

6. Literatura

- [1] MATTHEW S. GAST. *802.11 Wireless Networks: The Definitive Guide*. O'Reilly & Associates, Sebastopol, CA, April 2002.
- [2] VICTOR JONES. An Introduction to Handoff in Mobile Cellular Communications. March, 2004. dostupno na
http://people.deas.harvard.edu/~jones/cscie129/nu_lectures/lecture7/cellular/handoff/handoff.html
- [3] OpNet Modeler Online Documentation
- [4] PAOLO DI LORENZO. *Cellular IP: WLAN integration*. 2001. dostupno na
<http://labreti.ing.uniroma1.it/wine/papers/>
- [5] PAOLO DI LORENZO. *OPNET mobility simulation models*. 2001.
- [6] CAMPBELL, J. GOMEZ, C-Y. WAN, Z. TURANYI, A. VALKO. Cellular IP. *Internet Draft*, IETF, 1999
- [7] M. ZHAN, W. ZENG, Z. LIN. *M-TCP Performance Evaluation Using OPNET*. 2002.
- [8] <http://www.tcpipguide.com>
- [9] <http://www.bluetooth.com>
- [10] ERVIN ZENTNER. *Antene i radiosustavi*. Graphis, Zagreb, 2001.
- [11] RFC 793 Transmission Control Protocol. September 1981.
- [12] INTERLINK NETWORKS: Case study: Project Seoul , <http://interlinknetworks.com>
- [13] JAKŠIĆ, D. Analiza performansi protokola TCP-snoop u bežičnim lokalnim mrežama. Diplomski rad. FER-Zavod za telekomunikacije. 2005.
- [14] NEIL REID, RON SEIDER. *802.11 (Wi-Fi) Networking Handbook*. McGraw-Hill/Osborne. Berkley, CA, 2003.

Skraćenice

ACK	<i>Acknowledgement</i>
AP	<i>Access Point</i>
BSS	<i>Basic Service Set</i>
BSS ID	<i>Basic Service Set Identifier</i>
CDMA	<i>Code division Multiple Access</i>
CIP	<i>Cellular IP</i>
CSMA/CA	<i>Carrier Sense Multiple Access/Collision Avoidance</i>
DCF	<i>Distributed Coordination Function</i>
FTP	<i>File Transfer Protocol</i>
ICI	<i>Inter Control Information</i>
ICMP	<i>Internet Control Message Protocol</i>
IP	<i>Internet Protocol</i>
MAC	<i>Medium Access Control</i>
M-TCP	<i>Mobile TCP</i>
NAV	<i>Network Allocation Vector</i>
OSI RM	<i>Open System Interconnection Reference Model</i>
PCF	<i>Point Coordination Function</i>
PDU	<i>Protocol Data Unit</i>
PHY	<i>Physical Layer</i>
QoS	<i>Quality of Service</i>
RFC	<i>Request for Comments</i>
RTS/CTS	<i>Request To Send / Clear To Send</i>
TCP	<i>Transmission Control Protocol</i>
UMTS	<i>Universal Mobile Telecommunications System</i>
WLAN	<i>Wireless Local Area Networks</i>

Popis stranih izraza

<i>access point</i>	pristupna točka
<i>Bandwidth</i>	širina prijenosnog pojasa
<i>Beacon</i>	poruka, zraka
<i>client</i>	klijent
<i>congestion avoidance</i>	izbjegavanje zagušenja
<i>congestion window</i>	prozor zagušenja
<i>contention free</i>	bez sukoba
<i>contention free period</i>	vremenski interval bez sukoba okvira
<i>Downlink</i>	dolazni smjer
<i>end to end</i>	s kraja na kraj
<i>Fading</i>	propadanje signala
<i>handoff, handover</i>	prelaženje, prekapčanje
<i>hidden node</i>	skrivena stanica
<i>idle</i>	slobodno stanje
<i>idle time</i>	vrijeme neaktivnosti
<i>jitter</i>	kolebanje kašnjenja
<i>link capacity</i>	kapacitet linka
<i>medium access control</i>	podsloj MAC
<i>multipath fading</i>	Propadanje signala zbog višestaznog rasprostiranja signala
<i>multipath propagation</i>	višestazno rasprostiranje signala
<i>network allocation vector</i>	vektor rezervacije mreže
<i>physical carrier sense</i>	fizičko mjerjenje signala nosioca
<i>physical Layer</i>	fizički sloj
<i>point coordination function</i>	centralizirana koordinacijska funkcija
<i>protocol data unit</i>	protokolna podatkovna jedinica
<i>quality of service</i>	kvaliteta usluge
<i>real-time</i>	stvarno – vremensko
<i>router</i>	usmjerivač
<i>RTS Threshold</i>	RTS prag

<i>server</i>	poslužitelj
<i>Stream</i>	tok
<i>throughput</i>	propusnost
<i>timer</i>	vremenski brojač
<i>transmission control protocol</i>	protokol TCP
<i>transport layer</i>	transportni sloj
<i>uplink</i>	odlazni smjer
<i>virtual carrier sense</i>	virtualno mjerjenje signala nosioca
<i>wireless link</i>	bežični link
<i>wireless local area networks</i>	bežične lokalne mreže

Dodatak

Nakon instalacija programskog paketa OpNet, u korisničkom direktoriju (npr. *C:\Documents and Settings\Vedran Mikac*) stvore se dva poddirektorija: *op_models* i *op_admin*. U direktorij *op_admin* spremaju se datoteke koje OpNet interno koristi za vrijeme simuliranja. U taj direktorij, također se spremaju sigurnosne kopije svih snimljenih procesa i tu se nalaze datoteke u kojima su opisane greške koje su nastale prilikom prevođenja ili izvođenja simulacije (npr. *cc_err* datoteka). Direktorij *op_models* služi za spremanje svih datoteka vezanih uz neki projekt. U taj direktorij također se sprema izvorni kod svih vlastitih procesa. Kako bi se ponovila mjerena prezentirana u ovom radu, potrebno je izvršiti nekoliko krajnje jednostavnih koraka:

1. Zatvoriti OpNet 8.0.C paket
2. Iskopirati direktorij *op_models* sa CD-a predanog sa diplomskim radom u direktorij *op_models* na tvrdom disku
3. Sa direktorija *include* na predanom CD-u iskopirati datoteku *cip.h* na tvrdi disk u direktorij u kojem se nalaze sve *header* datoteke instalirane sa programskim paketom OpNet. Ako je OpNet instaliran na uobičajenom mjestu na C disku, tada je staza do tog direktorija *C:\Program Files\OPNET\8.0.C\models\std\include*.
4. Nakon uspješnog kopiranja, pokrenuti OpNet Modeler i otvoriti projekt diplomske i izvesti željene simulacije.

Prije izvođenja simulacije, potrebno je odabrati željeni scenarij pritiskom na kombinaciju tipaka Ctrl-1 za prvi scenarij Ctrl-2 za drugi itd. Simulacija se pokreće kombinacijom tipaka Ctrl-Shift-R ili iz izbornika Simulation. Svi parametri simulacije su već određeni kako bi se dobili isti rezultati kao i diplomskom radu, tako da nije potrebno ništa podešavati.