Matematički fakultet Univerziteta u Beogradu

Master rad

Alternativne arhitekture bežičnih sistema

Mentor:
Prof.dr Milan Tuba

Student:
Marko Brajović

Jul 2008, Beograd

Sadržaj:

Sažetak

Dizajn protokola zasnovan na narušavanju strogo definisane podjele komunikacije na sedam nezavisnih nivoa jeste Cross-Layer dizajn. Postoji više razloga za razmatranje alternativnih arhitektura za bežične sisteme, a jedan od njih su specifični problemi koji se pojavljuju razvojem različitih modela bežične komunikacije. Komunikacija između nivoa može da se organizuje na različite načine: uvođenje interfejsa između različitih nivoa, spajanje susjednih nivoa, dizajniranje parova različitih nivoa bez uvođenja novog interfejsa i vertikalno podešavanje duž nivoa.

Postoje različiti nacrti Cross-Layer dizajna, a jedan od njih je RBA arhitektura. Umjesto niza nivoa i njima pridruženih protokol hedera, RBA organizuje komunikaciju korišćenjem specifičnih funkcionalnih jedinica koje se nazivaju rolama. Glavna ideja na kojoj se zasniva RBA arhitektura je eksplicitno signaliziranje. Njihovo nepostojanje kod standardne OSI arhitekture je glavni razlog zbog kojeg middle-boxovi ne mogu biti uključeni u nju. Problemi kao što su prevođenje mrežnih adresa i bezbjednost komuniciranja i njihovo uključivanje u OSI model na jednostavan način se rešavaju prelaskom na RBA arhitekturu.

1. Uvod

Poslednjih godina došlo je do velikog razvoja bežičnih mreža. Mogućnost primjene takvih sistema je jako velika, a neke od oblasti primjene su mreže za povezivanje sistema i lokalne i regionalne mreže. Najznačajnija prednost bežičnih mreža jeste daleko jednostavniji pristup Internetu. Naravno, ovi sistemi imaju i određene mane, kao što su slabija propusna moć i niži stepen bezbjednosti. Bez obzira na sve, može se sa sigurnošću reći da će ovaj način povezivanja računara biti dominantan u budućnosti.

Dosadašnja primjena wireless sistema pokazala je da tradicionalna, slojevita organizacija mrežne arhitekture ne može u potpunosti da se prilagodi novom obliku umrežavanja. Iako je postojeći model služio više decenija i omogućio veliki razvoj Interneta i računarskih mreža uopšte, došao je trenutak da se razmotre i neki alternativni modeli koji bi riješili, sada već nagomilane probleme, koji su se pojavili primjenom savremenih načina umežavanja. Ovi problemi su jednim dijelom nastali ubrzanim razvojem "middle box"-ova, kao što su zaštitne barijere i NAT kutije. Takođe, razvojem novih oblika komunikacije, pojavljuju se neki karakteristični problemi koji se ne mogu riješiti u okvirima postojeće arhitekture.

Jedno od mogućih rešenja je da se postojeći model dodefiniše, na način što bi se omogućila komunikacija između nesusjednih nivoa ili redefinisale granice postojećih nivoa. Time se dolazi do novog dizajna protokola koji je zasnovan na narušavanju strogo definisane podjele komunikacije, i koji se naziva Cross-Layer dizajn. Razmjena informacija između nivoa može da se organizuje na više načina. Direktna komunikacija između nivoa primjenila bi se u slučaju kada je količina informacija koja se razmjenjuje mala. U suprotnom, uvela bi se dijeljena baza podataka.

Drugo rešenje je uvođenje potpuno nove arhitekture, što je, takođe, oblik Cross-Layer dizajna. Primjer takvog modela je Role-Based arhitektura. Umjesto niza nivoa i njima pridruženih protokola, komunikacija se organizuje uvođenjem novih funkcionalnih jedinica koje se nazivaju rolama. Svaka rola je opisana jedinstvenom oznakom i ima svoja unutrašnja stanja. U nekom konkretnom sistemu, role se pojavljuju u vidu različitog broja instanci koje se nazivaju čvorovima. Glavne karakterisitke RBA arhitekture su eksplicitno signaliziranje i organizacija metapodataka u vidu hipa. Uvođenje eksplicitnog signaliziranja omogućuje uključivanje middle-box-ova u novi model arhitekture.

2. Računarske mreže

Počeci razvoja računarskih mreža vezuju se za nastojanje Ministarstva odbrane SAD da obezbjedi komandno-kontrolni centar koji bi bio manje ranjiv od postojećeg. Do tada, sva vojna komunikacija se odvijala preko javnog telefonskog sistema. Njegova ranjivost se ogledala u tome što bi razaranjem nekoliko regionalnih centrala čitav sistem bio izdjeljen na veći broj nepovezanih ostrvaca. Agencija za napredne istraživačke projekte (engl. *Advanced Research Project Agency, ARPA*) dobila je zadatak da projektuje jedan takav sistem. Krajem 1969. god. puštena je u rad eksperimentalna mreža ARPANET. U prvo vrijeme, na nju su povezana četiri ustanove: UCLA, UCSB, SRI i Univerzitet Jute. Te ustanove su izabrane, između ostalog, i zato što su imale međusobno nekompatibilne računare. Mreža je ubrzo prekrila čitave Sjedinjene Države. Protokoli koji su prvobitno korišćeni pokazali su se neodgovarajućim za rad u više mreža, što je podstaklo razvoj modela TCP/IP i njegovih protokola. Tokom osamdesetih godina, na ARPANET su povezane dodatne mreže, naročito lokalne, i postajalo je sve teže pronaći računar na meži. Iz tog razloga, razvijen je **sistem imenovanja domena** (engl. *Domain Name System, DNS*), kojim je omogućeno prevođenje imena računara u njihove IP adrese.

Krajem sedamdesetih godina, od strane Američke nacionalne fondacije za nauku razvijana je još jedna mreža, nazvana **NSFNET.** Za razliku od ARPANETA-a, na čije priključivanje je bio neophodan ugovor sa Ministartstvom odbrane, NSFNET je omogućio slobodno povezivanje svih univerzitetskih istraživačkih grupa. To je ujedno bila i prva regionalna TCP/IP mreža. Uspjeh ove mreže bio je toliki da je ubrzo bila preopterećena.

Kada je TCP/IP postao jedini zvanični skup protokola mreže ARPANET, broj korisnika je sve više rastao. Povezivanje ARPANET-a i NSFNET-a označilo je početak eksponencijalnog razvoja. Ubrzo nakon toga, povezane su mnoge regionalne mreže, a uspostavljene su veze i sa mrežama u Kanadi, Evropi i na Pacifiku. Iz tog skupa mreža kasnije je nastao Internet.

Do početka devedesetih godina postojale su četiri glavne primjene Interneta:

- **Elektronska pošta** (engl. *e-mail*) - mogućnost primanja i slanja elektronskih poruka.
- **Diskusione grupe** (engl. *newsgroups*) - specijalizovani forumi za razmjenu poruka.
- **Daljinsko upravljanje** (engl. *remote login*) - prijavljivanje i rad na drugom računaru priključenom na Internet, korišćenjem programa kao što su: telnet, rlogin ili ssh.

- **Prenos datoteka** (engl. *file transfer*) - kopiranje datoteka sa jednog na drugi računar uz pomoć programa FTP.

Početkom devedesetih pojavljuje se nova oblast primjene **WWW** (World Wide Web). Time je omogućeno pravljanje informativnih strana sa tekstom, slikama, zvukom i videom. Takođe, uvedena je mogućnost pravljenja ugrađenih veza ka drugim stranama. Od tog trenutka, Internet počinju da koriste milioni neakademskih korisnika, što je suštinski izmjenilo njegovu dotadašnju svrhu. Vremenom su se pojavile mnoge specijalizovane vrste strana, kao što su razne vrtse tabela i mapa, katalozi, snimljeni radio i TV programi, lične strane kojima se pojedinci predstavljaju ostalim korisnicima i sl.

Iako ne postoji opšte prihvaćen sistem klasifikacije računarskih mreža, možemo izdvojiti dva najvažnija aspekta: tehnologija prenosa podataka i veličina. Što se tiče tehnologije prenosa, postoje dva najčešće korišćena tipa:

- *Veze za neusmjereno emitovanje.*
- *Veze od tačke do tačke.*

Mreže sa neusmjerenim (difuznim) emitovanjem (engl. *broadcast network***)** imaju jedinstveni komunikacioni kanal koji dijele svi umreženi računari. Bilo koji računar može da emituje kratke poruke, koje se ponekad nazivaju **paketi** (engl. *packets*), i njih primaju svi ostali umreženi računari. Unutar paketa postoji polje za adresu koje određuje primaoca. Računar obrađuje primljeni paket samo u slučaju da utvrdi da je namjenjen njemu. U suprotnom, paket se zanemaruje.

Mreže od "tačke do tačke" (engl. *point-to-point networks*) sadrže brojne veze između pojedinih parova računara. Da bi paket stigao do odredišne adrese, on prolazi kroz jedan ili više različitih računara. U većini slučajeva, postoji veći broj putanja različitih dužina. Kod ovog tipa mreža, jedna od bitnih stvari je pronalaženje optimalne putanje.

Drugi način klasifikacije mreža jeste prema njihovoj veličini. Razlikujemo lične, lokalne, gradske i regionalne mreže.

Mreže namjenjene jednoj osobi jesu **lične mreže** (engl. *personal area network*)**.** Razdaljina između sistema je oko 1m. Bežično povezivanje računara sa mišem, tastaturom ili štampačem jesu primjeri ovog tipa.

Lokalne mreže (engl. *Local area network, LAN*) jesu privatne mreže unutar jedne zgrade ili nekog manjeg područja, raspona do 2 km. Najčešće se uvode u firmama da bi se obezbjedila razmjena informacija i zajedničko korišćenje resursa (kao što je štampač). Postoje različite topologije lokalnih mreža. Topologija **magistrale** (engl. *bus*) povezuje računare linearnim kablom. U svakom trenutku, najviše jedan računar je u mogućnosti da emituje pakete. Da

bi se razriješile situacije u kojima istovremeno više računara želi da šalje poruke, uvode se različiti mehanizmi odlučivanja. Jedan od njih je sistem IEEE 802.3 ili **Ethernet**. Kod ovog mehanizma, računari mogu da emituju pakete kad god požele. U slučaju da dođe do sudaranja dva paketa, računari pauziraju slučajno izabrani period vremena, a zatim pokušavaju ponovno emitovanje. Kod topologije **prstena** (engl. ring), svaki bit kruži nezavisno od ostatka paketa kome pripada. I kod ove topologije postoje različiti mehanizmi odlučivanja, kao što je IBM-ova token ring mreža IEEE 802.5.

Gradske mreže (engl. Metropolitan Area Network, MAN) jesu mreže na području jednog grada. Najpoznatiji primjer je sistem kablovske televizije. Uvođenjem dvosmjerne Internet usluge, ovaj specijalizovani TV sistem je prerastao u pravu gradsku mrežu. Razvojem bežičnog pristupa Internetu nastala je i druga vrtsa gradskih mreža koja je standardizovana pod oznakom IEEE 802.16.

Reginalne mreže (engl. Wide Area Network, WAN) pokrivaju područje jedne države ili cijeli kontinent. Umreženi računari (engl. *hosts*) su povezani **podmrežom** (engl. *subnet*). Razdvajanje komunikacionog od aplikativnog aspekta, tj. podmreže od računara, omogućava jednostavnije projektovanje mreže. Dvije različite komponente čine podmrežu: linije prenosa (engl. *transmission lines*) i prekidački elementi (engl. *switching elements*). Linije prenosa služe za prenošenje bitova od jednog ka drugom računaru. Prekidački elementi su specijalizovani računari koji spajaju tri ili više linija prenosa. Nazivaju se još i usmjerivači ili ruteri (engl. *routers*). Kada podaci stignu do rutera, on odlučuje kojom linijom prenosa će nastaviti put ka odredišnom računaru. U situaciji kada dva rutera žele da komuniciraju, a pri tome nisu direktno povezani, koriste se međuruteri. Svaki međuruter prima cijeli paket, čuva ga dok se ne oslobodi odgovarajuća linija i dalje ga prosleđuje. Ovakav princip organizovanja podmreža naziva se "**čuvaj i proslijedi**" (engl. *store-and-forward*).

2.1 Bežične mreže

Bežični mreže se mogu podijeliti u tri osnovne kategorije:

- *Mreže za povezivanje sistema.*
- *Lokalne bežične mreže.*
- *Regionalne bežične mreže.*

Pod povezivanjem sistema podrazumjeva se povezivanje komponenata računara korišćenjem radio-talasa kratkog dometa. Na primjer, sistem **Bluetooth** omogućava priključivanje monitora, tastature, miša, digitalne kamere i drugih

uređaja, tako što se dovedu u domet emitovanja mreže**.** Jedan od načina koji se koristi za povezivanje sistema je obrazac nadređenog i podređenog uređaja. Nadređeni uređaj bila bi sistemska jedinica, dok su miš, tastatura i td. podređeni uređaji. Sistemska jedinica saopštava koje adrese treba da koriste ovi uređaji, koje frekvenije da koriste itd.

Kod lokalnih bežičnih mreža, svaki računar ima radio-modem i antenu pomoću kojih može da komunicira sa drugim sistemima. Standard koji opisuje ovu kategoriju mreža nosi naziv IEEE 802.11 ili **WiFi**. Njime su predviđena dva režima rada:

- *U prisustvu bazne stanice.*
- *U odsustvu bazne stanice.*

U prvom slučaju, komunikacija se odvija preko bazne stanice koja se naziva još i **pristupna tačka** (engl. *access-point*). U drugom slučaju, primjenjuje se **ad hoc umrežavanje** (engl. *ad hoc networking*), tj. uspostavljanje direktne veze dva računara. U vrijeme kada je razvijan standard IEEE 802.11, Ethernet je bio dominantan na polju lokalnih mreža, pa je i novi standard za bežične sisteme napravljen kompatibilnim sa postojećim. Time je omogućeno da se IP paket preko bežične lokalne mreže šalje na isti način kao i preko Ethernet-a.

Radiotalasna mreža, koja se koristi za mobilnu telefoniju, jeste primjer regionalne bežične mreže niske propusne moći. Prva generacija je bila analogna, a druga digitalna, ali su obje prenosile samo govor. Treća generacija koristi se i za prenos govora i za prenos podataka. Postoje takođe i bežične regionalne mreže visoke propusne moći, kojima se omogućava povezivanje na Internet, pri čemu se zaobilazi sistem telefonije.

3. Referentni modeli

Da bi projektovanje nekog kompleksnog sistema bilo jednostavnije, u računarskim naukama koristi se koncept **skrivanja informacija** (naziva se još i **apstraktni tipovi podataka, kapsuliranje, objektno orijentisano programiranje**). Ideja je da određena komponenta obezbjedi usluge korisnicima i da, pri tome, sakrije informacije o primjenjenim algoritmima i unutrašnjim stanjima. Iz tog razloga, mreže se većinom organizuju kao skup **slojeva** (engl. *layers*) ili **nivoa** (engl. *levels*). Sloj x na jednom računaru komunicira sa slojem x na drugom računaru koristeći **protokole** (engl. *protocols*) sloja x. Protokol se može shvatiti kao dogovor dvije strane o tome kako će da teče njihova komunikacija. U praksi, podaci se ne razmjenjuju direktno između dva sloja, već se, zajedno sa dodatnim informacijama, prosleđuju sloju neposredno ispod datog, sve dok ne stignu do **fizičkog medijuma,** kroz koji se odvija stvarna komunikacija. Svaki od slojeva obezbjeđuje niz operacija i usluga sloju koji se nalazi neposredno iznad njega. Taj skup operacija i usluga naziva se **interfejs** (engl. *interface*).

Prilikom projektovanja nekog sistema, od izuzetne je važnosti da se dobro definišu interfejsi između slojeva. Time je, pored skrivanja podataka, omogućena i kasnija zamjena nekog od slojeva unapređenom verzijom. Ako nije promjenjen skup interfejsa koje obezbjeđuje novi sloj, cijeli sistem nastavlja da funkcioniše bez problema.

Skup nivoa i interfejsa zajedno čine **arhitekturu mreže.** Važno je naglasiti da ni detalji realizacije ni specifikacija interfejsa nisu dio arhitekture jer se ne vide spolja.

Na razvoj nauke o računarskim mrežama veliki uticaj imala su sledeća dva modela: ISO OSI i TCP/IP. ISO OSI arhitektura je sveobuhvatna i još uvjek važeća, ali protokoli vezani za tu arhitekturu nemaju veću primjenu. Sa druge strane, TCP/IP model se rijetko koristi, ali su njegovi protokoli široko prihvaćeni.

3.1 Referentni model OSI

Referentni sistem ISO OSI (engl. *International Standards Organization - Open System Interconnection*) zasniva se na predlogu Međunarodne organizacije za standardizaciju. Sastoji se od sedam nivoa. ISO organizacija je predvidjela i standarde za svaki pojedinačni sloj (usluge i protokole), ali oni nisu dio modela.

Iz tog razloga, OSI model ne predstavlja arhitekturu mreže.

Prilikom određivanja broja nivoa, korišćeni su sledeći principi:

1. Kad god je neophodna nova apstrakcija, napraviti novi sloj.
2. Jasno definisati funkciju svakog sloja.
3. Prilikom odabira funkcije nekog sloja, imati u vidu definisanje međunarodno standardizovanih protokola.
4. Granice između slojeva definisati tako da se minimizuje protok informacija između slojeva.
5. Broj slojeva treba da bude dovoljno mali da arhitektura ne bude previše složena, a ipak dovoljno veliki da se funkcije, čije se namjene jasno razlikuju, ne smještaju u isti sloj.

Fizički sloj (engl. *physical layer*) je zadužen za prijem i prenos niza bitova duž komunikacionog kanala. Prilikom projektovanja ovog sloja, uglavnom se rešavaju mehanički i električni zahtjevi, kako bi se obezbjedilo da kada jedna strana pošalje bit 1, druga strana takođe primi bit 1. Na ovom nivou se odlučuje o tome koliki napon treba da predstavlja jedinicu, a koliki nulu, koliko nanosekundi treba da traje jedan bit, kako se na početku uspostavlja veza, a kako prekida, da li se prenos može istovremeno obavljati u oba smjera i sl.

U okviru **sloja veze podataka** (engl. *data link layer*), ulazni podaci se dijele na **okvire podataka** (engl. *data frames*) dužine od nekoliko stotina do nekoliko hiljada bitova, i šalju jedan za drugim. Primalac potvrđuje ispravan prijem svakog okvira slanjem **okvira za potvrdu** (engl. *acknowledgement frame*). Na ovom nivou neophodno je uvođenje mehanizma za regulisnje saobraćaja, kako ne bi došlo do neusaglašenosti između brzine slanja i brzine primanja podataka. Dodatni problem, koji se javlja kod mreža sa difuznim emitovanjem poruka, jeste kako upravljati pristupom zajedničkom kanalu. Iz tog razloga, uvodi se podsloj za upravljanje pristupom medijumima (engl. *medium access control sublayer, MAC*).

Za upravljanje radom podmreže koristi se **mrežni sloj** (engl. *network layer*). Ključna stvar, koju je neophodno odrediti prilikom njegovog projektovanja, jeste kako se paketi upućuju od izvora ka odredištu. Putanje se mogu određivati statički, korišćenjem tabela koje su ugrađene u mrežu, ili dinamički za svaki paket, u zavisnosti od trenutnog opterećenja. Važno je naglasiti da prilikom prolaska paketa sa jedne na drugu mrežu mogu da nastanu razni problemi. Na primjer, može se desiti da se razlikuju protokoli, da je drugačiji način adresiranja, da druga mreža ne prihvati paket zbog njegove veličine i sl. Osnovni zadatak mrežnog sloja jeste da prevaziđe navedene probleme i omogući povezivanje heterogenih mreža.

Transportni sloj (engl. *transport layer*) je zadužen za prihvatanje podataka od viših slojeva, eventualno razvrstavanje u manje grupe i prosleđivanje mrežnom sloju. Na ovom nivou se, takođe, definišu usluge namjenjene sloju sesije, a to su, u krajnjoj liniji, korisnici mreže. Primjeri transportnih usluga su kanal "od tačke do tačke" sa ispravljanjem grešaka, kojim se isporučuju poruke redom kojim su poslate, zatim prenošenje izolovanih poruka bez garancije redosleda pristizanja, kao i difuzno slanje poruka na više odredišta. Transportni sloj i slojevi koji se nalaze iznad njega obavljaju komunikaciju sa odgovarajućim slojevima odredišnog računara (doduše, virtuelnu, jer stvarna komunikacija ide jedino preko fizičkog medijuma). Za razliku od njih, niži slojevi su lančano povezani sa najbližim susjedima.

Uspostavljanje **sesije (**engl. *session***)** između korisnika na različitim računarima omogućeno je **nivoom sesije** (engl. *session layer*). Na ovom nivou se vodi računa o **upravljanju dijalogom** (engl. *dialog control*), tj. na koga je red da šalje poruke, **radu sa tokenima** (engl. *token management*), tj. sprečavanje istovremenog pokretanja iste kritične operacije, i **sinhronizovanju** (engl. *synchronization*), odnosno omogućavanju nastavljanja prenosa od tačke prekida u slučaju pada sistema.

Sintaksom i semantikom prenjetih informacija bavi se **sloj prezentacije** (engl. *presenation layer*). Računari predstavljaju podatke na različite načine. Da bi se omogućila njihova komunikacija, definišu se apstraktne strukture podataka, koje se standardno kodiraju u cilju prenosa. Na ovom nivou se obavlja njihova obrada, kao i definisanje struktura podataka višeg nivoa, koje se razmjenjuju sa aplikativnim slojem.

Sloj aplikacija (engl. *application layer*) sadrži veći broj protokola namjenjenih korisnicima. Jedan od njih je i **protokol za prenos hiper-teksta** (engl. *Hypertext Transfer Protocol, HTTP*), koji čini osnovu World Wide Weba. Postoje, takođe, i drugi protokoli koji definišu prenos elektronkse pošte, datoteka, poruka diskusionih grupa i sl.

3.2 Referentni model TCP/IP

Na ovom referentnom modelu je zasnovan ARPANET, preteča svih regionalnih mreža, a takođe i Internet, globalna svjetska mreža. Ime modela, TCP/IP (engl. *TCP/IP Reference Model*), nastalo je kombinovanjem imena njegova dva osnovna protokola. Model se sastoji od 4 nivoa.

Međumrežni sloj (engl. *internet layer*) je zadužen za isporuku paketa do njihovog odredišta. Pri tome, mora se voditi računa o usmjeravanju i izbjegavanju zagušenja. Paketi na odredište mogu da stignu u drugačijem

redosledu nego što su poslati. Zadatak viših nivoa je da izvrše kasnije spajanje paketa u odgovarajućem redosledu. Ovakav način slanja podataka naziva se **princip komutiranja paketa.** Na ovom nivou se definiše zvanični format paketa i tzv. **Internet protokol** (engl. *Internet protocol, IP*). Po svojoj funkcionalnosti, međumrežni sloj TCP/IP modela može se uporediti sa mrežnim OSI slojem.

Komunikacija između dva ravnopravna procesa na različitim računarima obavlja se preko **transportnog sloja** (engl. *transport layer*). Po svojoj funkciji, može se uporediti i sa transportnim slojem OSI modela. Na ovom nivou su definisana dva različita protokola. Prvi od njih je **protokol za upravljanje prenosom** (engl. *Transmission Control Protocol, TCP*), i on predstavlja pouzdan protokol sa uspostavljanjem direktne veze. Polazni podaci se dijele na niz poruka i svaka od njih se prosleđuje međumrežnom sloju. Sa druge strane, TCP primalac uređuje primljene poruke i obrazuje polazni niz bajtova. Ovaj protokol takođe upravlja i tokom podataka, tako da nije moguća situacija u kojoj brzi pošiljalac zatrpa sporog primaoca većim brojem poruka nego što ovaj može da obradi.

Drugi protokol ovog nivoa je **protokol za korisničke dijagrame** (engl. *User Datagram Protocol, UDP*). Za razliku od TCP protokola, UDP predstavlja nepouzdan protokol bez uspostavljanja direktne veze. Namjenjen je aplikacijama koje same uređuju svoje pakete i upravljaju tokom podataka. Vrlo često se koristi za jednostavne upite klijentsko-serverskog tipa (zahtjev-odgovor), kao i u slučajevima kada hitnost isporuke ima veću važnost nego tačnost.

U modelu TCP/IP nema slojeva sesije i prezentacije, jer je iskustvo pokazalo da su većini aplikacija kod OSI modela od male koristi.

Protokoli višeg nivoa su smješteni u **sloj aplikacija** (engl. *application layer*)**.** U početku**,** to su bili protokoli za virutelni terminal (TELNET), za prenos podataka (FTP) i za elektronsku poštu (SMTP). Vremenom su dodavani novi protokoli: sistem imenovanja domena (DNS), protokol za prenošenje poruka USENET-ovih diskusionih grupa, protokol za preuzimanje WWW strana (HTTP) i mnogi drugi.

Najniži sloj je **sloj za povezivanje računara sa mrežom.** TCP/IP modelom se ne precizira kako se tu odvijaju stvari, osim da se računar povezuje na mrežu i šalje IP pakete koristeći neki od protokola. Međutim, sam protokol nije definisan, i on se razlikuje od računara do računara i od jedne do druge mreže.

4. Definicija Cross-Layer dizajna i osnovni tipovi organizacije

Kao sto je već pomenuto, jedan od preduslova koji su omogućili ekspanziju Interneta je njegova zasnovanost na OSI modelu arhitekture. OSI model je apstraktni opis dizajna protokola komunikacionih i računarskih mreža, predstavljen u obliku sedam slojeva. Njime je opisana podjela cjelokupne komunikacije na datih sedam slojeva i definisana je hijerarhija servisa koji su obezbjeđeni na svakom pojedinačnom nivou. Direktna komunikacija između nesusjednih nivoa nije moguća, dok je komunikacija između susjednih ograničena na mali broj procedura. To ima za posljedicu da protokoli višeg nivoa koriste isključivo servise nižih protokola, i takođe, da su protokoli na svakom pojedinačnom nivou razvijeni nezavisno od protokola na ostalim nivoima.

Definicija1: Dizajn protokola zasnovan na narušavanju strogo definisane podjele komunikacije na sedam nezavisnih nivoa jeste Cross-Layer dizajn.

Komentar1: Primjeri napuštanja OSI modela komunikacije uključuju kreiranje novih interfejsa između nivoa, redefinisanje granica samih nivoa, dizajniranje protokola na pojedinačnim nivoima u zavisnosti od toga kako su razvijeni protokoli na nekom drugom nivou, omogućavanje podešavanja parametara duž većeg broja nivoa i sl.

Komentar2: Cross-Layer dizajn je definisan kako metodologija dizajniranja protokola. Međutim, protokol definisan korišćenjem takve metodologije se takođe naziva Cross-layer design.

Da bi navedena definicija bila jasnija, zamislimo model koji se sastoji od tri nivoa, označena sa L1, L2 i L3, pri cemu je L1 najniži a L3 najviši nivo. U slučaju da ovaj hipotetički model zadovoljava uslove koje propisuje OSI model komunikacije, tada ne bi postojala direktna komunikacija između nivoa L1 i nivoa L3. Međutim, zamislimo da je nivo L3 dizajniran tako da mu je u vremenu izvršavanja potrebno proslijediti neki parametar nivoa L1. To za posledicu ima uvođenje novog interfejsa, a samim tim i napuštanje OSI modela. Drugi primjer bio bi povezivanje nivoa L1 i L2 u jedan nivo, tzv. "super-nivo". Dizajniranje nivoa L3, pri čemu se imaju u vidu procesi na nivou L1, takođe dovodi do napuštanja nezavisnog dizajniranja protokola na različitim nivoima. Sve su ovo različiti primjeri uvođenja Cross-Layer dizajna.

4.1 Glavni motivi za uvođenje Cross-Layer dizajna

Tri su osnovna razloga za napuštanje strogo definisanog načina komunikacije između nivoa: jedinstveni problemi koji se pojavljuju razvojem različitih modela bežične komunikacije, mogućnost oportunističke komunikacije kod bežičnih veza i novi modeli komunikacije.

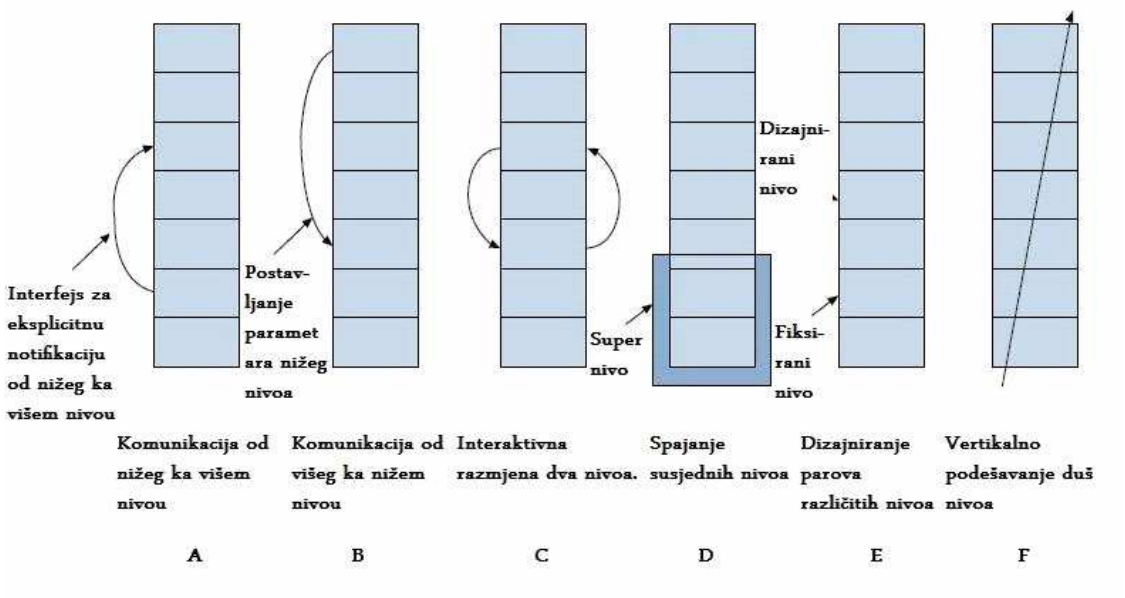
Primjer problema koji se pojavljuje uvođenjem bežične komunikacije, a koji se ne može riješiti klasičnim mehanizmima OSI modela je zagušenje u mrežnom saobraćaju. Zagušenje se signalizira od strane rutera (Network nivo), a kontrola brzine slanja paketa je na transportnom nivou. Jedno od mogućih rešenja je eksplicitno signaliziranje, što nas dovodi do Cross-Layer dizajna. Linkovi kod kojih se kvalitet tokom vremena mijenja, omogućuju oportunističko korišćenje veza, gdje se parametri prenosa prilagođavaju promjenama u kvalitetu kanala [1]. Bežični medijumi omogućavaju nove modalitete komunikacije, omogućene samom prirodom bežičnih medijuma, kojima klasične arhitekture ne mogu da se prilagode. Na primjer, fizički nivo se može osposobiti za primanje višestrukih paketa u isto vrijeme [2]. Takođe, neki od elemenata savremenih mreža, kao što su NAT kutije, zaštitne barijere, IP bezbjednost, ne uklapaju se u klasičnu OSI arhitekturu (odjeljak 5.6 i 5.7).

4.2 Načini organizovanja komunikacije između različitih nivoa

U zavisnosti od konkretne primjene, komunikacija između nivoa može da se organizuje na različite načine. Važno je naglasiti da se navedeni načini međusobno ne isključuju, odnosno da je moguća njihova koegzistencija u nekom sistemu. Na slici 1 su prikazani neki od oblika komunikacije, a zatim je dat i njihov opis. Tokom vremena, ako se bude ukazala potreba, ovom skupu će se dodati još neki modaliteti komunikacije.

[1] Z.Ji et al., "Exploiting Medium Access Diversity in Rate Adaptive Wireless LANs", *Proc. ACM Annual Int'l. Symp. Mobile Comp. and Net.*, Philadelphia, PA, Oct. 2004

[2] L. Tong, V. Naware, and P. Venkitasubramaniam, "Signal Processing in Random Access", *IEEE Sig. Proc., vol. 21, no. 5*, Sept. 2004, pp. 20-39



4.2.1 Kreiranje novih interfejsa

Nekoliko različitih Cross-Layer dizajna zahtjevaju uvođenje novih interfejsa, koji se koriste za razmjenu informacija u vremenu korišćenja. Pravac komunikacije moze da bude od nižeg nivoa ka višem, od višeg nivoa ka nižem i interaktivna razmjena informacija između dva nivoa (slika 1).

Na primjer, ako jedna TCP putanja (eng. *path*) sadrži wireless link, greška koja se pojavi na tom linku može da dovede do situacije u kojoj TCP pošiljalac to vidi kao zagušenje u mreži, što za posledicu ima pogoršanje performansi. Kreiranje interfejsa od rutera ka network nivou omogućilo bi slanje informacije da li je došlo do zagušenja u mreži, čime bi TCP pošiljalac mogao da preduprijedi gore opisanu situaciju.

Uvođenje interfejsa od višeg ka nižim nivoima omogućilo bi tzv. *delay-sensitive* aplikacijama da proslijede informaciju do link nivoa o njihovim specifičnim zahtjevima, čime bi bilo omogućeno da link nivo obrađuje takve pakete sa većim prioritetom. Jedan od načina kako bi mogao da se posmatra tok informacija od višeg ka nižim nivoima, odnosno od nižeg ka višim, jeste da se oni tretiraju kao sistem obavještenja i znakova (engl. *notifications and hints*), kao što je predloženo u [3]. Informacije ka višim nivoima imaju za cilj obavještavanje o trenutnom stanju mrežnog sistema, dok informacije ka nižim nivoima imaju za cilj signaliziranje kako da se obrade podaci potrebni određenoj aplikaciji.

[3] L.Larzon, U.Bodin, and O.Schelen, "Hints and Notifications", *Proc. IEEE Wireless Commun. and Net. Conf.*, Orlando, FL, Mar. 2002.

Dva nivoa, koja obavljaju različite zadatke, mogu međusobno da sarađuju u vremenu izvršavanja. To se manifestuje uspostavljanjem iterativne petlje između dva nivoa, pri čemu se informacije razmjenjuju u oba smjera. Kao primjer, može se uzeti NDMA (engl. *Network-assisted diversity multiple acces*, opisano u [4]), gdje fizički nivo (PHY) i MAC nivo uspostavljaju saradnju u razrešavanju kolizija nastalih u wireless LAN sistemu. Nakon što se detektuje kolizija, bazna stanica prvo procjeni broj korisnika koji su došli u koliziju, zatim zahtjeva ponovno slanje određenog broja paketa od tih korisnika. Nakon toga, PHY omogućuje baznoj stanici da razdvoji njihove signale, čime se razrešava kolizija.

Spajanje susjednih nivoa - Još jedan način kako bi mogao da se primjeni Cross-Layer dizajn je da se dva ili više različitih nivoa dizajniraju tako da zajedno čine jedan super-nivo, pri čemu su servisi tog super-nivoa unija servisa pojedinačnih nivoa koji ga čine. Ovakav pristup ne zahtjeva kreiranje novih interfejsa. Posmatrano sa strane arhitekture, takav jedan super-nivo bi ostvarivao komunikaciju sa ostatkom steka koristeći vec postojeće interfejse originalne arhitekture. Kao primjer može se navesti već pomenuta ideja NDMA, iz razloga što takvo dizajniranje PHY i MAC ima tendenciju rušenja granica između ta dva nivoa.

Dizajniranje parova različitih nivoa, bez uvođenja novog interfejsa - Ovo je još jedna kategorija Cross-Layer dizajna, koja uključuje dizajniranje grupe dva ili više nivoa bez kreiranja dodatnih interfejsa za razmjenu informacija u vremenu izvršavanja. Sa tačke gledišta arhitekture, cijena koja se ovdje plaća je nemogućnost promjene jednog nivoa bez ostvarivanja odgovarajućih promjena na nekom drugom nivou. Kao primjer, zamislimo wireless LAN kod kojeg je PHY nivo dizajniran tako da može da primi više od jednog paketa istovremeno. Takva jedna mogućnost PHY nivoa neizbježno dovodi do promjene uloge MAC nivoa, pa smim tim i on mora da se redizajnira.

Vertikalno podešavanje duž nivoa - Poslednja kategorija Cross-Layer dizajna opisana u literaturi je vertikalno podešavanje duž nivoa. Motivacija za uvođenje ove kategorije je očigledna, jer performanse sistema, posmatrane sa tačke aplikativnog nivoa, jesu funkcija parametara svih nižih nivoa. Samim tim, omogućavanje podešavanja parametara duž većeg broja nivoa dovodi do ostvarivanja boljih performansi sistema, nego u slučaju kada se vrši pojedinačno podešavanje parametara na različitim nivoima.

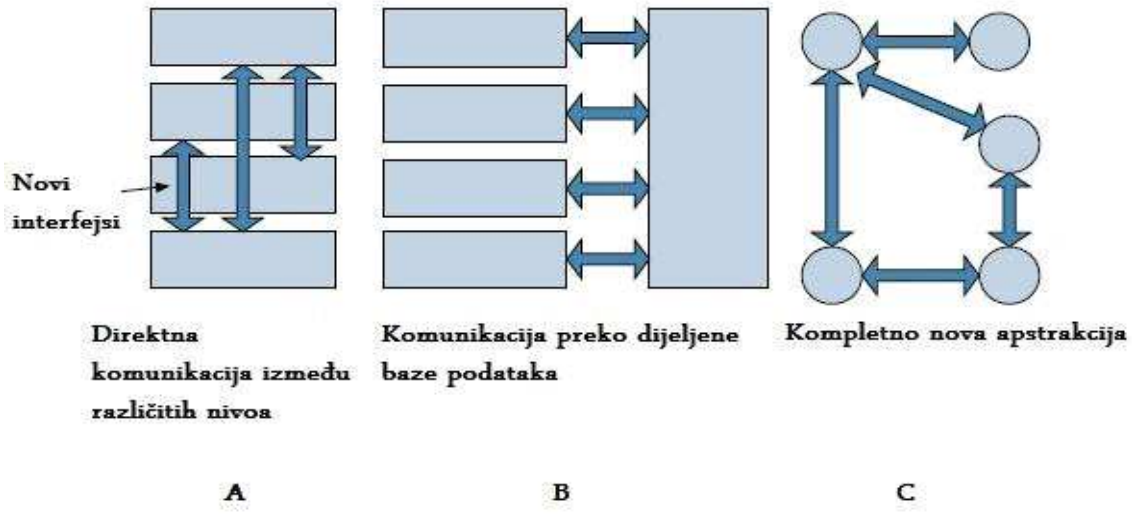
Vertikalno podešavanje duž nivoa može da se ostvari na statički i na dinamički način. Statički način podrazumjeva podešavanje parametara u vremenu dizajniranja, pri čemu se vodi računa o optimizaciji određenih metrika. Time ne dolazi do značajnijih komplikacija u implementaciji jer su parametri jednom podešeni u vremenu dizajniranja i nakon toga se ne mijenjaju. Sa druge strane, dinamički način podrazumjeva podešavanje u vremenu izvršavanja, što se može

[4] G.Dimic, N.D. Sidiropoulos, and R.Zhang, "*Medium Access Control - Physical Cross-Layer Design*", IEEE Sig. Proc., vol. 21, no. 5, Sept. 2004, pp. 40-50

posmatrati kao prilagodljivi protokol stek koji reaguje na sveukupno stanje sistema. To takođe zahtjeva uvođenje mehanizama za primanje i podešavanje vrijednosti parametara na različitim nivoima, kao i striktno definisanje samog procesa, kako bi u svakom trenutku imali precizan uvid u trenutno stanje na ostalim nivoima sistema.

4.3 Načini razmjene informacija između različitih nivoa

Razmjena informacija između različitih nivoa kod Cross-Layer dizajna može se ostvariti na više načina. Neki od njih su prikazani na slici 2. Koji od navedenih oblika će se primjeniti zavisi od konkretne primjene. Direktna komunikacija i komunikacija preko dijeljene baze mogle bi se primjeniti u slučaju kada se OSI model narušava uvođenjem novih interfejsa u postojeću arhitekturu, dok je treći oblik razmjene primjenljiv u slučaju uvođenja kompletno nove arhitekture.



Slika 2

Direktna komunikacija među nivoima - Najjednostavniji način razmjene informacija je direktna komunikacija među nivoima. Ovaj način je primjenljiv u slučaju da postoji potreba za komunikacijom u vremenu izvršavanja (npr. ako su kreirani novi interfejsi ili ako je omogućeno vertikalno podešavanje nivoa). U praksi, to znači učiniti promjenljive jednog nivoa vidljivim od strane drugih nivoa, u vremenu izvršavanja. To je u direktnoj suprotnosti sa OSI arhitekturom, gdje svaki nivo upravlja svojim promjenljivim i one su nedostupne ostalim nivoima sistema.

Postoje različiti načini kako se može ostvariti direktna komunikacija. Na primjer, protokol hederi mogu biti korišćeni za protok informacija između različitih nivoa. Uporedna studija različith nacrta direktne komunikacije je predstavljena u [5], gdje se, takođe, predlaže i jedan dodatni nacrt direktne komunikacije, tzv. CLASS model (engl. *cross-layer signaling shortcuts*). CLASS omogućava direktnu komunikaciju bilo koja dva nivoa sistema.

Direktna komunikacija je primjenljiva u slučaju kada je količina informacija koja se razmjenjuje između različitih nivoa sistema mala, a sam sistem je originalno dizajniran u skladu sa OSI modelom arhitekture. U tom slučaju, implementira se par "rupa" u steku kako bi se omogućila direktna komunikacija.

Dijeljena baza podataka - Sledeći način razmjene informacija između nivoa propisuje korišćenje dijeljene baze podataka, što je ilustrovano na slici 2. Takva jedna baza može se posmatrati kao novi nivo sistema, koji omogućuje funkcije smještanja i prihvatanja informacija, i koji je dostupan ostalim nivoima. Primjetimo da je opisani sistem razmjene informacija prilagođen vertikalnom podešavanju parametara, jer bi ono moglo da se ostvari implementacijom programa za optimizaciju, koji bi komunicirao sa različitim nivoima preko dijeljene baze. Takođe, uvođenje novih interfejsa može da se ostvari korišćenjem ovog modela razmjene informacija. Glavni izazov ovdje bio bi dizajn interakcije između različitih nivoa sistema sa jedne strane, i dijeljene baze sa druge strane.

Kompletno nova arhitektura - Treći način razmjene informacija je kompletno nova arhitektura, što je šematski prikazanao na slici 2. Primjer takve arhitekture je Role-Based arhitektura, koja je detaljnije opisana u petoj glavi. Novi modeli organizacije protokola imali bi veliku prednost iz razloga što bi omogućili daleko veću interakciju između različitih grupa protokola od kojih je izgrađen sistem. Pored toga, potencijalno bi bila omogućena velika fleksibilnost, kako u vremenu dizajniranja, tako i u vremenu izvršavanja. Sam način organizacije protokola bio bi suštinski promjenjen, pa bi to zahtjevalo potpuno novu implementaciju sistema i nivoa od kojih je izgrađen.

[5] Q.Wang and M. A. Abu-Rgheff, "Cross-Layer Signalling for Next-Generation Wireless Systems", *Proc. IEEE Wireless Commun. and Net. Conf., New Orleans*, LA, Mar. 2003.

4.4 Cross-Layer dizajn - otvorena pitanja

Nakon što je uvedena definicija Cross-Layer dizajna i prikazani osnovni nacrti organizacije sistema, neophodno je navesti pitanja koja do sada nisu u potpunosti razjašnjena u literaturi i čije bi rešavanje uticalo na krajnji izgled arhitekture bežičnih sistema. Neka od tih pitanja su:

- Kako da se omogući koegzistencija različitih nacrta cross-layer dizajna?
- Da li bi neka od postojećih ideja organizacije bežičnih sistema mogla negativno uticati na razvoj u budućnosti?
- Koji dizajn Cross-Layer modela bi imao najveći uticaj na performanse sistema, kako bi se u budućnosti što više fokusirali na njegov razvoj?
- Za koju mrežu i za koje uslove okoline bi trebalo primjeniti konkretni nacrt Cross-Layer dizajna?
- Da li je moguće standardizovati mehanizme koji će se koristiti za razmjenu informacija između različitih nivoa komunikacije?
- Koja bi bila uloga fizičkog nivoa kod bežičnih sistema?
- Da li je tradicionalno shvatanje mreže kao kolekcije point-to-point linkova odgovarajuće za bežične sisteme?

Iako danas u literaturi postoji veći broj različitih nacrta Cross-Layer arhitekture, nije jasno koji od njih je najznačajniji. Da bi moglo da se odgovori na ovo pitanje, neophodno je napraviti njihovu uporednu analizu, na način što bi se uzeli u obzir i kompleksnost implementacije i poboljšanje performansi sistema. Generalno gledano, sledeći zaključci se mogu izvući iz postojeće literature koja se bavi ovim pitanjima: cross-layer dizajn je neophodno primjeniti između network i MAC nivoa, kako bi se omogućila njihova uzajamna interakcija (odeljak 4.2.1); neophodno je uvesti novi interfejs ka transportnom nivou, jer bi se time poboljšale end-to-end performanse sistema; u slučaju cross-layer dizajna za mobilne uređaje, uključiti specifične zahtjeve korisnika u protokol stek (npr, bilo bi korisno omogućiti korisniku da dinamički promjeni prioritet aplikacija koje se izvršavaju).

Rečeno je da je koegzistenija različitih nacrta jedno od važnih pitanja. Pretpostavimo da MAC nivo reguliše određene parametre u cilju prilagođavanja promjenama nastalim u toku izvršavanja. Da li dodavanje jedne ovakve mogućnosti link nivou može da utiče na dalja unapređivanja sistema? Konkretno, da li bi neko kasnije dodavanje mogućnosti kontrole parametara link nivoa od strane, npr. transportnog nivoa, moglo funkcionisati bez problema? Navedimo još jedan primjer. Neka su fizički i link nivo optimizovani u cilju unapređenja nekog aspekta performanse sistema. Da li će nakon primjene ovog konkretnog cross-layer uparivanja moći da se primjeni neki drugi nacrt, koji je zasnovan na različitom cross-layer uparivanju, ili neki koji podrazumjeva da ne postoje već primjenjena uparivanja? Odgovor na ova i slična pitanja jednim dijelom bi odgovorio i na još jedno važno pitanje - da li neki od postojećih nacrta može

ugušiti dalji razvoj bežičnih sistema u budućnosti? Arhitekture čija je dobra utemeljenost i stabilnost omogućila ogroman razvoj različitih oblasti računarstva su, na primjer, Fon-Nojmanova arhitektura i OSI arhitektura. Ovi primjeri nam ukazuju koliko je bitno naći odgovore na navedena pitanja, kako ne bismo došli u situaciju da neka od primjenjenih rešenja uguše dalji napredak.

Jedna od važnih karakteristika OSI modela jeste razdvajanje sistema na niz nezavisnih nivoa, sa tačno određenim granicama između njih i definisanim mehanizmima komunikacije. U situaciji kada narušavamo date principe podjele, postavlja se pitanje kako bi trebalo da izgledaju granice između nivoa? Da li trebamo da se pridržavamo već utemeljenog principa podjele i da u okviru njega definišemo nove mehanizme komunikacije među nivoima? Ili da definišemo potpuno novi apstraktni model, kao što je Role-Based arhitektura? Da li je kombinacija ta dva pristupa prihvatljivo rešenje? Organizacija modula sistema (u vidu nivoa ili na neki drugi način) i interfejsi između njih određuju koliko efektivno se mogu razmjenjivati informacije. To u krajnjoj liniji određuje i efikasnost samog nacrta Cross-Layer dizajna.

Kod klasične "žične" mreže, uloga fizičkog nivoa je dosta sužena: slanje i primanje paketa na zahtjev nekog od viših nivoa sistema. Međutim, primjena naprednih metoda obrade signala omogućila bi veću ulogu fizičkom nivou. Takođe, promjene na višim nivoima sistema dovele bi do potrebe mijenjanja uloge i fizičkog nivoa.

Sve do pojave bežičnih sistema, definisanje mreže kao kolekcije point-to-point linkova odgovaralo je samoj prirodi računarskih mreža. Iz razloga sto se bežični medijumi zasnivaju na emitovanju signala, takva definicija je danas teško primjenjiva. Poslednjih godina rađena su istraživanja emisione prirode bežičnih medijuma u cilju korišćenja inovativnih komunikacionih šema za bežične sisteme[6]. Novi modaliteti komunikacije, omogućeni samom prirodom bežičnih medijuma, ne mogu biti usvojeni korišćenjem klasičnih arhitektura, već zahtjevaju uvođenje Cross-Layer dizajna.

[6] B.Zhao and M. C. Valenti, "Practical Relay Networks: A Generalization of Hybrid-ARQ", *IEEE JSAC, vol.23, no. 1*, Jan 2005, pp. 7-18.

5. RBA arhitektura

Kao što je rečeno, klasična OSI arhitektura organizuje komunikaciju u vidu niza nivoa, a metapodaci koji kontrolišu prijem i slanje paketa su organizovani u vidu protokol hedera, po jedan za svaki nivo. Ovakav način organizacije je dobro služio kod originalne Internet arhitekture. Već su navedeni mnogi razlozi zašto je takav model danas teško održiv. Pomenimo jos da je situacija dodatno zakomplikovana razvojem niza middle-boxe-ova, kao što su zaštitne barijere i NAT kutije. U ovoj glavi je opisana sasvim nova arhitektura - Role-Based arhitektura. Umjesto niza nivoa i njima pridruženih protokol hedera, RBA organizuje komunikaciju korišćenjem funkcionalnih jedinica koje su nazvane rolama (eng. *roles*). Same role nisu organizovane striktno hijerarhijski, pa samim tim mogu da budu međusobno više povezane nego standardni protokol nivoi. Ulaz i izlaz iz pojedinačne role su podaci neke aplikacije i metapodaci za kontrolu, adresovani nekoj narednoj roli. Ideja je da role budu blokovi od kojih će biti izgrađen sistem, pri čemu će postojati nekoliko desetina dobro poznatih i standardizovanih rola. Broj rola, lokalno definisanih u nekom specifičnom sistemu, mogao bi biti znatno veći.

5.1 Osnovne karakteristike nove arhitekture

Glavna ideja na kojoj se zasniva RBA arhitektura je eksplicitno signaliziranje. Njihovo nepostojanje kod standardne OSI arhitekture je glavni razlog zbog kojeg middle-boxovi ne mogu biti uključeni u nju. Na primjer, ne postoji definisani način kojim bi se signaliziralo da je neki paket zaista prošao firewall protekciju. RBA takođe omogućuje svim komponentama uključenim u mrežu da budu jasno identifikovani, adresovani i da se sa njima ostvari komunikacija. Postojeći veliki protokoli, kao sto su IP, TCP, HTTP, primjenom ove arhitekture bili bi izdjeljeni na određeni broj manjih jedinica pridruženih specifičnim zadacima. Primjeri takvih zadataka mogli bi da budu "prosleđivanje paketa", "fragmentacija", "kontrola nivoa protoka", "zahtjev za web stranom", i td. (eng. "*packet forwarding*", "*fragmentation*", "*flow rate control*", "*request web page*"). Svaki od navedenih zadataka mogao bi se pridružiti specifičnoj roli.

Prilikom definisanja nove arhitekture, mora se voditi računa o sledećem - klasična OSI arhitektura je omogućavala modularnost, imala je definisanu strukturu metapodataka i pravila po kojima se obrađuju. Modularnost, sa osobinama skrivanja informacija i nezavisnosti, predstavlja alat od neprocjenjive važnosti za dizajn sistema. Bilo koji alternativni model mora da ima takvu jednu

osobinu. Za razliku od OSI arhitekture, gdje je struktura metapodataka predstavljena u vidu steka, kod RBA je primjenjena struktura hipa. Paket heder je zamjenjen kontejnerom koji može da sadrži blokove metapodataka promjenljive veličine. Tim blokovima može da se pristupa, mogu biti umetani, modifikovani ili uklonjeni, u proizvoljnom redosledu (što je omogućeno primjenom hip strukture). Što se tiče pravila po kojima role obrađuju podatke, ne postoje nikakva ograničenja u slučaju da su one međusobno nezavisne. U suprotnom, moraju se definisati pravila obrade za grupe zavisnih rola.

5.2 Opis pojedinačnih elemenata na kojima se zasniva RBA

Kao što je rečeno, osnovna funkcionalna jedinica RBA arhitekture je rola. Svaka rola je funkcionalni opis blokova od kojih je izgrađen sistem, pri čemu pojedinačni blokovi obavljaju specifične funkcije koje su od značaja za prosleđivanje i/ili obradu paketa. U konkretnom sistemu, role se pojavljuju u vidu različitog broja instanci ili čvorova (eng. *nodes*) koje se nazivaju actor-ima (engl. *actors*). Razlika između role i instance u mnogim slučajevima neće biti od važnosti, tako da možemo govoriti o roli, imajući u vidu konkretnu primjenu. Rola može da obuhvati veći broj instanci, tako da se rola može shvatiti kao apstraktni pojam, dok bi njene instance bile konkretni primjeri role u nekom sistemu.

Metapodaci u paketu (eng. *role data*) podjeljeni su na veći broj jedinica (eng. *chunk*) nazvanih RSH hederi (eng. *role-specific headers*). Na slici 3 je prikazan paket koji sadrži tri različita RSH-a u svom hipu i tri role koje čitaju i upisuju podatke u svakom od njih.

Bilo koja instanca u path-u može dodati novi RSH u postojeći hip paketa. Na primjer, pretpostavimo da firewall posjeduje jednostavan način zapisivanja informacije da je dati paket ispitan i da je u skladu sa njegovom sigurnosnom polisom (engl. *security policy*). On može da uključi dodatni RSH sa tom informacijom u postojeći paket i da ga proslijedi firewall roli domaćina.

RBA je dizajniran sa ciljem ostvarivanja sledećih osobina:

Mogućnost daljeg proširivanja (engl. extensibility): RBA je suštinski proširiv iz razloga što, sa jedne strane, podatke kodira koristeći mehanizam (*tip, dužina, vrijednost*), a sa druge strane, podržava koncept modularnosti.

Portabilnost (engl. portability): Apstraktni pojam role je dizajniran tako da bude nezavisan od konkretnog izbora instance, kojom će biti predstavljen u nekom sistemu. Ova portabilnost omogućuje fleksibilno dizajniranje iz razloga što se funkcije mogu grupisati na najpogodniji način.

Teorijska osnova za uvođenje middle box-ova: Osnovni elementi na kojima je zasnovan RBA dizajnirani su sa ciljem da arhitektura uključi middle-box-ove. Više o samom načinu uključivanja middle-box-ova u poglavlju 5.6.

Kontrolisani pristup metapodacima: RBA uključuje šemu kontrole pristupa metapodacima, kojom je definisano koji čvorovi mogu da čitaju i mijenjaju određeni podskup metapodataka.

5.3 Opis role

Role imaju dobro definisani ulaz i izlaz, predstavljen u vidu RSH elemenata, čija sintaksa i semantika može biti opisana u formi API-ja ostalim softverskim komponentama u lokalnom čvoru. Rola je predstavljena jedinstvenom **RoleID** oznakom. Kompletna RoleId oznaka može imati veći broj komponenata, kao što je slučaj kod imena fajlova. U ovom slučaju, hijerarhija bi označavala izvođenje specifične role iz neke generičke role (objašnjenje u daljem tekstu). Radi efikasnog transporta i uparivanja oznaka, u većini slučajeva koristi se kratka forma RoleID-a

RoleID služi da uputi meta-podatke elementu koji obezbjeđuje određenu funkcionalnost. Njime se ne određuje koji konkretni čvor će biti korišćen za obavljanje tražene funkcije. Čvorovi takođe imaju jedinstvenu oznaku, **NodeID,** koji bi odgovarao network adresi tradicionalne OSI arhitekture. Rol adresa se simbolički zapisuje u formi RoleID@NodeID, ili RoleID@*, ako NodeID nije određen.

Da bi mogli da obavljaju specifične zadatke, čvorovi mogu da imaju svoje unutrašnje stanje, koje se naziva **role state**. Postavljanje, promjena i brisanja role state-a zahtjeva postojanje nekog oblika signaliziranja, što je podržano kroz razmjenu RSH-ova.

Role se definišu i u obliku komplementarnih parova koji su nazvani **reflective roles**. Jednostavni primjeri takvih parova su (*Fragment, Reassemble*), (*Compress, Expand*) ili (*Encrypt, Decrypt*).

Moguće je formirati i familije srodnih rola, koje se razlikuju u detaljima, ali definišu istu generičku funkcionalnost. Takve familije rola predstavljaju se u vidu generičkih rola (eng. **generic role**). Analogni primjer ovakvog organizovanja elemenata bilo bi objektno-orijentisano programiranje. Ako bi uspostavili paralelu između rola u RBA arhitekturi i klasa u OO konceptu, onda bi čvorovima ili role actor-ima odgovarale instance klasa.

5.4 Organizovanje podataka u paketu

Svi podaci u jednom paketu su podjeljeni u niz RSH-ova. Broj RSH-ova varira u zavisnosti od servisa koje zahtjeva klijent, a takođe varira i dinamički, u toku prolaska paketa kroz mrežu. Svaki pojedinčni RSH može biti adresovan većem broju rola, i svaka pojedinačna rola može primati i obrađivati veći broj RSH-ova.

Kao što se rolama ostvaruje podjela algoritama za komunikaciju na niz čvorova, tako se i uz pomoć RSH-ova izvršava podjela meta-podataka sadržanih u nekom paketu. Ta podjela je izvršena u skladu sa granicama između različitih rola, tako da se uz pomoć RSH-ova formiraju logičke jedinice organizovanja i kontrole pristupa nad meta-podacima.

Format pojedinačnih RSH-ova je specifičan za neku rolu. On može da bude u skladu sa konvencionalnim heder protokolima, ili može da ima drugačiju organizaciju, npr. (dužina, tip, vrijednost). RSH sadrži listu adresa različitih rola kojima je upućen i tijelo koje obuhvata podatke namjenjene tim rolama. Simbolički, to se zapisuje u obliku;

RSH(<*RoleAddressList*>;<*RoleBody*>)

Na primjer, zapis RSH(Expand@N3, Decrypt@*;...) predstavlja RSH adresovan čvoru *N3* role *Expand*, i bilo kojem čvoru role *Decrypt*.

5.5 Otvorena tehnička pitanja i mogući pravci rešavanja

Buduća nadgradnja definicije RBA arhitekture mora da odgovori na niz tehničkih pitanja. Ovdje su navedena samo najbitnija od njih i predloženi načini na koje bi mogli da budu riješeni u praksi. Pored navedenih, postoji i veliki broj manje bitnih problema koja će se pojaviti prilikom razvoja RBA arhitekture, ali njihovo rešavanje će biti u velikoj mjeri determinisano načinom na koji će se riješiti ovdje navedeni problemi.

Raspored izvršavanja pojedinačnih actor-a - Nephodno je ustanoviti pravila koja će spriječiti neželjeni niz operacija, kao što je *Encrypt, Compress* (kompresija nije korisna nakon enkripcije), ili *Expand, Compress* (pogrešan redosled), ili *Compress, Encrypt, Expand, Decrypt* (pogrešan redosled). Jedan od načina kako bi mogao da se prevaziđe ovaj problem jeste uvođenje bitova kontrole. Na primjer, u slučaju da imamo dva actor-a *Encrypt i Compress*, dva dodatna bita u RSH paketu (ili čak samo jedan) omogućila bi kontrolu izvršavanja navedenih actor-a. Actor zadužen za enkripciju bi, nakon obrade

RSH paketa, postavio na jedinicu odgovarajući kontrolni bit. Time bi proslijedio informaciju actor-u za kompresiju da može da pristupi obradi datog RSH paketa.

Kontrola pristupa RSH-ovima - RBA predviđa dva nivoa kontrole pristupa, de jure i apsolutna kontrola. De jure kontrola pristupa je obezbjeđena bitovima kontrole u pojedinačnom RSH-u, i njima su određena prava čtianja i pisanja pojedinačnim rolama. Pravo pisanja obezbjeđuje mogućnost promjene ili brisanja RSH-a iz paketa.

De jure kontrola je dovoljna dok čvorovi slijede RBA pravila. Inače, čvorovi mogu imati apsolutnu kontrolu nad nekim RSH-om tako što bi izvršili njegovu enkripciju. Ovakvi viši nivoi kontrole imali bi i veću cijenu. Međutim, postavlja se pitanje da li postoji stvarna potreba za uvođenjem drugog nivoa kontrole. Prvo, time bi se dodatno iskomplikovao dizajn arhitekture, kako u pogledu kompleksnosti definisanja uzajamne interakcije među rolama, tako i dodavanjem većeg broja novih algoritama. Drugo, dodavanje takve jedne mogućnosti moglo bi da dovede do sledeće situacije. Ako bi neki od actor-a iskoristio pravo apsolutne kontrole i enkriptovao RSH paket, a zatim pozvao svoju unutrašnju funkciju koja bi odgovarala destruktoru u OO programiranju, dati RSH paket ostao bi bez mogućnosti da mu pristupi neki drugi actor ili čak da ga obriše. Time bismo došli do problema koji je u programiranju poznat pod imenom **curenje memorije** (engl. *memory leak).*

Definicija rola - Da bi se u potpunosti definisala specifična rola, neophodno je definisati njena unutrašnja stanja, algoritme i RSH-ove koji bi bili primani i slati. Takođe, neke role mogu imati non-network interfejse koji moraju biti definisani. Ostaje da se vidi da li je koncept RBA arhitekture moguće definisati korišćenjem neke od tehnika formalne protokol specifikacije. Iz do sada izloženog, može se zaključiti da bi postojeći objektno-orijentsani koncept bio primjenljiv i u slučaju RBA arhitekture. Kao prvo, to je uspješno primjenjeni koncept čije su pozitivne i negativne strane u velikoj mjeri proučene. Kao drugo, način na koji je zamišljena sama priroda actor-a i rola ukazuje da postoji velika sličnost sa osnovnim pojmovima OO paradigme, klasama i njihovim instancama.

Kompozicija rola - Dvije različite role, *Ra* i *Rb*, koje međusobno direktno komuniciraju korišćenjem RSH-ova, mogu da formiraju novu rolu *Rc*, koja bi predstavljala njihovu kompoziciju. Time bi komunikacija između *Ra* i *Rb* bila zamjenjena internom komunikacijom pomoću dijeljenih podataka.

Sa druge strane, kompleksne role bi mogle da se dekomponuju u manje složene role, zamjenjujući dijeljene podatke komunikacijom preko RSH-ova. Time bi se omogućila modularnost same arhitekture, a već je pomenuto (i poznato) od kolike je to važnosti za uspjeh nekog koncepta u računarskim naukama. Ako bi prilikom zasnivanja RBA arhitekture koristili OO paradigmu, zahtjevi za kompozicijom i dekompozicijom bi bili automatski primjenjivi.

5.6 Prevođenje mrežnih adresa i RBA arhitektura

Objasnimo ukratko problem iscrpljivanja IP adresa. Davalac Internet usluga mogao je imati adresu klase B koja mu je omogućavala usluživanje 65534 računara. Za kućne korisnike koji su se povezivali modemom, IP adresa se mogla dodjeljivati dinamički u trenutku prijavljivanja na mrežu i ponovo oduzimati kada se korisnik odjavi. Na taj način, moguće je imati 65534 istovremeno aktivna korisnika, što je bilo dovoljno za davaoca Internet usluga koji ima više stotina hiljada korisnika. Ova strategija bila je primjenljiva dok postoji veći broj kućnih korisnika. Problem nastaje kada davalac ima prvenstveno poslovne korisnike. Poslovni korisnici očekuju da stalno budu na mreži tokom radnog vremena, što ima za posledicu da svaki računar čitavog dana ima istu IP adresu. Takođe, veliki broj kućnih korisnika koristi ADSL liniju ili kablovski Internet, tako da se oni uopšte ne odjavljuju sa mreže i time dodatno uvećavaju problem nedostatka IP adresa.

Sistem za **prevođenje mrežnih adresa** (engl. *Network Address Translation, NAT*) svakoj kompaniji dodjeljuje samo jednu IP adresu (ili najviše manji broj adresa, tzv. **eksternih adresa**). Unutar kompanije, svaki računar dobija jedinstvenu IP adresu koja služi za interni saobraćaj. Kada paket napušta kompaniju i odlazi ka davaocu Internet usluga, on prolazi kroz **NAT kutiju** (engl. *Nat box*) i njegova interna adresa prevodi se u eksternu adresu. Nat kutija se često kombinuje u istom uređaju sa **zaštitnom barijerom** (engl. *firewalls*).

Problem nastaje kada stigne spoljašnji paket sa Interneta koji je, naravno, adresiran na eksternu adresu. Postavlja se pitanje kako NAT kutija zna kojom internom adresom da je zamjeni? Taj problem rešava se korišćenjem **izvorišnog i odredišnog priključka** (engl. *source and destination port*). Kada proces želi da uspostavi TCP vezu sa udaljenim procesom, on se priključi na nezauzet TCP priljučak na svom računaru. To je izvorišni priključak koji TCP kodu saopštava gdje da šalje dolazne pakete koji pripadaju priključku. Proces obezbjeđuje i odredišni priključak - mjesto na udaljenom računaru na kome treba predavati pakete. Svaka poslata TCP poruka sadrži i izvorišni i odredišni priključak. Oni zajedno identifikuju procese koji koriste vezu, i to na oba kraja. Iako je ovaj sistem privremeno riješio problem manjka IP adresa (do trenutka prelaska na IPv6), postoje mnoge kritike na njegov račun. Prvo, NAT narušava arhitekturu IP modela u kome svaka IP adresa jedinstveno identifikuje samo jedan računar na svijetu. Drugo, Internet se pretvara u mrežu sa izvjesnim uspostavljanjem direktne veze (Nat kutija mora da održava informacije o preslikavanju za svaku vezu koja prolazi kroz nju). Treće, NAT narušava osnovno pravilo raspoređivanja protokola po slojevima: sloj k ne smije da pravi nikakve pretpostavke o tome šta je sloj k+1 smjestio u polje za korisničke podatke. Ako bi se pojavila nova verzija protokola TCP, npr. TCP-2 koji koristi drugačiji raspored zaglavlja (npr. 32-bitne priključke), NAT će da zakaže. Četvrto, NAT podrazumjeva da se koristi TCP ili UDP protokol (novi transportni protokol za

npr. multimedijalne aplikacije bi onemogućio NAT da ispravno locira TCP izvorišni priključak).

Iz navedenog je jasno da je NAT jedna od komponenata koja se ne uklapa u standardnu OSI arhitekturu. Međutim, prelazak na RBA arhitekturu riješio bi neke od navedenih problema i omogućio jednostavno uključivanje NAT-a u arhitekturu mreže. Prije opisa kako bi se to implementiralo, navedimo da postoje dvije vrste prevođenja IP adresa, **jednostavni NAT** (engl. *pure NAT*) i **sistem za prevođenje adresa i portova** (engl. *Network address port translation*). Jednostavni NAT obavlja dinamičko mapiranje (tipa *one-to-one*) između velikog broja internih adresa i malog broja eksternih adresa. **Prevođenje portova** (engl. *Netork address port translation*) omogućava pristup određenim računarima u mreži na način što se saobraćaj usmjerava preko određenih portova.

Protokoli aplikativnog nivoa nisu svjesni postojanja NAT komponente sistema. Neke aplikacije, kao što su **protokol za prenos datoteka** (engl. *File Transfer Protocol, FTP*) i protokol za Internet telefoniju H.323 umeću IP adrese u sam tekst. Primalac tada preuzima adrese i koristi ih. Pošto NAT ništa ne zna o tim adresama, on ih ne može zamjeniti, pa će propasti pokušaji da se one koriste na udaljenom računaru. NAT bi mogao da se prepravi tako da sarađuje sa navedenim protokolima, ali "krpeljenje" koda svaki put kada se pojavi nova aplikacija nije preporučljivo. RBA arhitektura na jednostavan način rešava ovaj problem. Prilikom obrade odlazećeg paketa (engl. *outcoming packet*), NAT prosleđuje dodatni RSH paket roli koja se naziva **NAT-primalac** (engl. *Nat-receiver*) i koja se nalazi na strani servera, dajući joj originalnu adresu. Na taj način server postaje svjestan da je došlo do date promjene i može bez problema da obavi zahtjevani transfer. U slučaju NAPT-a, situacija je nešto složenija. Pored dodavanja nat-primalac RSH paketa, neophodno je dodati i tzv. Cookie rolu. Kod odlazećih paketa, NAPT prosleđuje dodatni RSH paket ovoj roli, koji sadrži cookie koji je jedinstven za svaku internu adresu. Svi sistemi bi trebalo da budu svjesni postojanja cookie role tako što će emitovati svoj cookie dodavanjem tzv. *cookie_echo* RSH-a u svaki response paket. Na taj način se prevazilazi i navedeni problem nemogućnosti lociranja TCP izvorišnog priključka u slučaju prelaska na neki novi protokol.

5.7 Bezbjednost komuniciranja i RBA arhitektura

Problem bezbjednosti može se jednostavno opisati sledećim pitanjem - kako da bitove diskretno i bez mijenjanja sprovedemo od izvorišnog do odredišnog računara i da, pri tome, neželjene bitove zadržimo napolju? Uvođenje dodatne bezbjednosti pratila je polemika oko toga gdje je treba smjestiti. Stvarna bezbjednost, po mišljenju većine stručnjaka, može se postići samo šifrovanjem i provjerom integriteta od jednog do drugog kraja (tj. u sloju aplikacija). Drugim

riječima, izvorišni proces treba da šifruje podatke i da zaštiti njihov integritet, a zatim da ih pošalje odredišnom procesu gdje će biti dešifrovani i provjereni. Glavni problem kod ovog pristupa je što većina korisnika ne shvata u potpunosti bezbjednost i ne bi bila sposobna da je koristi na pravi način. Manje dobar pristup je da se podaci šifruju u transportnom sloju ili u novom sloju između sloja aplikacija i transportnog sloja. Argumenti u korist drugog pristupa su vremenom prevagnuli, i kao rezultat toga nastao je projekat **IP bezbjednost** (engl. *IP security, IP-sec*). Jedna od bitnih karakteristika IPSec sistema je to što on radi sa uspostavljanjem direktne veze iako se nalazi u IP sloju (neophodno je uspostaviti ključ tokom određenog vremena, a to jeste neka vrsta veze).

IPSec se može koristiti u dva režima. Kod **transportnog režima,** IPSec zaglavlje se umeće neposredno iza IP zaglavlja. Novo zaglavlje služi za prenos bezbjednosnog identifikatora, podataka za provjeru integriteta i drugih informacija. U **tunelskom režimu,** IP paket se kapsulira u tijelo novog paketa sa potpuno novim IP zaglavljem. Ovaj režim je naročito zgodan u slučaju kada se tunel završava prije nego što podaci dođu do izvorišnog računara. Na primjer, ako bi se tunel završio u zaštitnoj barijeri neke lokalne mreže, u njoj bi se obavljalo kapsuliranje i dekapsuliranje paketa. Računari koji se nalaze u toj mreži ne bi morali da budu svjesni postojanja IPSec-a.

Jedan od osnovnih principa na kojima je izgrađen Internet je **end-to-end** prinicip - svaki čvor u mreži može da šalje pakete svim ostalim čvorovima, pri čemu međuelementi ne moraju da obavljaju interpretaciju tih paketa. IPSec, NAT i zaštitne barijere su neki od elemenata savremenih mreža koji ne zadovoljavaju ovaj princip. Pored toga, IPSec uvodi novi "međusloj" između sloja aplikacija i transportnog sloja, čime se narušava originalna OSI podjela komunikacije na sedam nivoa. RBA ekvivalent, kojim bi se obezbjedila IP bezbjednost, sastojao bi se od četiri role: *Encrypt, Decrypt, Send i Receive*. Ove role bi bile definisane na svakom čvoru mreže.

```
typedef ADDRESS_LEN 128
typedef enum {false, true} boolean;
#define  INTERMEDIATE_ELEMENT  -1
#define  DESTINATION_ELEMENT    11
```

RSH bi mogao biti definisan u vidu strukture, koja bi sadržala adresu role i RSH podatke (specifične za svaku rolu). Takođe, paket se može predstaviti kao struktura koja sadrži integer koji predstavlja broj RSH-ova, zatim niz RSH-ova i korisni teret (tipa unsigned char *).

```
typedef struct
{
    unsigned char [ADDRESS_LEN] role_address;
    unsigned char *rsh_data;
} RSH;
typedef struct
{
    int number_of_rsh;
    RSH *rsh_array;
    unsigned char *payload;
} Packet;
```

Klase Encrypt i Decrypt, pored javnih podataka, definisale bi i funkcije za enkripciju (tj. dekripciju), za dodavanje odgovarajućih RSH-ova, za razmjenu ključeva, kao i funkcije za provjeru i promjenu odgovarajućih kontrolnih bitova.

```
class Encrypt
{
public:
    ...
    /*Javni podaci*/

private:

    /*F-ja za dodavanje RSH-a*/
    int addEncryptRsh (Packet &p);
    /*F-ja za razmjenu ključeva*/
    int exchangeKeys (int *key);
    /*F-ja za enkripciju*/
    int encrypt (Packet &p);
    /*F-ja koja provjerava da li je zahtjevana primjena IPSec-a*/
    boolean checkRequestedIPSecControlBit (Packet p);
    /*F-ja za postavljanje send kontrolnog bita*/
    void incrementSendControlBit (Packet &p);
    /*F-ja za postavljanje kontrolnog bita "primjenjen je IPSec"*/
    void incrementIPSecAppliedControlBit (Packet &p);
}
class Decrypt
{
    /*F-ja za dodavanje RSH-a*/
    int addDecryptRsh (Packet &p);
    /*F-ja za razmjenu ključeva*/
    int exchangeKeys (int *key);
    /*F-ja za dekripciju*/
    int decrypt (Packet &p);
    /*F-ja za prosleđivanje paketa narednoj roli*/
    void decrementIPSecAppliedControlBit (Packet &p);
}
```

U nekoj konkretnoj razmjeni paketa, međuelementi bi koristili isključivo Send i Receive role, kako bi obezbjedili prosleđivanje paketa od izvorišnog do odredišnog računara. Na izvorišnom računaru, paket bi prvo stigao do Encrypt role, koja bi nakon obrade promjenila vrijednot dva kontrolna bita. Prvim kontrolnim bitom naznačilo bi se Send roli da je paket spreman za slanje. Drugi

kontroni bit služio bi da se odredišnom računaru proslijedi informacija da je primjenjen IPSec na dati paket, tj. da li je paket kriptovan. Time bi se ostavila mogućnost da se u specijalnim slučajevima isključi primjena IP bezbjednosti. Npr, međusobna komunikacija računara u lokalnoj mreži ne zahtjeva njegovu primjenu, dovoljno je da bude obezbjeđena komunikacija ka računarima izvan lokalne mreže. Send rola, nakon što provjeri da li je kontrolni bit postavljen na jedinicu, obavila bi slanje paketa. Kad paket stigne do odredišnog računara, njegovu obradu bi preuzela Receive rola. Provjerom vrijednosti kontrolnog bita odredilo bi se da li se paket prvo prosleđuje Decrypt roli ili direktno ide na dalju obradu.

```
void exchangePackets (Packet &p)
{
   /*Deklaracija različitih actor-a*/
 Encrypt   encrypt_actor;
 Decrypt   decrypt_actor;
 Send      send_actor;
 Receive   receive_actor;
 ...
 while (true)
 {
 /*Čekamo informaciju o nekom događaju. Ako uslijedi zahtjev za
  *pripremu paketa za slanje, provjeriti da li je zahtjevana
  *primjena IPSec-a, i preduzeti odgovarajuće akcije*/

      wait_for_event (&event);
      if (PACKET_PREPARE == event)
      {
           preparePacket (&p);
           if (encrypt_actor.checkRequestedIPSecControlBit  (&p))
           {
           /*Ako je zahtjevana primjena IPSec-a, generisati i
            *razmjeniti ključeve i izvršiti enkripciju. Zatim
            *dodati novi RSH i proslijediti informaciju da je paket
            *spreman za slanje, kao i da je primjenjen IPSec
            *(promjenom odgovarajućih kontrolnih bitova)*/

                 int *key;
                 encrypt_actor.exchangeKeys (key);
                 encrypt_actor.encrypt (p);
                 encrypt_actor.addEncryptRsh (p);
                 encrypt_actor.incrementSendControlBit (p);
                 encrypt_actor.incrementIPSecAppliedControlBit(p);
                 encrypt_actor.toSendRole (p);
           }
           else
           {
                /*Ako se ne primjenjuje IPSec, dodati novi RSH i
                 *naznačiti da je paket spreman za slanje*/

                encrypt_actor.addEncryptRsh (p);
                encrypt_actor.incrementSendControlBit (p);
           }
           if (send_actor.checkSendControlBit (p))
```

```
            {
                 /*Dodati novi RSH i pozvati funkciju za slanje
                  *paketa*/

                 send_actor.addSendRsh (p);
                 send_actor.sendPacket (p);
            }
       }
       else if (PACKET_RECEIVED == event)
       {
           /*Ako je događaj "stigao je novi paket", prvo provjeriti
            *da li je on upućen nekoj drugoj adresi. Ako jeste, paket
            *se bez promjene prosleđuje u mrežu. U suprotnom,
            *preduzeti odgovarajuće akcije u zavisnosti od toga da li
            *je primjenjen IPSec*/

            if (INTERMEDIATE_ELEMENT ==
                         receive_actor.checkAddress(p))
            {
                 receive_actor.incrementSendControlBit (p);
            }
            else if (DESTINATION_ELEMENT ==
                         receive_actor.checkAddress (p))
            {
                 if (receive_actor.checkIPSecAppliedControlBit (p))
                 {
                 /*Ako smo na odredišnoj adresi i primjenjen je
                  *IPSec, dodati novi RSH i izvršiti dekripciju
                  *paketa. Nakon toga, paket se upućuje sledećoj
                  *roli*/

                       receive_actor.addReceiveRsh (p);
                       decrypt_actor.decrypt (p);
                       decrypt_actor.addDecryptRsh (p);

                    decrypt_actor.decrementIPSecAppliedControlBit(p);
                       decrypt_actor.toNextRole (p);
                 }


                 else
                 {
                      /*IPSec nije primjenjen, paket je spreman za
                       *dalju obradu bez prethodne promjene*/

                      receive_actor.addReceiveRsh (p);
                      receive_actor.toNextRole (p);
                 }
            }
       }

 }
}
```

Encrypt i Decrypt rola bi mogle da budu definisane kao generičke role. Iz njih bi bile izvedene specifične role, npr. u zavisnosti od toga koji je algoritam za šifrovanje primjenjen (AES, Rijnadael i td.).

Da bi se spriječilo prodiranje digitalne gamadi i uljeza u lokalnu mrežu, uvodi se sistem zaštitnih barijera (engl. *firewalls*). Ovaj sistem sastoji se od dvije osnovne komponente: dva usmjerivača koji filtriraju pakete i mrežnog prolaza za aplikacije. Svaki paket koji želi da napusti lokalnu mrežu ili da uđe u nju mora prvo da prođe kroz zaštitnu barijeru. Postojeće komponente mogle bi vrlo jednostavno da se prilagode RBA arhitekturi. Funkciju filtera za pakete obavljale bi **dolazna filter rola** i **odlazna filter rola**. Kao i u postojećim sistemima, i ovdje bi role radile na osnovu tabela koje konfiguriše administrator sistema. Paketi koji zadovoljavaju definisane kriterijume bili bi prosleđivani roli koja obavlja mrežni prolaz za aplikacije, gdje bi se vršila dalja provjera paketa. Prilikom slanja paketa iz neke lokalne mreže, paketi takođe prolaze kroz zaštitnu barijeru. Ako odlazna filter rola ili rola za mrežni prolaz za aplikacije zaključe da dati paket ne zadovoljava neki od kriterijuma, u paket se dodaje posebno RSH zaglavlje koje sadrži informaciju o tome da paket nije prošao barijeru i, eventualno, informaciju o tome koji kriterijum nije zadovoljen. Na ovaj način se omogućava eksplicitno signaliziranje aplikaciji da paket nije poslat.

6. Zaključak

Postoji stalna potreba za poboljšanjem performansi bilo kog sistema. U većini slučajeva, to dovodi do suprostavljenih zahtjeva arhitekture, sa jedne strane, i performansi sa druge strane. Dobro utemeljena arhitektura je od primarne važnosti za razvoj nekog sistema, i u trenutku kada se razmatraju alternativni modeli za bežične sisteme, ova činjenica ne smije biti zaobiđena. Mora se izbjeći situacija u kojoj bi novi model doveo do tzv. "špageti dizajna", gdje bi kasnije uvođenje novih osobina dovelo do jako velikog broja međusobnih interakcija sa postojećim elementima sistema. Nedovoljno razmotreni prelazak na novu arhitekturu može zagušiti dalji razvoj sistema, ako bi kasnija poboljšanja performansi zahtjevala potpunu reorganizaciju i zamjenu nekih djelova sistema.

Glavna ideja ovog rada je da predstavi RBA arhitekturu, kao jedan od oblika Cross-layer dizajna. Cilj uvođenja nove arhitekture jeste pojednostavljivanje dizajna i razvoja protokola za komunikaciju u današnjem svijetu, gdje uzajamno dejstvo između različitih elemenata sistema često izlazi iz okvira klasične OSI arhitekture. RBA obezbjeđuje uniforman način za strukturiranje protokola i njihovu međusobnu interakciju.

Napuštanje postojećeg modela arhitekture, naravno, ima i svoju cijenu. OSI model arhitekture je veoma moćna "alatka" za dizajniranje sveobuhvatnih protokola. Taj model je veoma uspješno izdržao probu vremena. Neophodno je sa sigurnošću ustanoviti da bi RBA arhitektura obezbjedila najmanje podjednako dobar, ako ne i bolji način za razvoj protokola. Takođe, opisani model podrazumjeva složeniju organizaciju podataka u paket hederima, sto ima uticaja na implementaciju, veličinu paketa i performanse samog sistema. Dalji razvoj modela mora pokazati da je cijena prelaska na novi model opravdana.

Literatura:

[1] **Tanenbaum, Andrew S.**: "Računarske mreže", *Prevod četvrtog izdanja*, *Mikro knjiga*, 2005
[2] **Srivastava, V.,Motani, M.**: "Cross-Layer design: A Survey and the Road Ahead", *IEEE Commun. Magazine*, December 2005
[3] **Braden, R., Faber, T., Handley, M.**: "From Protocol Stack to Protocol Heap - Role-Based Architecture", *HotNets-I*, October 2002
[4] **Kawadia, V., Kumar, P.R.**: "A Cautionary Perspective on Cross-Layer Design", *IEEE Wireless Communications*, February 2005