SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 290

Virtualizacija prevoditelja mrežnih adresa visokih performansi realizirana programskim okvirom Data Plane Development Kit

Karlo Miličević

Zagreb, veljača 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 290

Virtualizacija prevoditelja mrežnih adresa visokih performansi realizirana programskim okvirom Data Plane Development Kit

Karlo Miličević

Zagreb, veljača 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 2. listopada 2023.

DIPLOMSKI ZADATAK br. 290

Pristupnik:	**Karlo Miličević (0036506449)**
Studij:	Računarstvo
Profil:	Znanost o mrežama
Mentor:	izv. prof. dr. sc. Miljenko Mikuc

Zadatak: **Virtualizacija prevoditelja mrežnih adresa visokih performansi realizirana programskim okvirom "Data Plane Development Kit"**

Opis zadatka:

Uređaji koji obavljaju prevođenje mrežnih adresa (engl. Network Address Translation, NAT) ključni su za rad današnjih IPv4 mreža. Zbog jako sporog prijelaza na IPv6 i potrebe za održavanjem starijih sustava koji ne mogu prijeći na moderniji standard, neizbježno je korištenje funkcije prevoditelja mrežnih adresa. Postoje specijalizirana sklopovska rješenja, ali ona su skuplja i puno manje fleksibilna od programske izvedbe. Tehnologija "Data Plane Development Kit" (DPDK) omogućuje obradu mrežnog prometa bez korištenja mrežnih funkcija jezgre i time omogućuje efikasnije iskorištenje resursa i kraće odzive. Vaš je zadatak implementirati prevođenje mrežnih adresa korištenjem programskog okvira DPDK te usporediti performanse s postojećim implementacijama koje pružaju moderni operacijski sustavi.

Rok za predaju rada: 9. veljače 2024.

ZAHVALA

Želim zahvaliti mentoru, izv. prof. dr. sc. Miljenku Mikucu na pomoći pri izradi rada.

Hvala dr. sc. Denisu Salopeku na motiviranju i podršci tijekom cijelog procesa pisanja rada.

Također, hvala i Pet Puta Devet d.o.o. na pruženoj opremi za izradu rada, opremi za mjerenje i pruženoj prilici učenja od iskusnih stručnjaka.

Na kraju, želim zahvaliti Ivoni i Medi, roditeljima, baki, bratu, sestri i šogoru na neizmjernoj potpori koju su mi pružali godinama tijekom studiranja.

Sadržaj

Uvod

Sve je više korisnika u internetu i nije izgledno da će se broj korisnika u budućnosti smanjiti. Tome doprinosi eksplozivan porast broja IoT (engl. *Internet of things*) uređaja i sustava koji zahtjevaju pristup internetu. Osim toga, sve više ljudi je u svijetu i sve lakša je dostupnost uređaja koji se mogu povezati na internet.

Iako gotovo trideset godina postoji noviji standard IPv6[1], IPv4[2] se ne prestaje koristiti. Kako bi se zabišao problem iscrpljivanja IPv4 adresa, definirana je tehnika prevođenja adresa (NAT[3], engl. *Network Address Translation*). To se često koristi u kućnim mrežama i malim uredima gdje postoji više uređaja, poput računala, pametnih telefona i televizora, koji trebaju pristup internetu. Danas većina kućnih mreža ne koristi globalno dostupne adrese, nego privatne adrese, a operater kućanstvu dodjeljuje samo jednu javnu adresu. U tom se slučaju prevođenje adresa lokalne mreže događa na uređaju koji operater postavlja.

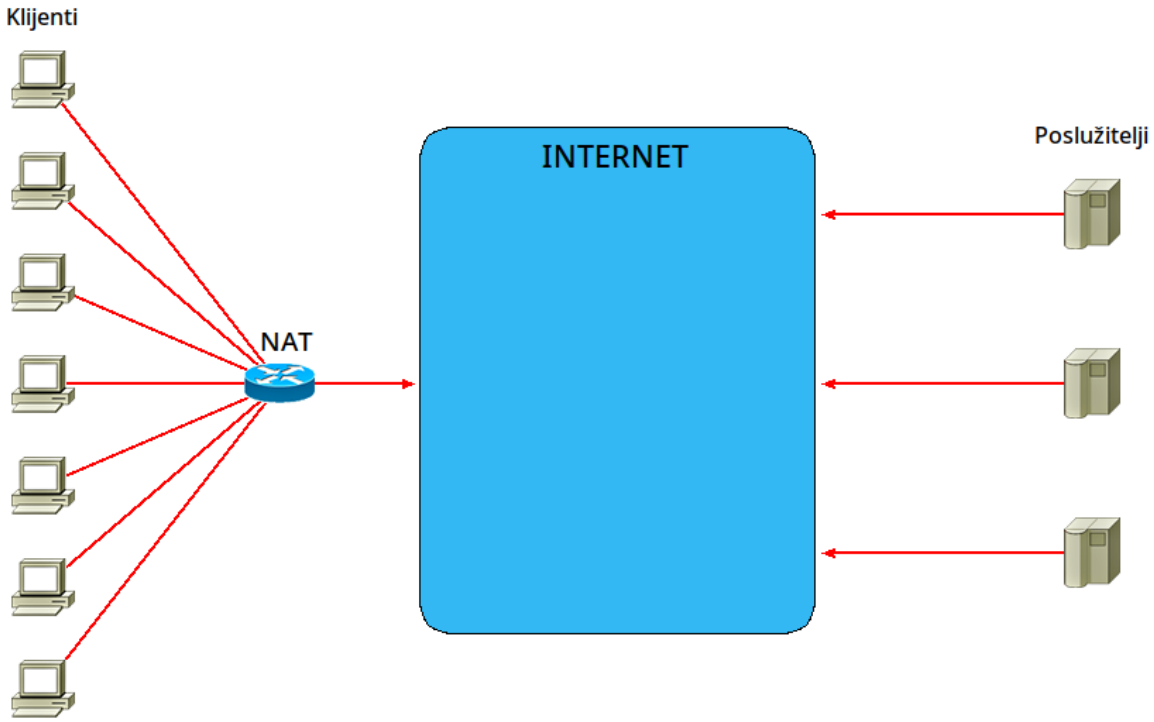
Neki operateri povezuju više kućanstava u jednu mrežu te rade dodatno prevođenje adresa (Carrier-grade NAT), time dodatno smanjujući broj korištenih IPv4 adresa.

U sklopu ovog rada implementirana je funkcionalnost prevoditelja mrežnih adresa korištenjem platforme DPDK. Opisana je implementacija prevoditelja i primjer obrade prometa. Opisano je okruženje u kojemu su rađena mjerenja. Isprobana je funkcionalnost i performanse. Sva mjerenja su napravljena i za popularne prevoditelje iptables i ipnat te su uspoređeni rezultati.

[1] https://datatracker.ietf.org/doc/html/rfc1883
[2] https://datatracker.ietf.org/doc/html/rfc760
[3] https://datatracker.ietf.org/doc/html/rfc1631

1 NAT

Prevoditelj mrežnih adresa je uređaj koji je nastao kao jedan od odgovora problemu velikog porasta broja uređaja na globalnoj IPv4 mreži, odnosno ponestanku slobodnih adresa. Sredinom 1996. godine je opisana funkcionalnost ovog uređaja u dokumentu RFC 1631. Broj globalno dostupnih IPv4 adresa iznosi oko 3.7 milijardi, no on je malen u usporedbi s brojem ljudi i uređaja u svijetu. Iako postoji drugoročnije i bolje rješenje - korištenje IPv6 protokola - postojeće mreže, odnosno operateri relativno sporo prelaze na njega. Slika 1 prikazuje tipičan položaj uređaja NAT u internetu.



Slika 1: Položaj NAT-a u mreži

Činjenica koja omogućuje korištenje NAT-a je to što nije potrebno da su svi uređaji uvijek dostupni svima ostalima u mreži. Većina uređaja u internetu se može svrstati u dvije kategorije - one koji traže nešto od drugih uređaja, i one koji poslužuju druge uređaje svojim uslugama. Samo poslužitelji zapravo trebaju imati neki način da se komunikacija prema njima započne.

Ostali uređaji zapravo samo trebaju privremene adrese za vrijeme svojeg komuniciranja s poslužiteljima. Uređaj NAT, dakle, može privremeno, odnosno za vrijeme trajanja sjednice s poslužiteljem, pružiti uređaju korištenje neke od njegovih adresa. Za UDP[4] i TCP[5] je relativno jednostavno prepoznati sjednicu - jedinstveno je definirana parom (<izvorišna adresa, izvorišni port>, <odredišna adresa, odredišni port>).

ALG (engl. *application-level gateway*) je dio prevoditelja koji je zaslužan za ispravno prepoznavanje i prevođenje komunikacije određenog protokola. Kod UDP i TCP prometa, bitno je prepoznati sjednice, zamijeniti privatnu adresu adresom uređaja, te popraviti kontrolni zbroj (engl. *checksum*). TCP i UDP predstavljaju većinu prometa u internetu. Drugi protokoli se isto mogu prevoditi, a ovisno o protokolu prevođenje može biti nešto kompliciranije.

NAT46 i NAT64

Jedna od mogućih funkcija prevoditelja mrežnih adresa je prevođenje iz privatnog IPv4 prometa u javni IPv6 promet, odnosno NAT46. Ova funkcija omogućuje suživot IPv4 mreža s IPv6 mrežama i time olakšava postepeni prijelazak na IPv6. Jedan od značajnih dijelova ove funkcije je prevođenje DNS odgovora iz IPv6 u IPv4. NAT64[6] ima istu ulogu kao i NAT46, ali sa zamijenjenom javnom i privatnom stranom. NAT46 i NAT64 je nešto zahtjevnija funkcionalnost i u ovom radu neće biti istraživana.

CGNAT

CGNAT (engl. *Carrier-grade* NAT) je vrsta NAT-a koja se koristi u telekomunikacijskim mrežama. Glavna razlika u odnosu na obični NAT je podrška za većom količinom korisnika, većom količinom prometa i redundancijom. Osim toga, u telekomunikacijskim mrežama nužna je dobra podrška za bilježenje korisničkih akcija. Za pružanje tih funkcionalnosti CGNAT se ponaša nešto drukčije od običnog NAT-a. Prilikom dodjeljivanja porta korisniku je nužno i dodati u zapisnik da tom korisniku pripada taj port. Radi smanjenja količine

[4] https://datatracker.ietf.org/doc/html/rfc768
[5] https://datatracker.ietf.org/doc/html/rfc793
[6] https://datatracker.ietf.org/doc/html/rfc6146

zapisa portovi se ne dodjeljuju zasebno, nego iz većih uzastopnih regija - `bucketing`. Ako pretpostavimo `bucket` veličinu od 500, korisniku će osim tog porta biti rezervirano i idućih 499 portova za buduće dodjele.

2 Korišteni alati

C++

Kao jezik implementacije rješenja odabran je C++. C++ je čest izbor u razvoju programske potpore visokih performansi zbog svoje raširenosti i brzine. Brzinu ostvaruje prevođenjem u izvršni kod te izvođenjem istoga direktno. Osim toga, moderni prevoditelji rade puno različitih optimizacija i koriste znanje o sklopovlju na kojemu će se program izvršavati. Omogućuje direktan rad s memorijom što daje mogućnost bolje iskoristivosti arhitekture memorije, odnosno priručnih spremnika. U implementacijama je korišten ograničen dio jezika C++ koji se gotovo izravno može prenijeti u programski jezik C.

git

Git je alat za distribuirano upravljanje izvornim kodom i verzioniranje. U današnje vrijeme, git[7] je najčešći odabir za alat za upravljanje izvornim kodom. Omogućuje lako pregledavanje projekta kroz verzije i distribuirani rad na projektu.

CMake i Ninja

Problem s jezikom C++, a i puno drugih jezika je što je prevođenje programa izvan specifikacije jezika. Taj se problem uglavnom zaobilazi korištenjem drugih jezika i alata. Ti alati se koriste samo za opisivanje prevođenja programa. CMake[8] je korišten u svrhu opisa prevođenja programske potpore. CMake pokušava samo apstraktno opisati prevođenje nekog projekta i ne može direktno izvršiti opis prevođenja. Na temelju opisa prevođenja generira program za prevođenje. Pokretanjem programa za prevođenje prevodi se izvorni program u skladu s opisom koji je definiran pomoću CMakeLists.txt datoteke. Može se odabrati jezik koda programa za prevođenje kojeg će CMake generirati. U izvedbi programske

[7] https://git-scm.com/
[8] https://cmake.org/

potpore koja je napravljena u sklopu ovog rada odabran je jezik Ninja[9]. Taj jezik i alat je sličan poznatom Make[10] alatu, ali je puno više ograničen, s fokusom na jednostavnost i brzinu interpretiranja. Ninja skripte nisu zamišljene da se ručno pišu, nego se generiraju alatima poput CMake.

Data Plane Development Kit (DPDK)

U ovom radu je korištena Intelova platforma Data Plane Development Kit[11] za prihvat i slanje mrežnih paketa. DPDK omogućuje zaobilaženje mrežnog podsustava jezgre i direktan pristup mrežnim paketima iz korisničkog programa. Time je znatno smanjena količina posla prilikom obrade paketa te povećana količina paketa koje možemo obraditi. Osim toga, ova platforma sadrži biblioteke s implementacijama raznih struktura koje omogućuju bolje iskorištenje priručne memorije procesora te neke korisne strukture za višejezgreni rad programa.

Linux

Razvijena programska potpora se ne oslanja na puno specifičnosti operacijskog sustava. Pretpostavlja se okolina koja nalikuje UNIX operacijskom sustavu. Osim toga, očekuje se podrška za Intel DPDK i mogućnost stvaranja virtualnih (TAP) mrežnih sučelja. S time na umu, odabran je Debian Linux 12.4 kao operacijski sustav na kojemu je napravljena naša implementacija.

iptables

iptables je alat na Linux operacijskim sustavima koji omogućuje upravljanje vatrozidom (engl. *firewall*). Predstavlja program koji je stvoren za izvođenje u korisničkom načinu rada, a upravlja ugrađenim *firewall* sustavom u jezgri Linux. Linuxov *firewall* ima mnoge mogućnosti koje uključuju filtriranje paketa, izmjenu podataka, označavanje paketa, uprav-

[9] https://ninja-build.org/
[10] https://www.gnu.org/software/make/
[11] https://www.dpdk.org/

ljanje prometom i prevođenje adresa. Ovaj *firewall* je danas često korišten te ima puno lako dostupnih primjera korištenja za različite primjene. Ostvaren je kao skup tablica (engl. *tables*) koje sadrže lance (engl. *chains*) pravila (engl. *rules*). Dodatno, omogućuje vanjske nadogradnje modulima. Moduli dodaju nove vrste pravila koja proširuju mogućnosti *firewalla*. U novijim jezgrama `iptables` je zamijenjen s `nftables`, no moguće je koristiti korisnički alat `iptables` za upravljanje jednim ili drugim sustavom u jezgri. Za ovaj rad, najbitnija mogućnost je prevođenje mrežnih adresa. `iptables` će biti korišten u usporedbi performansi.

ipfilter

`ipfilter` je jedan of *firewalla* koji je ugrađen u `FreeBSD` operacijski sustav. U dokumentaciji se ponekad i spominje pod kraćim imenom `ipf`. Dostupan je i na drugim `BSD` inačicama poput `NetBSD` i `OpenBSD`, te na operacijskom sustavu `Solaris`. Sintaksa pravila je dosta jednostavna i intuitivna, a pruža slične mogućnosti kao `iptables`. Bitna mogućnost za ovaj rad je mogućnost prevođenja mrežnih adresa. Za postavljanje `NAT` pravila ovog *firewalla* koristi se program `ipnat`. `ipfilter` će biti korišten u usporedbi performansi.

IXIA

`IXIA`[12] je uređaj za testiranje mrežnih uređaja i generiranje mrežnog prometa. Podržava mnoge protokole i mogu se zadati proizvoljne topologije. Zatim se u te topologije smjesti uređaj kojeg testiramo (engl. *Device Under Test*), odnosno `DUT`. Ima mogućnost generiranja velike količine prometa u bezkonekcijskom radu, odnosno radu bez stanja (engl. *stateless*). Osim toga, može i generirati promet protokola s konekcijama, no performanse su tada znatno ograničene. Ovaj uređaj je korišten kao generator prometa za isprobavanje velike količine paketa na ulazima prevoditelja mrežnih adresa.

[12] `http://www.ixiacom.com/`

SR-IOV

Single Root I/O Virtualization (SR-IOV) je tehnologija koja omogućuje virtualizaciju funkcija uređaja. U ovom radu koristi se SR-IOV podrška mrežne kartice. SR-IOV omogućuje fizičkim PCIe (*Peripheral Component Interconnect Express*) uređajima, odnosno PF (engl. *physical functions*) da se prikažu kao više virtualnih PCIe uređaja. Virtualni uređaji, odnosno VF (engl. *virtual functions*) mogu biti korišteni kao i stvarni uređaj - mogu biti direktno predani virtualnim strojevima. Na ovaj način je virtualnim strojevima omogućen direktniji rad s PCIe uređajima i znatno veće performanse.

3 Implementacija

Programsko rješenje je izvedeno pomoću jezika C++ i biblioteke DPDK. Prije pokretanja potrebno je upariti sučelja, odnosno odrediti koja privatna sučelja će biti prevedena u koja javna sučelja. To se radi pomoću konfiguracijske JSON[13] datoteke. Na Slici 2 je prikazan primjer te datoteke.

```
{
  "uuid": "nat-1-uuid",
  "identity": "NAT 1",
  "dashboard": "10.0.2.2",
  "interfaces": [
    {
      "local": {
        "mac": "aa:bb:cc:dd:ee:ff",
        "tap": "enp6s0"
      },
      "internet": {
        "mac": "00:11:22:33:44:55",
        "tap": "enp10s0"
      }
    }
  ]
}
```

Slika 2: Primjer konfiguracije parova sučelja

Nije implementirana dinamička alokacija sučelja. U konfiguraciji se postavljaju parametri kojima identificiramo NAT. Polje `interfaces` je lista parova sučelja: `local` za privatnu stranu prevoditelja, a `internet` za javnu. Sučeljima definiramo samo MAC adrese i imena

[13] https://www.rfc-editor.org/rfc/rfc8259

`TAP` sučelja. Ostatak konfiguracije sučelja i rada prevoditelja se dohvaća od poslužitelja koji je zadan poljem `dashboard`. Osim dohvaćanja konfiguracije, tom poslužitelju se prijavljuje i stanje prevoditelja.

Slika 3 prikazuje sučelje za postavljanje adresa naše implementacije prevoditelja. Polje `Local address` i `Internet address` određuju adresu sučelja uređaja. Implementirani `NAT` ne radi usmjeravanje prometa, nego samo prevođenje adresa. Poljem `Gateway address` određujemo kamo će biti prosljeđivan sav promet privatne strane. Kod povratnog prometa od interneta, nakon prevođenja se odredište na podatkovnom sloju postavlja u ono koje je zabilježeno kod stvaranja zapisa prevođenja. `Public IP pool addresses` je popis mreža iz kojih će biti dodijeljene adrese u koje se prevodi klijentski promet.

Slika 3: Postavljanje adresa sučelja

Za svako sučelje je zadužena jedna jezgra procesora - "jezgra sučelja". Jezgra sučelja prihvaća pakete s tog sučelja, a pakete šalje jedino na upareno sučelje ili na pomoćne jezgre za dodatnu obradu. Za bolje performanse kod asimetričnog prometa potrebno je istražiti korištenje više jezgri po jednom sučelju te više sučelja po jednoj jezgri. Svaka jezgra ima svoje brojače za brojanje preuzetih i poslanih okteta i paketa. Razlog tomu je smanjenje opterećenja na sinkronizaciju priručnih spremnika. Prilikom dohvaćanja brojača sučelja, prikupljaju se i zbrajaju brojači svake jezgre koja šalje ili prima pakete na tom sučelju.

TAP sučelja

U normalnom radu, *driver* koji brine o radu mrežne kartice je zadužan za stvaranje sučelja koja su vidljiva korisničkim programima - sučelja na koja se mogu postaviti adrese i kroz koja korisnički programi mogu slati pakete. DPDK zahtjeva pokretanje posebnih *drivera* koji ne stvaraju takva sučelja i koji omogućuju korisničkom programu potpuno preuzimanje sučelja bez sudjelovanja jezgre. U tom slučaju, drugi korisnički programi neće vidjeti ta sučelja kroz uobičajene mehanizme, a rad programa koji zahtjevaju pristup mreži kroz ta sučelja neće biti moguć. Jezgra Linux omogućuje stvaranje virtualnih (TAP) sučelja iz korisničkih programa korištenjem specijalnog uređaja /dev/net/tun. Na taj način naš program postaje posrednik između DPDK *drivera* i korisničkih mrežnih sučelja. U implementaciji rješenja, za svako DPDK sučelje koje program preuzima stvara se jedno TAP sučelje. Svi su paketi kojima je odredište adresa samog prevoditelja predani pomoćnoj jezgri zaduženoj za upravljanje TAP sučeljima. Ta jezgra takve pakete prosljeđuje na TAP sučelja te ih time prikazuje Linux mrežnom sustavu i korisničkim programima. Osim toga, pomoćna jezgra čita pakete koji su poslani na TAP sučelja od strane Linux jezgre, odnosno drugih korisničkih programa, i šalje ih na DPDK sučelja. Na taj način, omogućen je nesmetan rad ostatka sustava i dozvoljeno je, ukoliko je potrebno, drugim programima korištenje sučelja čijim radom upravlja naš prevoditelj adresa.

Jezgre za prihvaćanje prometa

Jezgre sučelja moraju biti spremne prihvatiti veliku količinu prometa i zbog toga se na njima vrši što manje nepotrebne obrade. Provjerava se ispravnost primljenih paketa te protokol, a zatim se traži klijent i tok (engl. *flow*) kojemu taj paket pripada. Na lokalnoj, odnosno privatnoj strani je korisnik određen IP adresom. Ukoliko se radi o javnoj strani, tok kojemu taj paket pripada je određen pretraživanjem tablice prevođenja za protokol paketa.
Na primjer, za protokol TCP koji dolazi od privatne strane:

1. provjeri se kontrolni zbroj IP i smanji se TTL u IP zaglavlju
2. provjeri se kontrolni zbroj TCP
3. potraži se korisnika čiji je to paket po MAC i IP adresi

4. potraži se postoji li već zapis u tablici prevođenja za taj *flow*

   Ukoliko ne postoji zapis, šalje se pokazivač na taj paket na pomoćnu jezgru koja je zadužena za stvaranje zapisa prevođenja i nastavlja se idući paket obrađivati. Na pomoćnoj jezgri za taj paket:

   (a) potraži se postoji li već zapis u tablici prevođenja za taj *flow* i ako postoji se prevede kako je zabilježeno u tablici
   (b) ukoliko korisnik smije stvoriti još prevođenja, zabilježi se izvorišna adresa i port na privatnoj strani
   (c) prevede se izvorišna adresa i port u neku od slobodnih adresa i portova prevoditelja mrežnih adresa
   (d) ispravi se kontrolni zbroj
   (e) ukoliko je potrebno, primijene se dodatni `ALG`
   (f) paket se šalje na javno sučelje

5. prevede se izvorišna adresa i port kako je zabilježeno u tablici prevođenja
6. ispravi se kontrolni zbroj
7. ukoliko je potrebno, primijene se dodatni `ALG`

   Na primjer, u slučaju `FTP` protokola, potrebno je ispraviti adrese u tekstualnom obliku unutar podatkovnog dijela paketa i pratiti koliko okteta je dodatno ubačeno ili izbačeno iz podataka zbog toga.

8. paket se šalje na javno sučelje

Koraci označeni slovima se odrađuju na pomoćnoj jezgri i nisu sinkronizirani s radom jezgre sučelja. Jedino što znamo je da korak (a) započinje nakon što se 4. korak odradi. Koraci od (c) nadalje su isti kao i koraci 5. nadalje, s razlikom da se odrađuju na pomoćnoj jezgri za stvaranje zapisa prevođenja. Paketi se ne vraćaju od pomoćne jezgre prema jezgri sučelja kako bismo izbjegli nepotrebna komuniciranja između jezgri.
Kod povratnog `TCP` prometa:

1. provjeri se kontrolni zbroj `IP` i smanji se `TTL` u `IP` zaglavlju

2. provjeri se kontrolni zbroj `TCP`
3. potraži se tok kojemu pripada paket po njegovoj `IP` adresi i `TCP` portu
4. ako nije pronađen tok, paket se odbacuje i ako je postavljeno, odgovara se da tok ne postoji
5. ako tok postoji, iz zapisa se čitaju originalna `IP` adresa i `TCP` port privatne strane
6. adresa i port se prevedu u privatnu
7. ispravi se kontrolni zbroj
8. ukoliko je potrebno, primijene se dodatni `ALG`
9. paket se šalje na privatno sučelje

Pomoćna jezgra za stvaranje zapisa prevođenja

Kod stvaranja zapisa prevođenja potrebno je izvesti neke operacije čije je trajanje manje predvidivo. Potrebno je provjeriti informacije o korisniku i njihovim dozvolama, pronaći slobodnu adresu i port iz javnog skupa adresa te stvoriti sam zapis prevođenja. Osim što ovaj postupak može nešto dulje trajati u odnosu na samu obradu zahtjeva - ovaj postupak radi s više memorije nego što je potrebno za samo proslijeđivanje. Da bismo bolje iskoristili priručnu memoriju jezgri i dobili bolje performanse, potrebno je smanjiti količinu priručne memorije s kojom jezgre aktivno rade. S time na umu, stvaranje zapisa prevođenja u jezgrama sučelja nije opcija.

Predavanje paketa od jezgre sučelja prema pomoćnoj jezgri se radi pomoću strukture `rte_ring`.

Provjeravanje tablice prevođenja na jezgri za stvaranje prevođenja

Na prvi pogled provjeravanje tablice prevođenja u pomoćnoj jezgri koja stvara zapise izgleda nepotrebno. To je potrebno jer se može dogoditi da više paketa iz istog toka dođe na jezgru sučelja prije nego što pomoćna jezgra stigne obraditi prvi paket tog toka. U tom će slučaju svi paketi iz tog toka biti predani pomoćnoj jezgri za stvaranje zapisa prevođenja. Kako bismo osigurali da se više paketa iz tog toka ne prevede u više različitih javnih adresa i portova - i time izgubi ideja toka na javnoj strani - potrebno je provjeriti postoji li već zapis prije stvaranja novog zapisa.

Prihvaćanje i slanje paketa

Mrežne kartice imaju više redova (engl. *queue*) od kojih možemo istovremeno s više jezgri dohvaćati pakete na dolaznoj strani (engl. *rx queue*) i više redova na koje možemo predavati pakete za slanje (engl. *tx queue*). Mrežne kartice imaju ograničen broj redova. Više jezgri ne smije istovremeno koristiti isti red, ali različite redove smiju. Kako bi se izbjegla skupa sinkronizacija jezgri, potrebno je iskoristiti redove koje kartica pruža.

Pakete će prihvaćati samo jezgra koja je zadužena za to sučelje. Zbog toga je dovoljan samo jedan red za prihvaćanje paketa.

U implementaciji je postavljeno da su javna i privatna sučelja dodijeljena 1:1. Na svako javno sučelje će moći doći samo paket s jednog privatnog sučelja te isto tako u povratnom smjeru. Osim toga, paket će morati moći poslati i jezgra za stvaranje zapisa prevođenja kako bismo izbjegli nepotrebno slanje paketa između jezgri. Zbog načina na koji je izvedena komunikacija s `TAP` sučeljima i za njih je dodijeljena zasebna jezgra. I ta će jezgra isto morati moći slati pakete na sučelja. Zato je dovoljno napraviti tri reda za slanje paketa na svakom sučelju na javnoj i dva reda na privatnoj strani.

4 Mjerenja

Virtualnim strojevima je dano jedno mrežno sučelje za upravljanje (engl. *management interface*) te dva sučelja namijenjena za podatkovni promet. Sučelja za podatkovni promet su `VF SR-IOV` sučelja.

`VF` sučelja koja su dana virtualnim strojevima su na različitim mrežnim karticama. Fizička sučelja imaju definiranu brzinu od `100 Gb/s`. Korištena mrežna kartica je `Intel E810`. Korišteni procesor na kojemu su mjerene performanse prevoditelja je `Intel Xeon Gold 6430`.

Sva mjerenja su rađena nad virtualnim strojem s jednakim brojem istih jezgri te jednako memorije. Virtualnim strojevima su dane četiri jezgre i 8 GiB memorije.

Na istom sklopovlju je postavljen:

- virtualni stroj s implementiranim rješenjem,
- virtualni stroj za testiranje `iptables` sa sustavom `Debian 12.4`,
- virtualni stroj za testiranje `ipfilter` sa sustavom `FreeBSD 14.0`.

Kako bismo postigli što sličnije okruženje, na sva tri virtualna stroja je namješteno da za svako sučelje koriste samo jedan *rx queue* i jedan *tx queue*.

Funkcionalna mjerenja

Zbog načina na koji je postavljen laboratorij u kojemu su rađena mjerenja, ne postoji mogućnost direktnog spajanja fizičkog stroja (poput laptopa) na privatnu stranu prevoditelja te isprobavanja njegovog ponašanja na taj način. Zato je stvoren virtualni stroj koji je spojen na privatnu stranu prevoditelja i s njega je isproban pristup internetu. Osim toga, isproban je i pristup internetu te "surfanje" raznim stranicama tako što je na tom virtualnom stroju s privatne strane prevoditelja upaljen `SOCKS5` posredni server, a laptopom je ostvaren pristup prema internetu preko posrednika. Isprobano je ponašanje stranica uz podešavanje nekih parametara poput vremena isteka `TCP` sjednica. Većina isprobanih stranica je pokazala impresivnu otpornost na prekidanje sjednica. Na primjer, stranica `Youtube`[14] je uz *timeout* od

[14] `https://www.youtube.com`

1 sekunde normalno prikazivala videe - samo bi poneke slike ostale neučitane. Isprobane su i stranice raznih web portala te `Speedtest`[15].

Mjerenja performansi

Za brzu provjeru stabilnosti prevoditelja prilikom obrade velike količine prometa korišten je alat `iperf`. Mjerenja performansi su rađena pomoću uređaja `IXIA`. Slika 4 prikazuje topologiju za testiranje izrađenu u alatu `IXIA`. Prikazan ja slučaj s 1000 klijenata i 1000 poslužitelja. Na privatnoj strani je postavljeno da svaki klijent ima zasebnu adresu na podatkovnom sloju. Na javnoj strani je samo jedna adresa podatkovnog sloja s 1000 mrežnih adresa. Tako je postavljeno zato što naša implementacija nema tablicu usmjeravanja, nego sav promet za javnu stranu prosljeđuje istom uređaju.

Slika 4: `IXIA` topologija za mjerenje performansi prevoditelja

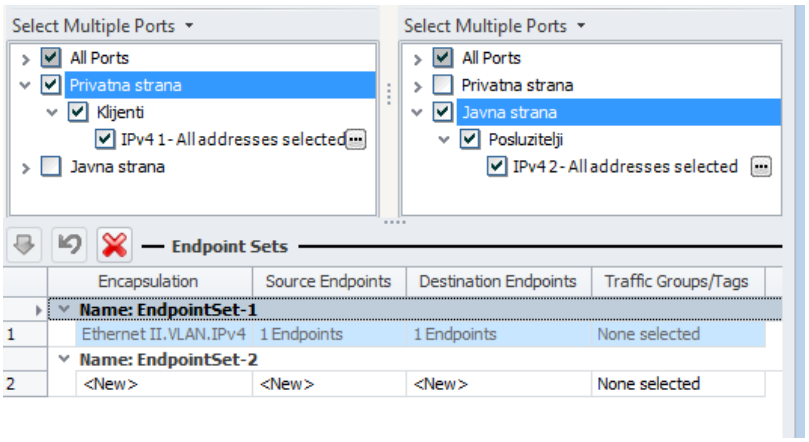
Definirani su `IXIA` scenariji koji mjere propusnost prevoditelja mrežnih adresa. Isprobani scenariji su:

1. 1 klijent, 1 poslužitelj
2. 10 klijenata, 1 poslužitelj
3. 100 klijenata, 1 poslužitelj
4. 1000 klijenata, 1 poslužitelj
5. 1 klijent, 10 poslužitelja
6. 1 klijent, 100 poslužitelja
7. 1 klijent, 1000 poslužitelja

[15] https://speedtest.net/

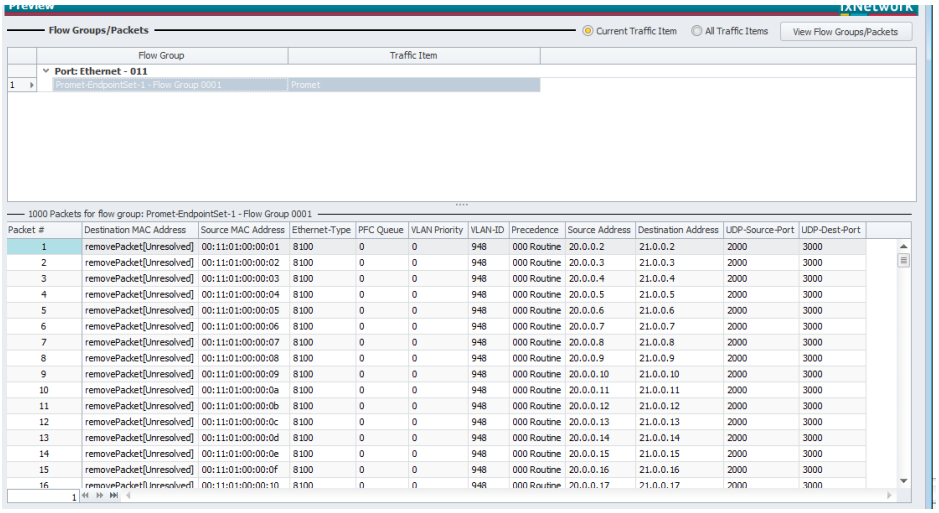
8. 10 klijenata, 10 poslužitelja
9. 100 klijenata, 100 poslužitelja
10. 1000 klijenata, 1000 poslužitelja

Slika 5 prikazuje postavljanje toka prometa koji je korišten u testiranju performansi.



Slika 5: `IXIA` definicija toka prometa

Slika 6 prikazuje generirane pakete koji će biti slani prilikom mjerenja performansi.



Slika 6: prikaz generiranih paketa

Izvršena su mjerenja s paketima veličine `100`, `500` i `1500` okteta. Mjerenja `1500` oktetnim paketima su u nekim slučajevima bila ograničena brzinom fizičkog sučelja mrežne kartice.

Propusnosti prevoditelja su izražene mjernom jedinicom `Mpps` (milijuna paketa po sekundi). Nakon mjerenja je bilo vidljivo da veličina paketa nije značajno utjecala na performanse prevoditelja. Radi lakšeg pretvaranja podataka u mjernu jedinicu `Gb/s`, na grafovima su s desne strane dodane oznake s odgovarajućim veličinama.

Podaci

Jednosmjeran promet

Korišten je `UDP` promet nepoznatih protokola s visokim portovima i slučajnog sadržaja u podatkovnom dijelu.

1. Naša implementacija

   (a) Veličina paketa: `100 okteta`

Tablica 1: Propusnost `NAT`-a s povećanjem broja klijenata

broj izvorišta	propusnost (Mpps)
1	10.093
10	9.603
100	6.725
1000	6.463

Tablica 2: Propusnost `NAT`-a s povećanjem broja poslužitelja

broj odredišta	propusnost (Mpps)
1	10.092
10	10.081
100	10.093
1000	10.113

Tablica 3: Propusnost `NAT`-a s povećanjem broja klijenata i poslužitelja

broj izvorišta i odredišta	propusnost (Mpps)
1	10.093
10	9.440
100	6.985
1000	6.476

(b) Veličina paketa: `500 okteta`

Tablica 4: Propusnost `NAT`-a s povećanjem broja klijenata

broj izvorišta	propusnost (Mpps)
1	10.046
10	9.599
100	6.719
1000	6.409

Tablica 5: Propusnost `NAT`-a s povećanjem broja poslužitelja

broj odredišta	propusnost (Mpps)
1	10.046
10	9.861
100	10.085
1000	9.910

Tablica 6: Propusnost `NAT`-a s povećanjem broja klijenata i poslužitelja

broj izvorišta i odredišta	propusnost (Mpps)
1	10.041
10	9.224
100	6.952
1000	6.472

(c) Veličina paketa: `1500 okteta`

Tablica 7: Propusnost `NAT`-a s povećanjem broja klijenata

broj izvorišta	propusnost (Mpps)
1	8.224
10	8.224
100	6.714
1000	6.439

Tablica 8: Propusnost `NAT`-a s povećanjem broja poslužitelja

broj odredišta	propusnost (Mpps)
1	8.224
10	8.223
100	8.223
1000	8.223

Tablica 9: Propusnost `NAT`-a s povećanjem broja klijenata i poslužitelja

broj izvorišta i odredišta	propusnost (Mpps)
1	8.224
10	8.224
100	6.949
1000	6.469

2. `iptables`

Prije svega, postavljamo da se koristi samo jedan red za prihvat paketa. Time želimo napraviti što sličniju okolinu za sve testirane implementacije. Ukoliko se koristi jedan red za prihvat paketa, neće biti korišteno više jezgri za prihvat prometa. Na taj način sve implementacije koriste samo jednu jezgru i rezultati se lakše mogu usporediti. Naredba kojom je osigurano da se koristi samo jedan red za prihvat paketa:

```
ethtool -X enp6s0 equal 1
```

Naredba kojom je uključeno prevođenja adresa na sučelju `enp10s0`:

```
iptables -t nat -A POSTROUTING -o enp10s0 -j MASQUERADE
```

(a) Veličina paketa: `100 okteta`

Tablica 10: Propusnost `iptables` s povećanjem broja klijenata

broj izvorišta	propusnost (Mpps)
1	1.151
10	1.104
100	1.063
1000	1.064

Tablica 11: Propusnost `iptables` s povećanjem broja poslužitelja

broj odredišta	propusnost (Mpps)
1	1.150
10	0.996
100	0.964
1000	0.958

Tablica 12: Propusnost `iptables` s povećanjem broja klijenata i poslužitelja

broj izvorišta i odredišta	propusnost (Mpps)
1	1.151
10	0.997
100	0.995
1000	0.961

(b) Veličina paketa: `500 okteta`

Tablica 13: Propusnost `iptables` s povećanjem broja klijenata

broj izvorišta	propusnost (Mpps)
1	1.151
10	1.102
100	1.062
1000	1.063

Tablica 14: Propusnost `iptables` s povećanjem broja poslužitelja

broj odredišta	propusnost (Mpps)
1	1.151
10	1.001
100	0.963
1000	0.962

Tablica 15: Propusnost `iptables` s povećanjem broja klijenata i poslužitelja

broj izvorišta i odredišta	propusnost (Mpps)
1	1.152
10	1.001
100	0.999
1000	0.963

(c) Veličina paketa: `1500 okteta`

Tablica 16: Propusnost `iptables` s povećanjem broja klijenata

broj izvorišta	propusnost (Mpps)
1	1.150
10	1.106
100	1.064
1000	1.069

Tablica 17: Propusnost `iptables` s povećanjem broja poslužitelja

broj odredišta	propusnost (Mpps)
1	1.150
10	0.999
100	0.965
1000	0.963

Tablica 18: Propusnost `iptables` s povećanjem broja klijenata i poslužitelja

broj izvorišta i odredišta	propusnost (Mpps)
1	1.151
10	1.002
100	0.998
1000	0.962

3. `ipnat`

Ograničavanje na jedan red za prihvat paketa je ostvareno podešavanjem varijable `override_nrxqs` sustava `iflib` na vrijednost 1.

Naredba kojom je uključeno prevođenja adresa na sučelju `iavf1`:

```
echo 'map iavf1 0/0 -> 0/32' | ipnat -f -
```

(a) Veličina paketa: `100 okteta`

Tablica 19: Propusnost `ipnat` s povećanjem broja klijenata

broj izvorišta	propusnost (Mpps)
1	0.873
10	0.866
100	0.865
1000	0.870

Tablica 20: Propusnost `ipnat` s povećanjem broja poslužitelja

broj odredišta	propusnost (Mpps)
1	0.873
10	0.879
100	0.879
1000	0.878

Tablica 21: Propusnost `ipnat` s povećanjem broja klijenata i poslužitelja

broj izvorišta i odredišta	propusnost (Mpps)
1	0.873
10	0.865
100	0.865
1000	0.869

(b) Veličina paketa: `500 okteta`

Tablica 22: Propusnost `ipnat` s povećanjem broja klijenata

broj izvorišta	propusnost (Mpps)
1	0.855
10	0.844
100	0.842
1000	0.850

Tablica 23: Propusnost `ipnat` s povećanjem broja poslužitelja

broj odredišta	propusnost (Mpps)
1	0.855
10	0.859
100	0.853
1000	0.850

Tablica 24: Propusnost `ipnat` s povećanjem broja klijenata i poslužitelja

broj izvorišta i odredišta	propusnost (Mpps)
1	0.855
10	0.851
100	0.846
1000	0.848

(c) Veličina paketa: `1500 okteta`

Tablica 25: Propusnost `ipnat` s povećanjem broja klijenata

broj izvorišta	propusnost (Mpps)
1	0.844
10	0.842
100	0.839
1000	0.849

Tablica 26: Propusnost `ipnat` s povećanjem broja poslužitelja

broj odredišta	propusnost (Mpps)
1	0.844
10	0.854
100	0.852
1000	0.853

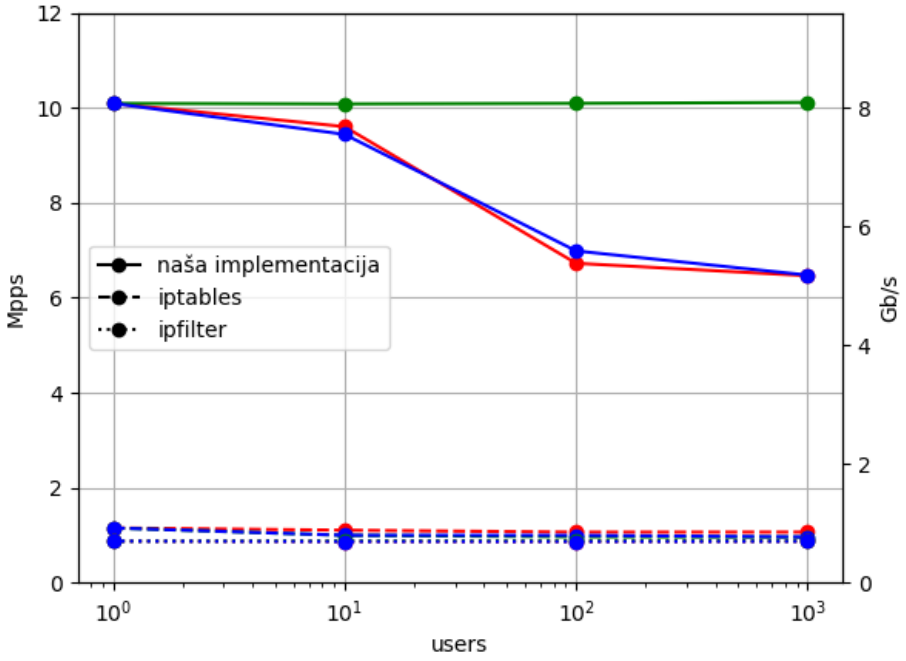
Tablica 27: Propusnost `ipnat` s povećanjem broja klijenata i poslužitelja

broj izvorišta i odredišta	propusnost (Mpps)
1	0.844
10	0.841
100	0.845
1000	0.850

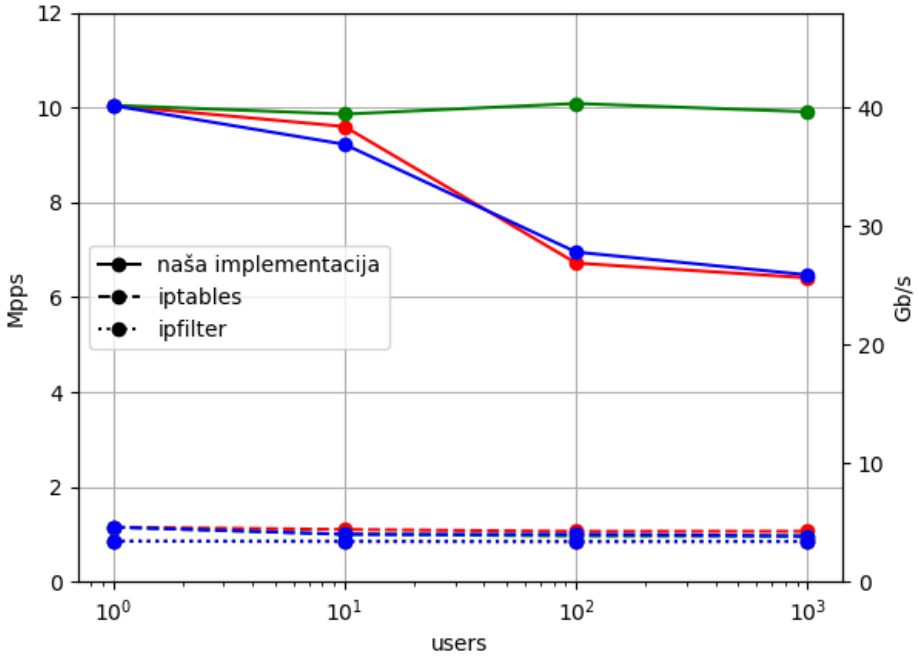
Grafovi

Na grafovima je različitim bojama označena promjena različite varijable. Crvenom bojom je označena promjena količine izvorišta paketa. Zelenom bojom je označena promjena količine odredišta paketa. Plavom bojom je označena promjena količine izvorišta i odredišta paketa. Punom linijom su označena mjerenja našeg prevoditelja. Isprekidanom linijom su označena mjerenja `iptables` prevoditelja. Točkama su označena mjerenja `ipnat` prevoditelja.
Na grafovima *y* os označava propusnost prevoditelja. S lijeve strane su dane oznake s mjernom jedinicom milijuna paketa u sekundi. S desne strane je označena propusnost u gigabitima po sekundi. Os *x* je na logaritamskoj skali te označava broj korisnika.

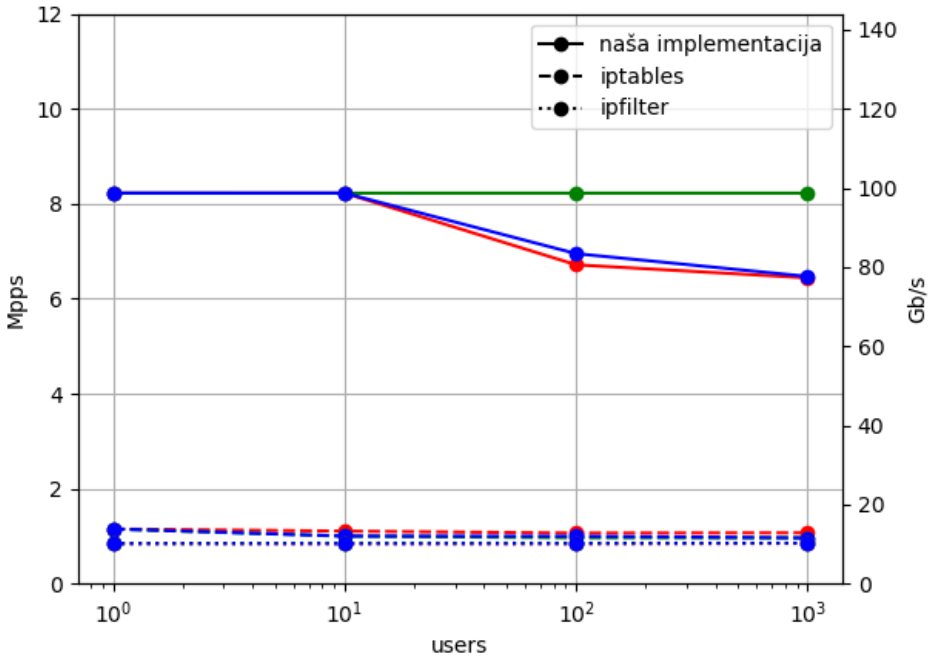
1. Paketi veličine 100 okteta



2. Paketi veličine 500 okteta



3. Paketi veličine 1500 okteta



Kod paketa veličine 1500 okteta dolazimo do ograničenja propusnosti sučelja od `100 Gb/s`. Zbog toga je manje izražen pad propusnosti naše implementacije prilikom povećanja broja klijenata.

Općenito je u našoj implementaciji vidljiv pad propusnosti s povećanjem klijenata. Razlog tome je što s povećanjem performansi utjecaj priručnih memorija postaje sve izraženiji. Kad je pušten promet od samo jednog ili deset klijenata, većina podataka potrebnih za obradu prometa je dostupna u bržoj priručnoj memoriji. Promet `100` i `1000` klijenata zahtjeva više memorije te se prelazi veličina priručnih spremnika što se očituje kao pad performansi.

`iptables` je u svim mjerenjima održavao propusnost od oko `1 Mpps` uz slab pad propusnosti prilikom povećavanja količine klijenata. `ipnat` je u svim mjerenjima održavao propusnost od oko `0.85 Mpps` uz slab pad propusnosti prilikom povećavanja količine klijenata.

Naše rješenje je znatno brže, ali je i znatno specijaliziranije. Radimo pretpostavku da

se jedno sučelje s privatne i jedno javne strane uvijek povezuju i da se promet za svaki par sučelja odvojeno obrađuje. Osim toga, ne možemo s paketima raditi puno toga što mrežni podsustavi operacijskih sustava `Linux` i `FreeBSD` mogu.

Postavljanje kontrolnog zbroja je moguće prepustiti mrežnoj kartici i zbog toga nije potrebno čitati niti mijenjati pakete dublje od nekoliko početnih zaglavlja. Povećanje veličine paketa nije gotovo nimalo utjecalo na ponašanje prevoditelja.

Dvosmjeran promet

Korišteni generator prometa `IXIA` nema mogućnost dinamičkog stvaranja paketa. Ovaj generator može raditi neka ograničena mjerenja takvih protokola jednim drugim programom, no tim programom ne može mjeriti performanse. Generiranje popisa i sadržaja paketa se radi samo prilikom pokretanja mjerenja. Generiraju se odlazni paketi i određuje se na temelju čega će dolazni paketi biti prihvaćani. Uređaji koje mjerimo dinamički odabiru adrese i portove na javnoj strani u koje će prevoditi promet s privatne strane. `IXIA` nema mogućnost korištenja te dinamičke informacija dolaznih paketa u svrhu generiranja povratnog prometa. Zbog toga, visoke performanse su mjerene samo u smjeru privatne strane prema javnoj strani prevoditelja adresa.

Zaključak

NAT je tehnika koja će se zasigurno koristiti dokle god se koristi IPv4. Ova mrežna funkcija je izvediva na razne načine i ugrađena je u mnoge današnje operacijske sustave.

U radu je pokazano da je moguća efikasna implementacija prevođenja mrežnih adresa pomoću platforme DPDK. Isproban je pristup u kojemu jedna procesorska jezgra obrađuje promet jednog mrežnog sučelja. Na taj način je korišten samo jedan red za prihvat paketa od mrežne kartice. Moguće je da je zbog toga brzina prihvata ograničena. Osim toga, obrada paketa sučelja na jednoj jezgri isto ograničava performanse implementacije. Ostvarena je propusnost oko 10 Mpps s jednim klijentom i oko 6.4 Mpps s tisuću klijenata.

Unatoč tome što ne mogu raditi pretpostavke i optimizacije koje su u našoj implementaciji korištene, iptables i ipnat pokazuju impresivne rezultate. Implementirani NAT je pokazao bolje performanse od jer je usko specijaliziran. U nekim slučajevima visoke performanse puno više znače od široke funkcionalnosti.

Sažetak

Ovaj rad započinje kratkim uvodom i motivacijom za korištenje NAT-a. Opisuje ulogu i različite vrste NAT-a. Daje se popis i kratki opis tehnologija koje su korištene u praktičnom dijelu rada. Opisana je implementacija praktičnog dijela i objašnjeno ponašanje algoritma obrade paketa. Navedene su neke korištene tehnike optimizacije. Izvršena su funkcionalna mjerenja i mjerenja performansi praktičnog dijela. Prikazani su rezultati i uspoređeni s poznatim prevoditeljima iptables i ipnat. Na kraju rada je dan kratki zaključak.

Ključne riječi: prevoditelj mrežnih adresa, DPDK, visoke performanse, iptables, ipnat

Summary

This paper begins with a brief introduction and motivation for the use of NAT. It describes the role and different types of NAT. A list and brief description of the technologies used in the practical part of the paper are provided. The implementation of the practical part is described and the behavior of the packet processing algorithm is explained. Some optimization techniques used are described. Functional measurements and performance evaluations of the practical part are conducted. The results are presented and compared with well-known translators iptables and ipnat. In conclusion, a brief summary is given.

Keywords: NAT, DPDK, high performance, iptables, ipnat

Skraćenice

ALG	Application-level gateway	
BSD	Berekley Software Distribution	
CGNAT	Carrier-grade NAT	NAT za telekomunikacijske sustave
DPDK	Data Plane Development Kit	
DUT	Device under test	uređaj kojeg testiramo
FTP	File Transfer Protocol	
Gb/s	Gigabits per second	Gigabita u sekundi
IP	Internet Protocol	
IPv4	Internet Protocol version 4	IP protokol verzija 4
IPv6	Internet Protocol version 6	IP protokol verzija 6
JSON	JavaScript Object Notation	
MAC	medium access control	
Mpps	Million packets per second	milijuna paketa u sekundi
NAT	Network address translation	prevođenje mrežnih adresa
PCIe	Peripheral Component Interconnect Express	
PF	Physical function	Fizički uređaj
SR-IOV	Single Root I/O Virtualization	
TCP	Transmission Control Protocol	
TTL	Time-to-live	
UDP	User Datagram Protocol	NAT za telekomunikacijske sustave
VF	Virtual function	Virtualizarana funkcija uređaja