Analiza performansi load balancera korištenjem mrežnog simulatora

Slavić, Filip

Sveučilište Jurja Dobrile u Puli

Fakultet informatike u Puli

FILIP SLAVIĆ

Analiza performansi load balancera korištenjem mrežnog simulatora

ZAVRŠNI RAD

Pula, rujan, 2024. godine

Sveučilište Jurja Dobrile u Puli

Fakultet informatike u Puli

FILIP SLAVIĆ

Analiza performansi load balancera korištenjem mrežnog simulatora

ZAVRŠNI RAD

JMBAG: 0303094753, redovan student

Studijski smjer: Informatika

Kolegij: Mrežni sustavi

Mentor: izv. prof. dr. sc. Siniša Sovilj

Pula, rujan, 2024. godine

Sadržaj

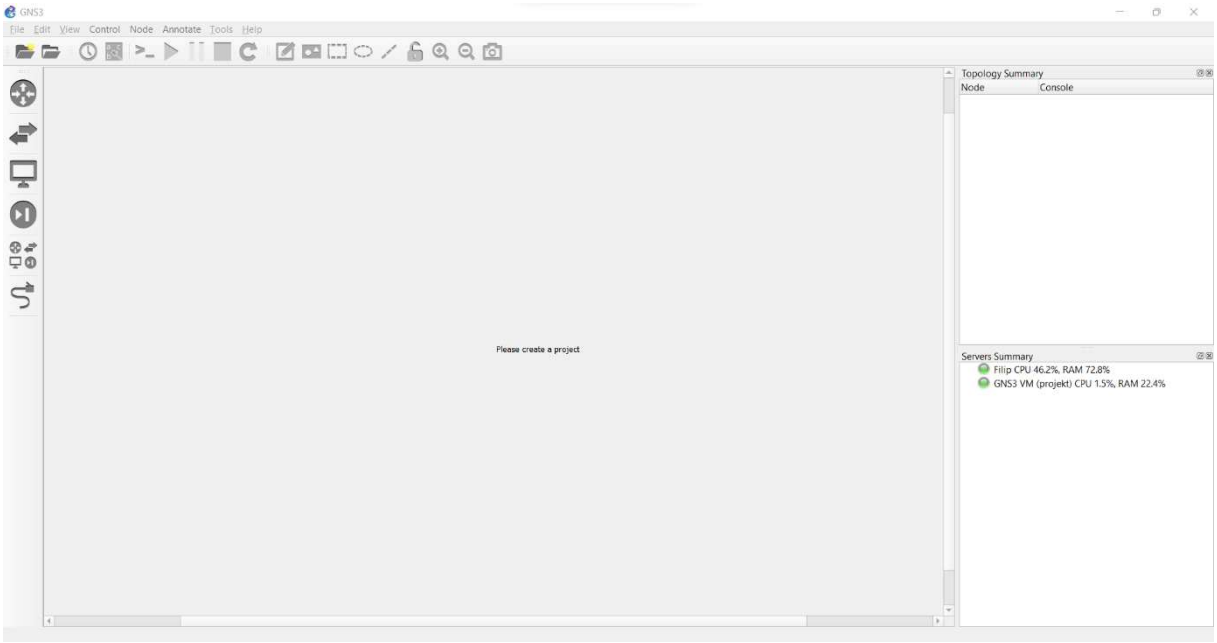
1. Uvod

1.1. Mrežni simulatori

Network Simulator (NS) jednostavno je diskretni alat za mrežnu simulaciju vođen događajima za proučavanje dinamičke prirode komunikacijskih mreža (Mohammad S. Obaidat et. al 2015). Jedan od glavnih razloga korištenja mrežnih simulatora je taj što štede vrijeme i resurse pri testiranju protokola na stvarnoj mreži. Bez potrebe za skupom fizičkom infrastrukturom, oni omogućuju inženjerima, istraživačima i programerima repliciranje složenih mrežnih topologija i procesa u kontroliranom okruženju. To omogućuje uočavanje i rješavanje mogućih problema prije nego što se mreža postavi u stvarnom svijetu. Sposobnost ponavljanja i fleksibilnosti jedna je od najznačajnijih prednosti korištenja mrežnih simulatora (Gomez, J. et al 2023). Simulacije se mogu ponavljati s različitim postavkama za procjenu ponašanja mreže u različitim scenarijima. Inženjeri, na primjer, mogu testirati kako se mreža ponaša pod velikim opterećenjem, kako različiti protokoli međusobno komuniciraju i kako sigurnosne mjere utječu na ukupnu izvedbu mreže, također ih koriste za ispravnu analizu i optimizaciju mrežnih protokola te testiranje rješenja koja povećavaju performanse i pouzdanost mreže. Može se reći da su mrežni simulatori jedni od značajnijih alata u mrežnoj tehnologiji.

1.2. Gns3 simulator

Jedan od najpopularnijih mrežnih simulatora je GNS3 (Graphical Network Simulator-3 GNS3, ili Graphical Network Simulator 3, je grafički mrežni simulator otvorenog koda, licenciran GNU General Public License (GPL), razvijen u Pythonu koji omogućuje simulaciju kompliciranih mreža. Može se besplatno preuzeti s interneta. GNS3 koriste stotine tisuća mrežnih inženjera diljem svijeta za emulaciju, konfiguraciju, testiranje i rješavanje problema virtualnih i stvarnih mreža. GNS3 omogućuje pokretanje male topologije koja se sastoji od samo nekoliko uređaja na vašem prijenosnom računalu, do onih koji imaju mnogo uređaja smještenih na više poslužitelja ili čak smještenih u oblaku (Gns3, 2024).



Slika 1 Grafičko korisničko sučelje (GUI).

Izvor: Samostalni rad

GNS3 pruža grafičko korisničko sučelje (GUI) jednostavno za korištenje za postavljanje mrežnih komponenti u virtualni stroj koji pokreće isti operativni sustav kao i izvorna mrežna komponenta. Ova tehnika omogućuje brzu i učinkovitu konstrukciju mrežnih scenarija, čineći GNS3 izvrsnim alatom za obrazovanje, laboratorijske vježbe i pripremu mrežnih certifikata kao što su CCNA, CCNP i CCIE. (Wikipedia, 2024). GNS3 je posebno koristan za mrežne inženjere i studente budući da podržava širok raspon mrežnih uređaja, uključujući usmjerivače, preklopnike i vatrozide, te se može integrirati s alatima kao što su Wireshark za analizu mrežnog prometa i SolarWinds za upravljanje mrežom. GNS3 je virtualizacijska platforma koja se sastoji od tri softverska programa koji rade na uobičajenom PC hardveru i može se instalirati na Microsoft Windows, Linux i MAC operativne sustave.

1.3. Reverse proxy load balancer

Reverse proxy balanser opterećenja je komponenta u modernim mrežnim topologijama koja se nalazi između korisnika i pozadinskih poslužitelja, točnije iza vatrozida na rubu privatne mreže radi optimizacije distribucije mrežnog prometa, jačanja sigurnosti i poboljšanja ukupnih mrežnih performansi. Kao reverse proxy, ova komponenta raspodjeljuje promet među brojnim poslužiteljima kako bi se osigurao stabilan protok prometa i spriječilo preopterećenje bilo kojeg pojedinačnog poslužitelja.

Reverse proxy balanser opterećenja poboljšava sigurnost filtriranjem prometa, autentifikacijom korisnika i pružanjem zaštite od DDoS napada, čime se podiže njihov rang u sigurnosti za određenu mrežu. To je tako jer reverse proxy balanser radi kao prvi sigurnosni obrambeni sustav pri čemu proces filtrira neželjene zahtjeve i skriva stvarne IP adrese pozadinskih poslužitelja. Zbog toga je napadaču znatno teže pokrenuti napad na stvarne poslužitelje koji se koriste u stvarnim DDoS napadima. Kibernetički kriminalac može ciljati samo reverse proxy na primjer, Cloudflareov CDN opterećen dodatnom sigurnošću i više resursa za obranu od kibernetičke prijetnje. Stoga reverzni proxy balanseri opterećenja nude višeslojnu zaštitu koja štiti mrežu od svih vrsta sigurnosnih prijetnji (Cloudflare, 2024).

Hypertext Transfer Protocol (HTTP) sesije mogu se usmjeravati pomoću reverse proxyja na različite načine i mjesta. Opterećenje se može distribuirati putem reverse proxyja na način koji optimizira iskustvo krajnjeg korisnika. Osim toga, uravnoteženje opterećenja stvara funkcionalniju i učinkovitiju mrežu.

Još jedna važna sigurnosna značajka reverse proxy balansera opterećenja je SSL završetak. Bavi se sigurnosnim stvarima, dešifriranjem podataka prije nego što se proslijede. To znači da se poslužitelji fokusiraju samo na isporuku sadržaja, što pomaže da sve radi brže i bolje (System Design School, 2023).

1.3.1. HTTP

HTTP je protokol koji programima omogućuje komunikaciju preko World Wide Weba. Protokol ima nekoliko primjena, no najpoznatiji je po omogućavanju dvosmjerne komunikacije između web preglednika i web poslužitelja te prijenos HTML stranica, fotografija, videa i drugih materijala (Brian Totty et. al 2002). On je jedan čestih jezika moderne globalne mreže. HTTP je važan u balansiranju mrežnog opterećenja jer je većina prometa koji se prenosi kroz load balancer upravo HTTP promet. Kada korisnik pošalje HTTP zahtjev, revese proxy balanser

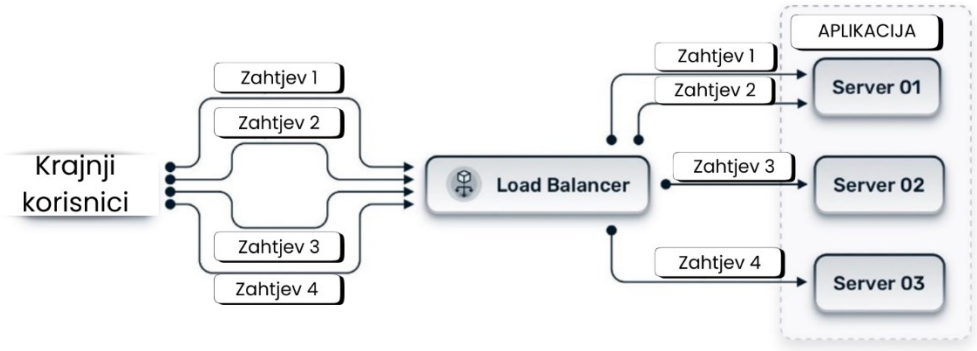
opterećenja ga prvo obrađuje prije nego što ga preusmjeri na jedan od pozadinskih poslužitelja, ovisno o korištenoj metodi balansiranja opterećenja. Međutim, jedan od najvažnijih značajki HTTP-a u kontekstu reverse balansera opterećenja je mogućnost izmjene HTTP zaglavlja. Reverse proxy može dodati, ukloniti ili promijeniti HTTP zaglavlja kako bi osigurao ispravnu isporuku prometa ili pružio dodatnu sigurnost.

1.4. Algoritmi uravnoteženja

Tehnika i postupak poznat kao balansiranje opterećenja koristi uređaj temeljen na mreži za raspodjelu prometa stranice na više poslužitelja (Bourke, Tony 2001). Neki od najčešće korištenih algoritama za uravnoteženje opterećenja uključuju:

1.5. Round robin

Round-Robin algoritam jedan je od najjednostavnijih algoritama za uravnoteženje opterećenja koji se koriste u posljednje vrijeme. Koristi kružni popis i pokazivač na zadnji odabrani poslužitelj za donošenje odluke o otpremi. Šalje korisničke zahtjeve web poslužitelju od prvog web poslužitelja do posljednjeg s naredbom (Z. Xu and X. Wang 2015) . Tehnika balansiranja opterećenja s RR algoritmom jedna je od najklasičnijih tehnika i još uvijek se široko koristi. Tehnika je vrlo jednostavna. Koncept je distribuirati klijentske zahtjeve preko poslužitelja (A. Erzurumluoğlu 2018). RR load balancer prosljeđuje klijentske zahtjeve svakom poslužitelju jedan po jedan (S. Abdallah et. al 2012). Nakon što dođe do kraja popisa dostupnih poslužitelja, vraća se i ponovno počinje postavljati klijentske zahtjeve s prvog poslužitelja. Glavna prednost ovog algoritma je njegova jednostavnost implementacije. Međutim, u situaciji kada zahtjevi za opterećenjem jako variraju, nije u mogućnosti učinkovito distribuirati opterećenja (L. Wang and G. Lu 2016). Tako je i u sljedećoj ilustraciji prikazan round robin algoritam.

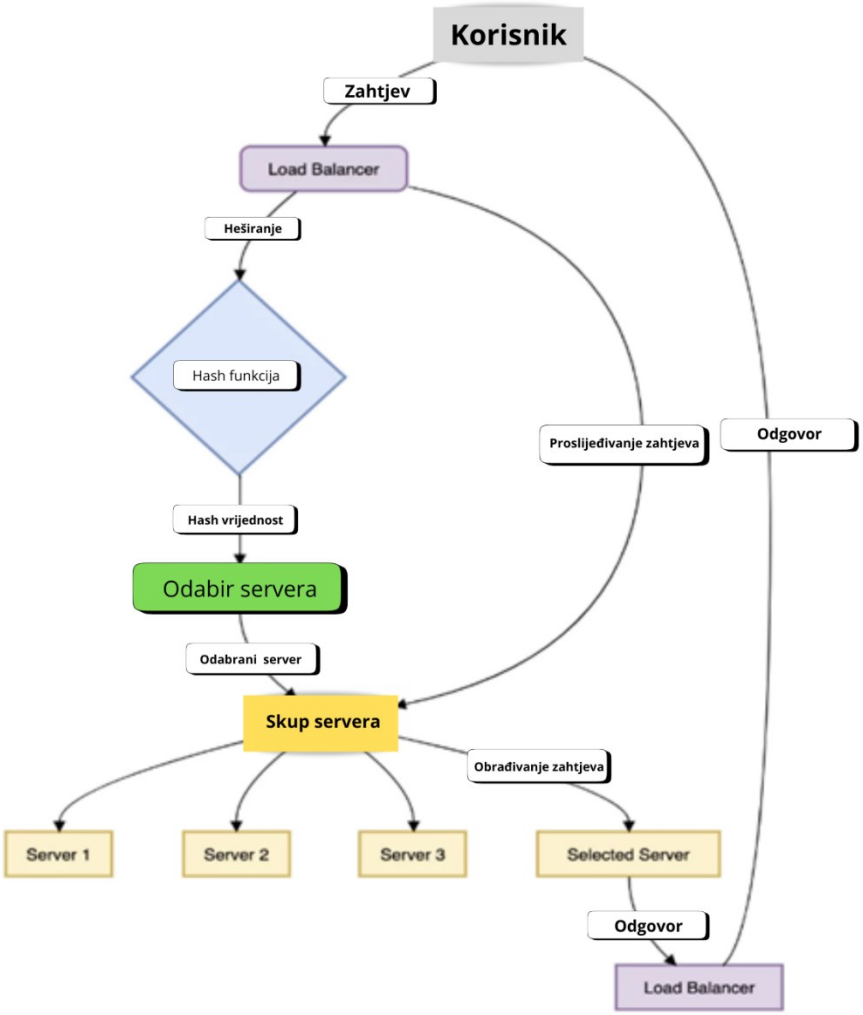


Slika 2 Round robin algoritam.

Izvor : https://traefik.io/glossary/round-robin-load-balancing/

1.6. Ip hash

Algoritam se koristi u balansiranju hash opterećenja za stvaranje jedinstvenog hash ključa pomoću izvorne i odredišne IP adrese klijenta i poslužitelja. Klijent se ovim ključem dodjeljuje određenom poslužitelju. Ova tehnika balansiranja opterećenja može jamčiti da je klijent preusmjeren na isti poslužitelj koji je prethodno korišten budući da se ključevi mogu ponovno generirati u slučaju da je sesija oštećena. Ovo je korisno ako je važno da se klijenti povežu s aktivnim sesijama nakon prekida i ponovnog povezivanja (Ju-Yeon Jo and Yoohwan Kim 2004). Ova metoda također uzima u obzir parametar težine prilikom distribucije hasha (DeJonghe, Derek 2020).



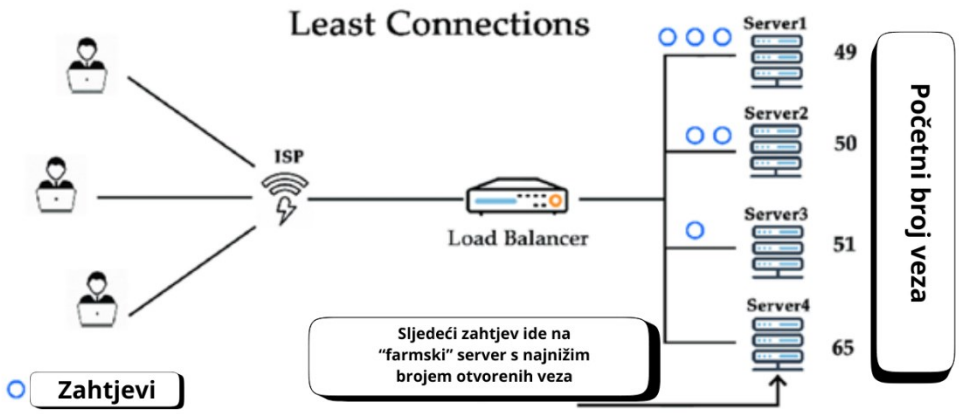
Slika 3 IP hash algoritam.

Izvor: https://webhostinggeeks.com/blog/what-is-ip-hash/

1.7. Least connections

Koristeći tehniku poznatu kao "least connections algoritam“ Nginx usmjerava ulazne zahtjeve na pozadinski poslužitelj koji ima najmanje aktivnih veza u trenutku podnošenja zahtjeva. Kako bi se zajamčilo da su zahtjevi ravnomjerno raspoređeni među svim poslužiteljima, nastoji se podijeliti radno opterećenje na temelju trenutnog radnog opterećenja poslužitelja. Zahtjevi se usmjeravaju prema poslužitelju s najmanje aktivnih veza, što mu omogućuje dinamičku prilagodbu trenutnom stanju svakog poslužitelja. Time se izbjegava preopterećenje poslužitelja i optimizira korištenje resursa za cijeli skup poslužitelja. Ako vaša aplikacija održava stalne veze, kao što su WebSocket ili stalne HTTP veze, tu je least connection algoritam onda koristan.

Budući da ove veze imaju kapacitet održavanja resursa poslužitelja tijekom dugih vremenskih razmaka, njihova podjela među poslužiteljima može pomoći da neki poslužitelji ostanu bez resursa (Chyrvon, Andrii et. al 2023). Ispod se nalazi ilustracija least connections algoritma.



Slika 4 Least connections algoritam.

Izvor: Alankar, Bhavya et. al 2020

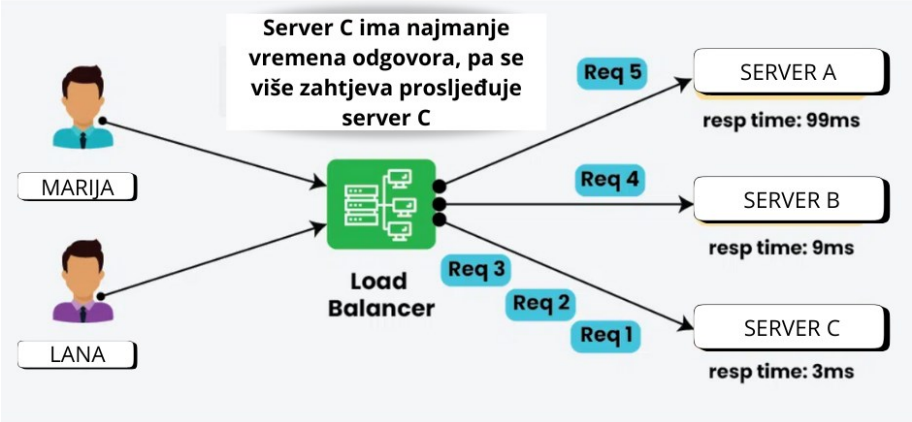
1.8. Weighted round robin

Ovaj pristup dodjeljuje težinu za svaki put, zatim distribuira zahtjeve sekvencijalno s obzirom na dodijeljene težine (Albowarab, Mustafa et. al 2018). Veće vrijednosti težine dodjeljuju se onim poslužiteljem s većim kapacitetom ili performansama, što implicira da će primiti veći broj zahtjeva u usporedbi s poslužiteljima nižeg kapaciteta. Weighted round robin tako puno bolje iskorištava resurse u heterogenim okruženjima gdje se sustavi razlikuju po svojim sposobnostima (Afrianto, Yuggo et. al 2018).

1.9. Least response time

Least response time algoritam je ukupno vrijeme potrebno poslužitelju za obradu dolaznog zahtjeva i slanje odgovora. Kombinirajući vrijeme odgovora poslužitelja s aktivnim vezama, metodom najmanjeg vremena odgovora moglo se utvrditi koji bi poslužitelj bio najbolji. Ovaj

algoritam koriste balanseri opterećenja kako bi osigurali da svaki korisnik dobije bržu uslugu (Harjanti, Trinugi et. al 2022).



resp time = vrijeme odgovora

req = zahtjev

Slika 5 Least response time algoritam.

Izvor: https://www.geeksforgeeks.org/load-balancing-algorithms/

2. METODOLOGIJA

Procjena performansi u simuliranom mrežnom okruženju zahtjeva niz metodologija koje koriste alate uključujući GNS3 mrežni simulator, NGINX reverse proxy kao balanser opterećenja, alate za usporedbu wrk, wireshark mrežni analizator, među ostalim koji se pokazao kao učinkovit alat otvorenog koda u proučavanju mrežnih paketa i njihovog ponašanja. Ovi elementi i procesi istraženi su u nastavku.

2.1. Topologija mreže

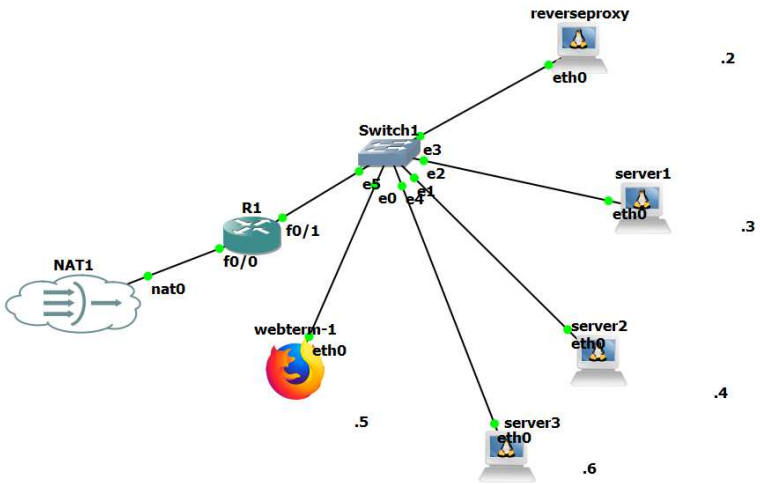
Topologija korištena u ovom radu prikazana je na slici te predstavlja jednostavnu mrežu sa routerom, switch-em i više servera povezanih na lokalnu mrežu. U mreži ove topologije korišteni su uređaji kao router (R1), switch (Switch1) te više servera sa različitim funkcijama. U osnovi, ova je mrežna arhitektura osmišljena da služi kao testna platforma za različite tehnike uravnoteženja mrežnog prometa i procjenjuje kako se te metode ponašaju pod različitim opterećenjima. Povezivanje sa internetom nam je omogućilo instaliranje wrk alata kojeg ćemo koristiti kao alat za benchmark.

Topologija uključuje sljedeće ključne elemente koji podržavaju simulaciju scenarija koji uključuju balanser opterećenja. Router (R1) je središnja točka koja pomaže povezati više uređaja s internetom i povezati uređaje međusobno (Cisco, 2024). Njegov f0/0 port povezuje ga s NAT uređajem, NAT1, dok njegov f0/1 port vodi do Switch1, koji zatim distribuira promet na druge uređaje unutar mreže.

Poslužitelji, koji oponašaju stvarno okruženje, povezani su s preklopnikom i pokazuju kako balanser opterećenja raspoređuje dolazne zahtjeve. Reverse proxy djeluje kao balanser opterećenja koji transparentno predaje zahtjeve drugom poslužitelju (Reese, Will 2008). Server1, Server2 i Server3, s IP-ovima 192.168.0.3, 192.168.0.4 i 192.168.0.6, predstavljaju pozadinske poslužitelje koji pružaju usluge i pokazuju kako balansiranje prometa utječe na njihove performanse.

Radna stanica: Webterm-1 (IP adresa 192.168.0.5) Uređaj simulira korisnički promet i testira distribuciju zahtjeva u sučelju balansera opterećenja. Pristupa mu se korištenjem prekidača, sa simulacijom virtualno povezanom s eth0, što omogućuje jednostavan pristup svim uslugama unutar mreže.

Ova topologija omogućuje simulaciju stvarnog mrežnog okruženja, što je ključno za preciznu evaluaciju performansi load balancera. Analiza će se uglavnom temeljiti na procjeni učinkovitosti, odgovoru sustava na različite zahtjeve i procjeni propusnosti u cilju optimizacije mrežnih resursa dostupnih u proizvodnom okruženju.



Slika 6. Toplogija projekta.

Izvor: Samostalan rad

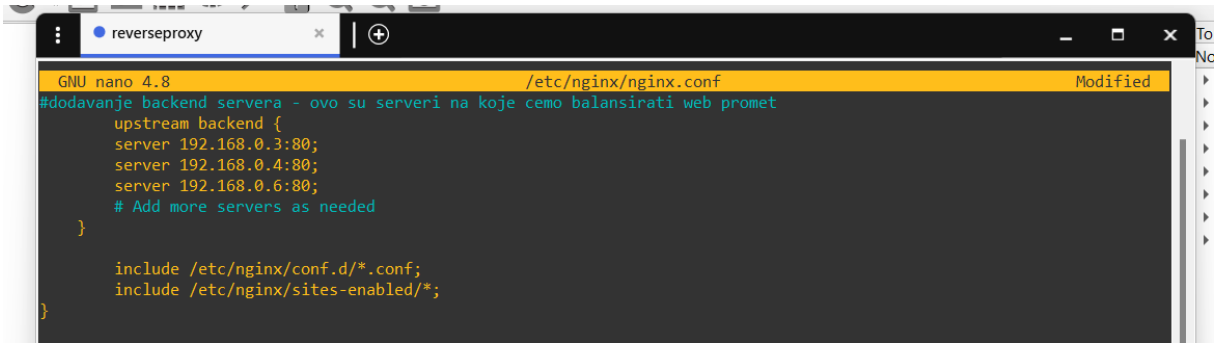
2.2. Korištenje Nginx-a kao load balancera

Budući da može djelovati kao balanser opterećenja i reverse proxy poslužitelj za HTTP i druge mrežne protokole, NGINX je jedan od najpopularnijih web poslužitelja na današnjem tržištu

(DeJonghe, Derek 2020). Nginx je poznat po svojoj laganoj prirodi i visokim performansama. Nginx može učinkovito obraditi veliki broj zahtjeva i omogućiti više klijenata pristup sustavu odjednom. Budući da Nginx može učinkovito podnijeti velike količine prometa, to je savršena opcija za uravnoteženje opterećenja (Chyrvon, Andrii et. al 2023).

Zbog toga smo koristili NGINX kao reverse proxy za učinkovitu distribuciju dolaznog web prometa između više backend servera opterećenja. Sljedeće što je bilo potrebno napraviti je konfigurirati datoteke za rad reverse proxy-a između nekoliko backend servera.

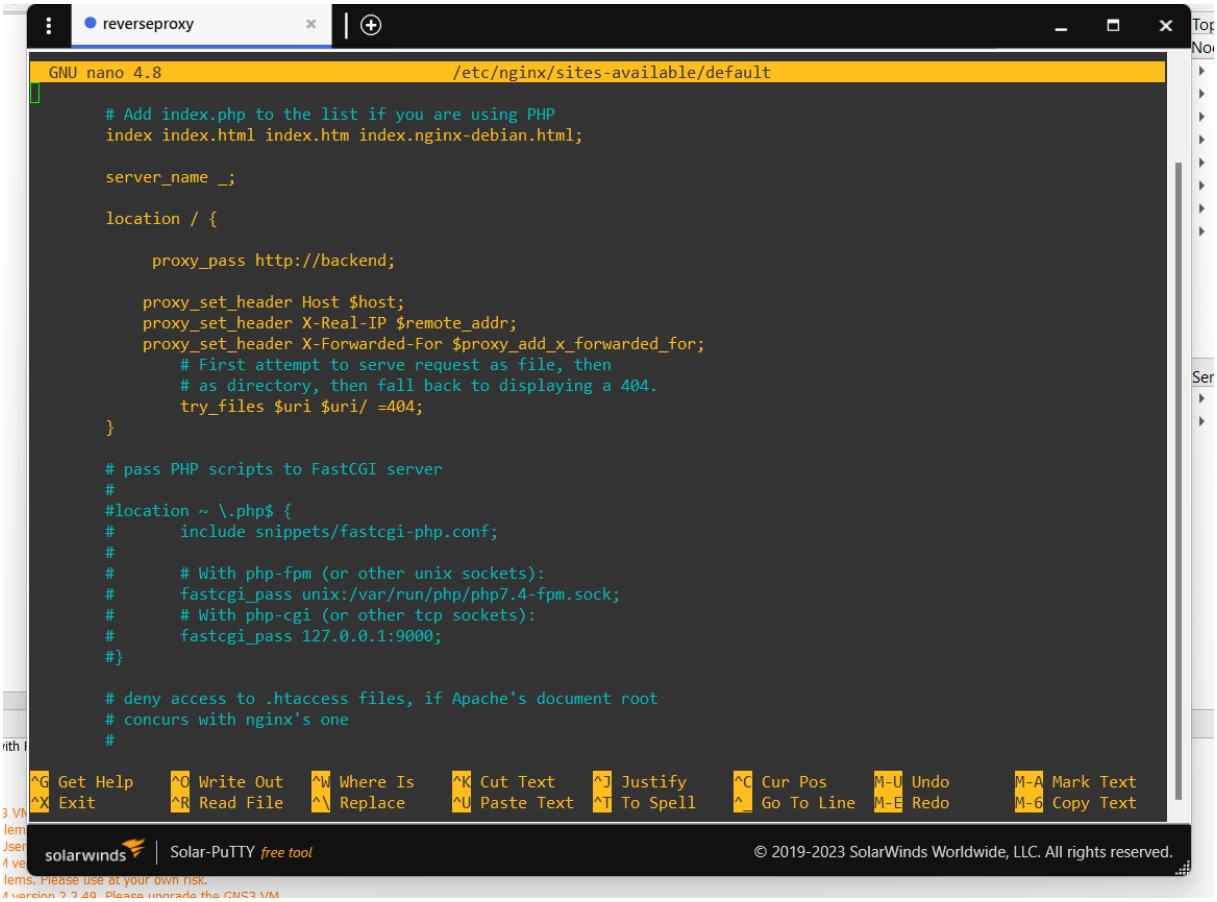
Konfiguracijska datoteka /etc/nginx/nginx.conf za NGINX uglavnom uključuje definiciju grupe pozadinskih poslužitelja. Ovi poslužitelji sudjeluju u balansiranju prometa, pri čemu se svaki nadolazeći zahtjev može proslijediti bilo kojem od definiranih poslužitelja, ovisno o postavljenom algoritmu balansiranja (DigitalOcean, 2022). Kada budemo radili benchmark, u ovoj datoteci ćemo mijenjati algoritme uravnoteženja, ovisno o tome koji želimo koristiti.



Slika 7. Definiranje grupe poslužitelja u nginx.conf datoteci.

Izvor: Samostalan rad

Default datoteka u direktoriju sites-available ima neke specifične konfiguracije u vezi s blokom poslužitelja, kao što su pravila za prosljeđivanje prometa definiranoj grupi pozadinskih poslužitelja.



Slika 8. Definiranje specifičnih konfiguracija za poslužiteljske blokove.

Izvor: Samostalan rad

U ovom slučaju NGINX sluša sve dolazne zahtjeve (bilo koji naziv poslužitelja, server_name _) i prosljeđuje promet definiranoj pozadinskoj grupi, kako je definirano u datoteci pod nazivom nginx.conf. Sljedeća direktiva je ponašanje u NGINX definirano proxy_pass http://backend; za koje se svi zahtjevi propagiraju na jedan od poslužitelja u pozadinskoj grupi (DigitalOcean, 2022).

Dodatna HTTP zaglavlja mogu se postaviti u NGINX, koja prenose ključne informacije od klijenta do pozadinskih poslužitelja. Neka od postavljenih zaglavlja uključuju sljedeće:

Host: ovo zaglavlje prenosi izvorni host iz zahtjeva. X-Real-IP: ovo zaglavlje sadrži stvarnu IP adresu klijenta. X-Forwarded-For: ovo zaglavlje sadrži informacije o IP adresama kroz koje je

zahtjev prošao. Zaglavlja omogućuju pozadinskim poslužiteljima da dobiju sve informacije od izvornog klijenta i zahtjeva (Nginx, 2024).

Nakon konfiguriranja datoteka za rad reverse proxy-a, preostaje samo konfigurirati datoteku index.html za svaki algoritam koji budemo ispitivali.

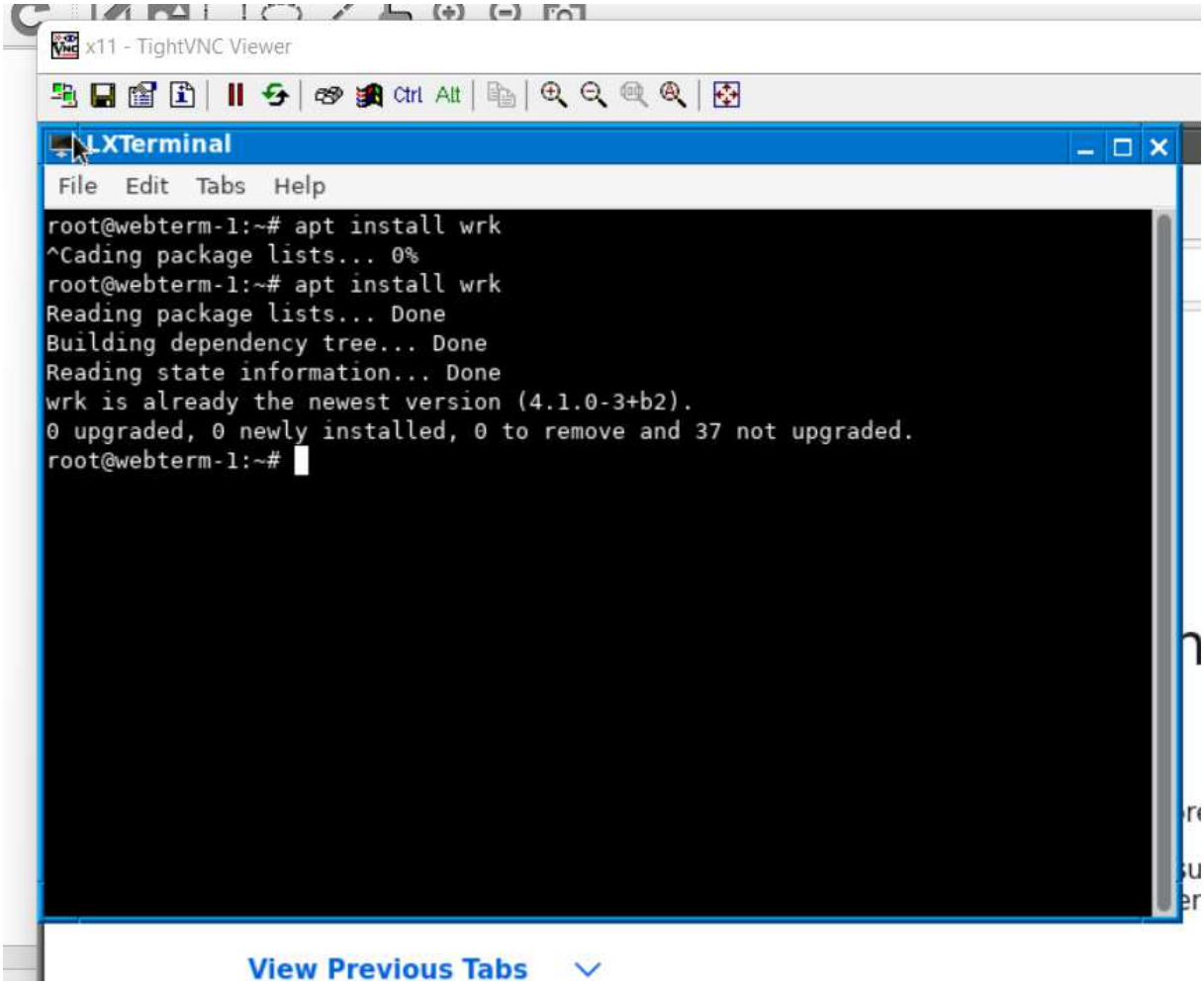
2.3. Alat za benchmark

Alati za usporedbu, u svakom slučaju, vrlo su ključni za testiranje balansera opterećenja, pa čak i mrežnih sustava. Ovi alati omogućuju stvaranje umjetnih opterećenja mreže i web poslužitelja za mjerenje njihovih performansi u takvim uvjetima. Stoga se u analizi koristio wrk alat kako bi se pregledala latencija, brzina prijenosa i propusnost koristeći različiti algoritam uravnoteženja.

Wrk je moderan HTTP alat za usporedbu koji kada se izvodi na jednom CPU-u s više jezgri, stvara vrlo značajno opterećenje (Github, 2024). Ističe se značajkom višenitnosti i može opteretiti web poslužitelje radi simulacije stvarnih situacija (CQR Tools, 2024).

Neke prednosti wrk alata su skalabilnost u kojem podržava veliki broj istovremenih veza i višestruke niti, što omogućuje generiranje značajnog opterećenja. Vrlo je brz i efikasan, sposoban je generirati veliki broj zahtjeva u kratkom vremenu te omogućuje prilagođavanje različitih parametara testa kako bi odgovarao specifičnim potrebama (CQR Tools, 2024).

Instalacija uslužnog programa wrk je jednostavna; koristi se upraviteljem paketa vašeg OS-a. Na primjer, u terminalu, možete lako instalirati wrk koristeći apt install wrk.



Slika 9. Instaliranje wrk alata.

Izvor: Samostalan rad

2.4. Wireshark za analizu prometa

Wireshark je analizator mrežnih paketa. Analizator mrežnih paketa pokušat će uhvatiti mrežne pakete i prikazati te paketne podatke sa što više detalja (Lamping, Ulf, and Ed Warnicke 2005). On je jedan od najmoćnijih alata za analizu mrežnog prometa; naširoko se koristi među mrežnim inženjerima i istraživačima za presretanje i daljnju inspekciju paketa koji se kreću mrežom. Također, može uhvatiti promet s mnogo različitih vrsta mrežnih medija i unatoč svom nazivu uključuje i bežični LAN. Koje su vrste medija podržane ovisi o mnogim stvarima poput operativnog sustava koji koristite (Lamping, Ulf, and Ed Warnicke 2005).

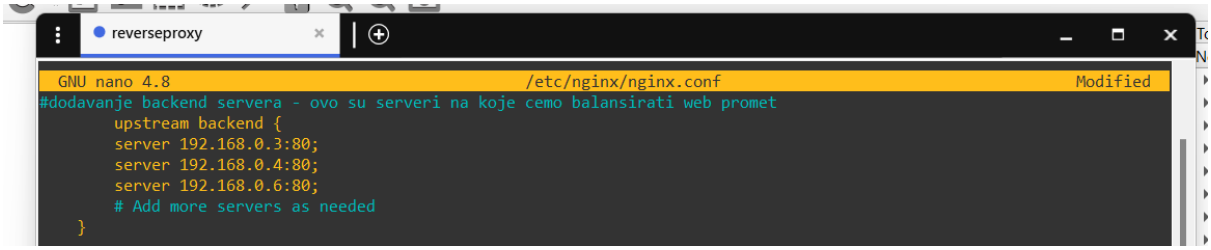
Dao je detaljan uvid u performanse i ponašanje mreže tijekom analize mrežnog prometa. Wireshark je iznimno važan alat za svakog mrežnog inženjera koji želi steći dubinsko razumijevanje i upravljanje složenim mrežnim okruženjima.

Instaliranje wireshark aplikacije se provodi na službenoj stranici wiresharka. Gns3 će ga automatski detektirati i integritati u njegov sustav.

3. Rezultati

Ovdje su predstavljeni rezultati testiranja algoritama balansiranja opterećenja u simuliranom mrežnom okruženju. Testirani algoritmi uključuju Round Robin, Least Connections i IP Hash. Neki ključni parametri performansi koji se nadziru su latencija, brzina prijenosa te propusnost kao što su navedeni u prethodnom dijelu rada. U svakom scenariju, promet je distribuiran na više poslužitelja, a rezultati su zabilježeni kako bi se identificiralo najboljeg algoritma uravnoteženja. Također, korištenje Wiresharka omogućilo je detaljnu analizu mrežnog prometa, pružajući dublji uvid u rad algoritama a i propusnost koja će biti vizualno prikazana putem grafikona.

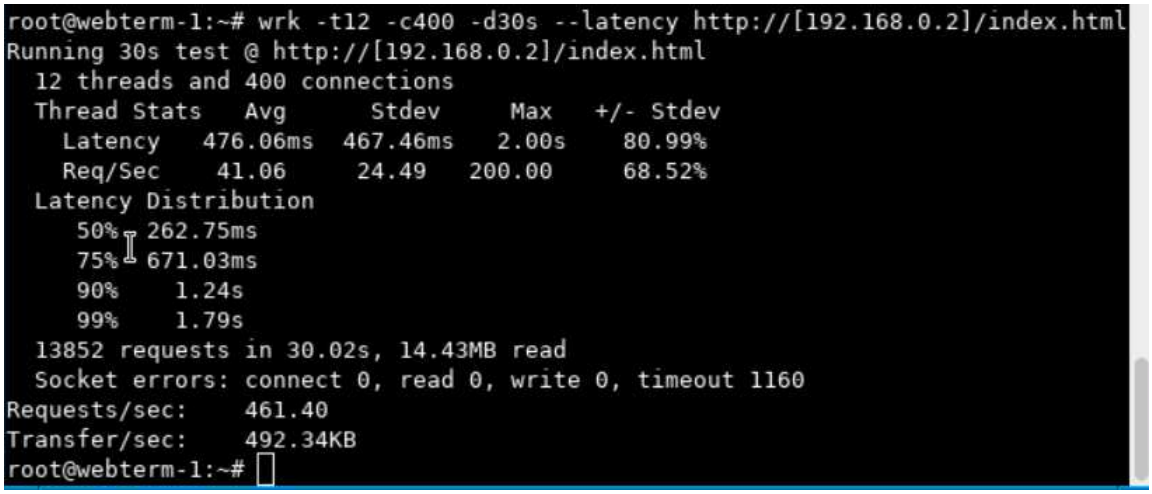
Prvi algoritam uravnoteženja koji je testiran je round robin. Datoteku /etc/nginx/nginx.conf konfiguriramo za rad.



Slika 10. Konfiguracija za round robin algoritam.

Izvor: Samostalan rad

U terminalu od webterma smo unijeli naredbu za wrk alat koja pokreće testiranje performansi web servera.



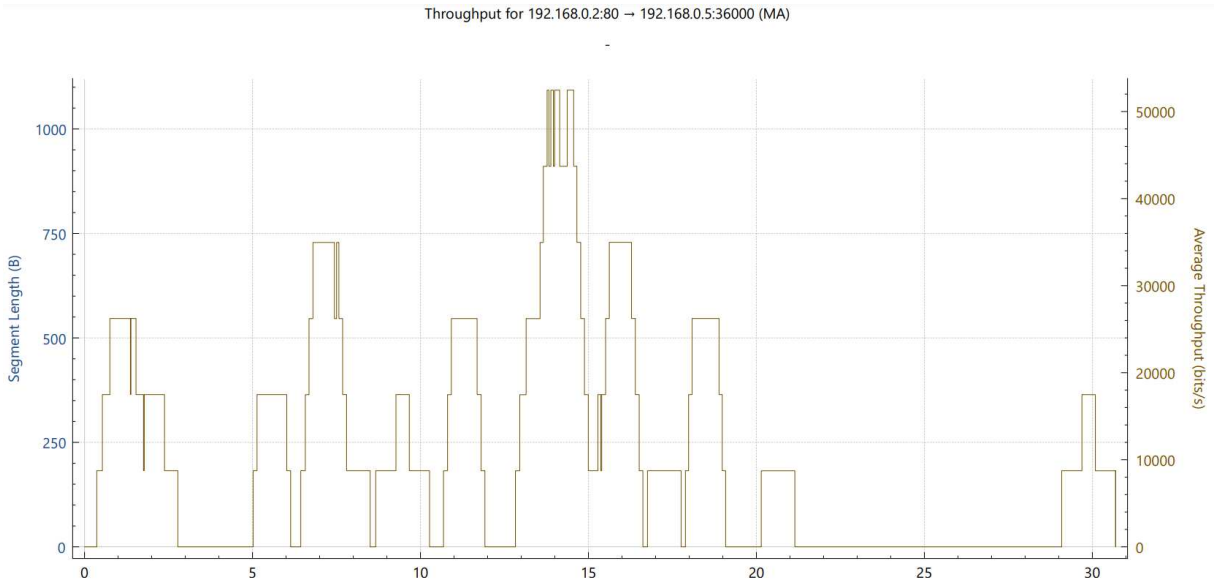
Slika 11. Round robin testiranje.

Izvor: Samostalan rad

Tijekom 30 sekundi (d30s), wrk će koristiti 12 niti (t12) kako bi održavao 400 (c400) istovremenih konekcija prema specificiranom URL-u (192.168.0.2).

Na kraju testa, wrk prikazuje statistiku koja uključuje brzinu prijenosa, kašnjenje (latenciju). Rezultati testiranja pokazuju da je prosječna brzina prijenosa, kada je poslužitelj pod opterećenjem, oko 492,34 KB/s, čime je osigurana dobra brzina protoka podataka. Stvarna brzina prijenosa može varirati ovisno o složenosti zahtjeva koje obrađuje, veličini odgovora i broju istodobnih veza. Prosječna latencija je velika i iznosi 476,06 ms za aplikacije koje zahtijevaju da sustav brzo reagira na korisnički unos. Ova distribucija pokazuje da se 50% zahtjeva izvrši unutar 262,75 ms, dok je kvantil kod 75% bio veći od 671,03 ms, a samo 99% zahtjeva obrađeno je nakon 1,79 sekundi, što može negativno utjecati na korisničko iskustvo.

Nakon testa sa wrk alatom, u wiresharku se pregledavalo kakva je propusnost koristeći round robin algoritam. S lijeve strane grafikona prikazana je duljina segmenata (u bajtovima), dok desna os prikazuje prosječnu propusnost u bitovima po sekundi



Slika 12. Grafikon propusnosti za round robin algoritam.

Izvor: Samostalan rad

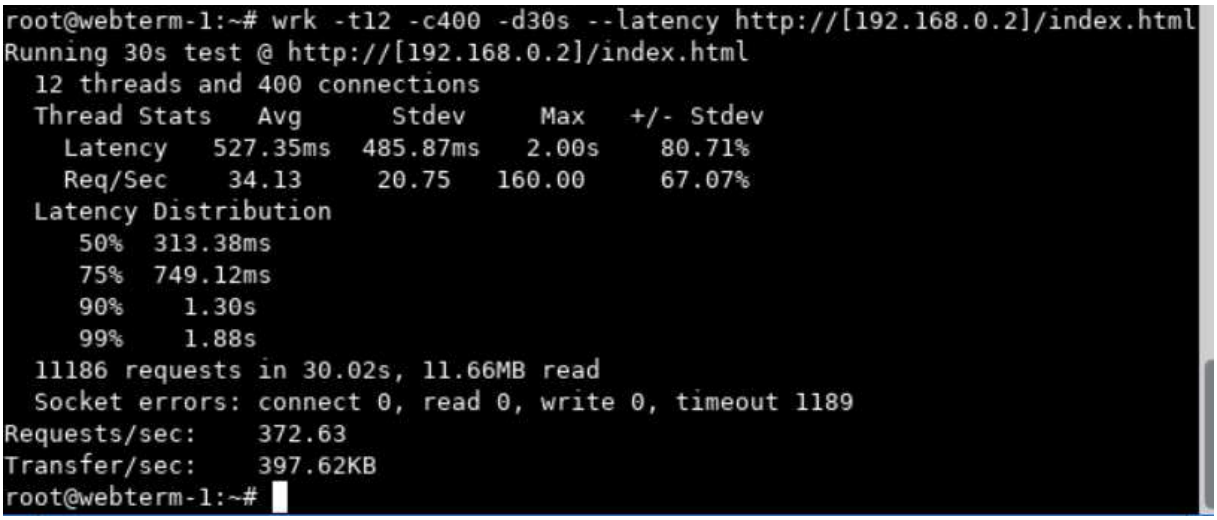
Graf analize propusnosti ima segmente promjenjive duljine, što znači da da se podaci šalju u različitim veličinama paketa zbog Round Robin algoritma koji ravnomjerno raspoređuje opterećenje između poslužitelja. Prosjek varira, u jednom ili drugom trenutku ima vrhunce između 30 000-50 000 bps, što je dokaz periodičnih promjena u brzini prijenosa. Također su vidljive praznine u prijenosu, što može biti rezultat čekanja između Round Robin ciklusa kada se promet preusmjerava. To su pokazatelji nejednake distribucije informacija na temelju oblika distribucije prometa.

Drugi algoritam uravnoteženja koji je testiran je least connections. Također smo konfigurirali /etc/nginx/nginx.conf datoteku za rad least connections algoritma.

Slika 13. Konfiguracija za least connection algoritam.

Izvor: Samostalan rad

Nakon konfiguracije datoteke smo unijeli istu naredbu za wrk u terminal webterma kako bi pokrenuli testiranje performansi web servera.

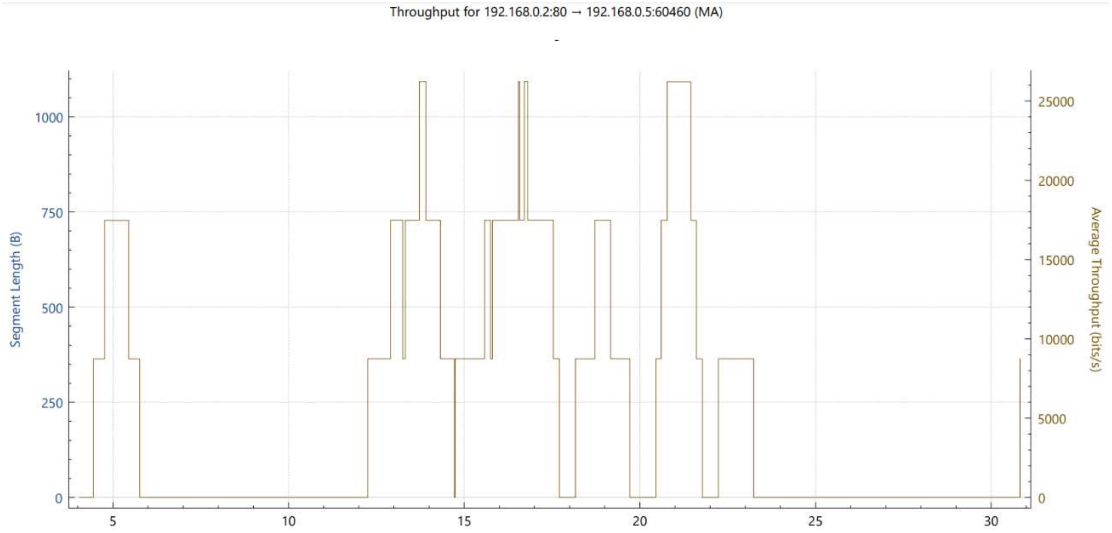


Slika 14. Least connections testiranje.

Izvor: Samostalni rad

Rezultati pokazuju da je prosječna brzina prijenosa bila 397.62 KB/s, što osigurava solidan protok podataka. Prosječna latencija iznosi 527,35 ms, što je vrlo visoko, posebno za aplikacije gdje su potrebni brzi odziv i fluidna interakcija. Detaljna raspodjela latencije pokazala je da je

50% zahtjeva obrađeno unutar 313,38 ms, dok je 75% zahtjeva obrađeno u 749,12 ms. Dok je za 99%, latencija bila 1,88 sekundi, što ozbiljno ugrožava korisničko iskustvo, posebno u onim aplikacijama koje se oslanjaju na nisku latenciju kako bi pružile učinkovitost i ugodnost.



Slika 15 Grafikon propusnosti za least connections algoritam.

Izvor: Samostalan rad

Varijacija duljine segmenta, kao što je naznačeno na lijevoj osi, ide do oko 1000 bajtova, duljina segmenta stoga varira s veličinom paketa podataka koji se prenose, pokazujući dinamiku u pristupu s least connections.

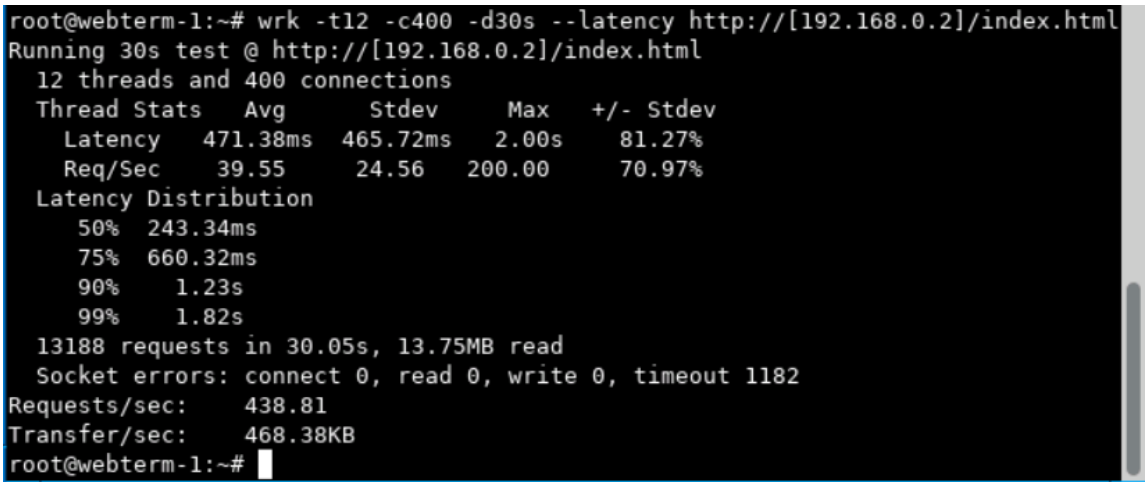
Suprotno tome, prosječna propusnost predstavljena desnom osi ima oštre oscilacije čije se vršne vrijednosti kreću od 15 000 do 25 000 bita u sekundi. Propusnost je nekonzistentna - što je vrlo vidljivo po velikim varijacijama u brzini prijenosa podataka. Takvi skokovi označavaju periodične skokove u opterećenju dok se podaci prenose.

Zadnji algoritam uravnoteženja koji je testiran je ip hash. Konfigurirali smo /etc/nginx/nginx.conf datoteku za rad ip hash algoritma.

Slika 16 Konfiguracija za ip hash algoritam.

Izvor: Samostalni rad

Nakon konfiguracije datoteke, pokrenuli smo testiranje performansi web servera unosom iste naredbe za wrk u terminal webterma.

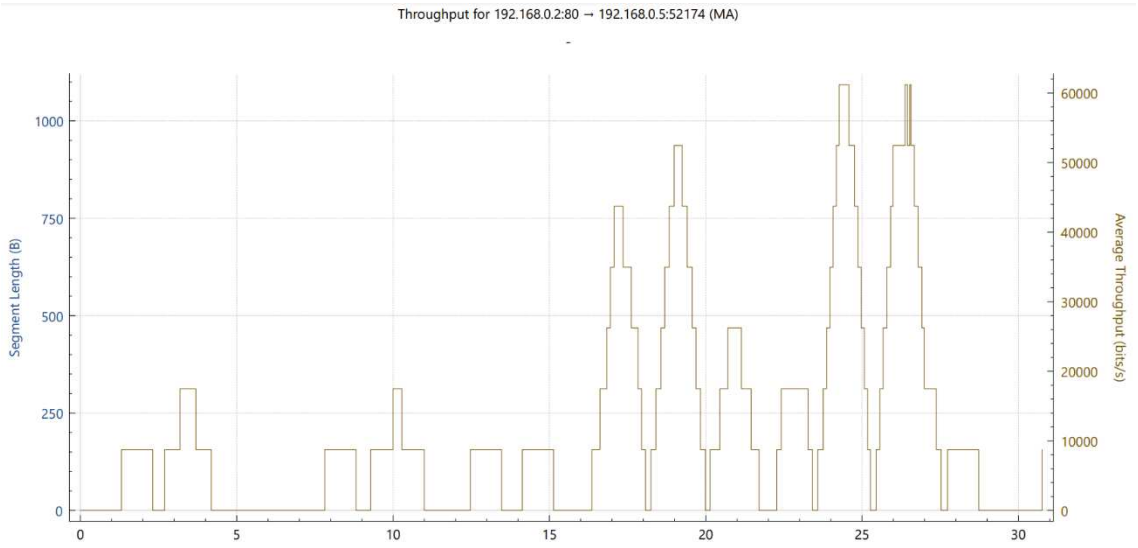


Slika 17 Ip hash testiranje.

Izvor: Samostalni rad

Što se tiče opterećenja, rezultati testova pokazuju da je prosječna brzina prijenosa bila oko 468,38 KB/s, što osigurava pristojan protok podataka Stvarna brzina prijenosa može varirati ovisno o složenosti zahtjeva, veličini odgovora i broju istodobnih veza koje poslužitelj obrađuje. Za aplikacije koje zahtijevaju brz odgovor sustava na korisničke zahtjeve, prosječna latencija je visoka i iznosi 471,38 ms. Ova se distribucija može detaljnije opisati: 50% zahtjeva

izvršeno je unutar 243,34 ms, 75% kvantila je 660,32 ms, a najsporijih 99% zahtjeva obrađuje se nakon 1,82 sekunde. Ovo može negativno utjecati na korisničko iskustvo, pogotovo kod aplikacija osjetljivih na kašnjenje.



Slika 18 Grafikon propusnosti za ip hash algoritam.

Izvor: Samostalni rad

Grafikon prikazuje varijacije u prijenosu podataka unutar sesije. Može se uočiti i povećanje i smanjenje duljine segmenta; na nekoliko značajnih vrhunaca, duljine prelaze 1000 bajtova po segmentu, dok prosječna propusnost doseže gotovo 60 000 bitova u sekundi. Ovi vrhovi mogu ukazivati na trenutke gušćeg prometa ili složenijih paketa koji se prenose preko mreže.

On je pokazao da postoji učinkovitost IP hash algoritma u raspodjeli opterećenja između različitih poslužitelja. Iako postoje slučajevi pada propusnosti, algoritam često održava visoke vršne tokove podataka na stabilnim razinama, čime se optimizira izvedba mreže minimiziranjem latencije zahtjeva.

4. Zaključak

Testiranje algoritama za uravnoteženje opterećenja: Round Robin, Least Connections i IP Hash daje uvid u njihovu učinkovitost s nekoliko scenarija mrežnog prometa. Svaki od algoritama ima svoje osobitosti u pogledu raspodjele zahtjeva, kontinuiteta propusnosti i upravljanja latencijom, a oni prolaze kroz određene prednosti i nedostatke.

Glavni nedostatak algoritma Round Robin je pojednostavljen proces distribucije zahtjeva koji dovodi do nepravilne distribucije propusnosti. Prosječna brzina prijenosa je dobra, 492,34 KB/s, ali analiza propusnosti pokazuje nestalne varijacije brzine prijenosa podataka. To je zato što prebacivanje opterećenja između poslužitelja slijedi ciklički proces. Periodični prekidi u prijenosu podataka također signaliziraju vrstu zastoja između ovih ciklusa, što je nepovoljno za dosljednost usluge.

Sljedeća dobra opcija je algoritam least connections, koji postavlja prosječnu brzinu prijenosa od 397,62 KB/s i uključuje neke prednosti i nedostatke. Iako ovaj pristup daje stabilnije opterećenje između poslužitelja, visoka prosječna latencija od 527,35 ms ozbiljno ugrožava aplikacije ovisno o niskoj latenciji. Fluktuacije u propusnosti i segmentima pokazuju dinamiku ovog algoritma, gdje će performanse ovisiti o trenutnom stanju poslužitelja i količini veza otvorenih u to vrijeme.

Najbolji kompromis između propusnosti i stabilnosti predstavlja IP Hash algoritam. Omogućujući visoku vršnu vrijednost protoka, njegova varijacija u distribuciji opterećenja je relativno stabilnija s prosječnom brzinom prijenosa od 468,38 KB/s s manje smetnji. Iako je latencija još uvijek visoka s prosjekom od 471,38 ms, rezultati pokazuju da ovaj algoritam ima optimalniju raspodjelu resursa, stoga će biti prikladan za mrežne sustave koji zahtijevaju veću pouzdanost u prijenosu podataka.

U zaključku, IP Hash algoritam može se smatrati najboljim rješenjem za sustave koji zahtijevaju stabilnost, brzinu prijenosa i smanjenje latencije. Algoritam osigurava kvalitetniju izvedbu u

usporedbi s drugima, što se odražava na stabilnu razinu propusnosti čak i pod promjenjivim opterećenjem. Prednost koju nudi je distribucija zahtjeva na temelju klijentovih relevantnih IP adresa, što omogućuje ravnomjerniju raspodjelu resursa i stoga smanjuje vjerojatnost zastoja u prijenosu podataka.

5. Sažetak

Ovaj rad raspravlja o ključnoj ulozi i učinkovitosti mrežnih simulatora u općem razvoju, s posebnim osvrtom na GNS3 i njegovu primjenu za analizu performansi. Mrežni simulatori poput GNS3 omogućuju crtanje topologije mreže s velikom preciznošću i, gotovo bez napora, testiraju različite scenarije bez upotrebe skupe fizičke opreme (Gomez, J et. al 2023).

Projekt je posvećen postavljanju virtualne mreže uz pomoć GNS3 jednostavnim stvaranjem usmjerivača, preklopnika i poslužitelja kako bi se stimuliralo okruženje ekvivalentno stvarnim mrežama. NGINX je konfiguriran kao reverse proxy za balansiranje opterećenja između poslužitelja. Istraženi su različiti algoritmi za balansiranje opterećenja Round Robin, IP Hash i Least Connections s ciljem razumijevanja njihovog ponašanja pod različitim uvjetima opterećenja mreže.

Performanse su mjerene pomoću wrk-a za generiranje umjetnih opterećenja, a s druge strane, Wireshark je korišten za analizu mrežnog prometa. Uz pomoć ovih alata, moguće je nadzirati svaki detalj o kašnjenju, brzini prijenosa i propusnosti, istovremeno pružajući uvid u različite algoritme za uravnoteženje opterećenja koji se mogu primijeniti na mreži.

Rezultati istraživanja skreću veliku pozornost na prikladnost izbora algoritma za uravnoteženje opterećenja i pokazuju kako mrežni simulatori i alati za analizu mogu pomoći u optimizaciji mrežnih resursa, poboljšanju performansi i jamčenju visoke dostupnosti usluge u modernim mrežnim okruženjima.

6. Literatura

Mohammad S. Obaidat, Faouzi Zarai, and Petros Nicopolitidis. 2015. Modeling and Simulation of Computer Networks and Systems: Methodologies and Applications (1st. ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Gomez, J., Kfoury, E. F., Crichigno, J., & Srivastava, G. (2023). A survey on network simulators, emulators, and testbeds used for research and education. *Computer Networks*, *237*, 110054.

Internetski izvor: Wikipedia (2024). Raspoloživo na: https://en.wikipedia.org/wiki/Graphical_Network_Simulator-3 (pristupljeno: 2. srpnja 2024.)

Internetski izvor: Gns3 (2024). Raspoloživo na https://docs.gns3.com/docs/ (pristupljeno: 2. srpnja 2024.)

Internetski izvor: Gns3 (2024). Raspoloživo na https://www.gns3.com/ (pristupljeno 5. srpnja 2024).

Internetski izvor: Fortinet (2024). Raspoloživo na https://www.fortinet.com/resources/cyberglossary/reverse-proxy (pristupljeno 7. srpnja 2024).

Internetski izvor: Cloduflare (2024). Raspoloživo na: https://www.cloudflare.com/learning/cdn/glossary/reverse-proxy/ (pristupljeno 15. srpnja. 2024).

Internetski izvor: System Design School (2023). Raspoloživo na https://systemdesignschool.io/blog/reverse-proxy-vs-load-balancer (pristupljeno 15. srpnja 2024).

A. Erzurumluoğlu, “Constructing day-balanced round-robin tournaments with partitions,” Discret. Appl. Math., vol. 235, pp. 81–91, 2018.

Neumann, Jason C. The book of GNS3: build virtual network labs using Cisco, Juniper, and more. No Starch Press, 2015.

S. Abdallah, E. Najjar, and A. Kayssi, “A round robin load balancing and redundancy protocol for network routers,” Proc. 11th IEEE Int. Conf. Trust. Secur. Priv. Comput. Commun. Trust. - 11th IEEE Int. Conf. Ubiquitous Comput. Commun. IUCC-2012, pp. 1741–1747, 2012.

L. Wang and G. Lu, “The Dynamic Sub-Topology Load Ba lancing Algorithm for Data Center Networks,” 2016 Int. Conf. Inf. Netw., pp. 268–273, 2016.

Ju-Yeon Jo and Yoohwan Kim, ”Hash-based Internet traffic load balancing,” Proceedings of the 2004 IEEE International Conference on Information Reuse and Integration, 2004. IRI 2004., 2004, pp. 204-209.

Internetski izvor: StrongDM (2024). Raspoloživo na https://www.strongdm.com/what-is/reverse-proxy-vs-load-balancer (pristupljeno 25. srpnja 2024).

Bourke, Tony. Server load balancing. " O'Reilly Media, Inc.", 2001.

Z. Xu and X. Wang, “A predictive modified round robin scheduling algorithm for web server clusters,” Chinese Control Conf. CCC, vol. 2015-Septe, pp. 5804–5808, 2015.

DeJonghe, Derek. *Nginx cookbook*. O'Reilly Media, 2020.

Chyrvon, Andrii & Lisovskyi, Kostiantyn & Kyryndas, Nikita. (2023). THE MAIN METHODS OF LOAD BALANCING ON THE NGINX WEB SERVER. 10.36074/logos-26.05.2023.040.

Albowarab, Mustafa & Zakaria, Nurul & Zainal Abidin, Zaheera. (2018). Load balancing algorithms in software defined network. 7. 686-693.

Afrianto, Yuggo, Heru Sukoco, and Sri Wahjuni. "Weighted round robin load balancer to enhance web server cluster in openflow networks." TELKOMNIKA (Telecommunication Computing Electronics and Control) 16.3 (2018): 1402-1408.

Harjanti, Trinugi & Setiyani, Hari & Trianto, Joko. (2022). Load Balancing Analysis Using Round-Robin and Least-Connection Algorithms for Server Service Response Time. Applied Technology and Computing Science Journal. 5. 40-49. 10.33086/atcsj.v5i2.3743.

Alankar, Bhavya & Sharma, Gaurav & Kaur, Harleen & Valverde, Raul & Chang, Victor. (2020). Experimental Setup for Investigating the Efficient Load Balancing Algorithms on Virtual Cloud. Sensors. 20. 7342. 10.3390/s20247342.

Ndatinya, Vivens, et al. "Network forensics analysis using Wireshark." International Journal of Security and Networks 10.2 (2015): 91-106.

Internetski izvor:Cisco (2024). Raspoloživo na https://www.cisco.com/c/en/us/solutions/small-business/resource-center/networking/howdoes-a-router-work.html (pristupljeno 2. kolovoza 2024).

Reese, Will. "Nginx: the high-performance web server and reverse proxy." *Linux Journal* 2008.173 (2008): 2.

Internetski izvor: Nginx (2024). Raspoloživo na: https://docs.nginx.com/nginx/admin-guide/web-server/reverse-proxy/ (pristupljeno 19. kolovoza 2024.)

Internetski izvor: DigitalOcean (2022). Raspoloživo na: https://www.digitalocean.com/community/tutorials/understanding-the-nginx-configuration-file-structure-and-configuration-contexts (pristupljeno 17. kolovoza 2024.

DeJonghe, Derek. Nginx cookbook. O'Reilly Media, 2020.

Internetski izvor: Github (2021). Raspoloživo na https://github.com/wg/wrk (pristupljeno 18. kolovoza 2024).

Internetski izvor: CQR Tools (2024). Raspoloživo na https://www.cqr.tools/tools/wrk (pristupljeno 21. kolovoza 2024).

Lamping, Ulf, and Ed Warnicke. "Wireshark user's guide." *Interface* 4.6 (2004): 1.

Slike