PD.TVZ.HR
POLYTECHNIC & DESIGN Vol. 14, No. 1, 2026.
Mrežna inačica: ISSN 2459-6302
Tiskana inačica: ISSN 1849-1995
DOI: 10.19279/TVZ.PD.2026-14-1-02

USPOREDBA PERFORMANSI TCP I QUIC PROTOKOLA U SIMULIRANIM MREŽNIM SCENARIJIMA

PERFORMANCE COMPARISON OF TCP AND QUIC PROTOCOLS ACROSS SIMULATED NETWORK SCENARIOS

Jakov Vukadin[1], Vedran Tadić[2], Ognjen Mitrović[2]

[1] Tehničko veleučilište u Zagrebu, Vrbik 8, 10000 Zagreb, Hrvatska, Student
[2] Tehničko veleučilište u Zagrebu, Vrbik 8, 10000 Zagreb, Hrvatska

SAŽETAK

Transportni protokoli čine temelj internetske komunikacije, u čemu TCP desetljećima dominira. Međutim, suvremene aplikacije zahtijevaju brz odaziv i otpornost na promjenjive mrežne uvjete, što je motiviralo razvoj QUIC protokola kao osnove za HTTP/3. Ovaj rad donosi usporedbu performansi TCP i QUIC protokola kroz niz mjerenja latencije, propusnosti, multipleksiranja i otpornosti na gubitke paketa u simuliranim mrežnim uvjetima. Testiranja i mjerenja su provedena u izoliranom Docker okruženju koristeći virtualne mrežne simulacije. Rezultati pokazuju da pod idealnim uvjetima protokoli imaju slične performanse, dok QUIC pod uvjetima veće simulirane latencije i gubitaka paketa značajno bolje upravlja protokolnim mehanizmima što rezultira nižom latencijom, stabilnošću i multipleksiranjem. Jedini aspekt u kojem TCP ostaje nadmoćan jest propusnost učitavanja, no QUIC držeći stabilan tok podataka pokazuje prednost u uvjetima složenih ili lošijih mreža.

Ključne riječi: *QUIC, TCP, HTTP/3, latencija, propusnost, HOL blokiranje, Docker*

ABSTRACT

Transport protocols form the backbone of internet communication, with TCP dominating for decades. However, modern applications demand low latency and resilience to variable network conditions, leading to the development of the QUIC protocol as the foundation for HTTP/3. This paper presents a performance comparison of TCP and QUIC protocols through measurements of latency, throughput, multiplexing, and packet loss resilience in simulated network conditions. Tests and measurements were conducted in an isolated Docker environment using virtual network simulations. Results show that under ideal conditions, the protocols perform similarly, while QUIC significantly better manages protocol mechanisms under increased simulated latency and packet loss, resulting in lower latency, stability, and multiplexing. The only area where TCP consistently outperforms QUIC is upload throughput, but QUIC's stable data flow offers advantages in complex or degraded network scenarios.

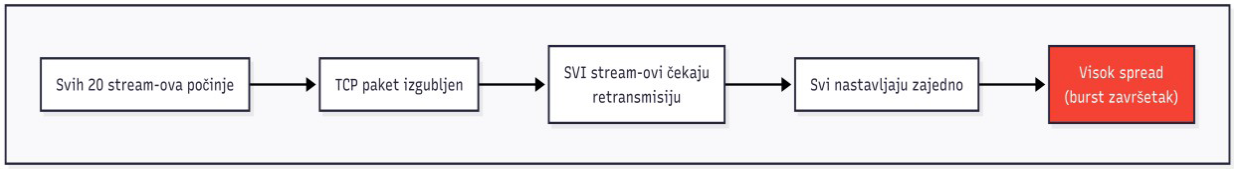
Keywords: *QUIC, TCP, HTTP/3, latency, throughput, head-of-line blocking, Docker*

1. UVOD
1. INTRODUCTION

Suvremeni Internet temelji se na protokolima koji su dizajnirani prije nekoliko desetljeća, u vrijeme kada su mrežni uvjeti i zahtjevi korisnika bili bitno drugačiji. Iako je TCP (engl. *Transmission Control Protocol*) godinama osiguravao pouzdan prijenos podataka, njegova arhitektonska ograničenja – poput implementacije unutar jezgre operacijskog sustava (engl. *kernel space*) i problema blokiranja na čelu reda (engl. *Head-of-Line blocking*) – postala su usko grlo za moderne web aplikacije [1].

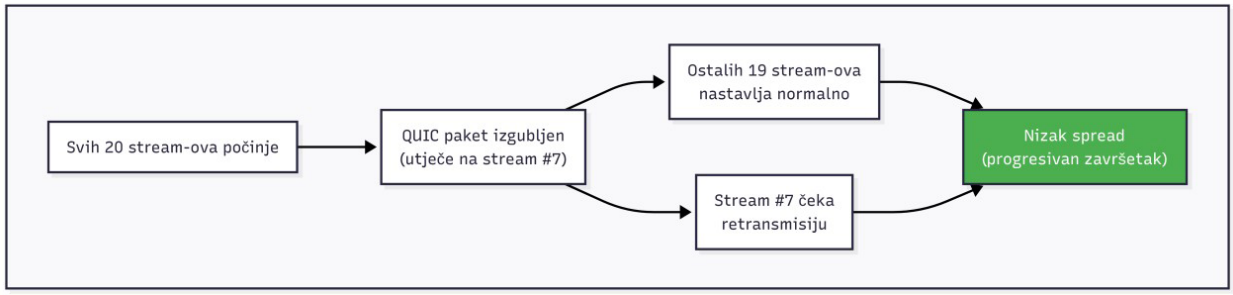
Svojom implementacijom multipleksiranja, HTTP/2 je razriješio problem *head-of-line* blokiranja na

Članak se nalazi u bazama: ● http://hrcak.srce.hr ● https://scholar.google.hr



Slika 1 *Utjecaj gubitka segmenta na multipleksiranje kod TCP protokola (autorski rad)*

Figure 1 *Impact of segment loss on multiplexing in the TCP protocol (source: author)*



Slika 2 *Utjecaj gubitka segmenta na multipleksiranje kod QUIC protokola (autorski rad)*

Figure 2 *Impact of segment loss on multiplexing in the QUIC protocol (source: author)*

aplikacijskoj razini, no s obzirom da TCP tretira sve podatke kao jedan uređeni tok (engl. *stream*) bajtova, problematika se prenijela na transportni sloj. Ako se izgubi jedan TCP segment, prijemna strana mora čekati ponovni prijenos neisporučenog segmenta prije nego li može proslijediti bilo koje podatke aplikaciji, uključujući i one iz drugih HTTP/2 tokova koji nisu ovisili o izgubljenom segmentu. Ovo ograničenje je poznato kao *head-of-line* blokiranje na transportnom sloju (Slika 1).

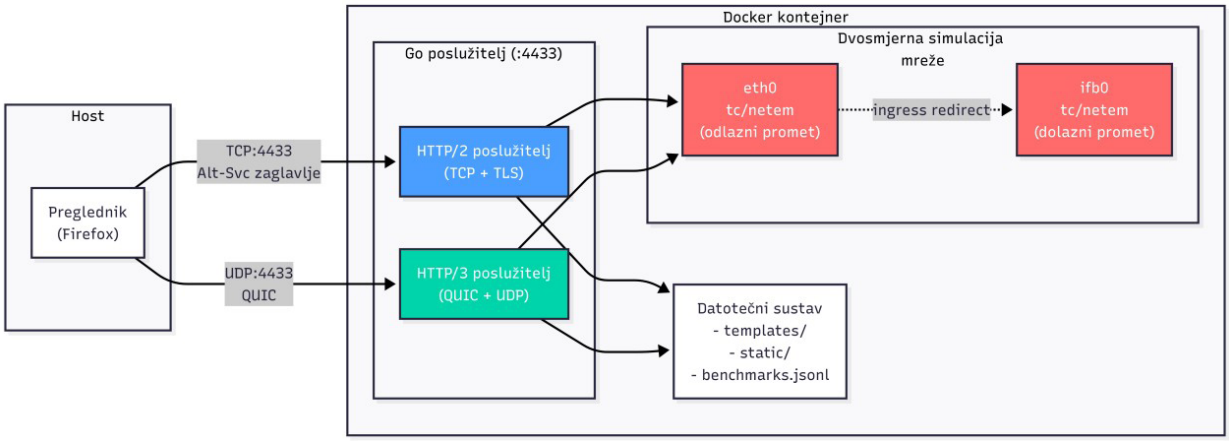
Kao odgovor na te izazove, razvijen je QUIC (*Quick UDP Internet Connections*), novi transportni protokol koji se temelji na UDP-u, ali uvodi napredne značajke poput brze uspostave veze, ugrađene enkripcije i poboljšane otpornosti na gubitak paketa [2] (Slika 2).

Važnost QUIC protokola najbolje oslikava njegova masovna primjena od strane tehnoloških divova koji upravljaju najvećim udjelom svjetskog mrežnog prometa. Google, kao začetnik protokola, integrirao je QUIC u preglednik Chrome i sve svoje ključne servise poput YouTubea i Gmaila. Prema njihovim službenim mjerenjima, uvođenje QUIC-a smanjilo je latenciju pretraživanja za 2 %, vrijeme ponovnog učitavanja videozapisa na YouTubeu smanjeno je za čak 9 % u uvjetima lošije mrežne povezanosti dok je propusnost povećana 3% za fiksne i 7% za mobilne mreže [3]. Slične rezultate zabilježila je i Meta (Facebook), čija istraživanja pokazuju da QUIC značajno smanjuje broj pogrešaka pri učitavanju sadržaja te skraćuje vrijeme odziva aplikacija na mobilnim mrežama gdje su gubici paketa učestali [4]. Također, vodeće mreže za isporuku sadržaja poput Cloudflarea navode QUIC kao ključan faktor za optimizaciju korisničkog iskustva na globalnoj razini, ističući prednosti brze (0-RTT) uspostave veze i bolje kontrole zagušenja [5]. Standardizacijom QUIC-a kroz RFC 9000 od strane IETF-a, on prestaje biti eksperimentalni projekt i postaje temelj HTTP/3 protokola [2].

Cilj ovog rada je kroz kontrolirane simulirane scenarije analizirati koliko te teoretske i već industrijski implementirane prednosti QUIC-a doista utječu na performanse u usporedbi s tradicionalnim TCP-om koristeći specifične testove koji mjere latenciju, propusnost, paralelizam tokova i ponašanje protokola pod mrežnom degradacijom u kontekstu posluživanja web sadržaja.

2. METODOLOGIJA ISTRAŽIVANJA
2. RESEARCH METHODOLOGY
2.1. USPOSTAVA TESTNE OKOLINE
2.1. ESTABLISHMENT OF A TEST ENVIRONMENT

Istraživanje je provedeno u kontroliranom izoliranom okruženju korištenjem Docker tehnologije kontejnerizacije (verzija 28.1.1), čime je osigurana ponovljivost eksperimenta i eliminiran utjecaj vanjskih mrežnih čimbenika.

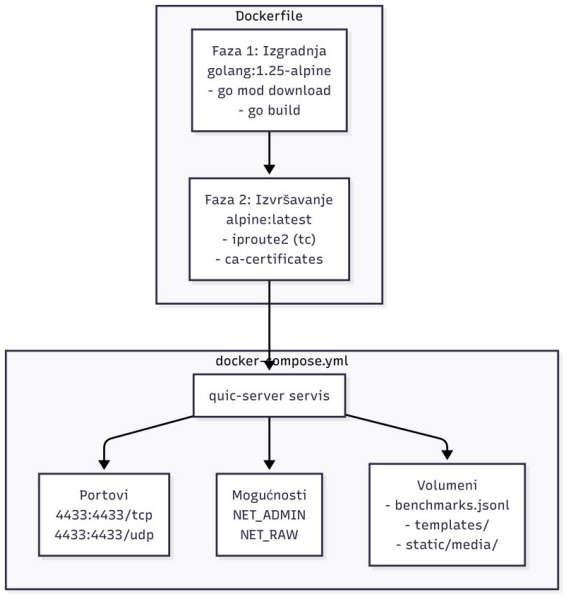


Slika 3 *Arhitektura testne okoline (autorski rad)*

Figure 3 *Architecture of the test environment (source: author)*

Arhitektura sustava sastoji se od dvije ključne komponente: poslužitelja i klijenta, povezanih unutar iste Docker mreže.

Poslužiteljska strana implementirana je u programskom jeziku Go (verzija 1.22), koristeći biblioteku quic-go za podršku HTTP/3 protokolu, te standardnu net/http bilblioteku za podršku HTTP/2 protokola. Za klijentsku stranu korišten preglednik Mozilla Firefox na samom host računalu. Firefox ima stabilnu HTTP/3 podršku od verzije 88 (2021.), no podrška za QUIC protokol se mora uključiti u postavkama preglednika. Za host je upotrebljen HP EliteBook 830 G7 (177D3EA), procesora Intel i7-10510U s 32GB RAM-a, NVMe SSD WDC PC SN630 od 512GB.



Slika 4 *Struktura Docker kontejnera (autorski rad)*

Slika 4 *Docker container structure (source: author)*

Kontejner se bazira na Alpine Linux distribuciji (oko 5 MB). Dodijeljena mu je NET_ADMIN ovlast jer je to preduvjet za izvršavanje tc naredbi. Isto tako, kontejneru je dodijeljena i NET_RAW ovlast koja omogućuje kreiranje IFB (engl. *Intermediate Functional Block*) virtualnog mrežnog uređaja potrebnog za dvosmjernu simulaciju mrežnih uvjeta. Docker Compose je konfiguracija definirana u YAML datoteci i pokreće cijeli sustav jednom naredbom, *docker-compose up --build -d*. Dodatno, na host sustavu potrebno je učitati IFB kernel modul naredbom *modprobe ifb*. Konfiguracija mapira port 4433 za oba transportna protokola (TCP i UDP) na host Dockera, montira datoteku s rezultatima mjerenja (benchmarks.jsonl) kako bi se podaci testiranja trajno pohranili izvan kontejnera, te montira predloške i medijske datoteke (Slika 3).

Kako bi se simulirali realni mrežni uvjeti, na mrežnom sučelju kontejnera korišten je alat tc (*Linux traffic control*) s modulom NetEem (*network emulator*), koji omogućuje precizno uvođenje latencije i gubitka paketa [6] (Slika 4).

2.2. NAČIN PRIKUPLJANJA I MJERENJA PODATAKA

2.2. METHOD OF DATA COLLECTION AND MEASUREMENT

Za usporedbu performansi TCP i QUIC protokola u ovom istraživanju fokus je stavljen na prikupljanje četiri ključne skupine metrika koje obuhvaćaju različite aspekte mrežne komunikacije:

1. Latencija i varijacija kašnjenja.
U prvoj skupini metrika, istraživanje koristi tri zasebna mjerenja kako bi se u potpunosti obuhvatila dinamika kašnjenja:

- Latencija - mjeri se osnovno mrežno kašnjenje paketa. Ova metrika služi kao indikator sirove brzine prijenosa na transportnom sloju i pokazuje koliko mrežni scenariji (latencija uvedena putem *netema*) utječe na kretanje podataka.
- Varijacija kašnjenja (engl. *jitter*) - mjeri se stabilnost veze, odnosno odstupanje u vremenu dolaska paketa. Ova metrika je ključna za razumijevanje predvidljivosti protokola. Visok jitter izravno narušava kvalitetu usluge, čak i ako je prosječna latencija niska.
- Vrijeme do prvog bajta TTFB (engl. *Time To First Byte*) - ova metrika mjeri ukupno vrijeme od slanja zahtjeva klijenta do primitka prvog bajta odgovora s aplikacijske razine. Za razliku od same mrežne latencije, TTFB uključuje vrijeme potrebno za uspostavu veze (handshake) i procesuiranje na poslužitelju, čime daje najtočniji prikaz inicijalne odzivnosti protokola.

2. Mjeri se efektivna brzina prijenosa podataka u oba smjera:

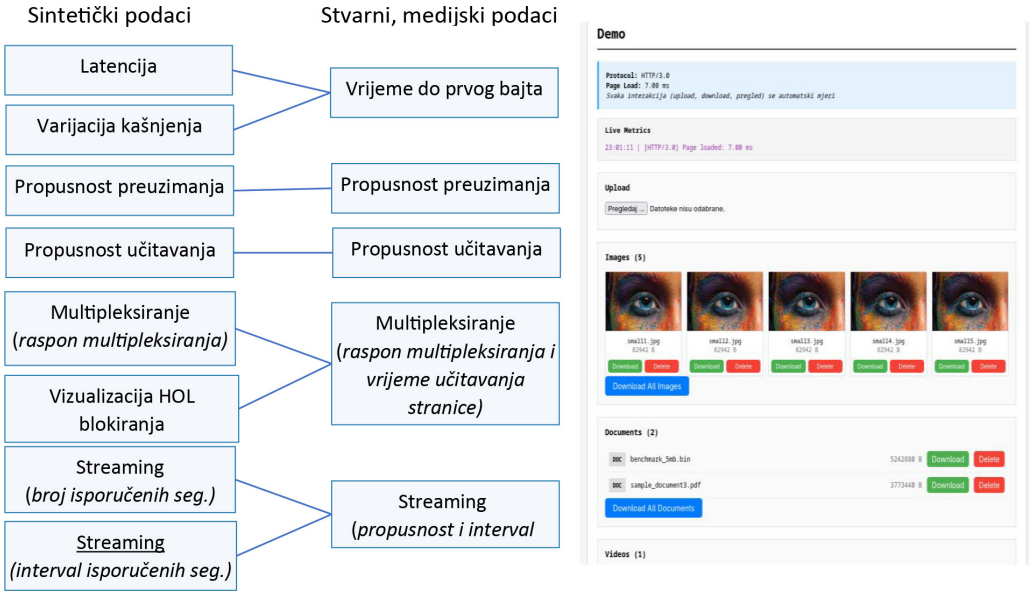
- Analizira se kako mehanizmi kontrole zagušenja (poput TCP Cubic i QUIC BBRv2) reagiraju na gubitke paketa.
- Metrika uključuje validaciju s alternativnim Chromium klijentom kako bi se izolirao utjecaj klijentske implementacije (programskih biblioteka) na ukupnu propusnost, posebno u otežanim mrežnim uvjetima.

3. Analizira se efikasnost protokola pri obradi istodobnih (engl. *concurrent*) zahtjeva:

- Mjeri se vrijeme završetka prijenosa više neovisnih resursa unutar iste sesije kako bi se utvrdila brzina paralelnog procesiranja.
- Fokus je na dokazivanju eliminacije HOL blokiranja kod QUIC-a, gdje gubitak paketa u jednom streamu ne bi smio zaustaviti isporuku podataka u drugim paralelnim streamovima. To se vizualizira kroz analizu vremenske propusnosti, koja detektira nagle zastoje i periode neaktivnosti uzrokovane retransmisijama na transportnom sloju.

4. Analiza performansi unutar jednog dugotrajnog toka podataka:

- Prati se broj isporučenih segmenata i vremenski intervali između njihove dostave klijentu. Ova metrika služi kao ključni parametar za aplikacije za video prijenos u stvarnom vremenu, gdje je ravnomjerna isporuka podataka važnija od vršne propusnosti, jer izravno utječe na stabilnost reprodukcije i veličinu međuspremnika (engl. *buffer*).



Slika 5 *Struktura testova i prikupljene metrike s prikazom testne mrežne aplikacije (autorski rad)*

Figure 5 *Test structure and collected metrics with the test web application (source: author)*

Ove metrike realizirane su kroz devet testova u kojima se mjeri latencija, varijacija latencije, propusnost preuzimanja i prijenosa na poslužitelj, ponašanje pri istovremenim zahtjevima, isporuka u dijelovima te opterećenje protokola. Osam testova koristi sintetičke podatke kontrolirane veličine kako bi izolirali pojedina protokolna svojstva, a deveti test ponavlja ključna mjerenja nad stvarnim multimedijskim sadržajem (fotografije od ~3.4 MB) kako bi se potvrdilo da rezultati testova nad sintetičkim podacima vrijede i u realnim uvjetima (Slika 5).

Svaki test ima pripadajuće metrike koje se agregiraju i zapisuju. Rezultati mjerenja potom se uspoređuju i tumače. Za svaki od provedenih testova mjerenja su obuhvatila četiri specifična mrežna scenarija: idealne mrežne uvjete bez degradacije (engl. *baseline*), scenarij s gubitkom paketa od 2 % bez dodatne latencije (2% *loss*), realistični scenarij (*realistic*) s 25 ms kašnjenja i 2 % gubitka, te scenarij otežanih uvjeta (*bad network*) s 50 ms kašnjenja i 5 % gubitka paketa (Tablica 1).

Ovakav pristup omogućio je izravnu usporedbu ponašanja TCP (HTTP/2) i QUIC (HTTP/3) protokola u identičnim, ali kontrolirano degradiranim mrežnim uvjetima.

Svaki test pokreće se iz klijentskog preglednika u privatnom načinu rada čime se osigurava da preglednik kreće od nule. Nema pohranjene TCP ili QUIC veze iz prethodnog testa, HTTP predmemorija je prazna te kolačići ne postoje. Firefox preglednik je konfiguriran da prihvaća samopotpisane certifikate (engl. *selfsigned certificate*) jer poslužitelj koristi vlastiti TLS certifikat.

Svaki test odrađen je u minimalno 30 iteracija po mrežnom scenariju, za svaki protokol. Kao mjere centralne tendencije prikazani su aritmetička sredina i medijan (P50). Medijan je otporniji na ekstremne vrijednosti, dok aritmetička sredina ulazi u izračun intervala pouzdanosti i veličine efekta. Uz njih, prikazan je i 95. percentil (P95) koji pokazuje ponašanje u najgorim slučajevima. Interval pouzdanosti od 95% izračunat je prema formuli:

$$CI = \bar{x} \pm t * SE, \text{ gdje je } SE = \sigma/\sqrt{n}$$

Svi testovi najprije se izvode za jedan protokol. Tek kad je prikupljen dovoljan broj uzoraka, ista se procedura ponavlja za drugi protokol pod identičnim uvjetima, bez promjena u konfiguraciji sustava ili mrežnog okruženja. Za početni broj uzoraka odabran je središnji granični teorem koji s 30 uzoraka konvencionalno pojednostavljuje analizu. Veći broj uzoraka prikazao je uže intervale pouzdanosti. Kod testova propusnosti preuzimanja i učitavanja bilo je dovoljno izvršiti tridesetak mjerenja jer daljnje uzorkovanje nije pokazalo odstupanje od 95% CI što ukazuje na pouzdanost broja uzoraka. Kod drugih mjerenja je izvršeno preko sto uzoraka za svaki protokol (primjeri mjerenja latencije, vremena do prvog bajta, streaming intervala i raspona multipleksiranja) te je interval pouzdanosti vrlo uzak.

Mjerenja su automatizirana korištenjem skripti koje su putem *Performance API* i *Fetch API* sučelja prikupljale podatke o vremenu odziva i brzini prijenosa.

Tablica 1 *Mrežni scenariji konfigurirani u tc/netem modulu*

Table 1 *Network scenarios configured in the tc/netem module*

Scenarij	tc naredba (oba smjera)	Latencija (po smjeru)	RTT (ukupni)	Varijacija	Gubitak paketa (po smjeru)
Bez degradacije	tc qdisc del dev eth0 root	0 ms	0 ms	-	0%
Gubitak paketa	netem loss 2%	0 ms	0 ms	-	2% ↕
Realistični	netem delay 25ms 10ms loss 2%	25 ms	~50 ms	±10 ms	2% ↕
Loši uvjeti	netem delay 50ms 15ms loss 5%	50 ms	~100 ms	±15 ms	5% ↕

2.3. OGRANIČENJA METODOLOGIJE
2.3. LIMITATIONS OF THE METHODOLOGY

Klijent i poslužitelj dijele isto računalo. Pod opterećenjem se natječu za procesor i memoriju, što može iskriviti mjerenja, naročito kod testova s visokim stupnjem paralelizma poput testa multipleksiranja ili vizualizacije HOL blokiranja. Uz to, u virtualiziranim ili dijeljenim okruženjima kod testova s većim datotekama i duljim prijenosima, dugotrajno korištenje diska može dovesti do I/O zagušenja, što unosi dodatnu varijabilnost u mjerenja propusnosti neovisnu o samom protokolu.

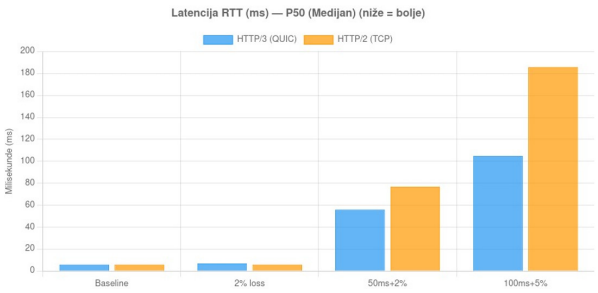
Nadalje, tc/netem simulira samo dio ponašanja stvarnih mreža. Alat dodaje kašnjenje i gubitak paketa, ali ne reproducira pojave prisutne u stvarnim mrežama poput prekomjernog punjenja međuspremnika na usmjerivačima (engl. *bufferbloat*), konkurentnog prometa drugih korisnika na mreži ni otkrivanja maksimalne veličine paketa na putu (engl. *Path MTU Discovery*) [7]. Razlike između protokola izmjerene u ovom radu odražavaju ponašanje pod kontroliranim uvjetima, ne nužno i pod stvarnim mrežnim opterećenjem. Sva mjerenja obuhvaćaju performanse na već uspostavljenoj vezi. QUIC-ov 0-RTT rukovanje, jedna od istaknutijih prednosti protokola, nije obuhvaćeno testovima jer preglednik sam upravlja životnim ciklusom veze. Klijentski JavaScript nema pristup tom koraku.

Cilj eksperimenta nije vjerna simulacija produkcijskog okruženja nego izolirana usporedba TCP-a i QUIC-a pod jednakim uvjetima. Za tu svrhu opisana metodologija pruža dovoljnu razinu kontrole.

3. REZULTATI I RASPRAVA
3. RESULTS AND DISCUSSION

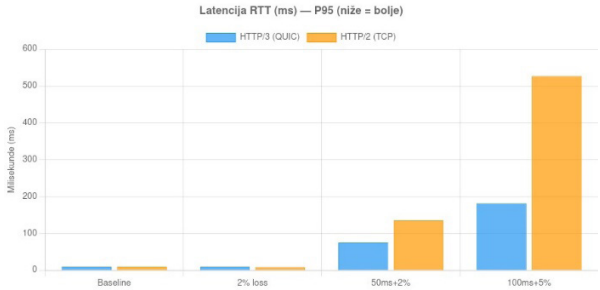
Zbog opsežnosti prikupljenih podataka, u ovom poglavlju fokus je stavljen na ključne metrike koje najjasnije ilustriraju arhitektonske razlike između HTTP/2 (TCP) i HTTP/3 (QUIC) protokola kroz četiri definirana mrežna scenarija.

3.1. LATENCIJA, VARIJACIJA KAŠNJENJA I VRIJEME DO PRVOG BAJTA
3.1. LATENCY, JITTER AND TIME TO FIRST BYTE



Slika 6 *Utjecaj mrežne degradacije na latenciju (autorski rad)*

Figure 6 *Impact of network degradation on latency (source: author)*



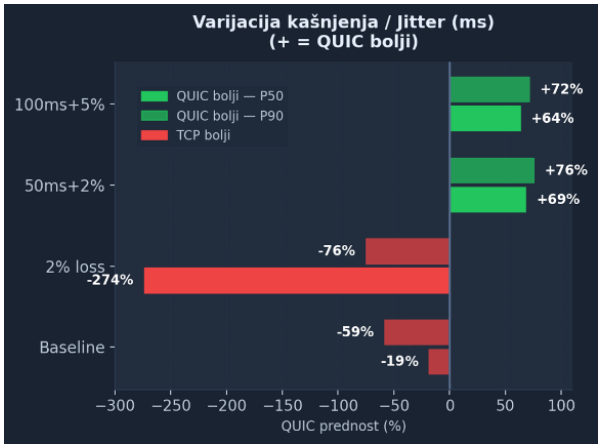
Slika 7 *Prikaz 95. percentila latencije (autorski rad)*

Figure 7 *P95 (95th percentile) latency (source: author)*

U idealnim mrežnim uvjetima (*baseline*), oba protokola pokazuju gotovo identične performanse s latencijom od približno 6 ms. Međutim, uvođenjem mrežnih ograničenja, razlika postaje progresivno vidljivija. Zanimljivo je primijetiti da je u scenariju s izoliranim gubitkom paketa od 2 % (bez dodatnog kašnjenja), HTTP/3 pokazao marginalno sporiji odziv (medijan 7 ms) u odnosu na HTTP/2 (6 ms). Ovo se pripisuje procesnom opterećenju QUIC protokola koji se izvodi u korisničkom prostoru (engl. *user-space*), za razliku od TCP-a koji je duboko integriran u jezgru operacijskog sustava.

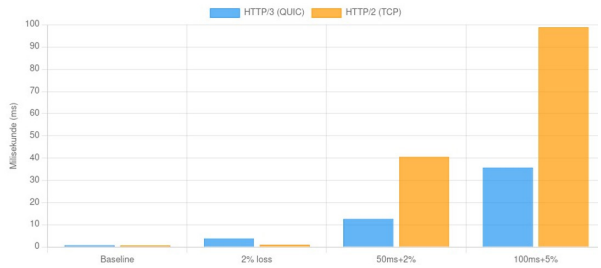
Prava prednost QUIC-a dolazi do izražaja tek kada se gubitku paketa pridruži i latencija. U scenariju s 50 ms kašnjenja i 2 % gubitka, medijan latencije za HTTP/3 iznosi 56 ms, dok za HTTP/2 raste na 77 ms. U najtežim uvjetima (*Bad Network*), razlike postaju drastične: na

95. percentilu (p95) latencija za HTTP/2 iznosi enormnih 527 ms, dok QUIC zadržava stabilnost na 182 ms (slike 6 i 7).



Slika 8 *Varijacija kašnjenja (ms) (autorski rad)*

Figure 8 *Jitter (ms) (source: author)*

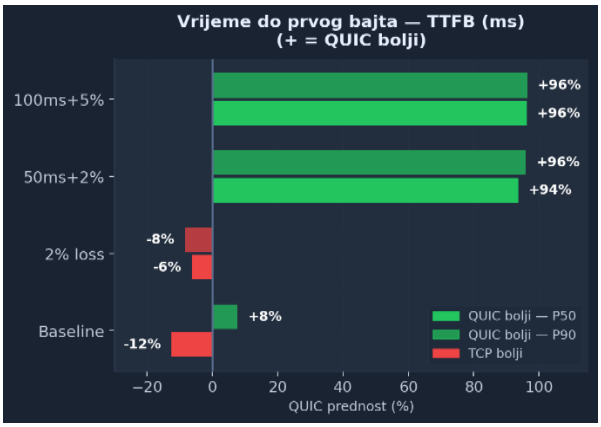


Slika 9 *Rezultat mjerenja varijacije kašnjenja (autorski rad)*

Figure 9 *Jitter measurement results (source: author)*

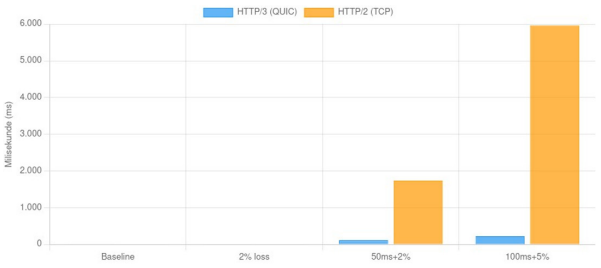
Varijacija kašnjenja (engl. *jitter*) prati sličan obrazac (slike 8 i 9). Iako u čistoj mreži QUIC ima nešto veći *jitter* (0.90 ms naspram 0.76 ms), u realnim uvjetima degradacije (50 ms latencije i 2 % gubitka) *jitter* kod HTTP/2 raste na 40.7 ms, dok kod QUIC-a pada na 12.8 ms. Ovi podaci potvrđuju da QUIC-ovi mehanizmi brzog oporavka nadmašuju inicijalno procesno opterećenje čim cijena retransmisije u mreži postane visoka.

Posebno se ističe nesrazmjer u varijaciji kašnjenja unutar scenarija s izoliranim gubitkom paketa od 2 % (bez dodatne latencije), gdje QUIC bilježi znatno veći *jitter* u usporedbi s TCP-om. Razlog leži u arhitektonskoj razlici: TCP-ovo upravljanje retransmisijama unutar jezgre operacijskog sustava omogućuje gotovo trenutnu reakciju u mrežama niske latencije. S druge strane, QUIC-ova implementacija u korisničkom prostoru uvodi mjerljivo procesno kašnjenje prilikom obrade gubitaka, što u ultra-brzim mrežnim uvjetima rezultira visokom relativnom varijacijom. Međutim, čim se u mrežu uvede bazna latencija (scenariji *realistic* i *bad network*), ta prednost kernela nestaje, a QUIC-ovi optimizirani algoritmi oporavka preuzimaju dominaciju, stabilizirajući *jitter* na razinama i do tri puta nižim od onih kod TCP-a.



Slika 10 *Vrijeme do prvog bajta (autorski rad)*

Figure 10 *Time to first byte (source: author)*



Slika 11 *Rezultat mjerenja vremena do prvog bajta (autorski rad)*

Figure 11 *Time to first byte measurement results (source: author)*

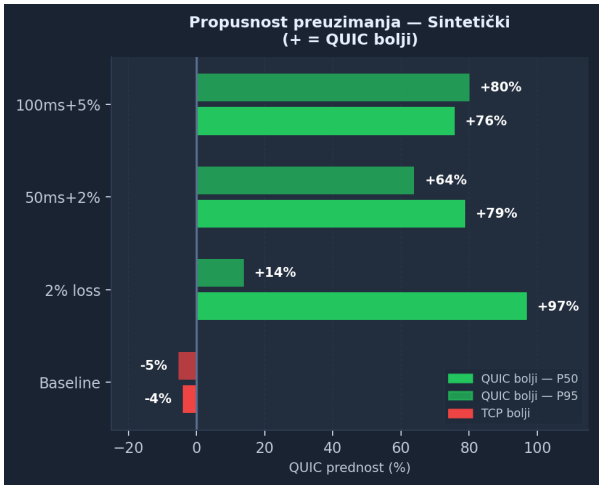
Validacija kroz testove nad stvarnim podacima (fotografija ~3.4 MB) potvrđuje sintetičke nalaze. Metrika vremena do prvog bajta (TTFB) u otežanim mrežnim uvjetima pokazuje da QUIC osigurava bržu inicijalnu isporuku sadržaja, što izravno utječe na percepciju brzine učitavanja kod krajnjeg korisnika (slike 10 i 11).

3.2. PROPUSNOST PREUZIMANJA I UČITAVANJA

3.2. DOWNLOAD AND UPLOAD THROUGHPUT

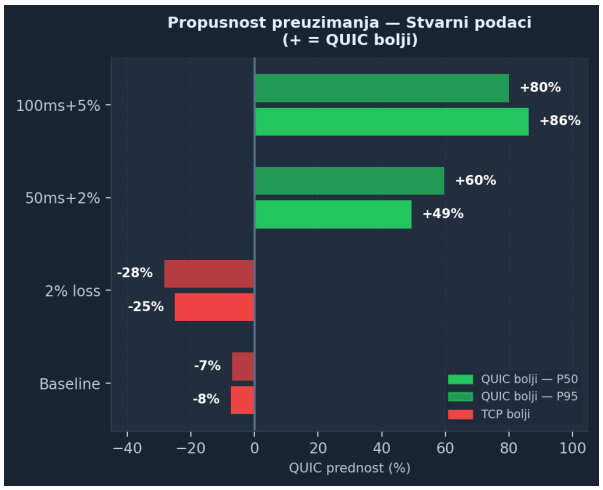
Analiza propusnosti (engl. *throughput*) otkrila je značajne razlike u načinu na koji protokoli

upravljaju mehanizmima kontrole zagušenja pod mrežnim opterećenjem.



Slika 12 *Propusnost preuzimanja sintetičkih podataka (autorski rad)*

Figure 12 *Synthetic data download throughput (source: author)*

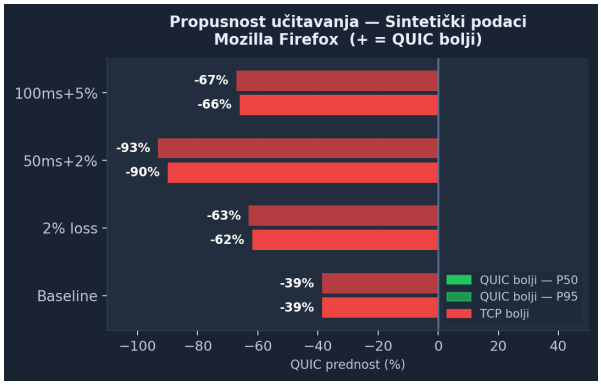


Slika 13 *Propusnost preuzimanja stvarnih podataka (autorski rad)*

Figure 13 *Real data download throughput (source: author)*

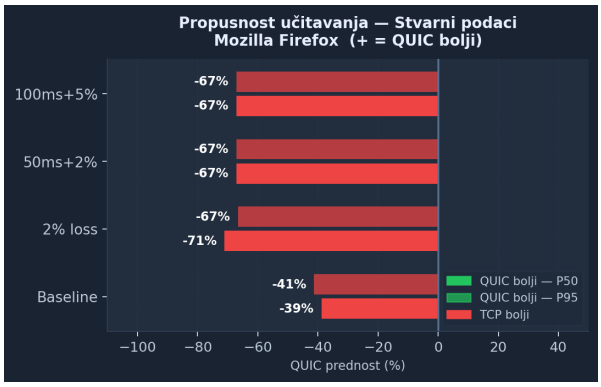
U idealnim mrežnim uvjetima (*baseline*), oba protokola pokazuju vrhunske performanse, pri čemu HTTP/2 ostvaruje blagu prednost od 4 % do 7.5 %. Međutim, čim se uvede gubitak paketa od 2 %, dolazi do drastične divergencije. U sintetičkim testovima HTTP/3 zadržava stabilnih 650 Mbit/s s vrlo uskom distribucijom (±16 Mbit/s), dok HTTP/2 pada na 330 Mbit/s uz visoku varijabilnost (±90 Mbit/s) (Slika 12).

Zanimljivo je da kod prijenosa stvarnih datoteka TCP u početku (2 % gubitka) pruža nešto veći otpor, no u najtežim uvjetima (*Bad Network*) QUIC postiže i do 86 % veću propusnost (Slika 13). Ekstremni pad TCP-a sa 700 Mbit/s na manje od 1 Mbit/s zorno demonstrira problem s mehanizmom "kliznog prozora" (engl. *sliding windows*) i retransmisijama, dok QUIC mehanizmi (BBRv2) osiguravaju kontinuitet isporuke. Ključna prednost HTTP/3 protokola leži u BBRv2 algoritmu koji, za razliku od TCP Cubica, gubitak paketa ne interpretira automatski kao zagušenje mreže, već kontinuirano procjenjuje stvarni kapacitet mrežne staze, čime se izbjegava nepotrebno usporavanje prijenosa u degradiranim uvjetima [8].



Slika 14 *Propusnost učitavanja sintetičkih podataka – Firefox (autorski rad)*

Figure 14 *Synthetic data upload throughput – Firefox (source: author)*

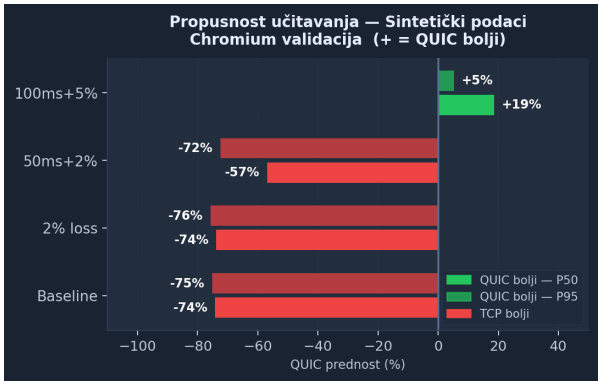


Slika 15 *Propusnost učitavanja stvarnih podataka – Firefox (autorski rad)*

Figure 15 *Real data upload throughput - Firefox(source: author)*

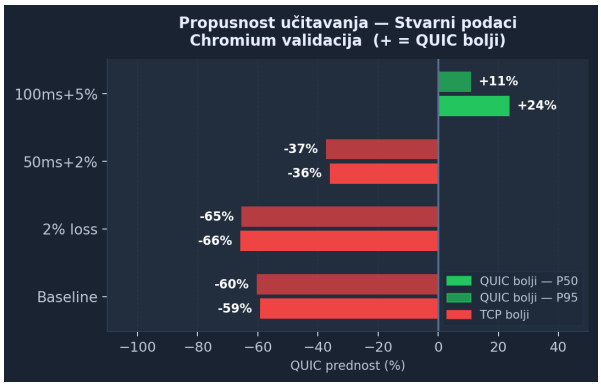
Rezultati učitavanja podataka u Firefoxu otkrivaju neočekivan trend: HTTP/3 (QUIC) kontinuirano ostvaruje značajno nižu propusnost u usporedbi s HTTP/2 (TCP) protokolom (Slike 14 i 15). To je neočekivano jer QUIC-ovi mehanizmi oporavka dolaze do izražaja upravo pod degradacijom, ali anomalija učitavanja se ne smanjuje. Uzrok

anomalije nije bio jasan iz trenutno dostupnih podataka. S obzirom na to da je klijent taj koji upravlja slanjem podataka i kontrolom zagušenja, ovi rezultati sugeriraju da pad performansi nije inherentno ograničenje QUIC protokola, već vjerojatna posljedica specifične implementacije unutar Firefox preglednika. Ovo opažanje postavilo je temelje za nužnu validaciju putem alternativnog klijenta kako bi se izolirao utjecaj softverskog stacka.



Slika 16 *Propusnost učitavanja sintetičkih podataka – Chromium (autorski rad)*

Figure 16 *Synthetic data upload throughput – Chromium (source: author)*



Slika 17 *Propusnost učitavanja stvarnih podataka – Chromium Chromium (autorski rad)*

Figure 17 *Real data upload throughput – Chromium (source: author)*

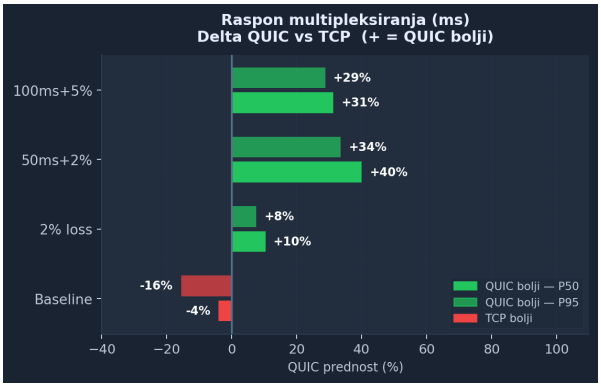
Ponovljeni testovi s preglednikom Chromium potvrđuju pretpostavku o utjecaju klijentske implementacije. Ključna razlika između preglednika je u QUIC implementaciji. Firefox koristi *neqo* (Mozilla, Rust) s Cubic algoritmom kontrole zagušenja[9], dok Chromium koristi *quiche* (Google, C++) s BBRv2 algoritmom[10].

U najtežem sintetičkom scenariju (100 ms latencije i 5 % gubitka), Chromiumov HTTP/3 ostvaruje 18,6 % bolju propusnost od HTTP/2 (Slika 16). Nad stvarnim podacima razlika je još drastičnija, gdje HTTP/3 postiže čak 51 % veću propusnost (Slika 17). Ova validacija je ključna jer dokazuje da je QUIC kao protokol arhitektonski superioran za oba smjera prijenosa, ali njegova stvarna učinkovitost uvelike ovisi o zrelosti i optimiziranosti klijentskog rješenja.

3.3. MULTIPLEKSIRANJE I HOL BLOKIRANJE

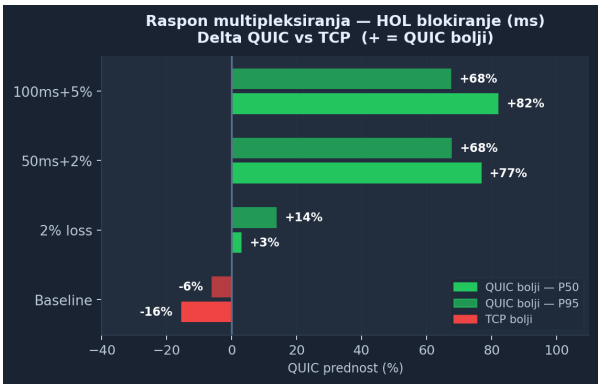
3.3. MULTIPLEXING AND HOL BLOCKING

Testovi multipleksiranja potvrdili su da QUIC uspješno eliminira HOL blokiranje na transportnom sloju. Gubitak jednog paketa u QUIC-u utječe samo na pripadajući stream, dok kod TCP-a zaustavlja sve ostale podatke dok se izgubljeni segment ne obnovi.



Slika 18 *Raspon multipleksiranja (autorski rad)*

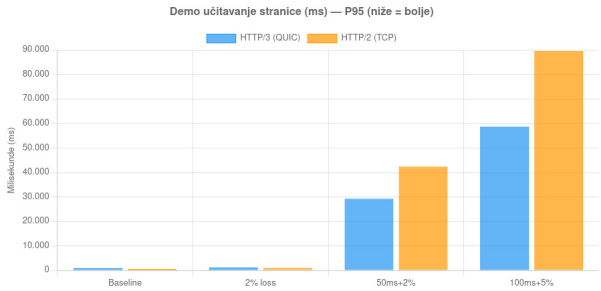
Figure 18 *Multiplexing range (source: author)*



Slika 19 *Raspon multipleksiranja testom HOL blokiranja (autorski rad)*

Figure 19 *Multiplexing range with HOL blocking test (source: author)*

Na čistoj mreži (*baseline*) raspon završetka paralelnih tokova očekivano je gotovo identičan za oba protokola. Bez gubitka paketa nema blokiranja, pa nema ni mjerljive razlike. Međutim, već uvođenje latencije od 50ms i 2% gubitaka za sintetički test ukazuje na 40 postotnu razliku (Slika 18). S realnim testom prijenosa slika izmjerena razlika je gotovo pet puta veća u korist QUIC protokola (Slika 19). Razlog leži u broju i veličini resursa: s više paralelnih tokova raste vjerojatnost da gubitak paketa pogodi barem jedan od njih, a veće datoteke drže TCP vezu blokiranom dulje nakon svakog ponovnog slanja.



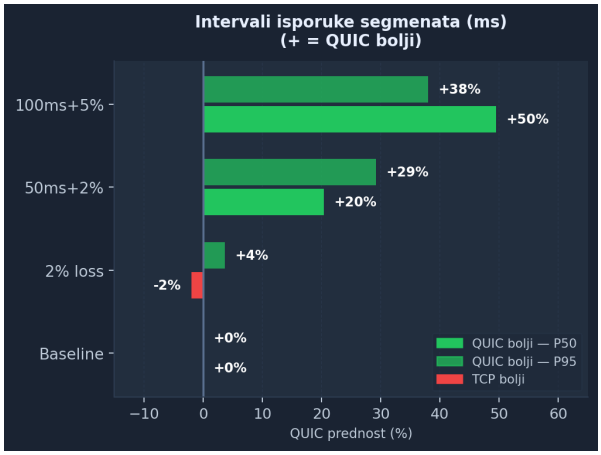
Slika 20 *Test učitavanja mrežne stranice (autorski rad)*

Figure 20 *Web page loading test (source: author)*

Prilikom testa učitavanja mrežne stranice na početnoj mreži gotovo i nema razlike. U uvjetima kašnjenja vrijeme učitavanja stranice iznosi 1947 ms za HTTP/3 u odnosu na 3036ms za HTTP/2, dok u najgorim uvjetima 3880 ms naspram 9227 ms. Razliku od pet sekundi krajnji korisnik svakako osjeti. Postotne razlike između protokola manje su nego kod raspona multipleksiranja jer ukupno učitavanje stranice obuhvaća i korake poput parsiranja HTML-a i pokretanja zahtjeva. Na 95. percentilu ukupno vrijeme učitavanja stranice za HTTP/2 dosegne 89538 ms dok HTTP/3 ostvaruje 58768 ms. U najgorem slučaju korisnik čeka gotovo 90 sekundi na učitavanje svih stranica (Slika 20).

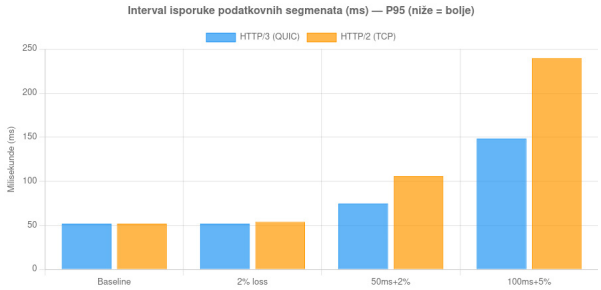
3.4. KONZISTENTNOST STREAMINGA
3.4. STREAMING CONSISTENCY

U testovima učitavanja streaminga na čistoj mreži nema mjerljivih razlika za oba protokola. Pod realističnim uvjetima slanja 20 segmenata HTTP/3 prima više segmenata nego ih server šalje. HTTP/2 ostaje na 20 ili manje. Razlog leži u načinu kako protokoli obrađuju izgubljene segmente. TCP čeka ponovno slanje izgubljenog segmenta i onda isporuči sve nakupljene segmente odjednom, stoga klijent dobije manji broj većih segmenata. QUIC isporučuje čim su podaci dostupni unutar toka, pa isti sadržaj stigne u više manjih dijelova s kraćim razmakom (Slika 21).



Slika 21 *Interval isporuke segmenata (autorski rad)*

Figure 21 *Segment delivery interval (source: author)*



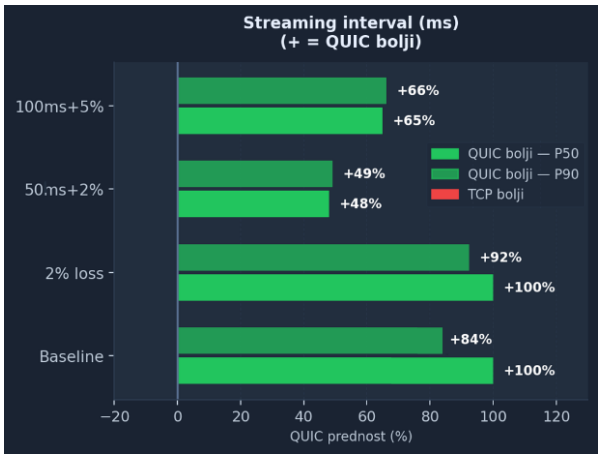
Slika 22 *Prikaz 95. percentila intervala isporuke segmenata (autorski rad)*

Figure 22 *Display of the 95th percentile of the segment delivery interval (source: author)*

Za streaming scenarije ključan je 95. percentil intervala između segmenata. Pod najgorim uvjetima HTTP/2 dosegne 240 ms između dva uzastopna segmenta, gotovo peterostruko duže od zadanih 50 ms. Za video stream to je zamjetna stanka. HTTP/3 drži 95. percentil na 149 ms što je gotovo 40% bolji rezultat (Slika 22).

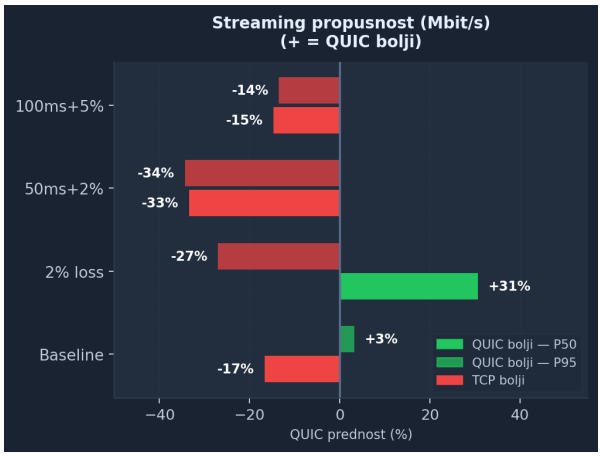
Za streaming aplikacije konzistentnost isporuke podjednako je bitna kao i brzina. Streaming metrika mjeri korisničko iskustvo. Preglednik preuzima sadržaj putem *ReadableStream* sučelja i bilježi interval između dolazećih segmenata te propusnost po segmentu. Na čistoj mreži (*baseline*) interval je ispod jedne milisekunde za oba protokola. Pod izoliranim gubitkom HTTP/2

prikazuje razliku između medijana i prosjeka mjerenja, što ukazuje na distribuciju: većina segmenata stiže brzo, ali kad TCP izgubi paket, čekanje na ponovno slanje stvori rijetke ali duge pauze koje podignu prosjek mjerenja (Slika 23). HTTP/3 protokol ne prikazuje takvu razliku. Pri najgorem scenariju HTTP/2 ima i do pola sekunde pauze između segmenata tijekom reprodukcije videa. HTTP/3 ostaje na 191 ms što je razlika između privremenog zastoja reprodukcije videa i prihvatljive degradacije kvalitete. HTTP/3 isporučuje sadržaj u više manjih segmenata s kraćim intervalom, pa je propusnost po pojedinom segmentu niža, ali je isporuka ravnomjernija (Slika 24). Za streaming to je bolji obrazac jer korisnik ne treba visoku propusnost u rijetkim naletima nego stabilnu isporuku bez pauza.



Slika 23 *Streaming interval isporuke segmenata (autorski rad)*

Figure 23 *Streaming segment delivery interval (source: author)*



Slika 24 *Streaming propusnost isporuke segmenata (autorski rad)*

Figure 24 *Streaming segment delivery throughput (source: author)*

4. ZAKLJUČAK
4. CONCLUSION

Provedeno istraživanje potvrdilo je da tranzicija s HTTP/2 na HTTP/3 (QUIC) ne donosi samo marginalna poboljšanja, već predstavlja temeljni zaokret u načinu na koji se transportni sloj nosi s nesavršenostima mreže. Ključni nalazi sugeriraju da je "cijena" QUIC protokola, izražena kroz inicijalno procesno opterećenje u korisničkom prostoru, zanemariva u usporedbi s dobicima koje donosi u degradiranim mrežnim uvjetima. Ovakvi rezultati nadovezuju se na prethodna istraživanja autora u kontekstu SMB protokola [11], gdje je također uočena korelacija između povećane potrošnje CPU resursa (zbog šifriranja i kompleksnosti transportnog sloja) i superiorne otpornosti na mrežne smetnje. Dok je u prethodnom radu fokus bio na stabilnosti prijenosa datoteka unutar poduzeća, ovaj rad potvrđuje da se isti obrasci ponašanja manifestiraju i u širem kontekstu mrežnog prometa (HTTP/3), čime se QUIC pozicionira kao univerzalno rješenje za prevladavanje ograničenja tradicionalnog TCP stoga. Analiza latencije i TTFB-a jasno je pokazala da QUIC-ova eliminacija HOL blokiranja na transportnom sloju transformira korisničko iskustvo u scenarijima s gubicima paketa, gdje p95 latencija ostaje i do tri puta niža u usporedbi s TCP-om. Iako je TCP i dalje dominantan u stabilnim mrežama ultra-niske latencije zbog efikasnosti unutar jezgre operacijskog sustava, rezultati propusnosti pod mrežnim stresom ukazuju na superiornost QUIC-ovih algoritama kontrole zagušenja, poput BBR-a. Poseban doprinos ovog rada je validacija utjecaja klijentske implementacije. Dokazano je da niska propusnost učitavanja nije inherentna mana QUIC-a, već odraz zrelosti softverskog *stacka*, što otvara put prema daljnjim optimizacijama preglednika. U konačnici, QUIC se nameće kao nužan standard za suvremeni web, posebno u mobilnim i zasićenim te promjenjivim mrežnim okruženjima gdje je predvidljivost performansi važnija od teoretskih maksimuma u idealnim uvjetima.

5. REFERENCE

5. REFERENCES

[1.] J. Erman, V. Gopalakrishnan, R. Jana, and K. K. Ramakrishnan, “Towards a SPDY’ier mobile web?,” IEEE/ACM Transactions on Networking, vol. 23, no. 6, pp. 2010-2023, Dec. 2015. doi: 10.1109/TNET.2015.2462737

[2.] J. Iyengar and M. Thomson, “QUIC: A UDP-based multiplexed and secure transport,” Internet Engineering Task Force, RFC 9000, May 2021. doi: 10.17487/RFC9000

[3.] D. Schinazi, F. Yang, I. Swett: „Chrome is deploying HTTP/3 and IETF QUIC“, Chromium Blog, [Online] Pristupano: 20.02.2026. Dosutpno: https://blog.chromium.org/2020/10/chrome-is-deploying-http3-and-ietf-quic.html

[4.] M. Joras, Y. Chi: „How Facebook is bringing QUIC to billions“ [Online] Pristupano: 20.02.2026. Dostupno: https://engineering.fb.com/2020/10/21/networking-traffic/how-facebook-is-bringing-quic-to-billions/

[5.] A. Ghedini, R. Lalkaka: „HTTP/3: the past, the present, and the future", Cloudflare Blog, [Online] Pristupano: 24.02.2026. Dostupno: https://blog.cloudflare.com/http3-the-past-the-present-and-the-future/

[6.] S.Hemminger, Linux foundation, [Online]. Pristupano:26.02.2026., Dostupno: https://www.linux.org/docs/man8/tc-netem.html

[7.] J. P. Achara, M. Mohiuddin, W. Saab, R. Rudnik, J.-Y. Le Boudec, and L. Reyes-Chamorro, “T-RECS: A virtual commissioning tool for software-based control of electric grids: Design, validation, and operation,” in Proceedings of the Ninth International Conference on Future Energy Systems (e-Energy ’18), New York, NY, USA: Association for Computing Machinery, Jun. 2018, pp. 303-313. doi: 10.1145/3208903.3208928

[8.] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh i V. Jacobson, “BBR: Congestion-Based Congestion Control,” Communications of the ACM, vol. 60, br. 2, str. 58–66, 2017. [Online]. Pristupano: 26.02.2026. doi:10.1145/3009824

[9.] O. Mansfeld, "The Fast and the Spurious: Congestion Control Experimentation in Firefox's QUIC stack," presented at FOSDEM 2026, Brussels, Belgium, Jan. 31, 2026. [Online]. Pristupano: 26.02.2026. Dostupno: https://fosdem.org/2026/schedule/event/VHW9HJ-firefox-quic-congestion-control/

[10.] K.Lepikhov: "BBR TCP (Bottleneck Bandwidth and RTT)," GÉANT Knowledge Base [Online] Pristupano: 26.02.2026. Dostupno: https://wiki.geant.org/pages/releaseview.action?pageId=121340614

[11.] Mitrović, O.; Tadić, V.: SMB Over QUIC: A Performance Evaluation // 2025 MIPRO 48th ICT and Electronics Convention / Car, Željka; Babić, Snježana; Čićin Šain, Marina (ur.). Piscataway (NJ): IEEE, 2025. str. 919-924. ISSN 1847-3946, doi: 10.1109/MIPRO65660.2025.11132048

AUTORI · *AUTHORS*

• Jakov Vukadin - rođen je 1999. godine u Livnu. Završio je prijediplomski studij Informatike na Tehničkom veleučilištu u Zagrebu gdje je trenutno student završne godine diplomskog studija. Interesi su mu sistemska i mrežna administracija te virtualizacijska okruženja.

Korespondencija · *Correspondence*

javukadin@gmail.com

• Vedran Tadić - Nepromijenjena biografija nalazi u časopisu Polytechnic & Design, svezak 11, br. 2 iz 2023. godine.

Korespondencija · *Correspondence*

vtadic@tvz.hr

• Ognjen Mitrović - Nepromijenjena biografija nalazi u časopisu Polytechnic & Design, svezak 5, br. 2 iz 2017. godine.

Korespondencija · *Correspondence*

ognjen.mitrovic@tvz.hr