Zbornik radova Fakulteta tehničkih nauka, Novi Sad

**UDK: 004.9**
**DOI: https://doi.org/10.24867/07BE29Krajinovic**

# ANALIZA PROTOKOLA HTTP 2.0

# ANALYSIS OF HTTP 2.0

Marko Krajinović, *Fakultet tehničkih nauka, Novi Sad*

**Oblast – RAČUNARSTVO I AUTOMATIKA**

**Kratak sadržaj –** *U ovom radu izvršena je analiza karakteristika komunikacionog protokola HTTP 2.0. Upoređen je ovaj protokol sa prethotnom verzijom protokola - HTTP 1.1. Analizirane su mogućnosti za primenu protokola HTTP 2.0 u serverskim radnim okvirimima Spring Boot, Microsoft ASP.NET Core i Node.js. Izvršena je analiza performansi protokola.*

**Ključne reči:** *Veb, HTTP*

**Abstract** – *In this paper the analysis of HTTP 2.0 communication protocol features has been done. Also, the comparison of this protocol with it's predecessor. Analysis of HTTP 2.0 implementation capabilities in Spring Boot, Microsoft ASP.NET Core, and Node.js server workspaces has been done as well as the analysis of the HTTP 2.0 performances.*

**Keywords:** *Web, HTTP*

## 1. KARAKTERISTIKE HTTP-A

Hypertext Transfer Protocol (HTTP) [1] je aplikativni protokol koji ne čuva stanje sesije, za distribuirane, kolaborativne, hipermedijske informacione sisteme. Predstavlja osnovu komunikacije podataka sa *World Wide Web*-a (WWW) [2].

Hipertekst podrazumeva da datoteke koje se prenose preko HTTP-a mogu sadržati reference na druge datoteke čiji će izbor izazvati dodatne zahteve za prenos. Pored veb stranica koje mogu da se serviraju, bilo koji server sadrži HTTP *daemon*, program koji je predviđen da čeka HTTP zahteve i obrađuje ih kad stignu.

HTTP funkcioniše kao protokol koji se zasniva na zahtev-odgovor mehanizmu u računarskom modelu klijent – server. Na primer, veb *browser* može biti klijent, a aplikacija koja radi na računaru koji *host*-uje veb stranicu može biti server. Klijent šalje serveru Http zahtev. Server, koji pruža resurse kao što su HTML datoteke i drugi sadržaj, ili obavlja druge funkcije u ime klijenta, vraća odgovor klijentu. Odgovor sadrži informacije o statusu završetka zahteva i može takođe da sadrži traženi sadržaj u telu http odgovora.

## 2. ISTORIJSKI RAZVOJ HTTP-a PRE VERZIJE 2.0

Originalni predlog za HTTP Tima Berners-Lee-a bio je osmišljen kako bi pomogao u usvajanju njegove druge ideje koja se tek rodila: *World Wide Web*. Berners Lee je 1991. izložio motivaciju za novim protokolom i nabrojao nekoliko dizajnerskih ciljeva na visokom nivou:

- funkcionalnost prenosa datoteka,
- mogućnost pretraživanja indeksirane arhive hiperteksta,
- pregovaranje o formatu, i
- sposobnost da klijenta uputi na drugi server.

Da bi se teorija dokazala na delu, izgrađen je jednostavan prototip koji je implementirao mali podskup predloženih funkcija:

- zahtev klijenta je jedan niz ASCII znakova,
- zahtev klijenta prekida se znakom za povratak prenosa,
- odgovor servera je HTML, i
- veza se prekida nakon završetka prenosa dokumenta.

Pošto je prototip bio podskup željenog protokola, nezvanično je dobio naziv HTTP 0.9 [3].

### 2.1. HTTP 1.0

Rastuća lista željenih mogućnosti veba i slučajeva njihove upotrebe na javnom vebu brzo su otkrili mnoga osnovna ograničenja HTTP-a 0.9. Potreban je bio protokol koji bi između ostalog mogao da:

- poslužiti više od samo hipertekstualnih dokumenata,
- pruži bogatije meta-podatke o zahtevu i odgovoru, i
- omogući pregovaranje o sadržaju.

Na osnovu ovih zahteva nastao je novi internet standard HTTP 1.0 [4]. U ovom standardu, zaglavlja zahteva i odgovora bila su čuvana kao ASCII karakteri ali sam objekat odgovora je mogao biti bilo koje vrste: HTML datoteka, obična teksualna datoteka, slika ili bilo koji drugi tip sadržaja.

Stoga je deo HTTP-a „hiperteksualni prenos“ postao pogrešan naziv nedugo nakon njegovog uvođenja. HTTP se zapravo brzo razvio u hipermedijski transport ali se originalni naziv zadržao.

### 2.2. HTTP 1.1

U nastavku će biti opisane prednosti ovog protokola u odnosu na prošli HTTP 1.0.

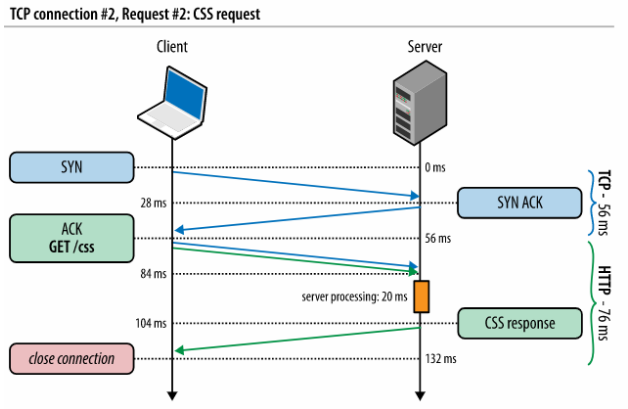
---

**NAPOMENA:**
**Ovaj rad proistekao je iz master rada čiji mentor je bio dr Branko Milosavljević, red. prof.**

### 2.3.1 Prednosti *keep-alive* konekcije

Svaka TCP konekcija počinje sa TCP *three-way handshake*-om, kojem je potreban vremenski period jednog obilaska između klijenta i servera. Ovo je fiksan trošak koji imaju sve http sesije koje nisu trajne odnosno *keep-alive*.

Dolazi se do jednostavne optimizacije: da se ponovo koristi tcp konekcija. Dodavanjem podrške za http trajnu konekciju koja je prikazana na slici 1, omogućeno je to da se eliminiše drugi tcp *three-way handshake* odnosno jedan čitav obilazak sa klijenta na server.




Slika 1. *Keep-alive konekcija*

U slučaju trajne konekcije, vreme koje se uštedi za N zahteva predstavlja (N – 1) * (vreme jednog celog obilaska) u odnosu na primer slanja zahteva bez trajne konekcije.

### 2.3.2 Korišćenje više TCP konekcija

Zbog nedostatka *multiplexing*-a kod Http-a 1.1, *browser* može naivno da stavi u red sve http zahteve na klijentu i šalje jedan za drugim putem jedne trajne veze. U praksi ovo je vrlo sporo. Zbog toga *browser*-i otvaraju više tcp sesija u paraleli. Ograničenje je da može da se otvori maksimalno šest konekcija za istog *host*-a.

Šest konekcija je odabrano bazirano na broju kompromisa koji se moraju uzeti u obzir. Što je veći broj konekcija to su veći opšti troškovi i na serveru i na klijentu. Međutim sa većim brojem konekcija je i veći broj zahteva koji se mogu obraditi u paraleli. Šest konekcija po *host*-u je siguran kompromis.

### 2.3.2 Troškovi protokola

Http 0.9 je bio jednostavan, jednolinijski ASCII zahtev za dobavljanje html dokumenata i imao je minimalan trošak. Http 1.0 je proširio ovaj protokol dodajući pojam zaglavlja zahteva i odgovora kako bi omogućio i jednoj i drugoj strani da razmene dodatne informacije o meta-podacima zahteva i odgovora. Konačno, http 1.1 je učinio ovaj format standardom: zaglavlja se lako proširuju i bilo koji server ili klijent ih može proširiti, i takođe se šalju kao običan tekst kako bi ostali kompatabilni sa prethodnim verzijama http-a.

Svaki zahtev koji je bio iniciran od strane *browser*-a će imati dodatnih 500-800 bajtova Http meta-podataka: *user-agent* string, *accept* i *transfer* zaglavlja koja se retko menjaju, *caching directives*, i tako dalje. Međutim u ovih 500-800 bajtova nisu uračunati HTTP *cookie*-i koji mogu najviše resursa zauzimati. Sa pomenutim *cookie*-ima, http meta-podaci mogu zauzimati i po nekoliko kilobajta dodatnih podataka za svaki http zahtev.

Rast liste http zaglavlja nije samo po sebi loša stvar, jer svako zaglavlje postoji s dobrim razlogom. Međutim, činjenica da su sva http zaglavlja običan tekst (bez ikakve kompresije), može dovesti do visokih troškova za svaki zahtev, što može dovesti do kašnjenja i lošijeg korisničkog doživljaja kod nekih aplikacija.

## 3. HTTP 2.0

Od svog objavljivanja, RFC 2616 je poslužio je kao temelj za neviđeni rast interneta. Milijarde uređaja svih oblika i veličina, od desktop računara do sitnih veb uređaja kao što su telefoni, svakodnevno koriste HTTP za isporuku vesti, video zapisa i miliona drugih veb sadržaja koji se koriste u svakodnevnom životu prosečne osobe.

Ono što je počelo kao jednostavan, jednolinijski protokol za preuzimanje hiperteksta brzo se razvilo u generički transport hipermedija, a sada se, može iskoristiti za isporuku gotovo svakog slučaja korišćenja koji se može zamisliti. Sveprisutnost servera koji mogu da koriste protokol i široka dostupnost klijenata dovele su do toga da su mnoge aplikacije sada dizajnirane isključivo za rad preko HTTP-a.

Jednostavnost HTTP protokola je ono što je omogućilo njegovo prvobitno usvajanje i brz rast. U stvari, sada nije neobično pronaći ugrađene uređaje – senzore, pokretače – koji koriste HTTP kao primaran protokol za kontrolu i podatke. Ali pod težinom sopstvenog uspeha i kako se sve više nastavlja sa migracijom svakodnevnih interakcija na internet – društvene mreže, e-pošta, vesti i video zapisi – takođe je počeo da pokazuje znakove slabosti. Korisnici i veb programeri sada traže reakciju u realnom vremenu i performanse od protokola http 1.1 koje on ne može da isporuči bez nekih izmena samog protokola.

Suočavajući se sa ovim novim izazovima, Http se nastavlja razvijati, i otuda je radna grupa HTTPbis najavila novu inicijativu za Http-om 2.0 [8] početkom 2012. godine. Primarni fokus http-a 2.0 je na poboljšanju transportnih performansi i omogućavanju manjeg kašnjenja i veće propusnosti. Povećanje glavne verzije zvuči kao veliki korak, ali važno je napomenuti da nije pogođena ni jedna semantika protokola na visokom nivou: sva HTTP zaglavlja, vrednosti i slučajevi korišćenja su ostali isti.

Http 2.0 [5] omogučava efikasnije korišćenje mrežnih resursa i smanjenu percepciju kašnjenja tako što uvodi kompresiju polja zaglavlja i omogućava više konkurentnih razmena na istoj konekciji. Tačnije, omogućava preplitanje poruka zahteva i odgovora na istoj konekciji i koristi efikasno kodiranje za http polja zaglavlja.

Takođe, omogućava dodeljivanje prioriteta zahtevima što omogućava da bitniji zahtevi budu pre obrađeni što dodatno poboljšava performanse. Omogućeno je i efikasnije procesiranje poruka kroz korišćenje binarnog *framing*-a poruka.

Važno je za napomenuti da http 2.0 proširuje, ne zamenjuje, prethodne http standarde. Semantike aplikacija koje koriste http su iste i nikakve promene se neće izvesti nad postojećim funkcionalnostima ili suštinskim

pojmovima kao što su http metode, status kodovi ili polja zaglavlja. API na visokom nivou ostaje isti ali su urađene izmene nad niskim nivoom koje utiču na nedostatke performansi prethodnih verzija http-a.
U nastavku će biti obrađeni pojmovi koji nadograđuju prethodni protokol (HTTP 1.1) i poboljšavaju performanse HTTP protokola.

### 3.1 Binarni *framing* sloj

U srži svih poboljšanja performansi http-a 2.0 jeste novi binarni *framing* sloj koji diktira kako su http poruke enkapsulirane i prenesene između klijenta i servera.
„Sloj" se odnosi na izbor da se predstavi nov, optimizovan mehanizam za šifrovanje između *socket* interfejsa i viših http api-a koji su izloženi aplikaciji. Http semantike kao što su metode, zaglavlja itd. su nepromenjene ali način na koji su šifrovani dok se vrši slanje je drugačije. Nasuprot novom redu unutar običnog teksta http-a 1.1, sva http 2.0 komunikacija je podeljena na manje poruke i *frame*-ove, od kojih je svaki šifrovan (enkodiran) u binarni format.
Kao posledica ovoga, i klijent i server moraju da koriste novi binarni mehanizam šifrovanja da bi se razumeli međusobno. Http 1.1 klijent neće razumeti server koji govori samo http 2.0 i obrnuto. Aplikacije ne moraju znati za ove promene jer klijent i server obavljaju sva neophodna *framing* posla umesto njih.

### 3.2 *Multiplexing* zahteva i odgovora

U verziji http-a 1.1, da bi klijent napravio više paralelnih zahteva kako bi poboljšao performanse, morao je koristiti više tcp konekcija. Ovakvo ponašanje je direktna posledica mehanizma rada http-a 1.1 pošto on obezbeđuje da se tačno jedan odgovor može dostaviti u toku jednog vremenskog perioda po konekciji. Ovo predstavlja problem ako je prvi odgovor koji treba da se dostavi dugačak pa ostali moraju čekati dok se prvi ne dostavi u celosti. Rezultat je neefikasno korišćenje tcp konekcije.
Binarni *framing* sloj u http-u 2.0 rešava ova ograničenja i omogućava preplitanje odnosno *multiplexing* zahteva i odgovora u celosti tako što dozvoljava klijentu i serveru da razbiju http poruku u posebne *frame*-ove, isprepliću ih i onda ih ponovo sastave na drugom kraju.
Mogućnost da se razbije http poruka u posebne *frame*-ove, isprepliće ih i onda ih ponovo sastavi na drugom kraju je najbitnije unapređenje http-a 2.0.
Novi binarni *framing* sloj u http-u 2.0 rešava problem blokiranja prvog zahteva ili odgovora koji je bio zastupljen u http-u verzije 1.1 i takođe eliminiše potrebu za više tcp konekcija kako bi se omogućio paralelizam pri procesiranju kao i paralelnom dostavljanju zahteva i odgovora. Kao rezultat ovoga, aplikacije su brže, jednostavnije i jeftinije za *deployment*.

### 3.3 *Server push*

Još jedna moćna novina koju uvodi http 2.0 jeste mogućnost servera da šalje više odgovora za isti klijentski zahtev. Znači, pored odgovora na originalni zahtev, server može da *push*-uje dodatne resurse na klijenta bez da ih je klijent zatražio eksplicitno.
Tipična veb aplikacija se sastoji od mnogo resursa, koji se otkrivaju od strane klijenta tako što klijent isčitava dokumente koji su poslati sa servera i kada naiđe u tom dokumentu na neku vezu sa drugim dokumentom ili resursom, onda ga zatraži od servera. Ovo zahteva, dodatno vreme koje bi se moglo smanjiti tako što će server sam da pošalje dodatne dokumente ili resurse koje klijent zahteva za rad sa inicijalnim dokumentom i time znatno smanjiti vreme čekanja klijenta.

### 3.4 Kompresija zaglavlja

Svaki http zahtev ili odgovor nosi sa sobom i zaglavlje koje opisuje resurse koji se prenose kao i njihova obeležja. U http-u 1.1, ovi meta-podaci se uvek šalju kao običan tekst što dodaje na zahtev oko 500 do 800 bajtova dodatnih podataka pri prenosu i nekada po više kilobajta ako postoje http *cookie*-i koji se koriste. Kako bi se smanjila ova količina podataka i samim tim i poboljšale performanse, http 2.0 vrši kompresiju zaglavlja http zahteva i odgovora koristeći HPACK kompresioni format koji koristi dve jednostavne ali i moćne tehnike:

- dozvoljava poljima zaglavlja koja se šalju da budu komprimovana omoću Huffman-ovog koda koji smanjuje veličinu svakog zahteva i odgovora koji se šalje,
- zahteva da i klijent i server održavaju i ažuriraju indeksiranu listu prethodno viđenih polja zaglavlja koja se koristi kao referenca za efikasno komprimovanje prethodno prenošenih vrednosti.

Huffman-ovo kodiranje dovoljava individualnim vrednostima da budu komprimovane kada se šalju, dok indeksirana lista prethodno prenošenih vrednosti dozvoljava da se komprimuju duplicirane vrednosti tako što se prenose indeksi vrednosti koji se mogu koristiti da efikasno pretraže i rekonstruišu ključeve i vrednosti zaglavlja.

## 4. PODEŠAVANJE HTTP-A 2.0 NA POPULARNIM SERVERSKIM TEHNOLOGIJAMA

### 4.1 Podešavanje HTTP-a 2.0 u *spring boot* aplikaciji

Podrška za http 2.0 u *spring boot* aplikaciji se može omogućiti sa konfiguracionim obeležjem *server.http2.enabled*. Ova podrška zavisi od odabranog servera i aplikacijskog okruženja pošto protokol nije podržan po *default*-u u JDK-u 8.

### 4.2 Podešavanje HTTP-a 2.0 u *node.js* aplikaciji

Core API pruža *low-level* interfejs koji je dizajniran da podrži karakterstike http 2.0 protokola. Nije dizajniran za kompatibilnost sa postojećim http 1 modul api-em, ali Compatibility API jeste.

### 4.3 Podešavanje HTTP-a 2.0 u ASP.NET Core aplikaciji

Veb server koji se koristi po *default*-u u asp.net Core projektu se naziva Kestrel. HTTP 2.0 je podržan u asp.net Core aplikacijama koje koriste Kestrel ako su zadovoljeni sledeći uslovi:

- Koristi se operativni sistem:
  - Windows Server 2016/Windows 10 ili noviji ili
  - Linux sa OpenSSL-om 1.0.2 ili noviji (npr. Ubuntu 16.04 ili noviji).
- Ciljani framework: .net Core 2.2 ili noviji,
- Application-Layer Protocol Negotiation (ALPN) konekcija, i
- TLS 1.2 ili novija konekcija.

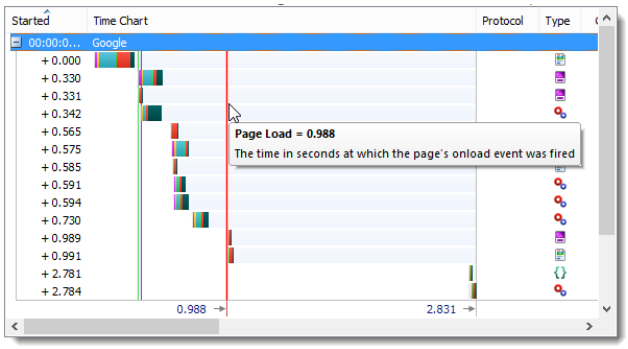
Ako su prethodno navedeni uslovi zadovoljeni HTTP 2.0 je omogućen po *default*-u i nisu potrebne nikakve dodatne konfiguracije.

## 5. PERFORMANSE HTTP-A 2.0

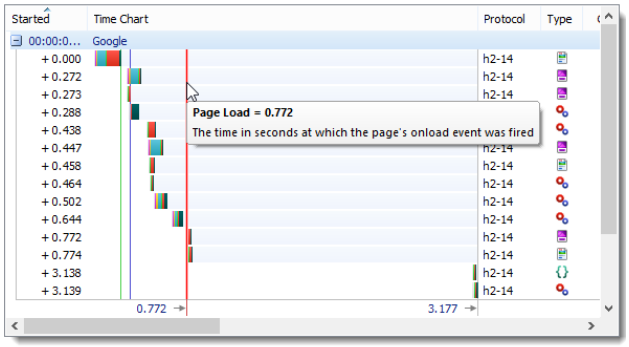
Za ocenu performansi HTTP-a 2.0, poredio se ovaj protokol sa njegovom prethodnom verzijom.

### 5.1 Vreme učitavanja stranice

Događaj učitavanja u *HttpWatch*-u pokazuje kada se stranica skinula u potpunosti i kada je spremna za korišćenje. U većini slučajeva ovo je razumna mera brzine stranice iz perspektive posetioca veb sajta.
Na slikama 2 i 3 su prikazane brzine učitavanja stranica za http 1.1 i 2.0.




Slika 2. *Brzina učitavanja stranice kod HTTP-a 1.1 [6]*




Slika 3. *Brzina učitavanja stranice kod HTTP-a 2.0 [6]*

Sa slika 2 i 3 se vidi da je brzina učitavanja stranice kod http-a 1.1 sporija za 0.216 sekundi u odnosu na HTTP 2.0 što je 21.86% (100 – (0.772 / 0.988 * 100)). Ovakva razlika u performansama je izražena zbog nekomprimovanih zaglavlja i dodatnih tcp konekcija i ssl *handshake*-ova kod protokola 1.1. Za kompleksnije stranice, brzina http-a 2.0 bi trebala da bude još izraženija.

## 6. ZAKLJUČAK

Na osnovu teorijske obrade protokola http 1.1 i http 2.0, može se ustanoviti da je http 2.0 bolji protokol u svakom pogledu jer samo popunjava nedostatke prošlog protokola i dodaje neke nove funkcionalnosti dok stare ostaju iste. Dodatna prednost ovog protokola jeste da unapređivanje protokola sa verzije 1.1 na 2.0 neće zahtevati promene u aplikaciji sem onih na visokom nivou kao što su konfiguracije.

Na osnovu poređenja performansi ova dva protokola se takođe može utvrditi da je http 2.0 superiorniji protokol. U testovima je pokazano da je vreme učitavanja stranice smanjeno za 21.86% sa upotrebom HTTP-a 2.0. Ovo su znatna poboljšanja performansi protokola, a za kompleksnije stranice, bi mogle doći još više do izražaja prednosti HTTP-a 2.0 što se tiče smanjenja vremena učitavanja stranica.
Podešavanje HTTP-a 2.0 za popularne *backend* tehnologije je relativno jednostavno ukoliko se koristi https dok je za nešifrovan http saobraćaj uglavnom nemoguće podesiti http 2.0. Ovo predstavlja veliku manu protokola, jer mnogi veb sajtovi ne zahtevaju šifrovan saobraćaj, a zbog toga ne mogu da koriste prednosti http-a verzije 2.0.
Iako je pokazano da se i *server push* može lako implementirati u popularnim *backend* tehnologijama, za razliku od ostalih poboljšanja protokola koje su pomenute, performanse koje se mogu dobiti (ili izgubiti) pomoću ove funkcionalnosti zavise od programera, odnosno od implementacije *endpoint*-a i odabira koji resursi će biti *push*-ovani.

## 7. LITERATURA


[1] HTTP: https://www.w3.org/History/19921103-hypertext/hypertext/WWW/Protocols/HTTP.html
[2] World Wide Web: https://en.wikipedia.org/wiki/World_Wide_Web
[3] HTTP 0.9: https://hpbn.co/brief-history-of-http/#http-09-the-one-line-protocol
[4] HTTP 1.0: https://hpbn.co/brief-history-of-http/#http10-rapid-growth-and-informational-rfc, https://hpbn.co/http1x/
[5] HTTP 2.0: https://tools.ietf.org/html/rfc7540, https://hpbn.co/brief-history-of-http/#http2-improving-transport-performance, https://hpbn.co/http2/
[6] Poređenje performansi HTTP-a 1.1 i 2.0: https://blog.httpwatch.com/2015/01/16/a-simple-performance-comparison-of-https-spdy-and-http2/


**Kratka biografija:**


**Marko Krajinović**, rođen 19.11.1995 u Somboru. Završio je osnovnu školu Nikola Vukićević u Somboru i srednju tehničku školu u Somboru. Diplomirao je na fakultetu tehničkih nauka u Novom Sadu.