CARNET

Transport Layer Security (TLS) verzija 1.3

CERT.hr-PUBDOC-2021-8-403

CERT.hr

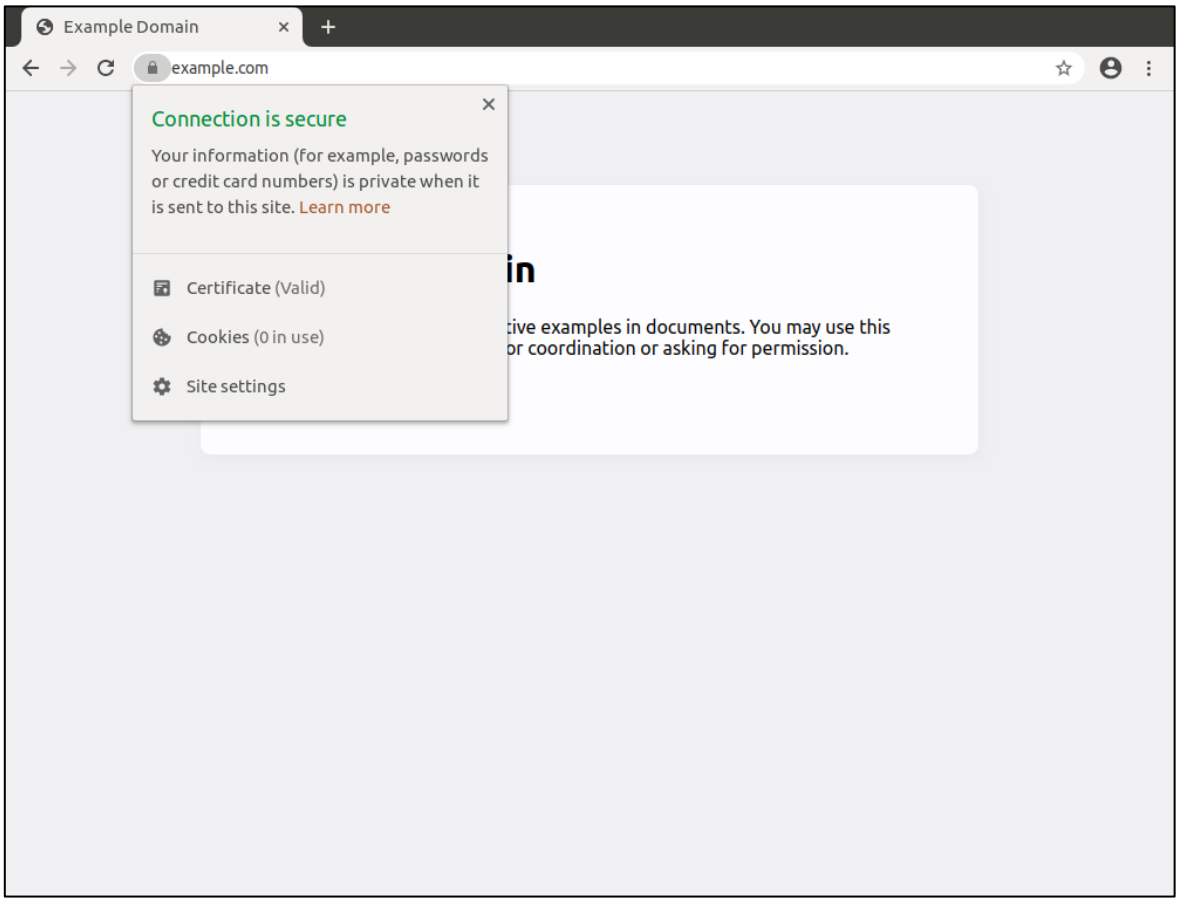
Sadržaj

Ovaj dokument izradio je Laboratorij za sustave i signale Zavoda za elektroničke sustave i obradbu informacija Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu.

1 Uvod

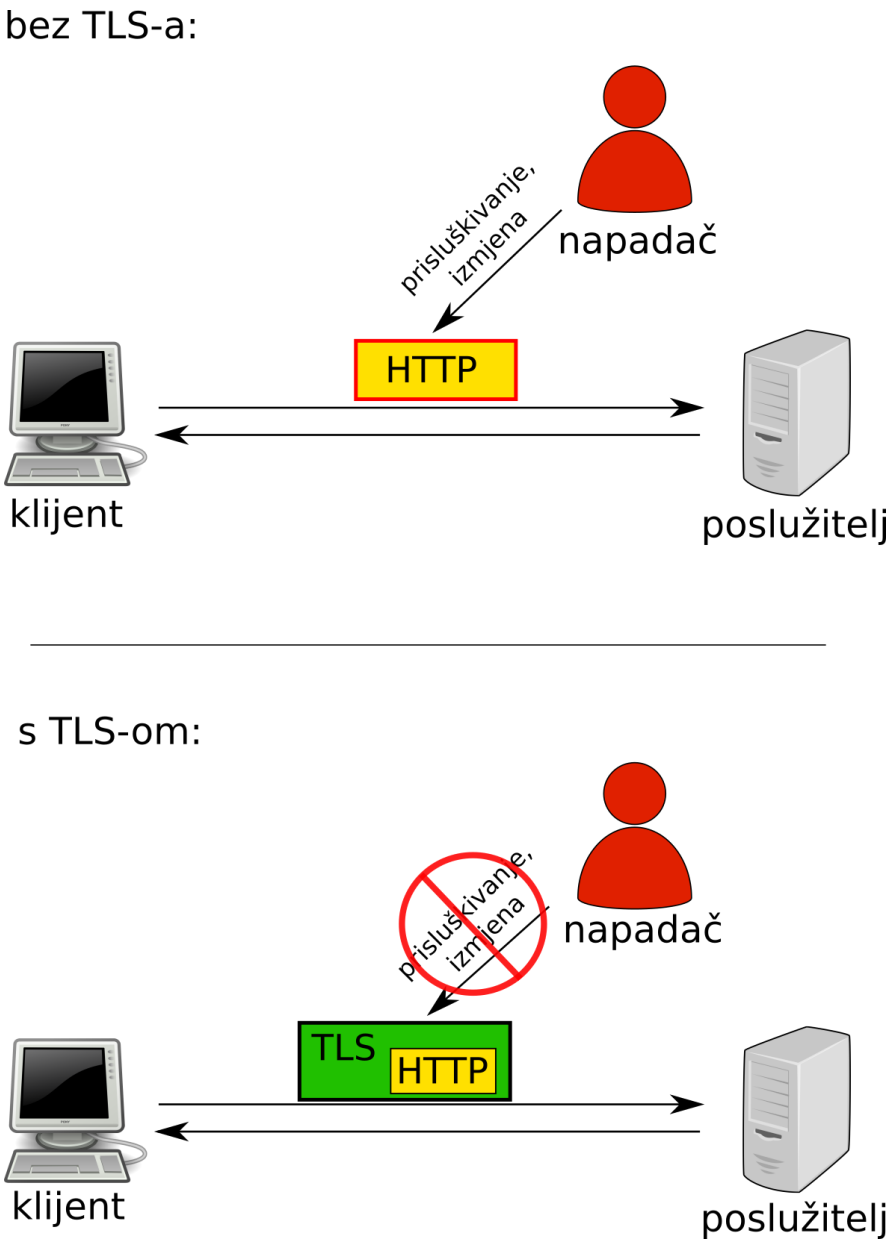
Transport Layer Security, skraćeno TLS, mrežni je protokol koji aplikacijama omogućava sigurnu komunikaciju preko nesigurnog kanala: interneta. Mrežni promet zaštićen TLS-om siguran je od prisluškivanja i krivotvorenja poruka, čak i kada bi napadač mogao presretati i izmjenjivati sav mrežni promet između aplikacija koje komuniciraju (tzv. *man-in-the-middle* napad) (1). Kada krajnji korisnici otvaraju web stranice, koriste mobilne aplikacije, šalju poruke e-pošte i slično, velika je vjerojatnost da u pozadini njihov uređaj koristi upravo TLS kako bi zaštitio poruke koje se razmjenjuju mrežom.

Primjerice, kod korištenja web preglednika, lokot pored URL-a web stranice označava da je veza zaštićena TLS-om. Klikom na lokot obično je moguće doći do više informacija o sigurnosti veze, kao što je prikazano na slici 1.



Slika 1 – klikom na lokot se u većini web preglednika prikazuje više informacija o sigurnosti veze

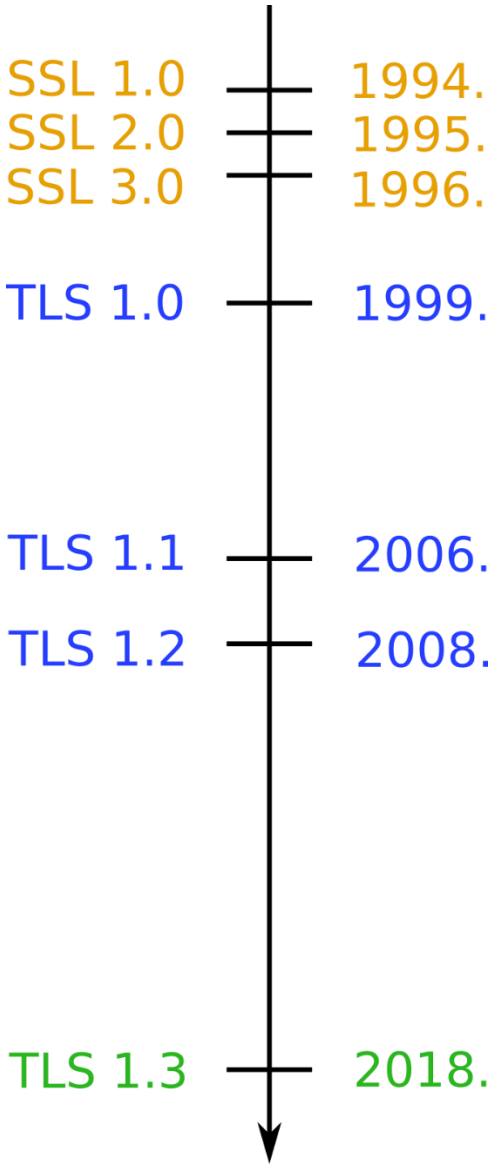
S tehničke strane, TLS se koristi kao zaštitni sloj oko aplikacijskih protokola poput HTTP-a i SMTP-a. Primjerice, dohvaćanje web stranica protokolom HTTP podložno je prisluškivanju i lažiranju poruka. Upravo zato se preporučuje zaštita protokola HTTP TLS-om, poznatije kao HTTPS, jer tada nije ranjiv na te napade (prikazano na slici 2).



Slika 2 – običan HTTP promet moguće je prisluškivati i izmjenjivati, dok je HTTP promet zaštićen TLS-om (HTTPS) otporan na takve napade

Verzija 1.0 protokola TLS uvedena je 1999. godine kao nasljednik protokola SSL (*Secure Sockets Layer*) kojemu su otkriveni sigurnosni propusti. S vremenom su uvedene novije verzije TLS-a koje su prvenstveno ojačavale sigurnosna svojstva i ispravile sigurnosne propuste protokola. TLS verzija 1.1 uvedena je 2006. godine, TLS 1.2 2008. godine, a najnovija verzija TLS-a 1.3 uvedena je 2018. godine (2). Vremenska crta verzija protokola SSL i TLS prikazana je na slici 3. Po standardu RFC8996 (3) sve verzije protokola TLS starije od verzije 1.2, i sve verzije starijeg protokola SSL, označene su kao zastarjele (engl. *deprecated*) i više se ne preporučuje njihovo korištenje.

Prethodni dokument Nacionalnog CERT-a: „TLS“ detaljno opisuje svojstva protokola TLS, korištene kriptografske algoritme, primjenu protokola i ostale općenite pojedinosti. Svrha ovog dokumenta je opisati izmjene protokola u najnovijoj verziji: TLS 1.3.



Slika 3 – vremenska crta verzija protokola SSL i TLS

2 Protokol rukovanja

Među najvećim izmjenama u verziji 1.3 protokola TLS su izmjene u protokolu rukovanja (engl. *handshake protocol*). Navedene izmjene poboljšale su sigurnost i brzinu uspostave TLS sjednice (engl. *session*), što za krajnje korisnike znači sigurnije i brže dohvaćanje web stranica, spajanje na poslužitelje e-pošte, korištenje nekih mobilnih aplikacija i slično.

Jedan od značajnih faktora koji određuje koliko će se brzo otvoriti web stranica je trajanje uspostave TLS sjednice. Trajanje uspostave TLS sjednice velikim dijelom ovisi o broju poruka koje je potrebno poslati i primiti prije nego klijent ili poslužitelj mogu početi slati aplikacijske podatke, poput HTTP zahtjeva i odgovora. Kako su razmijenjene TLS poruke relativno male, propusnost mreže (engl. *bandwidth*) u ovom slučaju predstavlja manje ograničenje u usporedbi s kašnjenjem/latencijom (engl. *latency*) koje se primijeti kod svake razmijenjene poruke.

Primjerice, kako bi klijent poslao čak i najmanji paket poslužitelju te dobio odgovor na njega (npr. HTTP zahtjev i odgovor), moraju proći dva vremena kašnjenja:

- za HTTP zahtjev od klijenta do poslužitelja
- te za HTTP odgovor od poslužitelja do klijenta.

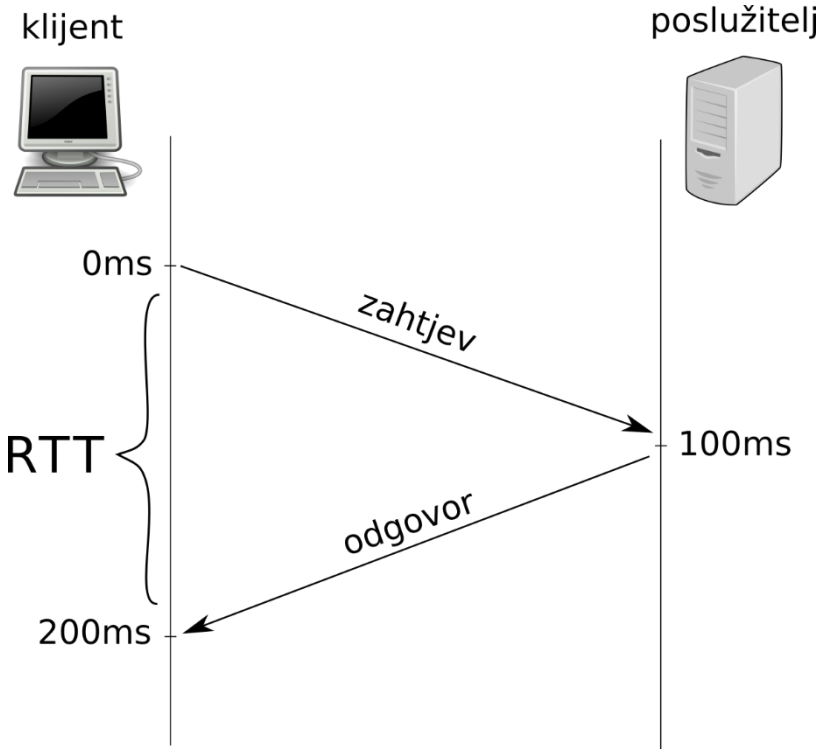
Drugim riječima, sveukupno treba proći jedno vrijeme obilaska (engl. *round-trip time*, RTT).

Primjerice, ako je vrijeme kašnjenja paketa od klijenta do poslužitelja 100ms, te vrijeme kašnjenja paketa od poslužitelja do klijenta isto 100ms, sveukupno vrijeme obilaska (RTT) bit će 200ms. Ako u tom primjeru klijent pošalje neki zahtjev paketom, odgovor na njega dobit će za 200ms (prikazano na slici 4). Iako je vrijeme kašnjenja na fiksnim mrežama danas relativno nisko, vrijeme kašnjenja na mobilnim mrežama, koje se s rastom popularnosti pametnih telefona, interneta stvari (engl. *Internet of Things*) i ostalih mobilnih uređaja sve više koriste, i dalje može biti prilično dugo.

Za usporedbu postupka uspostave sjednice u različitim verzijama protokola TLS:

- slika 5 prikazuje pojednostavljeni primjer uspostave sjednice protokolom TLS 1.2 (4),
- slika 6 prikazuje pojednostavljeni primjer uspostave sjednice protokolom TLS 1.2 uz korištenje dodatka (engl. *extension*) TLS *false start* (5),
- slika 7 prikazuje pojednostavljeni primjer uspostave sjednice protokolom TLS 1.3 (1).

U sva tri slučaja, prikazana je uspostava sjednice bez korištenja mehanizma nastavka sjednice (engl. *session resumption*) odnosno unaprijed razmijenjenog ključa (engl. *pre-shared key*). Taj mehanizam i izmjene u njemu bit će opisane u sljedećem poglavlju.



Slika 4 – u prikazanom primjeru, vrijeme obilaska (engl. *round-trip time*, RTT) između klijenta i poslužitelja je 200ms

Za uspostavu TLS 1.2 sjednice (prikazano na slici 5), nakon početnog TCP rukovanja na transportnom sloju (koje nije prikazano na slici), potrebno je razmijeniti četiri grupe poruka prije nego klijent može poslati prve aplikacijske podatke, npr. HTTP zahtjev.

Prve četiri grupe poruka na slici dio su postupka rukovanja i imaju sljedeće značenje:

1. Prvo, klijent šalje poruku u kojoj navodi koje verzije TLS-a i koje kriptografske algoritme podržava.

2. Zatim, poslužitelj šalje poruku u kojoj navodi koju TLS verziju i koje algoritme je odabrao, šalje svoj certifikat te svoj kriptografski materijal potreban za uspostavu zajedničke tajne.

3. Onda klijent šalje svoj kriptografski materijal potreban za uspostavu zajedničke tajne te šalje i završnu poruku pomoću koje poslužitelj može provjeriti je li bila narušena sigurnost postupka rukovanja.

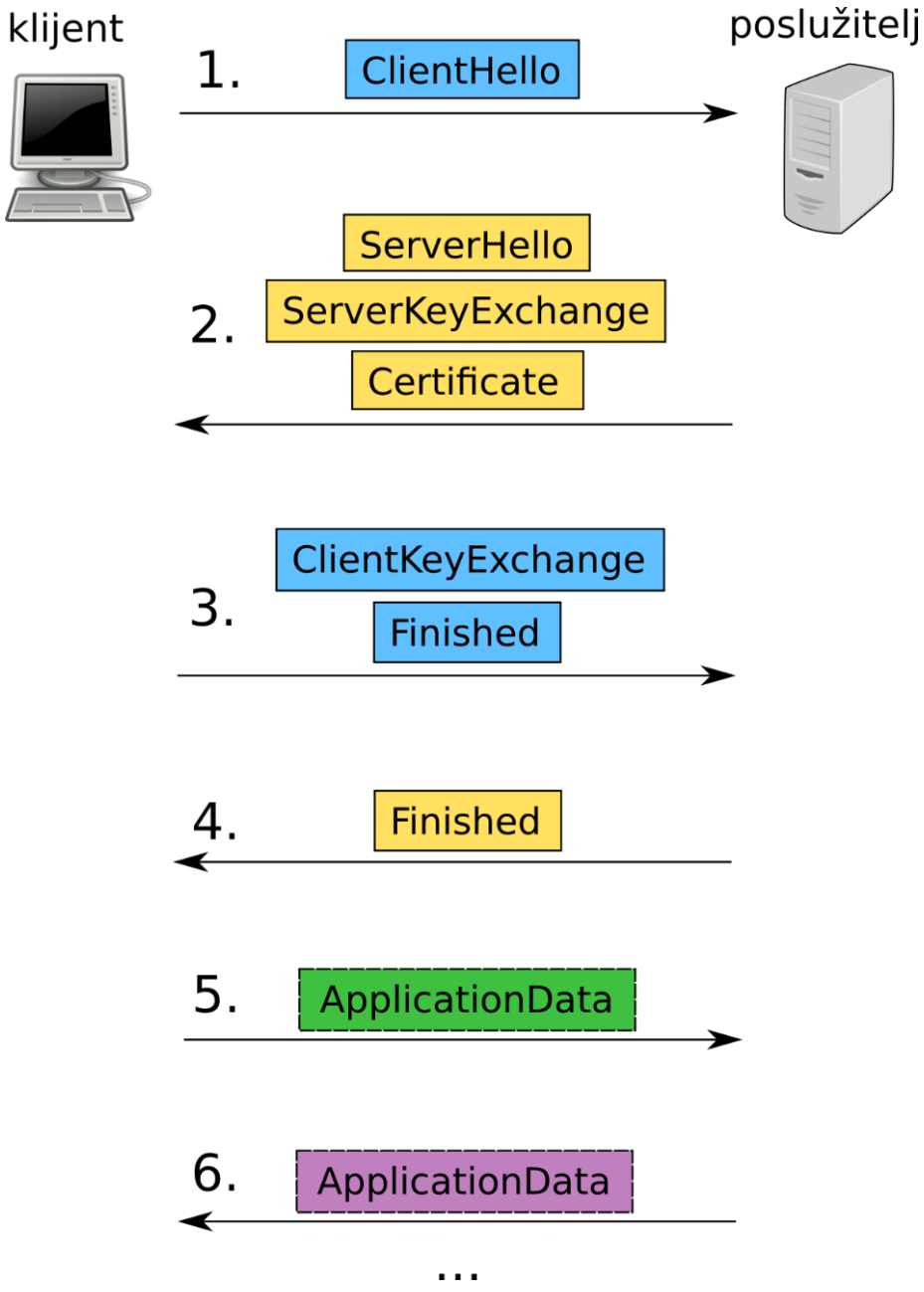
4. Konačno i poslužitelj šalje završnu poruku pomoću koje klijent može provjeriti je li bila narušena sigurnost postupka rukovanja.

Tek je tada TLS sjednica uspostavljena te klijent može slati prve aplikacijske podatke (npr. HTTP zahtjev).

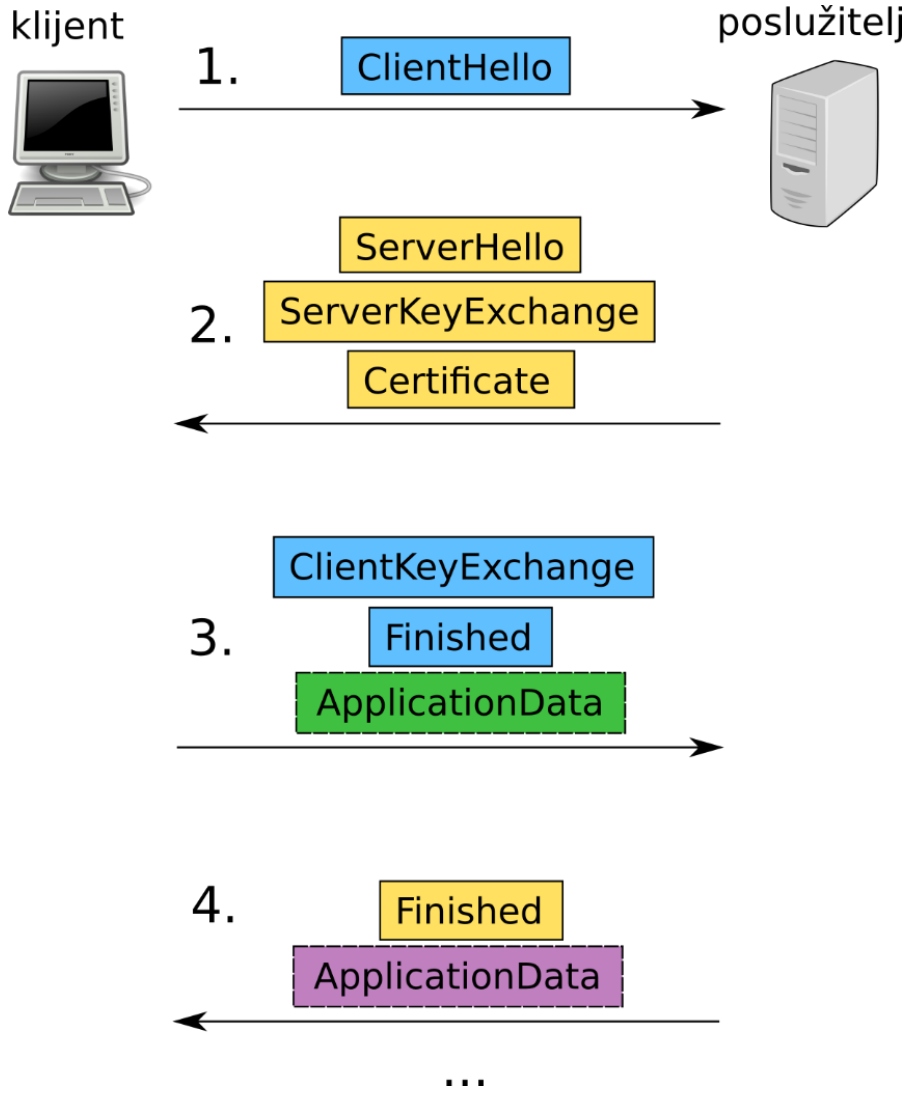
U ovom slučaju, ne uzimajući u obzir uspostavu TCP veze, moraju proći dva vremena obilaska (RTT) prije nego klijent može slati podatke. Kada se koristi ovakva uspostava TLS sjednice za dohvat web stranice protokolom HTTP, korisnik prvi odgovor od poslužitelja dobiva tek u šestoj grupi poruka na slici. Sveukupno, prije primanja prvih aplikacijskih

podataka (prvog HTTP odgovora) korisnik mora pričekati jedno vrijeme obilaska (RTT) za uspostavu TCP veze, dva vremena obilaska za uspostavu TLS sjednice (prve četiri grupe poruka označene na slici) i jedno vrijeme obilaska za HTTP zahtjev i odgovor.

Također, iako su integritet i autentičnost rukovanja očuvani, neke poruke ipak nisu šifrirane (poput poruke s certifikatom ili prve poslane poruke klijenta unutar koje može biti navedeno domensko ime poslužitelja), tako da privatnost nije u potpunosti očuvana jer napadač koji prisluškuje mrežu primjerice može otkriti domensko ime web stranice koju korisnik posjećuje.



Slika 5 – pojednostavljeni primjer uspostave sjednice protokolom TLS 1.2 (4)



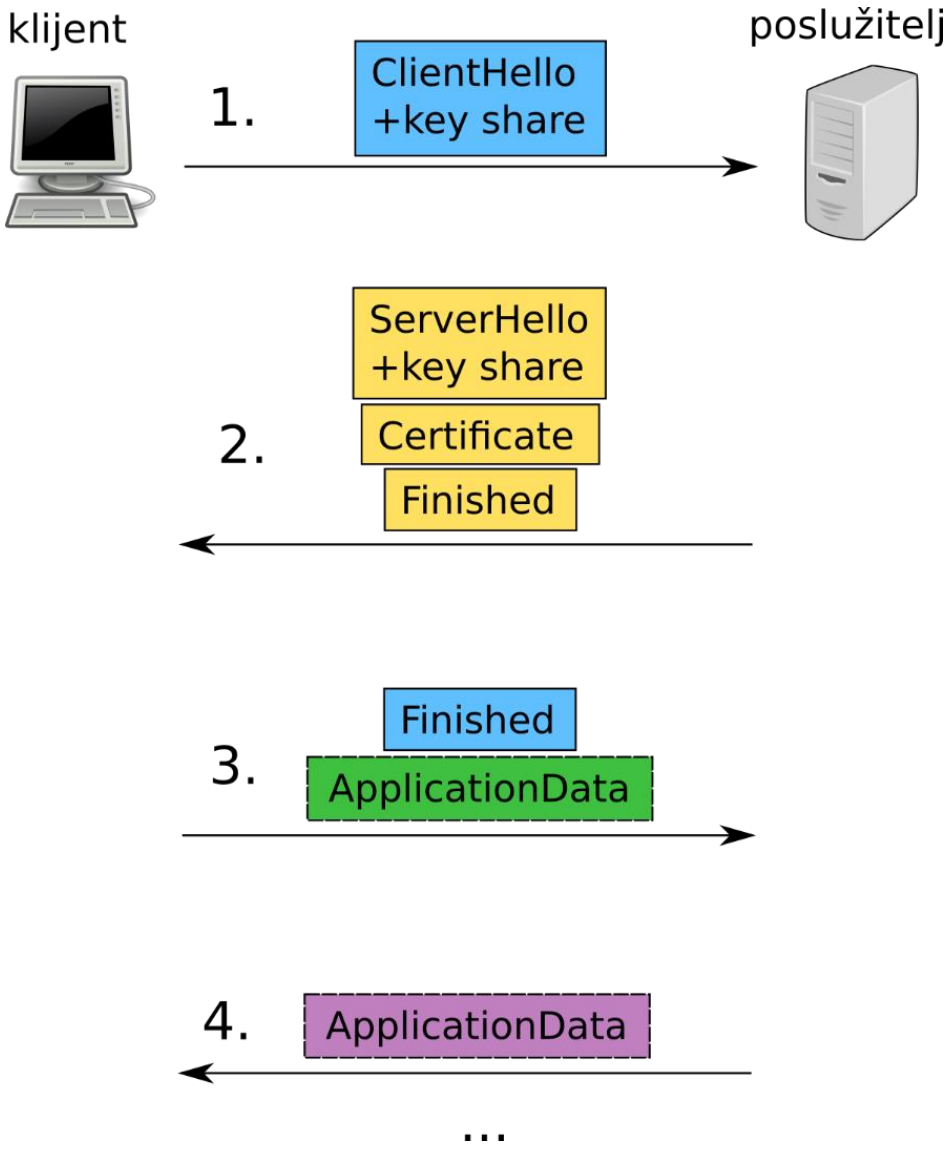
Slika 6 – pojednostavljeni primjer uspostave sjednice protokolom TLS 1.2 uz dodatak TLS *false start* (5)

Napredak od osnovnog rukovanja pruža dodatak (engl. *extension*) protokolu TLS 1.2 nazvan TLS *false start*. U usporedbi s osnovnim rukovanjem, TLS *false start* klijentu omogućava ranije slanje prvih aplikacijskih podataka, čime je ubrzano primjerice i otvaranje web stranica putem HTTPS-a (prikazano na slici 6). Konkretnije, rukovanje uz dodatak TLS *false start* dopušta klijentu da šalje prve aplikacijske podatke već u trećem skupu poruka, odmah nakon slanja svoje završne poruke, a prije primanja i provjere poslužiteljeve završne poruke (5).

Po pitanju sigurnosti, nedostatak takvog pristupa je činjenica da klijent šalje aplikacijske podatke prije nego je provjerio poslužiteljevu završnu poruku, tj. prije nego što se uvjerio da napadač nije izmijenio neke poruke unutar postupka rukovanja. Primjerice, napadač bi mogao utjecati na rukovanje tako da se za šifriranje odaberu slabiji kriptografski algoritmi. Ovim načinom rukovanja, klijent bi to saznao tek nakon što već pošalje prve aplikacijske podatke šifrirane potencijalno nesigurnim algoritmom kojega je napadač odabrao. (5)

Ovu opasnost moguće je minimizirati tako da klijent koristi dodatak TLS *false start* isključivo s kriptografskim algoritmima šifriranja koji se smatraju sigurnima. Na taj način, čak i da napadač utječe na odabir kriptografskih algoritma šifriranja, aplikacijski podaci ne bi smjeli biti ugroženi jer će i dalje biti šifrirani algoritmom koji se smatra sigurnim (5).

Što se tiče brzine, prednost rukovanja uz dodatak TLS *false start* prilično je jasna. Uz dodatak TLS *false start*, prvi aplikacijski podaci mogu se slati već u trećoj skupini poruka, a odgovor na njih u četvrtoj skupini, za razliku od pete i šeste skupine bez njegova korištenja Vremenski to znači da je TLS rukovanje, pa time i primjerice otvaranje web stranica, efektivno skraćeno za jedno vrijeme obilaska (RTT).



Slika 7 – pojednostavljeni primjer uspostave sjednice protokolom TLS 1.3

Osnovni postupak rukovanja u protokolu TLS 1.3 (prikazan na slici 7) pruža brzinu uspostave sjednice kao kada se koristi dodatak TLS *false start*, uz sigurnost koja je bolja od oba prethodno opisana rješenja.

Jedna od glavnih izmjena je to da klijent prilikom uspostave veze, prije bilo kakve komunikacije s poslužiteljem, pokuša pogoditi koji će kriptografski algoritmi biti

odabrani, te na temelju toga već u prvoj poruci šalje kriptografski materijal potreban za uspostavu TLS sjednice. Ako klijent uspješno pogodi odabrane kriptografske algoritme, poslužitelj već u drugoj poruci može šifrirati bilo kakav dodatan sadržaj, poput certifikata, te poslati svoju završnu poruku kojom potvrđuje sigurnost rukovanja (1).

Slično kao kod rukovanja koje koristi TLS *false start*, klijent može već u trećoj poruci slati aplikacijske podatke, pa će u konačnici otvaranje web stranice ili slična primjena biti jednako brza.

Što se tiče sigurnosti – u ovom slučaju klijent prije slanja aplikacijskih podataka već prima završnu poruku od poslužitelja, pa može potvrditi sigurnost rukovanja. Osim toga, s TLS-om 1.3 se u poslužiteljevoj drugoj poruci mogu čak i šifrirati podaci poput certifikata, čime se dodatno osigurava privatnost u usporedbi s TLS-om 1.2 (1).

Što ako klijent u prvoj poruci neuspješno pogodi kriptografski algoritam kojim će se nastaviti rukovanje? U tom slučaju poslužitelj šalje posebnu poruku (*HelloRetryRequest*) nakon koje rukovanje efektivno kreće ispočetka, ovaj put s algoritmom kojeg je poslužitelj odabrao. Time se dodaje još jedno vrijeme obilaska (RTT) duljini rukovanja, pa se tada gubi ostvarena vremenska prednost ovog pristupa (1). No takvi bi slučajevi trebali biti rijetki – osim što je u TLS-u 1.3 smanjen broj podržanih kriptografskih algoritama, u praksi je poznato i kako je većina poslužitelja konfigurirana i koje algoritme podržavaju. Zato klijentima (poput web preglednika) nije teško pogoditi koje kriptografske algoritme treba odabrati, pa se TLS 1.3 rukovanja mogu odvijati na prethodno opisani brzi način kao što je prikazano na slici 7.

3 Nastavak sjednica i unaprijed dijeljeni ključevi

Prethodno opisani osnovni postupak rukovanja, u sva tri navedena slučaja, relativno je „skup“. Korištena asimetrična kriptografija u postupku autentifikacije i uspostave zajedničke tajne zahtijeva puno rada procesora što može usporiti rukovanje i opteretiti poslužitelje s velikim brojem klijenata (npr. web stranice s puno posjetitelja) (6) (7).

Kada bi bilo moguće izbjeći taj skupi postupak uspostave zajedničke tajne, postupak TLS rukovanja manje bi opterećivao poslužiteljska i klijentska računala te bi bilo moguće napraviti brže rukovanje s manjim brojem razmijenjenih poruka. Upravo to je prednost mehanizma nastavka sjednice (engl. *session resumption*) odnosno unaprijed dijeljenih ključeva (engl. *pre-shared keys*, PSK).

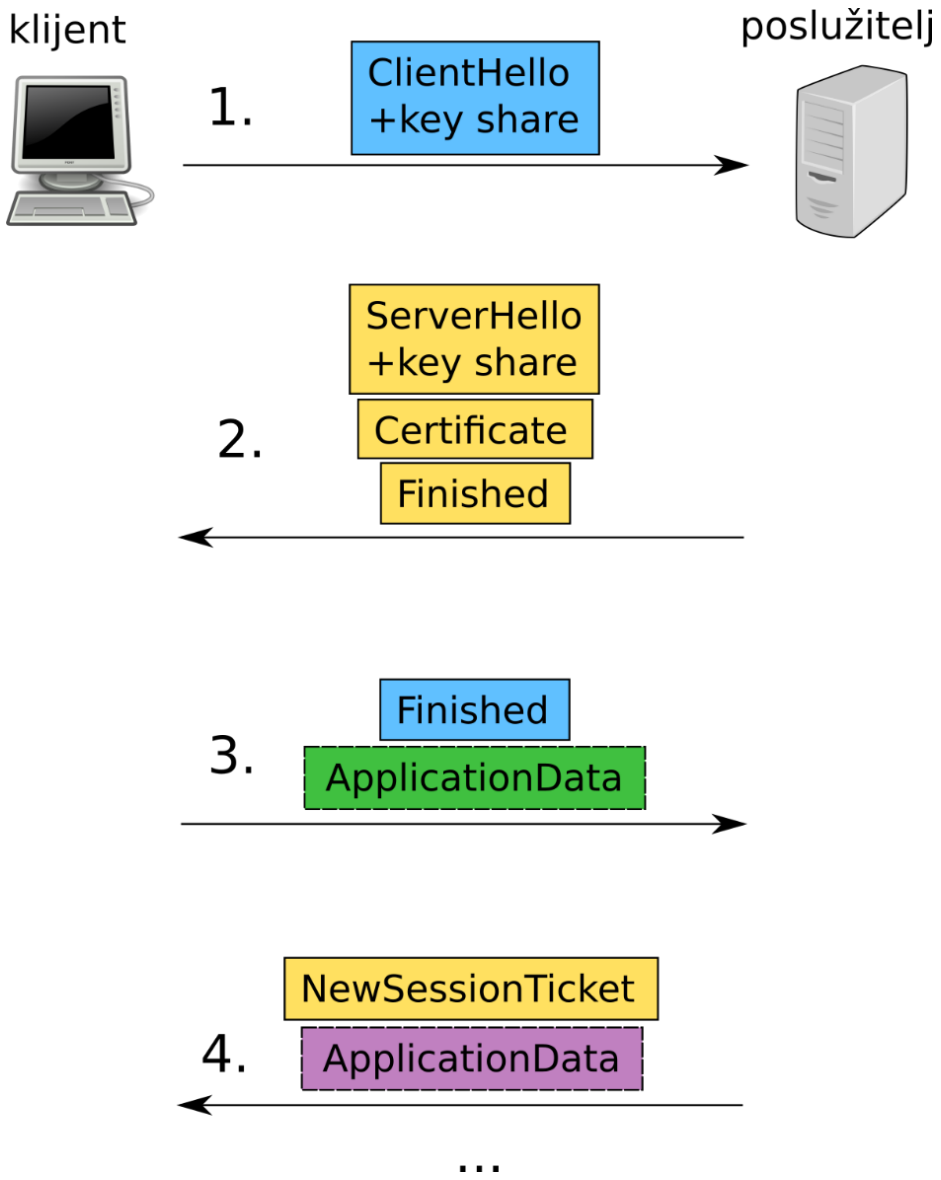
U TLS-u 1.2 postoji i mehanizam nastavka sjednice i mehanizam unaprijed dijeljenih ključeva (4). Navedeni mehanizmi su implementacijski slični, pa su u TLS-u 1.3 objedinjeni u jedan zajednički mehanizam pod imenom mehanizma unaprijed dijeljenih ključeva (1). Temeljna ideja tog mehanizma je sljedeća: jednom kada klijent i poslužitelj uspostave TLS vezu na uobičajeni način[1], onda nema potrebe da kod budućih spajanja opet prolaze skupi postupak potpunog rukovanja. Nakon prve uspostavljene TLS veze, klijent i poslužitelj već imaju uspostavljenu zajedničku tajnu, pa kod budućih spajanja mogu ubrzati rukovanje korištenjem te zajedničke tajne, umjesto da uspostavljaju novu.

Mehanizam unaprijed dijeljenih ključeva u TLS-u 1.3 prikazan je na slikama 8, 9 i 10 (1).

Slika 8 prikazuje potpuni postupak rukovanja u kojem, nakon uspostave TLS sjednice, poslužitelj klijentu šalje poruku *NewSessionTicket* s informacijama[2] pomoću kojih je ubuduće moguće uspostaviti TLS sjednicu skraćenim rukovanjem.

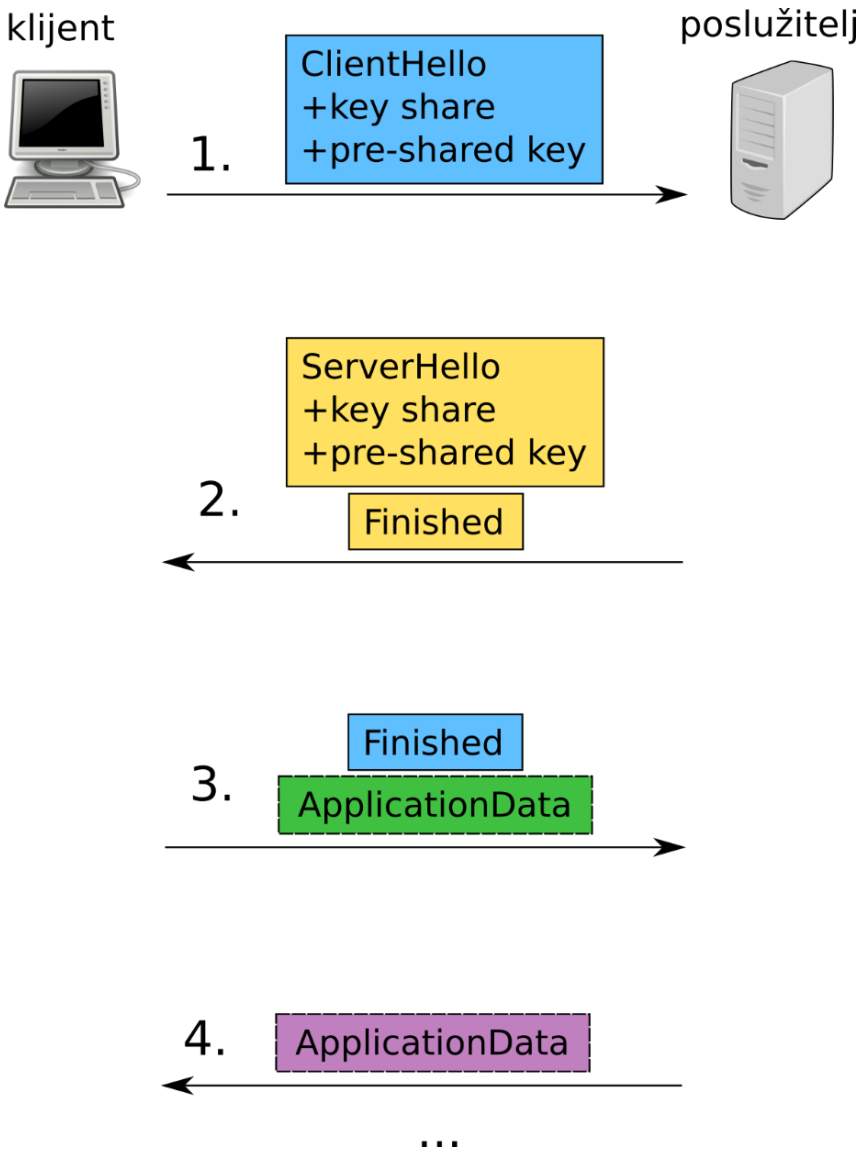
[1] dovoljno je da klijent i poslužitelj prethodno na bilo koji način uspostave siguran komunikacijski kanal (čak ne nužno TLS-om)

[2] poslužitelj u tu poruku može primjerice zapisati ključ za bazu podataka (na temelju kojega može doći do potrebnih informacija) ili u poruku može izravno zapisati šifrirane, autentificirane informacije o sjednici (kako poslužitelj ne bi morao kod sebe spremati te informacije)



Slika 8 – uspostava TLS 1.3 sjednice potpunim rukovanjem uz slanje poruke *NewSessionTicket* za buduće uspostavljanje sjednica skraćenim rukovanjem (1)

Nakon potpunog rukovanja prikazanog na slici 8, buduće TLS sjednice mogu biti uspostavljene skraćenim rukovanjem prikazanim na slici 9. Skraćeno rukovanje u postupku uspostave ključeva može, ali i ne mora koristiti algoritam Diffie-Hellman, odnosno Elliptic Curve Diffie-Hellman (označen na slici 9 tekstom „*key share*“). Korištenje tih algoritama dat će uspostavljenoj TLS sjednici sigurnosno svojstvo buduće tajnosti (engl. *forward secrecy*), no izbjegavanje korištenja tih algoritama manje će opteretiti računala klijenta i poslužitelja (jer se neće izvoditi zahtjevne operacije asimetrične kriptografije) pa će biti brže. Iako skraćeno rukovanje u TLS-u 1.3 prikazano na slici 9 nudi prednosti nad potpunim rukovanjem, prikazani primjer skraćenog rukovanja zapravo se ne razlikuje značajno od ekvivalentnog mehanizma u TLS-u 1.2.

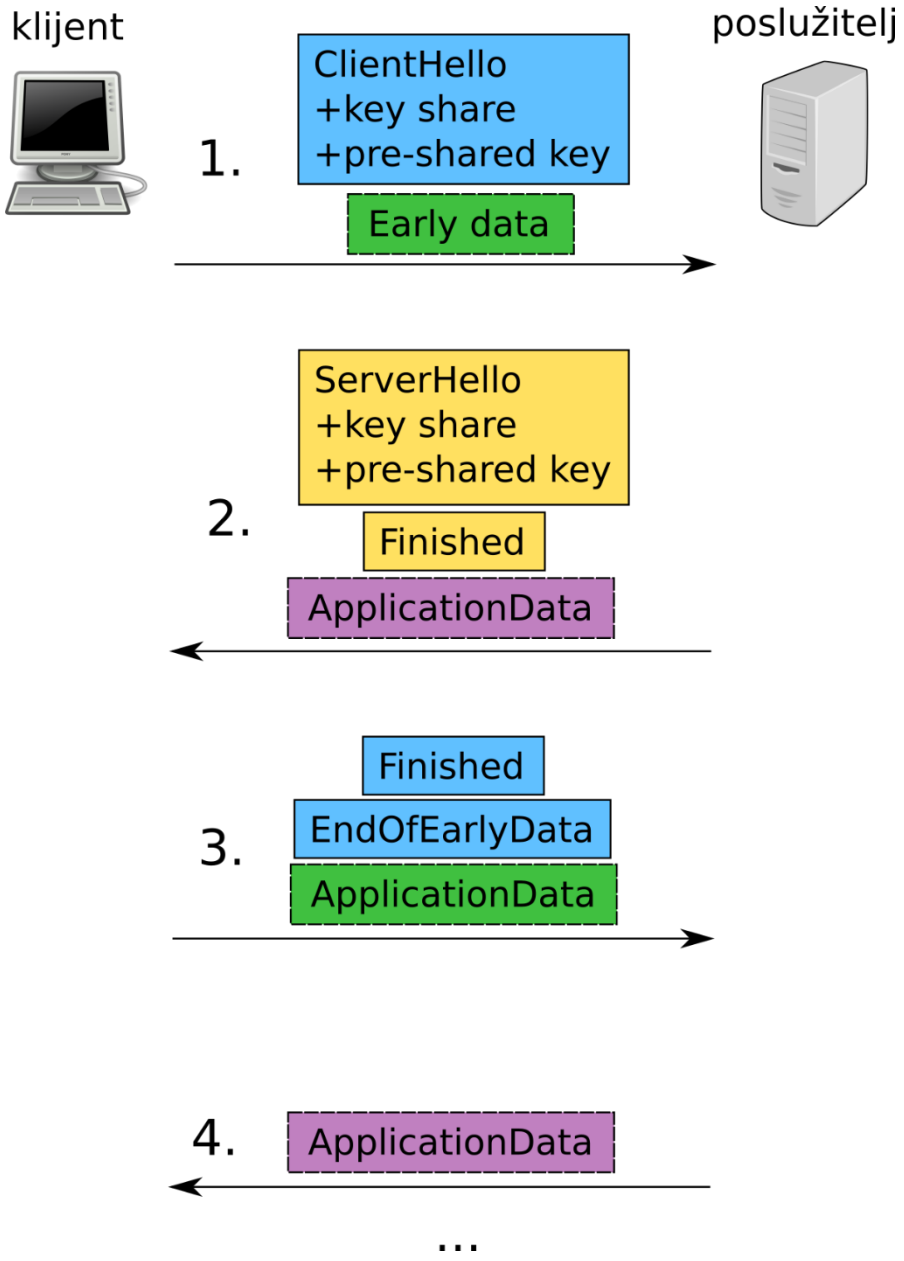


Slika 9 – uspostava TLS 1.3 sjednice skraćenim rukovanjem koristeći unaprijed dijeljeni ključ (engl. *pre-shared key*, PSK) (1)

Osim objedinjavanja mehanizama nastavka sjednice i unaprijed dijeljenih ključeva, TLS 1.3 ipak donosi jednu značajnu novost u tom kontekstu. U TLS-u 1.3, klijent prilikom skraćenog rukovanja može slati prve aplikacijske podatke (tzv. rane podatke, engl. *early data*) već od prvog skupa poruka, odmah nakon uspostave TCP veze, a prije nego je dobio bilo kakvu TLS poruku od poslužitelja. Ti se podaci nazivaju još i 0-RTT podaci, po tome što klijent ne treba čekati ni jedno vrijeme obilaska (engl. *round-trip time*, RTT) prije slanja tih podataka. Takva uspostava TLS sjednice u kojoj se šalju tzv. rani (0-RTT) podaci prikazana je na slici 10.

Takvo rukovanje primjerice znači da prilikom otvaranja web stranice klijent može poslati HTTP zahtjev praktički odmah nakon uspostave TCP veze, te ako poslužitelj prihvaća slanje ranih podataka, odgovor može dobiti odmah u prvom skupu poruka koji stiže s poslužitelja, nakon samo jednog vremena obilaska (1 RTT), kao što je prikazano na slici 10. Jedino što je još preostalo ubrzati u tom slučaju je početno TCP rukovanje za uspostavu veze na transportnom sloju – uz opisane promjene u TLS-u 1.3, na tome se radi kroz novi transportni protokol QUIC (8) i novu verziju 3.0 protokola HTTP (9).

Osim nedostatka svojstva buduće tajnosti (engl. *forward secrecy*), slanje ranih (0-RTT) podataka na prethodno opisani način podložno je napadima u kojima napadač snimi poslane rane podatke i ponovno ih pošalje poslužitelju (engl. *replay attacks*). Zato TLS 1.3 standard nalaže klijentima da šalju isključivo rane podatke koji se ne mogu zloupotrijebiti u takvim napadima, te predlaže i zaštitne mjere koje bi se trebale uvesti na poslužiteljima koji prihvaćaju rane (0-RTT) podatke. Potpuna zaštita od ovakvih napada zahtijeva i zaštitne mjere na aplikacijskoj razini, pa primjerice standard RFC8470 *Using Early Data in HTTP* predlaže kako koristiti TLS 1.3 rane podatke u HTTP protokolu te za tu svrhu definira novi HTTP statusni kod 425 *Too Early* i novo HTTP zaglavlje *Early-Data* (10).



Slika 10 – uspostava TLS 1.3 sjednice skraćenim rukovanjem koristeći unaprijed dijeljeni ključ i slanje ranih podataka (tzv. 0-RTT podaci) (1)

4 Kriptografski algoritmi

Temelj sigurnosti koju TLS pruža su kriptografski algoritmi poput algoritama šifriranja, digitalnog potpisivanja, razmjene ključa te računanja kriptografskog sažetka (engl. *hash*).

Zbog stalnog napretka u kriptografiji, neki kriptografski algoritam za kojega se godinama smatralo da je siguran može odjednom postati nesiguran ako se otkrije slabost u njemu. U takvom slučaju, odabir i standardizacija novih, sigurnih algoritama za TLS, nakon što se otkrije greška u starim, do tada korištenim algoritmima, te njihova implementacija u odgovarajuće programske biblioteke vjerojatno bi trajala dugo, zbog čega bi mnoge aplikacije u međuvremenu bile prisiljene koristiti TLS s nesigurnim algoritmima. Zato, kako bi se izbjegla takva situacija, tj. kako se ne bi cijela sigurnost TLS-a urušila preko noći zbog slabosti u jednom algoritmu, TLS podržava različite algoritme kako bi se, ako se otkrije slabost u jednom algoritmu, aplikacije lako mogle prebaciti na drugi već standardizirani i implementirani algoritam.

Povijesno, TLS je u verzijama prije verzije 1.3, službeno podržavao veliki broj različitih kriptografskih algoritama. S vremenom je postalo jasno da toliko veliki broj službeno podržanih algoritama zapravo ne pruža puno koristi, te ujedno čini TLS nesigurnijim. Problem je bio u tome što, zbog velikog broja podržanih algoritama i ostalih postavki, postalo je prilično teško konfigurirati TLS na siguran način (11) (12). Iako je u teoriji TLS 1.2 moguće konfigurirati na siguran način, u praksi su mnoge aplikacije koje koriste TLS 1.2 nesigurno konfigurirane jer programeri i administratori nisu upoznati sa svim slabostima različitih TLS algoritama i postavki.

Primjerice, pretpostavimo da web stranica *online* bankarstva želi osigurati da joj klijenti pristupaju isključivo preko sigurne veze. Zato će administratori web stranice postaviti da ta web stranica bude dostupna isključivo putem HTTPS-a, odnosno HTTP-a zaštićenim TLS-om. U tom slučaju, ako je web stranica dostupna isključivo preko HTTPS-a, administratori bi mogli pomisliti da su napravili sve što treba da veza bude sigurna. No kako su prethodne verzije TLS-a (prije 1.3) podržavale veliki broj kriptografskih algoritama, uključujući neke nesigurne algoritme, ako administratori nisu izričito isključili sve nesigurne algoritme i postavke, i dalje je moguće je da se uspostavi TLS veza s nesigurnim algoritmima. Takvu vezu, iako je ona formalno zaštićena TLS-om, napadači bi potencijalno mogli prisluškivati ili bi mogli izmijeniti njen promet zbog korištenih nesigurnih algoritama.

Time se fleksibilnost dobivena velikim brojem podržanih algoritama i postavki, koja je naizgled prednost, pretvara u ozbiljnu manu. Upravo zato, jedna od novosti u TLS-u 1.3 je znatno smanjen broj podržanih kriptografskih algoritama i postavki, što u konačnici čini TLS sigurnijim (1) (11) (12).

U prethodnim verzijama TLS-a, jedna grupa kriptografskih algoritama (engl. *cipher suite*) obuhvaćala je algoritam autentifikacije veze, razmjene ključa (uspostave zajedničke tajne), algoritme za zaštitu zapisa (engl. *record protection algorithms*) i algoritam kriptografskog sažetka (4).

Primjerice, u prethodnim verzijama TLS-a bi grupa kriptografskih algoritama označena oznakom TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 značila korištenje:

- algoritma RSA za autentifikaciju,
- algoritma Diffie-Hellman za razmjenu ključa,
- 128-bitne inačice algoritma AES, u načinu rada CBC, za zaštitu zapisa,
- i algoritma kriptografskog sažetka SHA256.

U TLS-u 1.3, koncept grupe kriptografskih algoritama (engl. *cipher suite*) promijenjen je tako da su odvojeni mehanizmi autentifikacije i razmjene ključa od algoritma zaštite zapisa i funkcije kriptografskog sažetka (1). Tako primjerice oznaka grupe kriptografskih algoritama u TLS-u 1.3 TLS_AES_128_GCM_SHA256 znači korištenje:

- 128-bitne inačice algoritma AES u načinu rada GCM za zaštitu zapisa
- i algoritma kriptografskog sažetka SHA256.

Algoritam autentifikacije veze (npr. RSA) i razmjene ključa (npr. Diffie-Hellman) specificira se zasebno. Odabir algoritama autentifikacije, razmjene ključa i zaštite zapisa time postaje neovisan. Taj neovisan odabir znatno olakšava prethodno opisano TLS 1.3 rukovanje, jer klijent u poruci *ClientHello* mora pogoditi samo algoritam razmjene ključa koji je poslužitelju prihvatljiv, a ostale algoritme (za autentifikaciju i zaštitu zapisa) može predložiti i čekati odgovor poslužitelja.

Preostali algoritmi za zaštitu zapisa u TLS-u 1.3 su svi vrste *Authenticated Encryption with Associated Data* (algoritmi koji istovremeno osiguravaju tajnost i autentičnost podataka), te svi preostali asimetrični algoritmi razmjene ključa sada osiguravaju svojstvo buduće tajnosti (engl. *forward secrecy*) (1). U konačnici, TLS 1.3 podržava:

- algoritme RSA, ECDSA i EdDSA za autentifikaciju (uz različite funkcije kriptografskih sažetaka i odgovarajuće sheme potpisivanja),
- algoritme Diffie-Hellman i Elliptic Curve Diffie-Hellman za razmjenu ključa/uspostavu zajedničke tajne,
- pet grupa kriptografskih algoritama (engl. *cipher suites*) za zaštitu zapisa i algoritma kriptografskog sažetka (engl. *hash*):
    - TLS_AES_128_GCM_SHA256
    - TLS_AES_256_GCM_SHA384
    - TLS_CHACHA20_POLY1305_SHA256
    - TLS_AES_128_CCM_SHA256
    - TLS_AES_128_CCM_8_SHA256

Osim uklanjanja podrške nesigurnim i potencijalno nesigurnim kriptografskim algoritmima, TLS 1.3 uklonio je i podršku za neke druge mogućnosti koje nisu nudile značajnu korist, a mogle bi biti izvor problema. Uklonjene su sljedeće mogućnosti:

- kompresija,
- dogovaranje različitih formata točaka za algoritme eliptičnih krivulja (engl. *elliptic curve*),
- ponovni dogovor sigurnosnih parametara (engl. *renegotiation*),
- nestandardne (engl. *custom*) grupe za uspostavu zajedničke tajne algoritmom Diffie-Hellman.

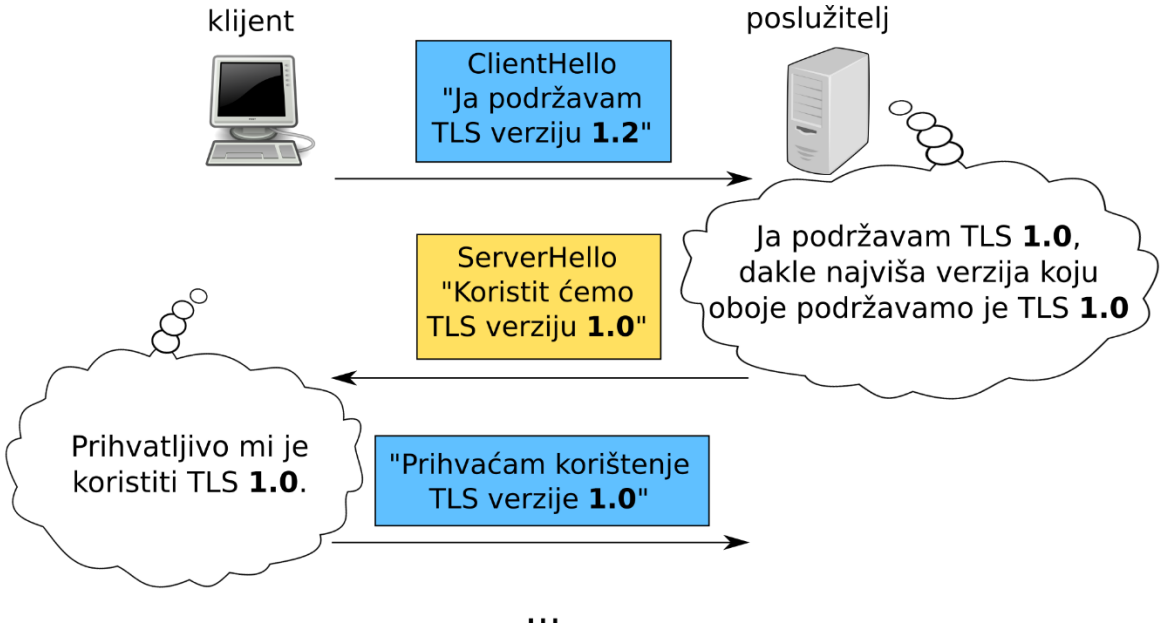
5 Zaštita od snižavanja korištene verzije TLS-a

Kako napadači ne bi mogli izmijeniti dogovorene sigurnosne parametre (verziju TLS-a, kriptografske algoritme i sl.) tijekom TLS rukovanja, završne poruke (*Finished*) na kraju rukovanja sadržavaju autentifikacijski kôd (engl. *message authentication code*, MAC) koji klijentu i poslužitelju potvrđuje da sadržaj poruka rukovanja nije mijenjan. Ako napadač ne može zaobići sigurnost koju pružaju završne (*Finished*) poruke, onda ne bi trebao moći ni mijenjati sadržaj poruka (npr. predložene algoritme ili TLS verziju) bez da klijent ili poslužitelj to otkriju (i zatim prekinu vezu).

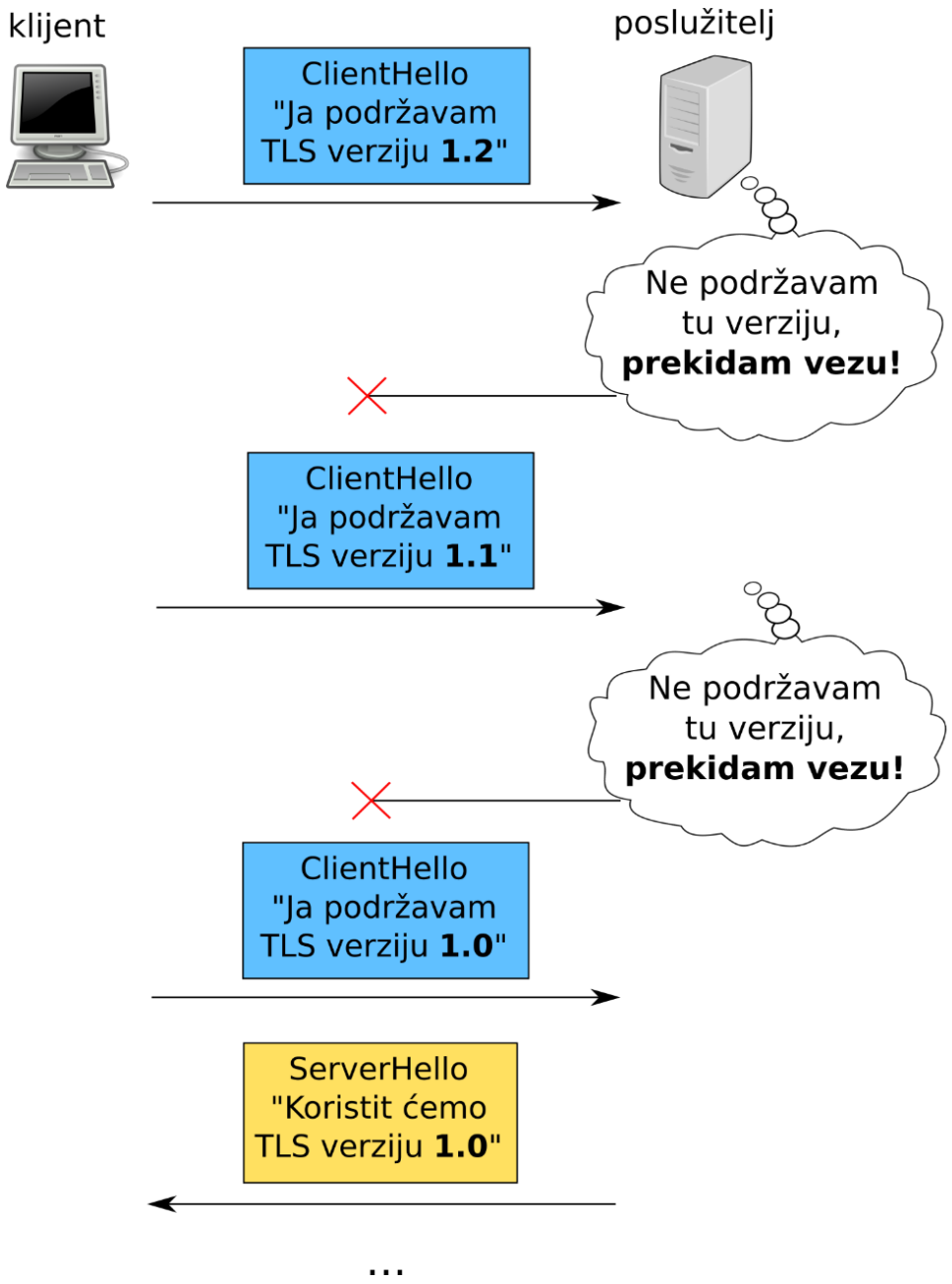
No i dalje u praksi postoje načini na koje bi napadač mogao utjecati na dogovor parametara između klijenta i poslužitelja. Primjerice, neki poslužitelji nisu ispravno implementirali mehanizam dogovora korištene verzije TLS-a (engl. *version negotiation*)[3] – kada bi im klijent predložio korištenje novije verzije TLS-a koju oni nisu podržavali (npr. TLS 1.2), umjesto da odgovore s najvišom verzijom TLS-a koju oni podržavaju (npr. TLS 1.0), ti poslužitelju su jednostavno prekinuli vezu. Kako bi ipak bili kompatibilni s takvim neispravnim poslužiteljima, neki klijenti (npr. web preglednici) su kod takvog prekida veze ponovno inicirali TLS rukovanje, a u tom novom rukovanju su predložili nižu verziju TLS-a kako bi veza s neispravnim poslužiteljima ipak bila uspostavljena. Takav je pristup riješio problem kompatibilnosti, no uzrokovao je sigurnosni problem – napadač koji može presretati mrežni promet može i namjerno uzrokovati prekid veze, te na taj način natjerati klijenta da ponovno inicira rukovanje s nižom, potencijalno nesigurnom verzijom TLS-a (13). Kako u tom slučaju napadač tehnički nije mijenjao sadržaj TLS poruka, ni klijent ni poslužitelj ne mogu znati radi li se o napadu ili o prekidu veze iz nekog drugog razloga (npr. neispravni poslužitelj).

Slike 11, 12 i 13 ilustriraju navedeni primjer. Slika 11 pojednostavljeno prikazuje dogovaranje korištene verzije TLS-a kod ispravnih implementacija TLS klijenta i poslužitelja. Slika 12 pojednostavljeno prikazuje dogovaranje korištene verzije TLS-a između neispravnog poslužitelja i klijenta koji se ponaša na prethodno opisani način kako bi zadržao kompatibilnost, koristeći tzv. nesigurno snižavanje korištene verzije TLS-a (engl. *insecure downgrade*). Slika 13 ilustrira kako napadač može prisilnim prekidima veze zloupotrijebiti mehanizam nesigurnog snižavanja verzije kako bi natjerao klijenta i poslužitelja da dogovore nižu (i nesigurniju) verziju TLS-a nego što bi inače dogovorili.

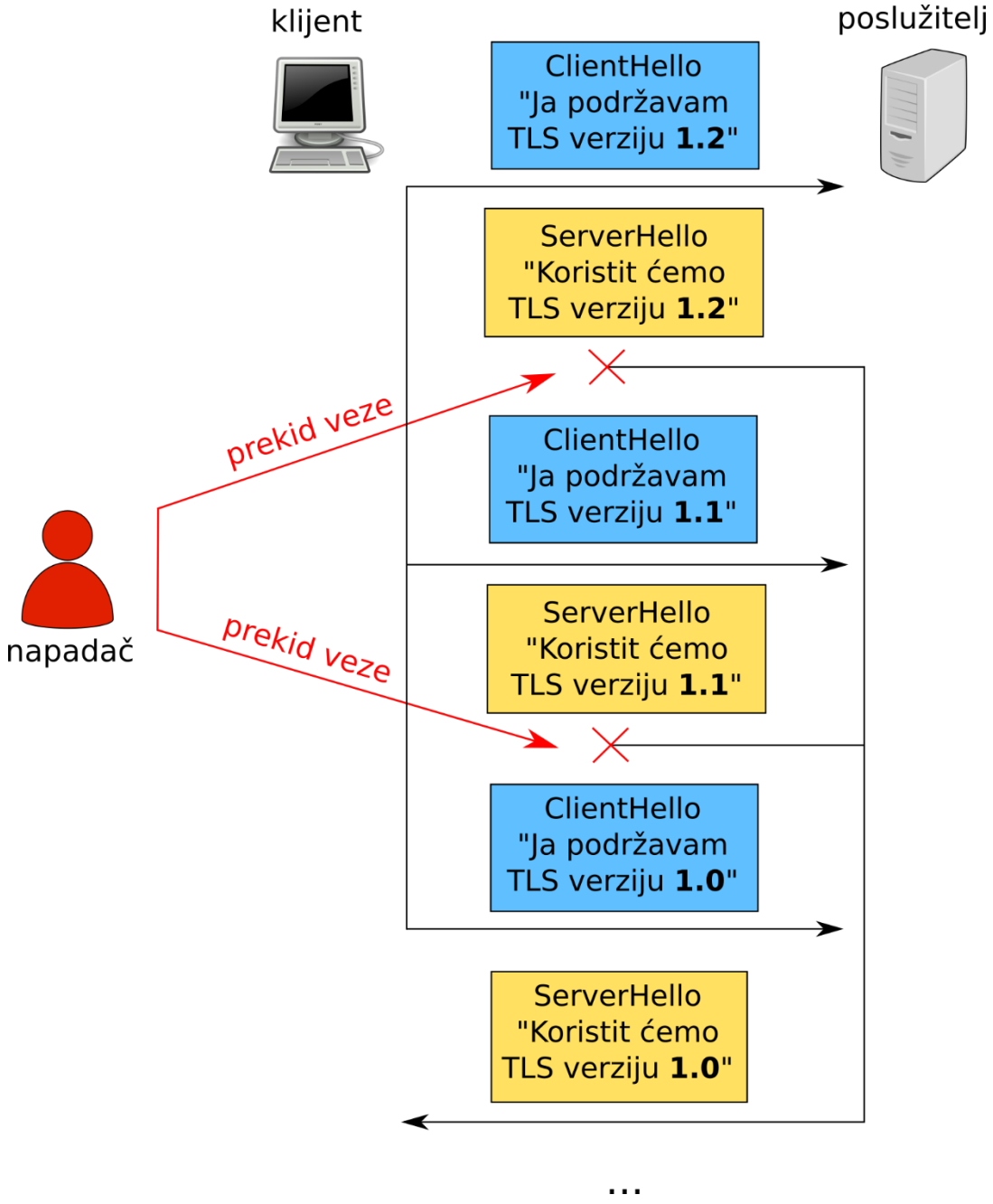
[3] detaljnije informacije o dogovoru korištene verzije (engl. *version negotiation*) i ostalim općenitim pojedinostima TLS-a dostupne su u prethodnom dokumentu Nacionalnog CERT-a: „TLS“



Slika 11 – pojednostavljeni prikaz dogovaranja korištene verzije TLS-a kod ispravnih implementacija TLS klijenta i poslužitelja



Slika 12 – pojednostavljeni prikaz dogovaranja korištene verzije TLS-a između neispravnog poslužitelja i kompatibilnog klijenta korištenjem tzv. nesigurnog snižavanja verzije TLS-a (engl. *insecure downgrade*)

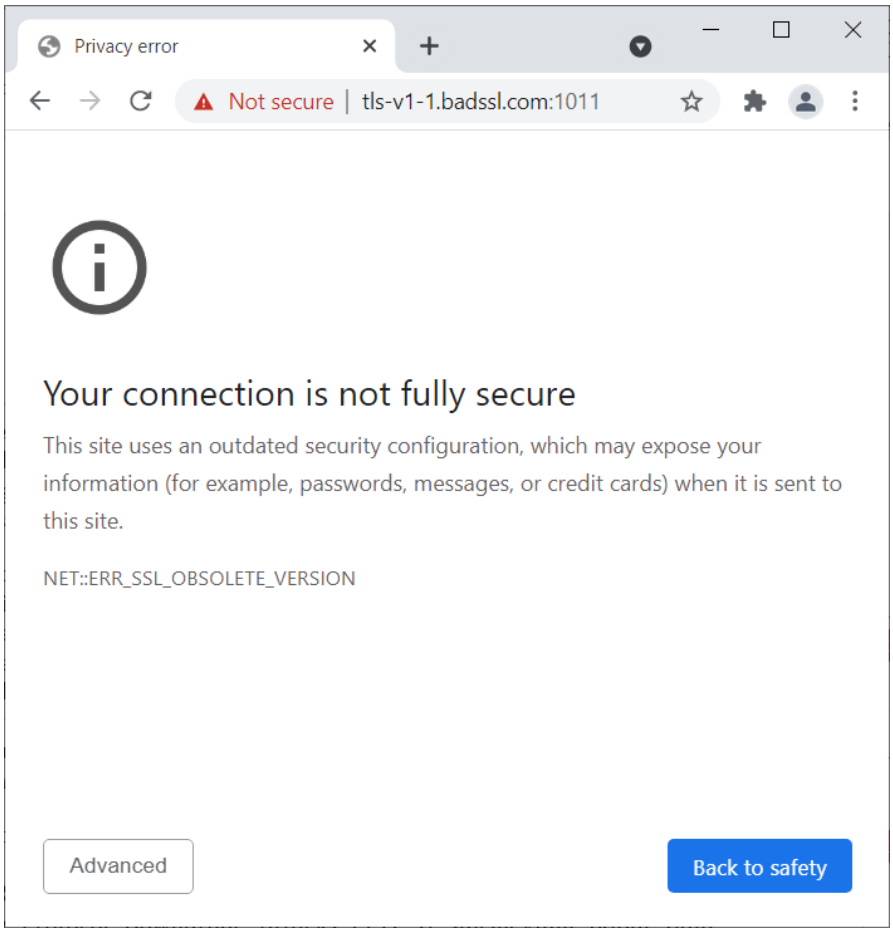


Slika 13 – uzastopnim prisilnim prekidima veze, napadač može zloupotrijebiti mehanizam nesigurnog snižavanja verzije (engl. *insecure downgrade*) kako bi natjerao klijenta i poslužitelja da dogovore nižu (i nesigurniju) verziju TLS-a

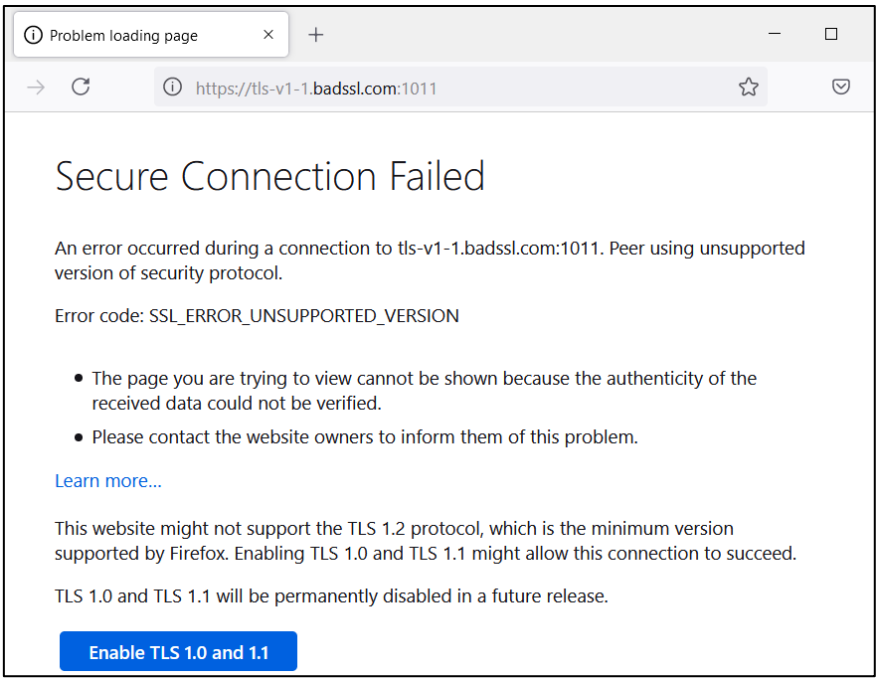
Jedino potpuno sigurno rješenje protiv takvog napada je izričito isključivanje podrške za ranije, potencijalno nesigurne verzije TLS-a. Primjerice, ako klijent ili poslužitelj strogo odbijaju koristiti bilo koju verziju TLS-a ispod 1.2, onda ih prethodno opisani napad ili slični napadi ne bi mogli natjerati da koriste raniju verziju, tj. TLS veza tada jednostavno ne bi bila uspostavljena.

Problem s tim pristupom je što se gubi kompatibilnost – npr. ako klijent isključi podršku za sve verzije TLS-a ispod 1.2, onda se neće moći spojiti na starije poslužitelje koji podržavaju samo te verzije. Iako su se proizvođači najvećih web preglednika (Chrome, Firefox, Microsoft Internet Explorer/Edge, Safari) dogovorili, te više od godinu dana unaprijed najavili, da će s početkom 2020. godine isključiti podršku za TLS verzije 1.0 i 1.1, to ipak tada nije napravljeno, već je odgođeno do sredine 2020. godine (14). Unatoč tome što je TLS 1.2 standardiziran još 2008. godine (4), isključivanje podrške za ranije verzije je čak i početkom 2020. godine, 12 godina kasnije, stvaralo probleme.

Od verzije 84 preglednika Google Chrome (15), te od verzije 78 preglednika Mozilla Firefox (16), korisnici koji otvaraju web stranicu koju štiti TLS 1.0 ili 1.1 vidjet će sigurnosno upozorenje prikazano na slikama 14 i 15, te se web stranica neće otvoriti bez da korisnici izričito dopuste korištenje starijih, manje sigurnih verzija TLS-a.



Slika 14 – sigurnosno upozorenje u pregledniku Google Chrome koje se pojavljuje kada korisnik pokuša otvoriti web stranicu zaštićenu TLS-om 1.0 ili 1.1



Slika 15 – sigurnosno upozorenje u pregledniku Mozilla Firefox koje se pojavljuje kada korisnik pokuša otvoriti web stranicu zaštićenu TLS-om 1.0 ili 1.1

Rješenja za ovakve probleme koji se javljaju kod prelaska na noviju verziju protokola bit će široko primijenjena samo ako se postigne razuman kompromis između sigurnosti i kompatibilnosti. Jedno rješenje koje postiže dobar kompromis standardizirano je kao neobavezni dodatak TLS-u kroz RFC7507 pod nazivom *TLS Fallback Signaling Cipher Suite Value (SCSV) for Preventing Protocol Downgrade Attacks*[4] (17). U slučajevima poput onih prikazanih na slikama 12 i 13, predloženo rješenje nalaže klijentima da u nastavak TLS rukovanja uključe oznaku da je predložena verzija TLS-a niža od one koju bi klijent inače predložio. Zatim, poslužitelji mogu na temelju te oznake detektirati da se potencijalno radi o napadu i prekinuti vezu. Time se sprječava napad prikazan na slici 13 bez negativnog utjecaja na kompatibilnost. Ovim rješenjem nije moguće spriječiti napadača koji u potpunosti može kompromitirati dogovorenu verziju TLS-a – jer tada bi napadač mogao ukloniti navedenu oznaku – no ipak je pružena viša razina sigurnosti za klijente kojima je kompatibilnost neophodna. Glavni nedostatak ovog rješenja je to što je ono naknadni, neobavezni dodatak TLS-u, zbog čega ga neki poslužitelji nisu implementirali, što na kraju ostavlja mnoge TLS veze bez zaštite (13).

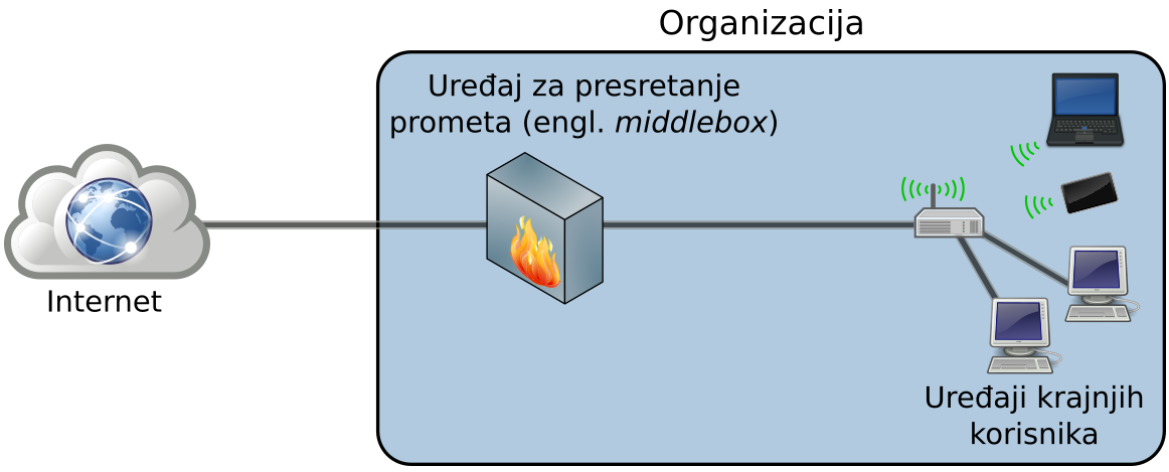
TLS 1.3 propisuje sličnu zaštitu protiv napada snižavanja korištene verzije TLS-a (engl. *version downgrade protection*). Kada poslužitelj koji podržava TLS 1.3 dogovori korištenje ranije verzije TLS-a, onda on unutar poruke *ServerHello*, unutar zadnjih 8 bajtova vrijednosti *random* postavlja oznaku da je dogovorena niža verzija TLS-a. Klijent koji podržava TLS 1.3 mora provjeriti tu vrijednost, te ako naiđe na navedenu oznaku, mora prekinuti vezu jer se po svemu sudeći radi o napadu. U usporedbi s prethodnim rješenjem koje je neobavezni dodatak TLS-u, značajna prednost ovog rješenja je činjenica da je ono obavezno za sve klijente i poslužitelje koji podržavaju TLS 1.3, tako da bi s vremenom taj mehanizam zaštite trebao postati gotovo univerzalan.

[4] specifikacija TLS-a 1.3 je kasnije predložila bolje rješenje, tako da je ovo rješenje po standardu RFC8996 označeno kao zastarjelo (engl. *obsolete*) i više se ne preporučuje njegovo korištenje

6 Kompatibilnost s uređajima za presretanje prometa

Jedan od neočekivanih problema prilikom oblikovanja i standardizacije TLS-a 1.3 bio je problem kompatibilnosti s uređajima za presretanje prometa. Kako bi testirali trenutnu verziju novog standarda, neki web preglednici (TLS klijenti) i neke web stranice (TLS poslužitelji) su implementirali tada aktualni, eksperimentalni standard TLS-a 1.3 te su mjerili postotak uspješnih uspostava TLS veza. Rezultat mjerenja bio je neočekivano loš – u usporedbi s TLS-om 1.2, pokušaji uspostave veze novom verzijom TLS-a 1.3 relativno često nisu bili uspješni (13).

Otkriveno je da su značajni dio problema nekompatibilni uređaji za presretanje prometa (engl. *middleboxes*). To su uređaji koji se nalaze između klijenta (korisnika) i poslužitelja (npr. web stranice) te im je svrha nadzor mrežnog prometa te eventualno i izmjena prometa ili prekid veze. Primjerice, takvi se uređaji koriste u tvrtkama koje žele nadzirati sigurnost svog mrežnog prometa i zaustaviti potencijalne napade, ili kao dio infrastrukture pružatelja mrežnih usluga kako bi mogli blokirati pristup zabranjenim web stranicama (13) (18). Na slici 16 prikazan je primjer mrežne pozicije uređaja za presretanje prometa (engl. *middlebox*) unutar organizacije.



Slika 16 – primjer mrežne pozicije uređaja za presretanje prometa (engl. *middlebox*) unutar organizacije

Po svemu sudeći, problem je bio u tome što ti uređaji nisu bili kompatibilni s potencijalnim novim verzijama TLS-a, uključujući tada eksperimentalnu verziju TLS-a 1.3. Neki od takvih uređaja su neispravno implementirali specifikaciju TLS-a, zbog čega bi jednom kada naiđu na noviju verziju TLS-a od očekivane jednostavno prekinuli vezu. Zato korisnici koji su se nalazili u organizaciji s nekompatibilnim uređajem za presretanje prometa nisu mogli uspješno uspostaviti vezu tadašnjom eksperimentalnom verzijom TLS-a 1.3 (13).

Slično kao i kod problema iz prošlog poglavlja, da bi se rješenje ovog problema smatralo uspješnim također bi se trebala zadržati i kompatibilnost i sigurnost. Idealno bi bilo da se uređaji za presretanje prometa ažuriraju, da im se isprave greške i da tako postanu kompatibilni s novim verzijama TLS-a, no taj proces ažuriranja bi vjerojatno trajao godinama. S druge strane, ignoriranje problema kompatibilnosti značilo bi da korisnicima koji se nalaze iza takvih uređaja efektivno ne bi radilo pristupanje web stranicama ako ne isključe TLS 1.3 u svojim web preglednicima i ostalom klijentskom softveru.

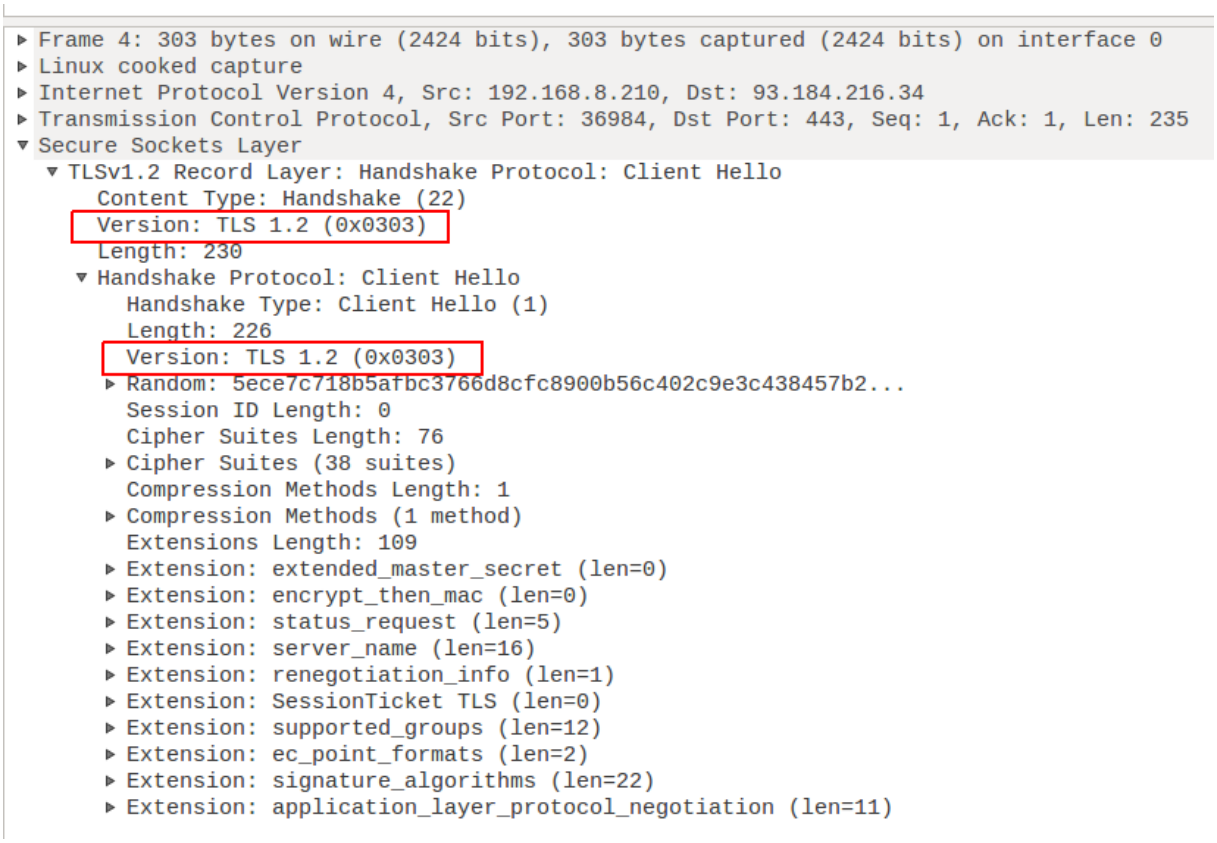
U konačnici, doneseno je jedno neobično rješenje koje je ipak uspjelo zadržati i kompatibilnost i sigurnost – poruke TLS-a 1.3 izmijenjene su tako da nalikuju porukama ranijih verzija TLS-a, što je bilo dovoljno da većina uređaja za presretanje prometa ipak ne zaustavlja uspostavu veza takvim izmijenjenim TLS-om 1.3 (13).

Najznačajnije su izmijenjene strukture poruka za dogovaranje korištene verzije TLS-a. Dok su u prethodnom mehanizmu klijent i poslužitelj signalizirali najvišu podržanu verziju TLS-a kroz jedno polje u početnim *ClientHello* i *ServerHello* porukama, sada je vrijednost u tom polju fiksirana, a stvarno polje za dogovaranje korištene verzije je preseljeno u novi obavezni TLS dodatak *supported_versions*. Stari i novi način dogovora korištene verzije TLS-a prikazan je kroz analizu snimke prometa alatom *Wireshark* na slikama 17, 18, 19 i 20:

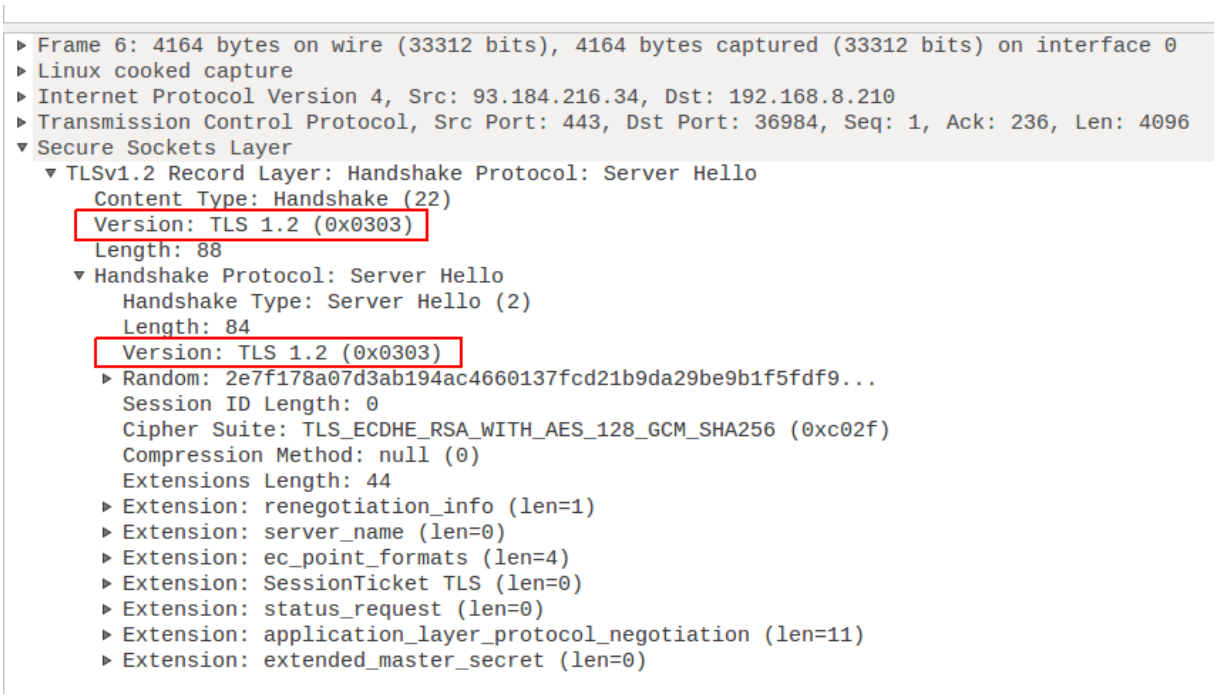
- slika 17 prikazuje poruku *ClientHello* za TLS verziju 1.2,
- slika 18 prikazuje poruku *ServerHello* za TLS verziju 1.2,
- slika 19 prikazuje poruku *ClientHello* za TLS verziju 1.3,
- slika 20 prikazuje poruku *ServerHello* za TLS verziju 1.3.

Na slikama 17 i 18 je vidljivo kako je i u vanjskom sloju (*Record Layer*) i u sadržaju poruke rukovanja (*Handshake Protocol*) označena TLS verzija 1.2. Oznaka verzije u vanjskom sloju zapravo nije posebno značajna, te u njoj može biti zapisana i neka ranija verzija protokola (npr. TLS 1.0), no oznaka verzije unutar poruke rukovanja je ključna za dogovor korištene verzije u svim verzijama TLS-a prije TLS-a 1.3.

Na slikama 19 i 20 je vidljivo kako, unatoč tome što se koristi verzija TLS-a 1.3, u vanjskom sloju (*Record Layer*) i u sadržaju poruke rukovanja (*Handshake Protocol*) je označeno korištenje ranijih verzija TLS-a zbog kompatibilnosti. Stvarna oznaka korištene verzije TLS-a vidljiva je niže u dodatku (engl. *extension*) *supported_versions*.

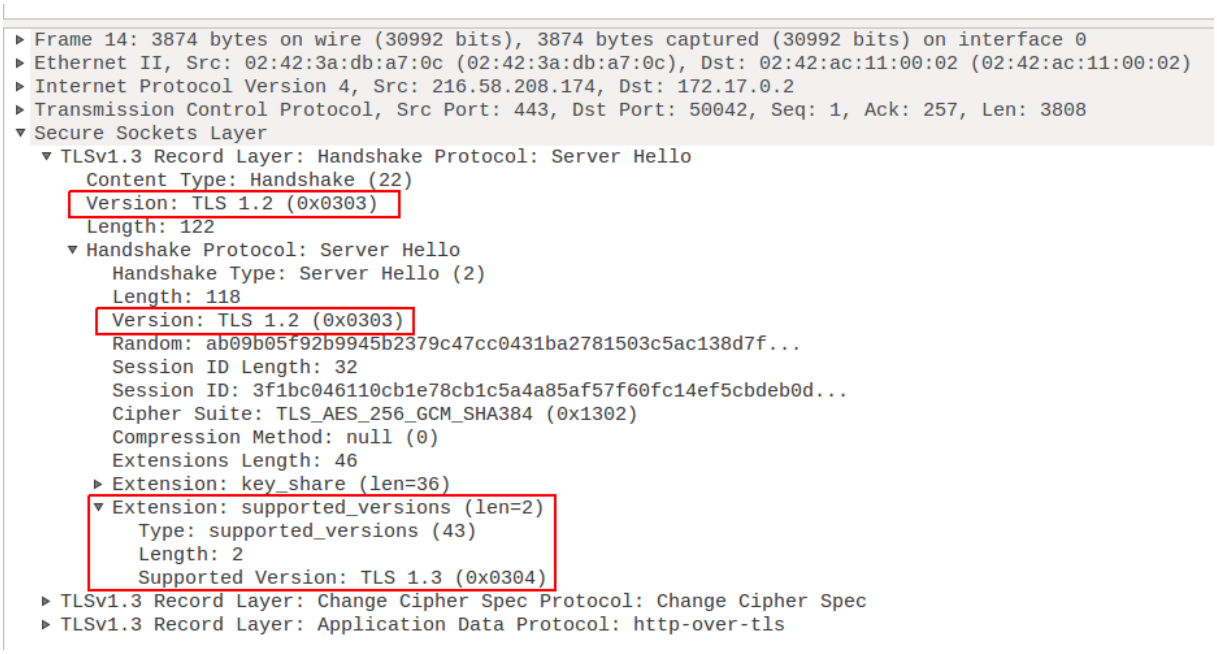


Slika 17 – poruka *ClientHello* za TLS verziju 1.2



Slika 18 – poruka *ServerHello* za TLS verziju 1.2

Slika 19 – poruka *ClientHello* za TLS verziju 1.3



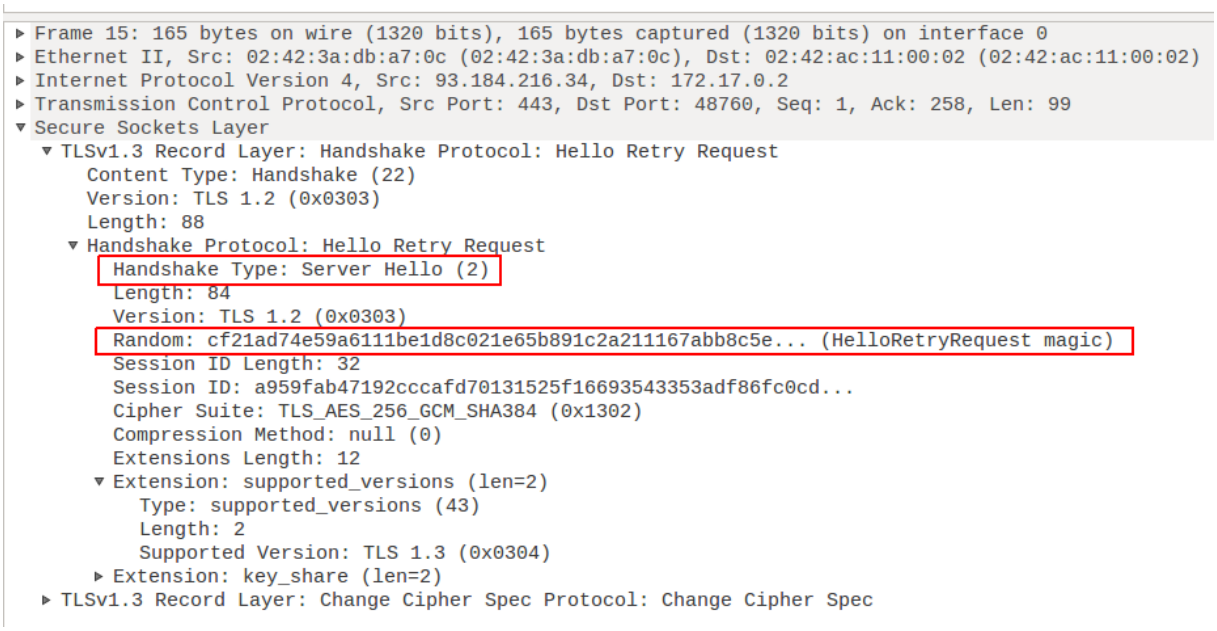
Slika 20 – poruka *ServerHello* za TLS verziju 1.3

Osim mehanizma dogovaranja korištene verzije, napravljene su i sljedeće izmjene kako ne bi bilo problema s kompatibilnosti s uređajima za presretanje prometa:

- struktura nove poruke *HelloRetryRequest* je prilagođena tako da nalikuje na poruku *ServerHello*; poruku *HelloRetryRequest* moguće je razlikovati od poruke *ServerHello* po tome što se u polju *random* nalazi vrijednost koja odgovara rezultatu funkcije SHA-256 za ulazni niz znakova "HelloRetryRequest" (prikazano kroz alat *Wireshark* na slici 21),

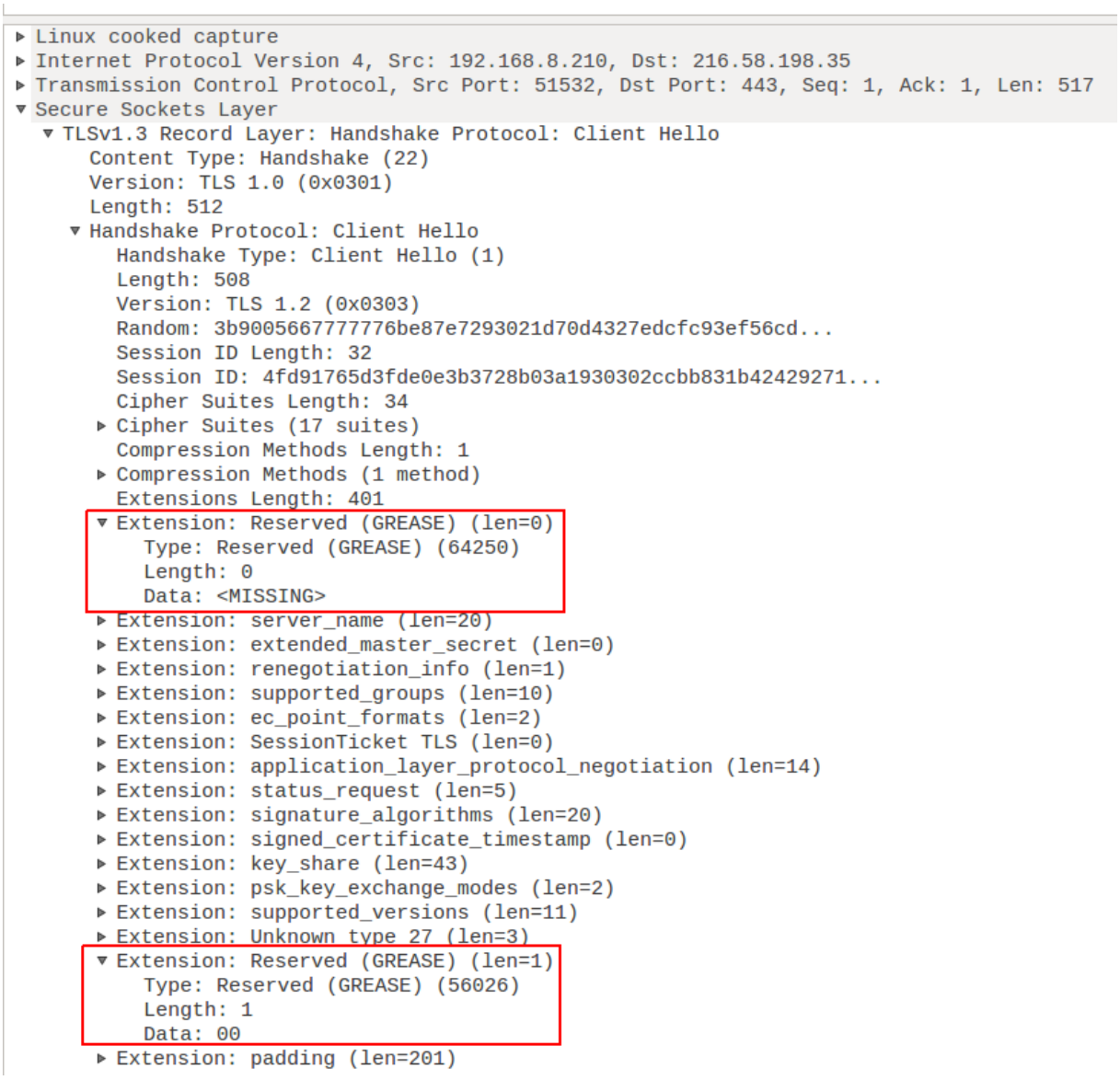
- iako se poruka *ChangeCipherSpec* u TLS-u 1.3 funkcionalno ne koristi ni za što, dopušteno je njeno slanje,

- slično tome, iako se u TLS-u 1.3 ne koristi identifikator sjednice (*session_id*) niti postoji mogućnost korištenja kompresije, TLS 1.3 poruke i dalje sadržavaju polje *session_id* koje bi u sebi trebalo imati naizgled nasumičnu vrijednost te polje za odabir metode kompresije koje bi trebalo biti prazno što označava da se ne koristi kompresija.



Slika 21 – TLS 1.3 poruka *HelloRetryRequest* nalikuje poruci *ServerHello* s posebnom vrijednosti unutar polja *random*

Kako bi se ubuduće spriječili ili barem ublažili slični problemi s kompatibilnosti, pripremljen je standard RFC8701 pod nazivom *Generate Random Extensions And Sustain Extensibility* (skraćeno GREASE). Taj standard klijentima i poslužiteljima dopušta slanje unaprijed definiranih, ali naizgled novih, nepredviđenih vrijednosti u gotovo svim poljima TLS poruka gdje bi ispravna implementacija TLS-a trebala ignorirati nepoznate vrijednosti zbog buduće kompatibilnosti. Takvim povremenim slanjem neočekivanih vrijednosti trebale bi se otkriti greške u implementacijama znatno ranije, idealno prije nego one postanu rasprostranjene i ugroze implementaciju novijih funkcionalnosti i verzija TLS-a (19). Na slici 22 je kroz alat *Wireshark* prikazan primjer kako web preglednik *Chromium* koristi naizgled nove, nepoznate dodatke (engl. *extensions*) unutar TLS *ClientHello* poruke kako bi se na vrijeme otkrile neispravne implementacije TLS poslužitelja koje ne ignoriraju takve nepoznate vrijednosti.



Slika 22 – web preglednik Chromium koristi naizgled nove, nepoznate dodatke (engl. *extensions*) unutar TLS *ClientHello* poruke kako bi se na vrijeme otkrile neispravne implementacije TLS poslužitelja koje ne ignoriraju takve nepoznate vrijednosti (standard RFC8701 – GREASE)

7 Zaključak

Protokol *Transport Layer Security*, TLS, temelj je sigurnosti mrežnog prometa koji putuje internetom, te izmjene u verziji 1.3 donose značajna sigurnosna poboljšanja i ubrzanje uspostave veze.

TLS 1.3 standardiziran je u kolovozu 2018. godine (1) te je podrška za verziju 1.3 implementirana:

- u široko korištenim programskim bibliotekama (*OpenSSL*, *GnuTLS*, *NSS*, *wolfSSL*...) (20) (21),
- u većini web preglednika (*Google Chrome*, *Mozilla Firefox*, *Microsoft Edge*, *Safari*...) (22) (21),
- u poslužiteljskom softveru poput Apache i NGINX HTTP poslužitelja (23).

Statistike tvrtke Qualys iz svibnja 2021. godine pokazuju da oko 45% web stranica podržava uspostavu veze TLS-om 1.3 (24), dok podaci IETF-a i često korištenih web preglednika iz kolovoza 2019. prikazuju da se tada, samo godinu dana nakon standardizacije, oko 30% TLS veza uspostavljalo verzijom 1.3 (21).

Izmjene u TLS-u 1.3 donijele su puno poboljšanja, no posao unaprjeđenja TLS-a i povezanih protokola još nije gotov. Dva značajna napretka očekuju se kroz razvoj:

- standarda za šifriranja sadržaja TLS poruke *ClientHello* (uključujući sadržaja vrijednosti *Server Name Indication*, SNI) (25),
- protokola HTTP/3 (9).

Šifriranje sadržaja poruke *ClientHello*, uz druge napretke poput šifriranja DNS prometa, predstavljalo bi značajan napredak privatnosti za krajnje korisnike. Kada bi te ključne poruke bile šifrirane, bilo bi značajno teže otkriti koje web stranice korisnik posjećuje samo na temelju mrežnog prometa.

Razvoj nove verzije 3 aplikacijskog protokola HTTP, koji se temelji na novom transportnom protokolu QUIC (8), dodatno bi ubrzalo učitavanje web stranica, između ostaloga i kroz uklanjanje jednog vremena obilaska (RTT) koje trenutno nije moguće izbjeći kada se koristi transportni protokol TCP.

Uz postojeća poboljšanja koja donosi TLS 1.3, i navedena buduća poboljšanja, otvaranje web stranica, korištenje mobilnih aplikacija, slanje poruka e-pošte i druge radnje na internetu postat će sve sigurnije i sve brže.

8 Literatura

1. **Rescorla, Eric.** The Transport Layer Security (TLS) Protocol Version 1.3. *Internet Engineering Task Force.* [Mrežno] kolovoz 2018. [Citirano: 28. siječnja 2020.] https://datatracker.ietf.org/doc/html/rfc8446.

2. **Prodromou, Agathoklis.** TLS Security 2: A Brief History of SSL/TLS. *Acunetix.* [Mrežno] 31. ožujak 2019. [Citirano: 27. svibnja 2020.] https://www.acunetix.com/blog/articles/history-of-tls-ssl-part-2/.

3. **Moriarty, Kathleen i Farrell, Stephen.** Deprecating TLS 1.0 and TLS 1.1. *IETF.* [Mrežno] ožujak 2021. [Citirano: 17. lipnja 2021.] https://datatracker.ietf.org/doc/html/rfc8996.

4. **Dierks, Tim i Rescorla, Eric.** The Transport Layer Security (TLS) Protocol Version 1.2. *IETF.* [Mrežno] kolovoz 2008. [Citirano: 27. svibnja 2020.] https://datatracker.ietf.org/doc/html/rfc5246.

5. **Langley, Adam, Modadugu, Nagendra i Moeller, Bodo.** Transport Layer Security (TLS) False Start. *IETF.* [Mrežno] kolovoz 2016. [Citirano: 27. svibnja 2020.] https://datatracker.ietf.org/doc/html/rfc7918.

6. **wolfSSL.** TLS 1.3 Performance Part 1 – Resumption. [Mrežno] 23. svibnja 2018. [Citirano: 10. lipnja 2020.] https://www.wolfssl.com/tls-1-3-performance-resumption/.

7. —. TLS 1.3 Performance Analysis – Full Handshake. [Mrežno] 15. travnja 2019. [Citirano: 10. lipnja 2020.] https://www.wolfssl.com/tls-1-3-performance-part-2-full-handshake-2/.

8. **Iyengar, Jana i Thomson, Martin.** QUIC: A UDP-Based Multiplexed and Secure Transport. *IETF.* [Mrežno] svibanj 2021. [Citirano: 17. lipnja 2021.] https://datatracker.ietf.org/doc/html/rfc9000.

9. **Bishop, Mike.** Hypertext Transfer Protocol Version 3 (HTTP/3) (draft-ietf-quic-http-34). *IETF.* [Mrežno] 2. veljače 2021. [Citirano: 17. lipnja 2021.] https://datatracker.ietf.org/doc/html/draft-ietf-quic-http-34.

10. **Thomson, Martin, Nottingham, Mark i Tarreau, Willy.** Using Early Data in HTTP. *IETF.* [Mrežno] rujan 2018. [Citirano: 4. listopada 2020.] https://datatracker.ietf.org/doc/html/rfc8470.

11. **Valsorda, Filippo.** An overview of TLS 1.3 and Q&A. *The Cloudflare Blog.* [Mrežno] 23. rujna 2016. [Citirano: 4. lipnja 2020.] https://blog.cloudflare.com/tls-1-3-overview-and-q-and-a/.

12. **Jackson, Brian.** An Overview of TLS 1.3 – Faster and More Secure. *Kinsta Managed WordPress Hosting.* [Mrežno] 2. lipnja 2020. [Citirano: 10. lipnja 2020.] https://kinsta.com/blog/tls-1-3/.

13. **Sullivan, Nick.** Why TLS 1.3 isn't in browsers yet. *The Cloudflare Blog.* [Mrežno] 26. prosinca 2017. [Citirano: 18. lipnja 2020.] https://blog.cloudflare.com/why-tls-1-3-isnt-in-browsers-yet/.

14. **Keizer, Gregg.** Browser makers cite coronavirus, restore support for obsolete TLS 1.0 and 1.1 encryption. *Computerworld.* [Mrežno] 3. travnja 2020. [Citirano: 23. lipnja 2020.] https://www.computerworld.com/article/3535806/browser-makers-cite-coronavirus-restore-support-for-obsolete-tls-10-and-11-encryption.html.

15. **Google.** TLS 1.0 and TLS 1.1 - Chrome Platform Status. [Mrežno] 2020. [Citirano: 17. lipnja 2021.] https://www.chromestatus.com/feature/5759116003770368.

16. **Mozilla.** Firefox 78.0, See All New Features, Updates and Fixes. [Mrežno] lipanj 2020. [Citirano: 17. lipnja 2021.] https://www.mozilla.org/en-US/firefox/78.0/releasenotes/.

17. **Moeller, Bodo i Langley, Adam.** TLS Fallback Signaling Cipher Suite Value (SCSV) for Preventing Protocol Downgrade Attacks. *IETF.* [Mrežno] travanj 2015. [Citirano: 18. lipnja 2020.] https://datatracker.ietf.org/doc/html/rfc7507.

18. **Bursztein, Elie.** Understanding the prevalence of web traffic interception. *The Cloudflare Blog.* [Mrežno] 12. rujna 2017. [Citirano: 18. lipnja 2020.] https://blog.cloudflare.com/understanding-the-prevalence-of-web-traffic-interception/.

19. **Benjamin, David.** Applying GREASE to TLS Extensibility. *IETF.* [Mrežno] siječanj 2020. [Citirano: 18. lipnja 2020.] https://datatracker.ietf.org/doc/html/rfc8701.

20. **Kario, Hubert.** Transport Layer Security version 1.3 in Red Hat Enterprise Linux 8. *Red Hat blog.* [Mrežno] 8. kolovoza 2019. [Citirano: 30. lipnja 2020.] https://www.redhat.com/en/blog/transport-layer-security-version-13-red-hat-enterprise-linux-8.

21. **Salowey, Joseph A., Turner, Sean i Wood, Christopher A.** TLS 1.3: One Year Later. *IETF.* [Mrežno] 17. prosinca 2019. [Citirano: 18. lipnja 2020.] https://ietf.org/blog/tls13-adoption/.

22. **caniuse.com.** Can I use... Support tables for HTML5, CSS3, etc. [Mrežno] 2020. [Citirano: 30. lipnja 2020.] https://caniuse.com/#feat=tls1-3.

23. **Saive, Ravi.** How to Enable TLS 1.3 in Apache and Nginx. *Tecmint.* [Mrežno] 16. srpnja 2019. [Citirano: 30. lipnja 2020.] https://www.tecmint.com/enable-tls-in-apache-and-nginx/.

24. **Qualys SSL Labs.** SSL Pulse. [Mrežno] 3. lipnja 2020. [Citirano: 24. lipnja 2020.] https://www.ssllabs.com/ssl-pulse/.

25. **Rescorla, Eric, i dr.** TLS Encrypted Client Hello (draft-ietf-tls-esni-11). *IETF.* [Mrežno] 14. lipnja 2021. [Citirano: 17. lipnja 2021.] https://datatracker.ietf.org/doc/html/draft-ietf-tls-esni-11.