SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2064

ANALIZA I KATEGORIZACIJA MREŽNOG PROMETA ŠIFRIRANOG KORIŠTENJEM PROTOKOLA TLS

KORISNIČKE UPUTE

Marko Plantić

Zagreb, veljača 2020.

Sadržaj

1.1. Uvod

Ovim korisničkim uputama je opisan rad skripta za analizu šifriranog mrežnog prometa u sklopu diplomskog rada na temu "Analiza i kategorizacija mrežnog prometa šifriranog korištenjem protokola TLS". Postoji sveukupno tri skripta koje analiziraju snimljeni mrežni promet. One su:

1. client_hello_analyzer.js – analizira client hello poruke snimljenih TLS paketa za identifikaciju web preglednika iz liste skupa sigurnosnih algoritama
2. tcp_syn_analyzer.js – analizira TCP SYN pakete za TLS promet, koristi se za dobivanje jedinstvenih vrijednosti TTL i veličine prozora
3. web_analyzer.js – analizira snimljen TLS promet preko Wireshark-a te uspoređuje sa snimljenim prometom u web pregledniku

1.2. Pokretanje skripti

Za pokretanje gore navedenih skripti su potrebni sljedeći alati:

- git - https://git-scm.com/downloads
- nodejs - https://nodejs.org/en/download/

Koraci za preuzimanje i pokretanje skripti:

1. $ git clone https://github.com/Kuyss/diplomski_rad.git
2. $ cd diplomski_rad
3. $ npm install
4. $ cd src/scripts
5. $ node [naziv_skripte] – pokretanje skripte

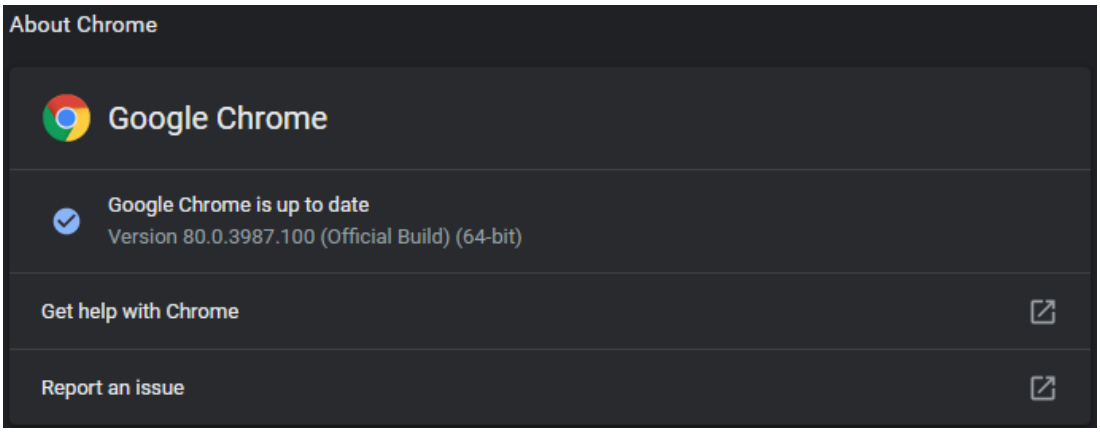
1.3. Korištenje skripte za analizu client hello poruka

Za analizu client hello poruka TLS prometa se koristi skripta client_hello_analyzer.js. Skripta se pokreće na sljedeći način:

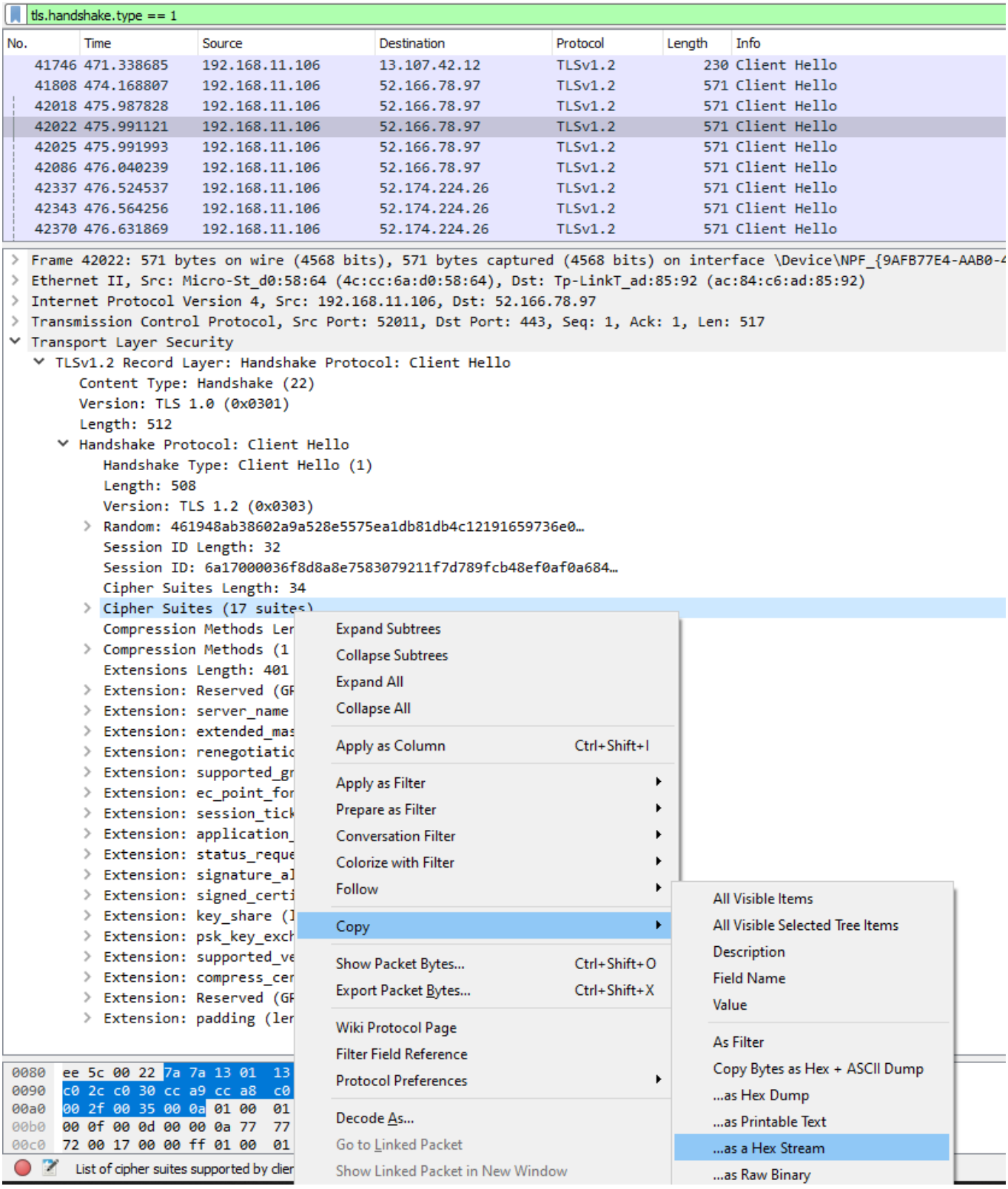
- $ node client_hello_analyzer.js

Ova skripta kao ulazne podatke sadrži sljedeće datoteke:

- tls_clientHello.pcap – pcap datoteka koja sadrži sve client hello poruke tijekom uspostave sigurne TLS veze. TLS client hello poruke se dobivaju korištenjem filtera *tls.handshake.type == 1* u Wireshark display filterima.
- cipher_suite_db.json – JSON datoteka koja sadrži bazu čiji elementi su verzija web preglednika i lista skupa sigurnosnih algoritama u heksadekadskom formatu. Baza se dobiva tako da se u web pregledniku pristupi nekoj web stranici sa HTTPS vezom te se u Wireshark-u snimi client hello poruka. Slika 1 prikazuje verziju korištenog preglednika Google Chrome. Do nje dolazimo preko options > Help > About Google Chrome. Zatim se pronađe client hello poruka za taj preglednik u Wireshark-u te se odabere u drugom prozoru Transport Layer Protocol > TLS Record Layer > Handshake Protocol > Cipher Suites. Zatim se desnim klikom na Cipher Suites odabere izbornik Copy > …as a Hex Stream. Slika 2. prikazuje ovaj korak. Zadnji korak je dodavanje vezije preglednika i liste skupa sigurnosnih algoritama u JSON datoteku koja predstavlja bazu podataka (Slika 3).



Slika 1. Korištena verzija Google Chrome web preglednika



Slika 2. Kopiranje liste skupa sigurnosnih algoritama

Slika 3. Baza verzije web preglednika i liste skupa sigurnosnih algoritama

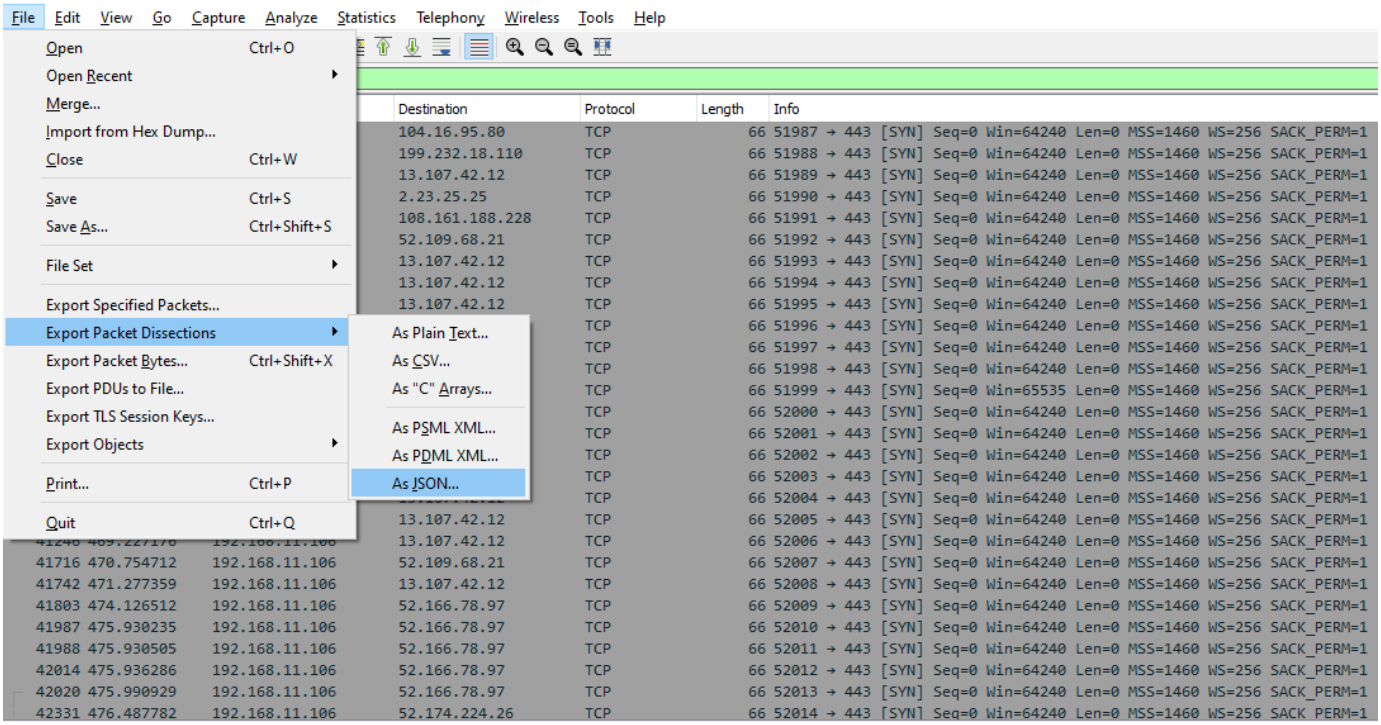
Skripta obrađuje svaki paket iz tls_clientHello.pcap datoteke, određuje listu skupa sigurnosnih algoritama te ju uspoređuje sa bazom spremljenom u cipher_suite_db.json. Rezultat usporedbe su verzije web preglednika te broj stvorenih veza od svakog preglednika te se printa u konzou.

1.4. Korištenje skripte za analizu TCP SYN paketa

Za analizu TCP SYN paketa se koristi skripta tcp_syn_analyzer.js. Pokreće se sljedećom naredbom:

- $ node tcp_syn_analyzer.js

Kao ulaz sadrži datoteku tcp_syn.json koja predstavlja snimljene TCP SYN pakete izvezene u JSON formatu. Korišteni filter za prikaz TCP SYN paketa *je tcp.flags.syn == 1 && tcp.flags.ack == 0*. Slika 4 prikazuje način stvaranja JSON formata snimljenih TCP SYN paketa. Zatim skripta za svaki paket odabire TTL i veličinu prozora te uspoređuje sa vrijednostima spremljenim u listi. Ako već postoje iste vrijednosti, ne ubacuje se u listu. Ako ne postoje, ubacuje se u listu. Rezultat je datoteka tcp_syn_db.json u kojoj su spremljene različite vrijednosti TTL i veličine prozora.



Slika 4. Izvoz TCP SYN paketa u JSON format

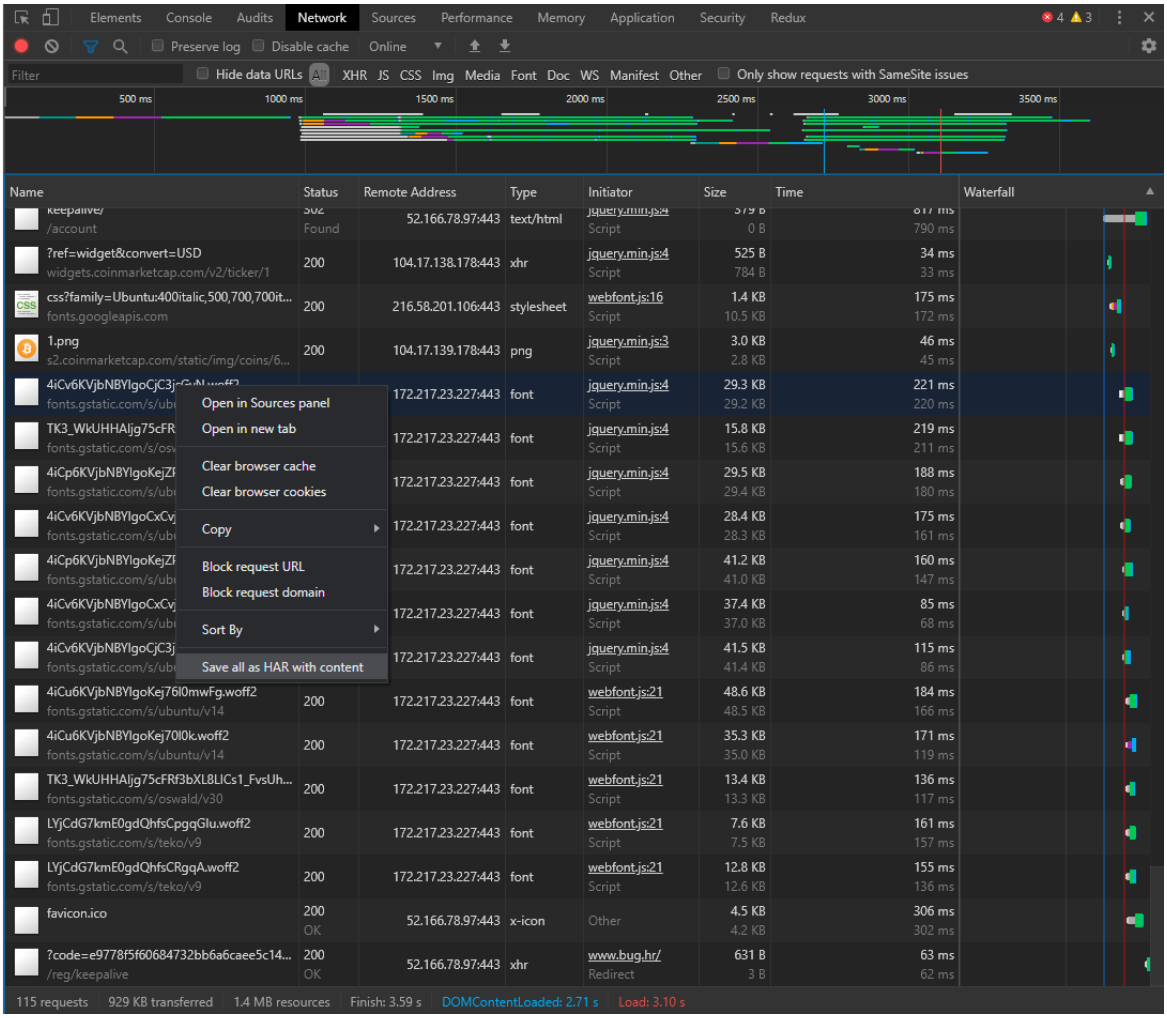
1.5. Korištenje skripte za analizu šifriranih podataka

Za analizu šifriranih podataka se koristi skripta web_analyzer.js. Njome analiziramo i identificiramo kojem web članku bug.hr portala je korisnik kojeg snimamo pristupio.

Pokreće se na dva načina:

- $ node web_analyzer.js calc – obrađuje spremljene har datoteke prometa snimljenog web preglednikom. Prvo se pozicioniramo na članak koji želimo snimati. Zatim pritisnemo F12 čime otvaramo Chrome DevTools. Zatim odaberemo refresh ikonu > desni klik mišem > Empty Cache and Hard Reload (Slika 5). Ovime omogućujemo da se svi resursi iz poslužitelja povuku, bez obzira jesu li bili spremljeni (keširani) u priručni spremnik. Nakon što se sav promet učita, odaberemo Network tab u DevTools-ima, desni klik mišem > save all as HAR with content (slika 6).

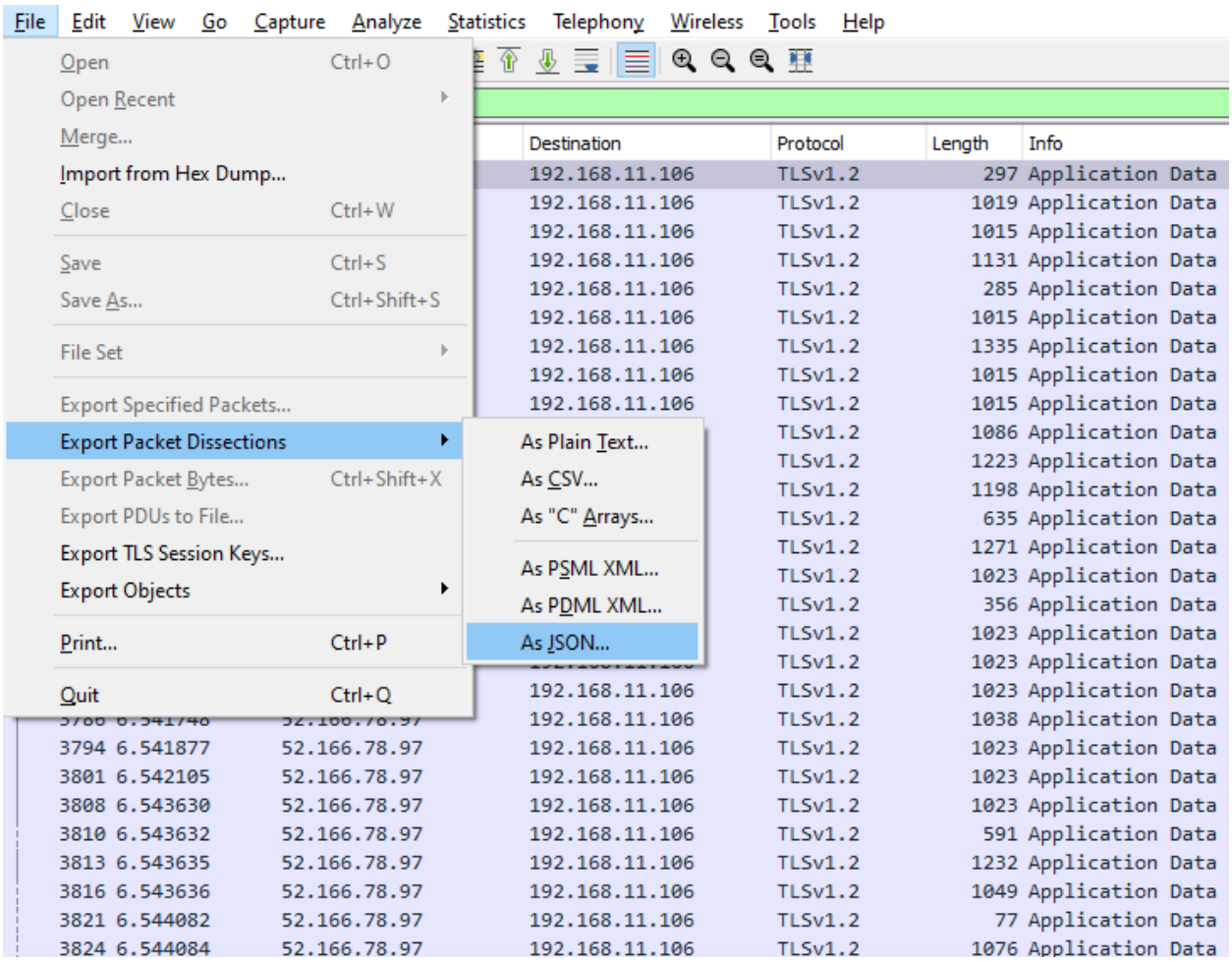
Slika 5. Odabir načina učitavanja web članka



Slika 6. Spremanje snimljenog prometa u HAR datoteku

Važno je napomenuti da je potrebno spremiti har datoteke u formatu data{num}.har, gdje je num broj od 1 do 10.

- $ node web_analyzer.js get – za ovaj način rada je potrebno snimiti promet preko Wireshark-a korištenjem sljedećeg filtra za svaku posjetu web članku: *ip.src == 52.166.78.97 && tls.app_data*. Time dobijemo odgovor od bug.hr poslužitelja sa svim vraćenim resursima. Sljedeći korak je spremiti snimljeni promet za pojedini članak u JSON formatu te nazvati datoteke capture{num}.json, gdje je num broj od 1 do 10 (slika 7). Zatim se za svaki snimljeni članak uspoređuje veličina vraćenih resursa sa bazom dobivenom prvim korakom te ispisuje najvjerojatnije pristupljeni članak.



Slika 7. Snimljeni promet za pojedini web članak bug.hr portala

1.6. Završne napomene

U GitHub repozitoriju diplomskog rada ne postoje datoteke sa snimljenim TCP SYN paketima jer zauzimaju ogromnu količinu prostora (500MB). Također postoji skripta crawler.js kojom sam pokušao automatizirati proces dohvata članka sa bug.hr portala i spremanja har datoteka. Na kraju sam odusato od tog načina analize veličine vraćenih resursa za svaki članak jer se koristio web preglednik iz komandne linije. Veličine vraćenih resursa web preglednika Google Chrome i onog iz komandne linije se nisu podudarale. Pretpostavio sam da je razlika zbog različitih implementacija preglednika te bi se možda odabirom drugog preglednika iz komandne linije mogli dobiti konzistentniji rezultati.