Covid-tracker

Tehnička dokumentacija

Verzija 1.0

Studentski tim: Adam Ban

Marko Filipović

Marino Rabuzin

Hrvoje Rom

Nastavnik: doc. dr. sc. Ante Đerek

**Sadržaj**

[1. Problematika i ciljevi projekta 3](#_Toc61221507)

[2. Načelo rada aplikacija za praćenje kontakata 4](#_Toc61221508)

[3. Razvoj aplikacije za praćenje kontakata temeljena na Googleovom API-u 5](#_Toc61221509)

[4. Sigurnosna analiza službene hrvatske aplikacije za praćenje kontakata Stop COVID-19 5](#_Toc61221510)

[4.1 Izgradnja i instalacija aplikacije iz izvornog koda 5](#_Toc61221511)

[4.2 Dekompajliran izvršni kod aplikacije i analiza 7](#_Toc61221512)

[4.3 Praćenje internetskog prometa 8](#_Toc61221513)

[4.4 Otpornost aplikacije na MITM napad pomoću Android uređaja 9](#_Toc61221514)

[4.5 Otpornost aplikacije na MITM napad pomoću Apple uređaja 10](#_Toc61221515)

[5. Zaključci i preporuke 20](#_Toc61221516)

[6. Literatura 20](#_Toc61221517)

# Problematika i ciljevi projekta

Prošlo je godinu dana od prve zabilježene zaraze korona virusom[[1]](#endnote-1), u prosincu 2019. u Wuhanu (Kina). Virus se brzo proširio svijetom i uveo nas u dugotrajno razdoblje svjetske pandemije i borbe sa novom i slabo poznatom bolešću. Dok su vlade i svjetski vođe vodili borbu s virusom informiranjem javnosti, provođenjem mjera poput fizičkog distanciranja, nošenjem maski, uvođenjem lockdowna i slično, tehnološka zajednica te sigurnosni stručnjaci su radili na tehnološki podržanom praćenju kontakata i pravovremenom obavještavanju korisnika da su bili u kontaktu s virusom. Nekoliko znamenitih primjera su Google/Apple Exposure Notification[[2]](#endnote-2), SwissCovid App[[3]](#endnote-3), te DP-3T[[4]](#endnote-4) protokol za praćenje kontakata. Uspješnost korištenih rješenja je upitna s obzirom na ograničene mjere kojima su vlade promovirale i poticale korištenja navedenih rješenja, što pokazuje činjenica da je android verziju hrvatske aplikacije preuzelo oko 50 tisuća korisnika. Jedan od mogućih razloga slabe korištenosti aplikacije je nepovjerenje korisnika u rad aplikacije te posljedično očuvanje privatnosti i tajnosti kontakata. Cilj našeg projekta bio je implementirati gotovo rješenje za praćenje kontakata u vlastitu aplikaciju uz testiranje funkcionalnosti te analizirati i usporediti sigurnosna svojstva prezentiranih rješenja i stvarnih implementacija na korisničkim uređajima.

# Načelo rada aplikacija za praćenje kontakata

Trenutno korištena rješenja[[5]](#endnote-5) temelje se na razmjeni podataka između korisnika aplikacije putem bluetooth odašiljača. Korisnik nakon instalacije aplikacije mora odobriti korištenje bluetootha za odašiljanje vlastitih nizova podataka. Te nizove podataka, nazovimo ih "ključevima", osluškuju drugi uređaji i pohranjuju u vlastitu memoriju uređaja. Aplikacija periodički povlači ključeve zaraženih korisnika sa centralnog poslužitelja i uspoređuje ih sa ključevima u vlastitoj memoriji. Kada pronađe poklapanje obavještava korisnika ili koristi to poklapanje za izračunavanje rizika za korisnika. Korisnik tad zna da je bio u kontaktu sa zaraženom osobom i može se samoinicijativno izolirati i tako spriječiti daljnje širenje virusa. Korisnici koji su zaraženi virusom mogu svoje ključeve prenijeti na udaljeni poslužitelj koji će ih onda podijeliti ostalim korisnicima. Tajnost kontakata očuvana je zbog činjenice da poslužitelj ne zna kome je podijelio ključeve te kojem je korisniku aplikacija dojavila da je bio u kontaktu sa "zaraženim" ključevima. Postoje varijacije na sam postupak jer je različitim algoritmima moguće dodatno zaštititi privatnost korisnika i smanjiti količinu podataka koja se izmjenjuje između korisnika i poslužitelja, npr. razmjenom "seedova" za generiranje ključeva umjesto razmjene svih generiranih ključeva. Dežurna zdravstvena organizacija mora zaraženom korisniku uz pozitivan test generirati verifikacijski ključ koji koristi prilikom prijenosa zaraženih ključeva na poslužitelj. Poslužitelj ne smije moći preko verifikacijskog koda identificirati osobu koja prenosi ključeve, već bi tu informaciju smjela znati samo dežurna zdravstvena organizacija. Trenutna rješenja se oslanjaju na dobru volju i savjest korisnika jer rade ispravno samo pod pretpostavkom da korisnici drže aplikaciju aktivnom na uređaju te da će podijeliti sa drugim korisnicima da su bili pozitivni jednom kad im test to potvrdi.

# Razvoj aplikacije za praćenje kontakata temeljena na Googleovom API-u

Nakon uvodnog upoznavanja sa problematikom zadatka i informiranja o postojećim rješenjima, započeli smo s pokušajem implementacije u vlastitu aplikaciju. Nakon izgradnje postojeće aplikacije iz izvornog koda i instalacije na uređaj došli smo do zaključka da aplikacija ne radi. Dodatno smo istražili i zaključili da je Googleov Exposure API otvoren i dostupan za razvoj samo "allowedlisted" računima tj. javnim zdravstvenim ustanovama koje su odlučile surađivati sa Googleom. Isprobana su druga rješenja no nismo ih uspjeli implementirati. S obzirom da je vlastito implementiranje praćenje kontakata vrlo zahtjevan problem odlučeno je da ćemo odustati od tog dijela projekta i posvetiti se analizi službene hrvatske aplikacije.

# Sigurnosna analiza službene hrvatske aplikacije za praćenje kontakata Stop COVID-19

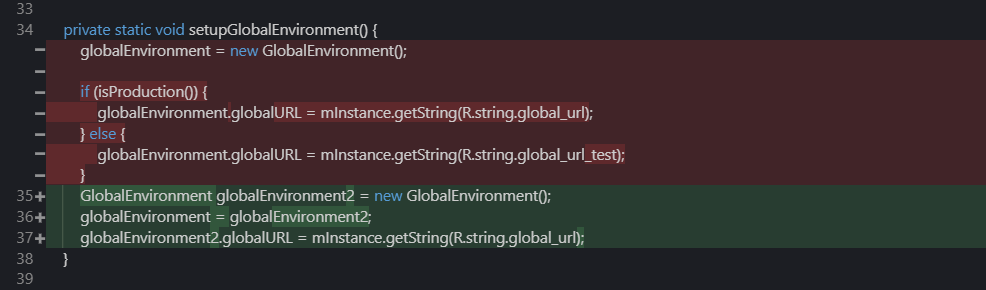
## Izgradnja i instalacija aplikacije iz izvornog koda

Izvorni kod aplikacije dostupan je udaljenom repozitoriju GitHub-a[[6]](#endnote-6). Svaki korisnik može besplatnim alatima sam izgraditi izvršni kod aplikacije i instalirati ju na uređaj. Korištenjem Android Studia[[7]](#endnote-7) i podešavanjem gradle postavki, lako je izgraditi aplikaciju i instalirati ju na uređaj. Aplikacija izgrađena iz izvornog koda izgleda i ponaša se identično kao i aplikacija instalirana sa Google Play Store-a[[8]](#endnote-8), no ukoliko pokušamo pokrenuti praćenje kontakata ili pokušamo unijeti verifikacijski kod, aplikacija će nas obavijestiti da uređaj nije kompatibilan s Exposure Api-jem (Googleov Exposure Notification Api)(Slike 1.1 i 1.2). To se događa jer, kao što smo ranije naveli, Exsposure Api je dostupan samo Googleovim partnerima u javnom zdravstvu[[9]](#endnote-9). Takav oblik zaštite pristupa api-ju vjerojatno je uveden kako bi se spriječila zlouporaba otvorenosti koda i umanjile mogućnosti instalacije krivotvorenih verzija službenih aplikacija koje bi ugrožavale zdravlje, sigurnost i privatnost korisnika. Na slikama 2.1 i 2.2 vidimo očekivano ponašanje aplikacije.

|  |  |  |  |
| --- | --- | --- | --- |
| Slika 1.1 Prikazuje početna stranica aplikacije kada je isključen API | | Slika 1.2 Prikazuje sučelje na kojem se šalju ključevi | |
| Slika 2.1 Prikazuje početnu stranicu aplikaciju kada je uključen API | Slika 2.2 Prikazuje poruku ako se upiše kirvi kod | |

## Dekompajliran izvršni kod aplikacije i analiza

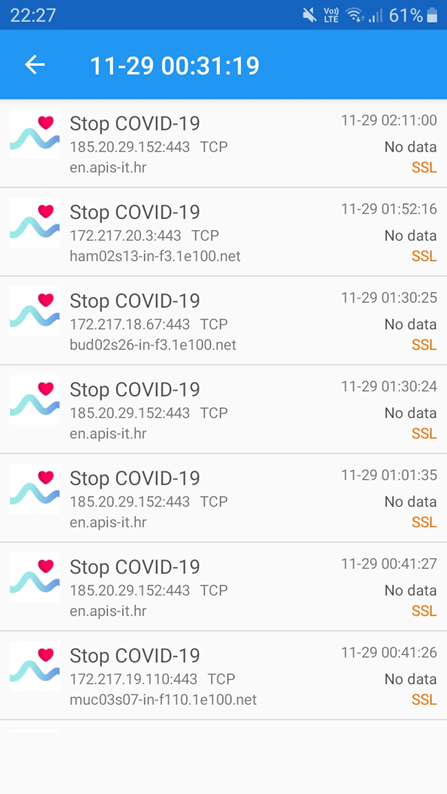
Otvorenost koda se često spominje u kontekstu sigurnosti programske podrške i u pravilu indicira da programska potpora radi ono za što se reklamira, bez tajnih motiva i skrivenih procesa, prateći pravila sigurnosti po dizajnu. Kako bi se uvjerili da aplikacija radi ono za što se reklamira, potrebno je prevesti izvršni kod u izvorni i usporediti ga sa službenim izvornim kodom dostupnim na GitHub-u. Izvršni kod Android aplikacija pakira se u apk datotečni format koji se zatim može lako prenositi i instalirati na android uređaje. Mi smo preuzeli apk datoteku, instalirali je na android uređaj kako bi provjerili je li to ta aplikacija te radi li. Nakon što smo se uvjerili da je apk datoteka valjana, korištenjem programa jadx[[10]](#endnote-10) i njegovog grafičkog sučelja dekompajlirali smo ju i dobili dekompajliran izvorni kod aplikacije. Usporedbom s kodom s GitHub-a, došli smo do zaključka da aplikacija ne radi ništa što nebi trebala, struktura projekta i klase su identične i ulazno/izlazna točka aplikacije pokazuje na isti centralni poslužitelj kao i referentni kod. Programski kod je malo drugačiji, no te razlike ostvaruju istu funkcionalnost kao što pokazuje primjer sa slike 3.1 gdje je crveno označen referentni izvorni kod, a zeleno dekompajliran izvorni kod. Vidimo da razlika vjerojatno proizlazi iz načina na koji je kod optimiran prilikom prevođenja.

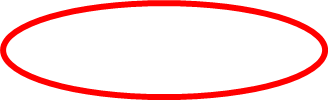
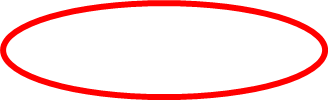


Slika 3.1 Prikazuje razliku dekompajliranih kodova

## Praćenje internetskog prometa

Prema navodima, aplikacija bi trebala komunicirati samo s centralnim poslužiteljem. U slučaju hrvatske aplikacije za praćenje kontakata to je poslužitelj tvrtke APIS-IT[[11]](#endnote-11) s IP-adresom 185.20.29.152. APIS-IT je tvrtka zadužena za implementaciju i održavanje hrvatske verzije aplikacije za praćenje kontakata. U izvornom kodu sa GitHub-a i dekompajliranom izvornom kodu definirana je ista adresa centralnog poslužitelja. No željeli smo se uvjeriti da aplikacija komunicira samo s tim poslužiteljem. Prvo smo snimali internetski promet koristeći aplikaciju Packet Capture[[12]](#endnote-12), ona nam omogućuje da zabilježimo s kojim poslužiteljima komunicira odabrana aplikacija. Zabilježila je komunikaciju s APIS-serverom, ali i s drugim nepoznatim poslužiteljima (Slika 4.1). Za ostale poslužitelje se ispostavilo da pripadaju Google-u i da se vjerojatno koriste u sklopu Google Play Service-ova koje je bilo neophodno prihvatiti za korištenje aplikacije za praćenje kontakata.





Slika 4.1 Prikazuje pračenje prometa aplikacije

Željeli smo istražiti sadržaj internetskog prometa te smo, koristeći alat Wireshark[[13]](#endnote-13) i osobno računalo kao Wi-FI Hotspot, snimali internetski promet aplikacije. Prema specifikaciji, očekivani promet bi trebao sadržavati: periodično preuzimanje ključeva s poslužitelja, lažno slanje vlastitih ključeva, te pravo slanje vlastitih ključeva (posljednje nije moguće provjeriti bez valjanog verifikacijskog koda).[[14]](#endnote-14) Aplikacija inicira komunikaciju s poslužiteljem svakih 20-ak minuta, sadržaj paketa je zaštićen na transportnom sloju te nije moguće iz sadržaja paketa odrediti svrhu komunikacije, ali vidljive su različite duljine sadržaja paketa pa iz toga zaključujemo da je zabilježen promet kakav smo i očekivali.

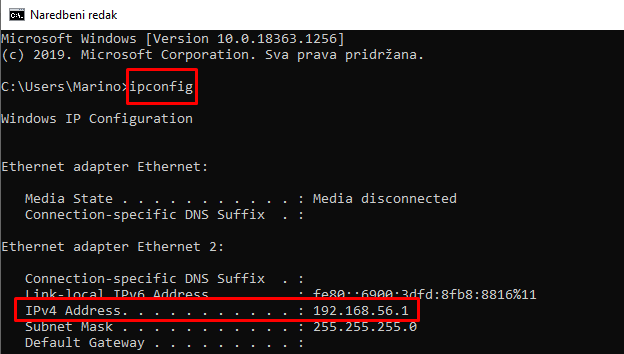
## Otpornost aplikacije na MITM napad pomoću Android uređaja

S obzirom da je promet zaštićen na transportnom sloju, pokušali smo doći do sadržaja paketa koristeći Man In The Middle[[15]](#endnote-15) napad. Ne ulazeći u detalje same zaštite, čitatelju je potrebno znati da u MITM napadu, MITM software gledajući sa strane poslužitelj glumi korisnika, a sa strane korisnika glumi poslužitelj i, koristeći vlastite certifikate, pokušava zavarati korisnika da ustvari komunicira sa pravim poslužiteljem i zavarava poslužitelj da komunicira direktno s korisnikom te bilježi sadržaj paketa kako bi se mogao analizirati.

Koristeći alate Fiddler[[16]](#endnote-16)ili mitmproxy[[17]](#endnote-17) postavljamo proxy poslužitelj na našem osobnom računalu, zatim je na mobilnom uređaju potrebno postaviti računalo kao proxy poslužitelj. Mobilni uređaj "ne vjeruje" proxy-u i internetski promet ne putuje preko njega. Potrebno je na mobilni uređaj instalirati certifikat odabranog alata. Na Android mobilnim uređajima, nakon dodavanja certifikata, moguće je pokrenuti internet preglednik i obavljati pretraživanje na internetu te je sav taj promet vidljiv u proxy software-u, no ostale aplikacije ne rade i ne snima se njihov promet. Razlog tome je što od Andriod Nougat-a nadalje nijedna aplikacije ne vjeruje korisničkim certifikatima. Rješenje tog problema je da modificiramo sam kod aplikacije, odnosno u manifest dodamo da vjeruje korisničkim certifikatima, no nakon izgradnje i instalacije na uređaj pri pokušaju korištenja aplikacija će nas obavijestiti s porukom da uređaj nije kompatabilan s Exposure API-jem te ponovo nije moguće zabilježiti promet te aplikacije. Iz svega zaključujemo da je aplikacija sigurna od MITM napada.

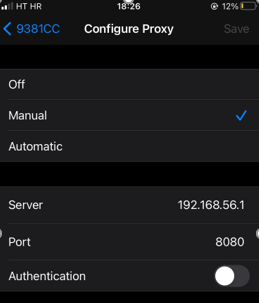
## Otpornost aplikacije na MITM napad pomoću Apple uređaja

Potrebne stvari za ispitivanje otpornosti aplikacije **Stop COVID – 19** pomoću MITM napad za početak su spajanje samog Apple uređaja na računalo preko pristupne točke. Tako će se dobiti efekt čovjeka u sredini kojeg će u ovom slučaju glumiti odabrano računalo na kojeg se Apple uređaj spojio. Kako bio spojili Apple uređaj potrebno je pronaći IP adresu samog računala tj. izvršavanjem **ipconfig** naredbe u naredbenom retku računala:



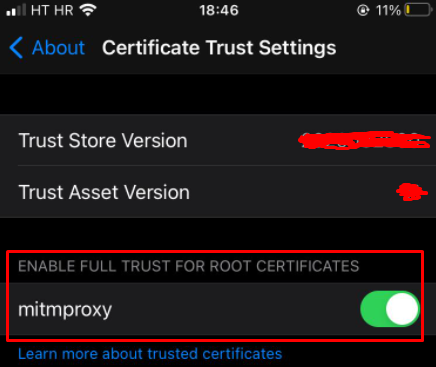
Slika 4.2 Slika prikazuje rezultata izvršavanja naredbe ipconfig

Nakon što je IP adresa računala poznata moramo konfigurirati proxy Apple uređaj. Konfiguracija proxya na Apple uređajima radi se u postavkama samog uređaja. Pritiskom na **Settings** 🡪 **Wi-fi** 🡪 pritisak na trenutnu spojenu vezu (pristpuna točka računala) 🡪 Configure Proxy prikaz je na slici 4.3. Kada se otvorilo sučelje sa slike 4.3 tada se popunjavaju podaci tj. **Server** što predstavlja prema kojem uređaju ćemo kreirati proxy, a to je IP adresa računala na kojeg je Apple uređaj spojen te **Port** na kojem će se osluškivati promet.



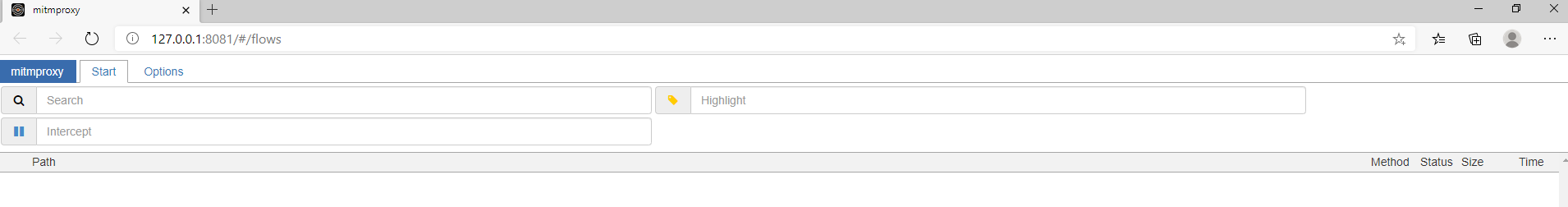
Slika 4.3 Prikazuje sučelje koje se otvara pritiskom na Configure proxy

Nakon što je na Apple uređaju konfiguriran proxy prema računalu na kojem je spojen, potrebno je instalirati certifikat za taj proxy kako bi mu aplikacije vjerovale. Na Apple uređaju u **Safari** upisuje se: ***mitm.it***te se instalirava certifikat za određeni sustav, konkretno u ovom primjeru preuzima se Apple certifikat (profil). Nakon što je certifikat preuzet potrebno ga je odobriti. Odobrenje preuzetog certifikata obavlja se tako da se na uređaju uđe u **Settings** te odabere **Profile Downloaded** tamo će biti prikazan mitmproxy certifikat koji je potrebno instalirati. Nakon što je certifikat instaliran, Apple uređaj počinje vjerovati tom certifikatu (djelomično) što je suprotno od Androida koji ne vjeruje korisničkim certifikatima. Kako bi Apple uređaj u potpunosti vjerovao certifikatu potrebno ga je dodati u provjerene certifikate. Odlaskom u **Settings** 🡪 **General 🡪 About 🡪 Cerificate Trust Settings** prikazat će se sučelje u kojem je potrebno pridružiti "Potpuno povjerenje" mitmproxy certifikatu.



Slika 4.4 Prikazuje sučjelje u kojem se dodjeljue potpuno povjerenje korisničkim certifikatima

Kada je podešeno "Potpuno povjerenje" nad mitmproxy certifikatom, može se početi hvatati promet na računalu preko mitmproxy sučelja koje mora biti instalirano na računalu. Ako mitmproxy nije instaliran na računalu tada je potrebno posjetiti mitmproxy web stranicu te preuzeti aplikaciju tj. sučelje, ako je to obavljeno potrebno je potražiti **mitmproxy ui** koji će otvoriti sučelje u kojem ćemo pratiti promet:

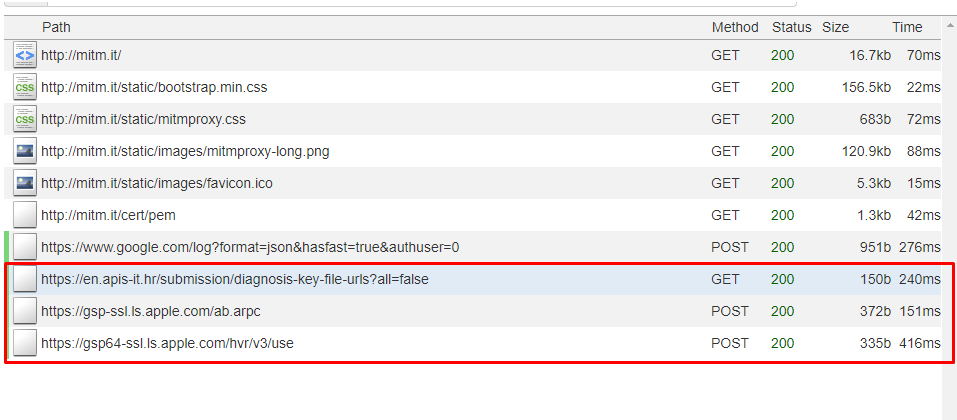


Slika 4.5 Prikazuje mitmproxy sučelje u kojem se prikazuje uhvaćeni promet

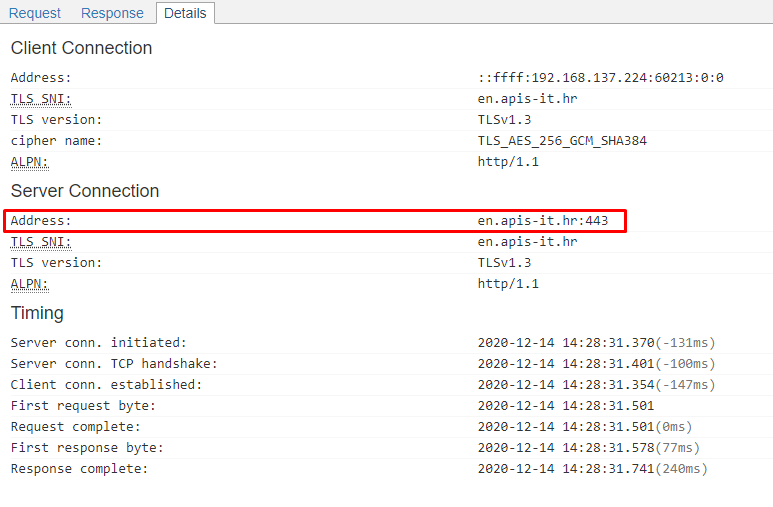
Potrebno je naglasiti u ovom modu rada u kojem je trenutno Apple uređaj nije moguće pristputi nekim aplikacijama kao što su npr. **Whatsapp**, **Snapchat**  koji imaju sustav čim otkriju da se na uređaju nalazi korisnički certifikat s potpunim povjerenjem tada se nabrojane aplikacije neće povezati sa svojim poslužiteljima te nema komunikacije klijenta i poslužitelja, ono što nam je ostalo učitano, prije nego li smo se spojili na vezu koja glumi čovjeka u sredini, će nam se prikazati, ali nikakve promjene u aplikaciji neće biti moguće (osvježavanje aplikacije) sve dok se ne spojimo na drugu mrežu koja ne koristi proxy. Samim time te aplikacije su zaštičene od napada oblika čovjek u sredini jer ne postoji komunikacija između klijenta i servera pa samim time ne postoji paket kojeg se može preusmjeriti i pregledati njegov sadržaj.

Aplikacija **Stop COVID – 19** nema takav oblik zaštite kao gore navedene aplikacije, stoga je moguće preusmjeriti, tj. uhvatiti promet između servera i klijenta. Prilikom ulaska u samu aplikaciju potrebno je omogućiti komunikaciju API-a i Apple uređaja tako da se odobri spajanje.

Prilikom tog odobrenja zabilježen je promet od 3 zahtjeva (1 GET zahtjev i 2 POST zahtjeva), podatci unutar tih zahtjeva su dostupni, ali su kriptirani. Svaki od zahtjeva koji je preuzet ima svoj: **Details**, **Request** i **Response**. Trenutno u **Detailsu** jedini korisni podatci su oni po kojima vidimo da aplikacija komunicira s APSI-ITem i Appleom.

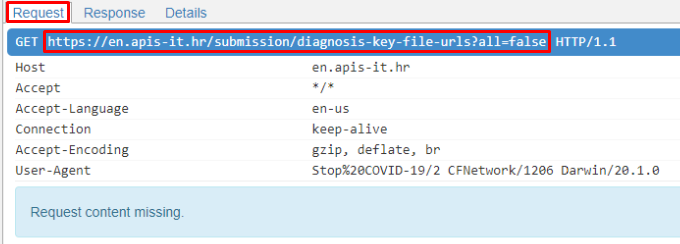


Slika 4.6 Prikazuje 3 zahtjeva koja su uhvaćena prilikom spajanja Apple uređaja i APIja



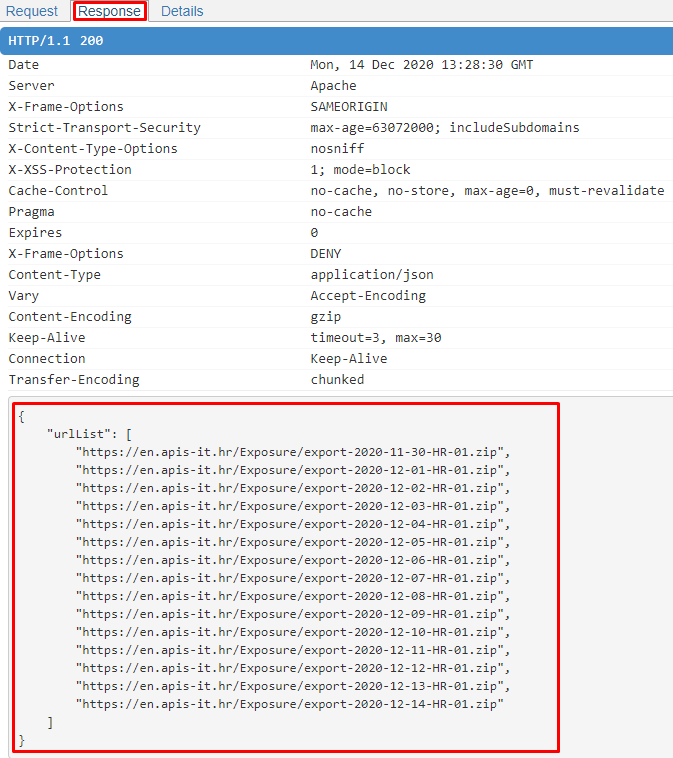
Slika 4.6 Prikazuje Details od prvog zahtjeva (GET zahtjev)

Unutar **Request** taba od prvog GET zahtjeva postoji URL na kojem se traže određeni podatci koji se dobivaju u tabu **Response,** trenutno informacije koje su dobivene nisu previše korisne, ukoliko je potrebna neka autentifikacija u ovom tabu, taj bi se token za autentifikaciju ispisao te bi se s tim tokenom moglo dalje manipulirati, ali aplikacija ima zaštitu od toga jer ne koristi nikakve tokene niti zaštitu tog tipa.



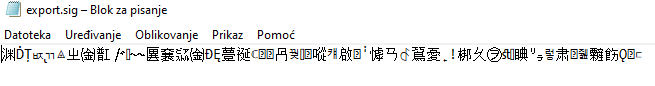
Slika 4.7 Prikazuje Request od prvog zahtjeva (GET zahtjev)

Unutar **Response** taba dobiven je rezultat GET zahtjeva te je uspješno zabilježen promet u JSON formatu koji prikazuje URL-ove zip datoteka koje se nalaze na web poslužiteljima tvrtke APIS-IT, moguće ih je preuzeti tako da se URL upiše u web preglednik te će se automatski skinuti određena zip datoteka.



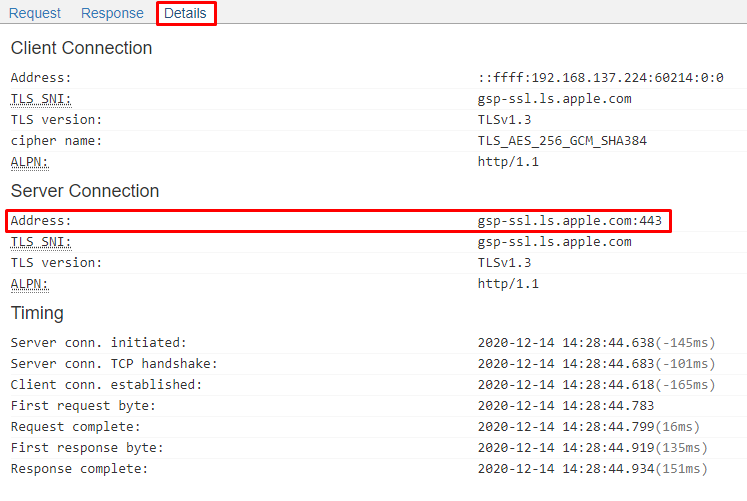
Slika 4.8 Prikazuje Response od prvog zahtjeva (GET zahtjeva)

Unutar zip datoteka nalazi datoteka **export.sig** kojoj je sadržaj kriptiran **SHA–256** algoritmom. Sadržaj te datoteke izgleda ovako:



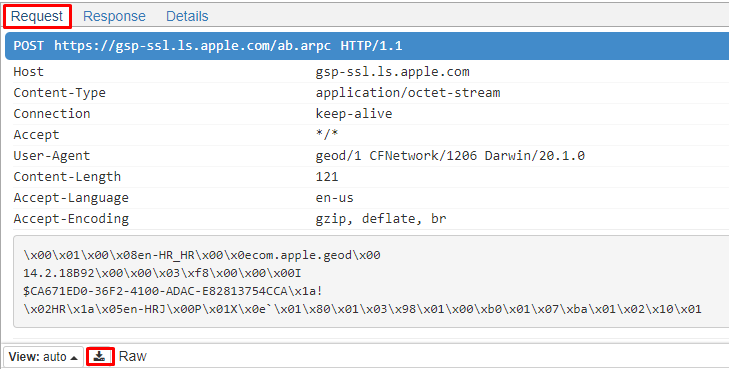
Slika 4.9 Prikazuje sadržaj datoteke export.sig

Kad je prvi zahtjev obrađen prelazimo na drugi zahtjev koji je POST zahtjev, drugi zahtjev i treći zahtjev dijele isti **Details.**

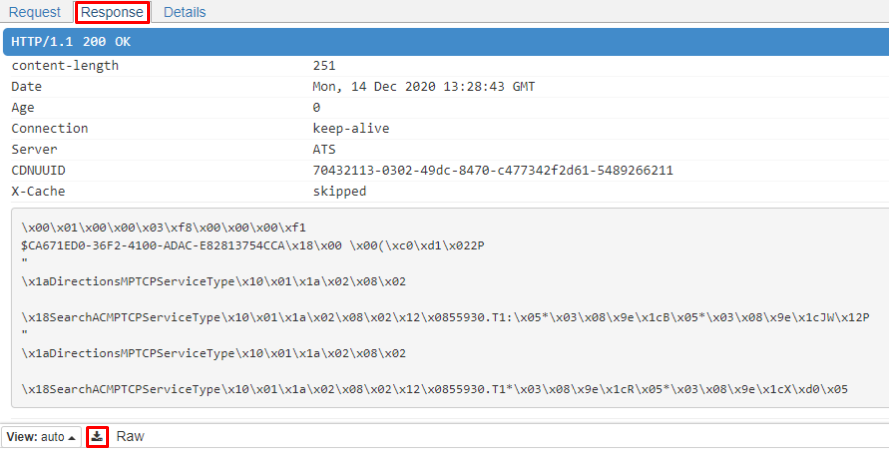


Slika 4.10 Prikazuje Details koji dijele 2. i 3. zahtjev (2 POST zahtjeva)

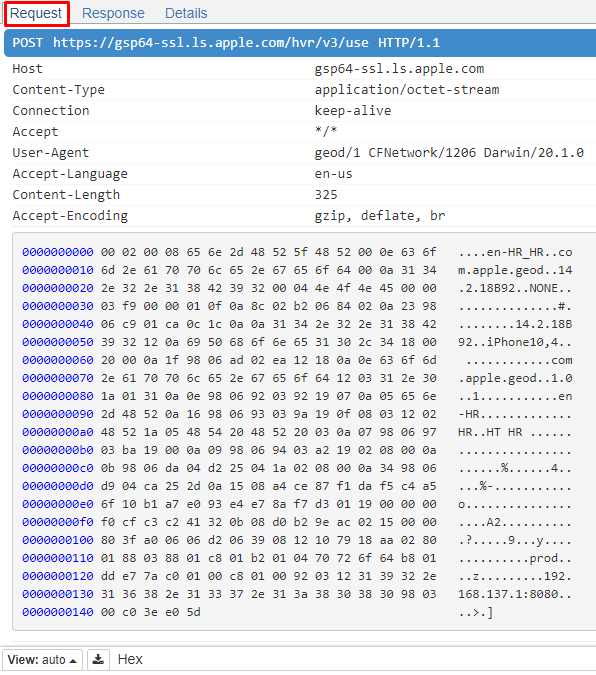
Mitmproxy takoder nudi jednu mogućnost a to je da se automatski skine cijeli Body koji se šalje u POST paketu te se pospremi u datoteku s predodređenim imenom, te nudi mogućnost skidanja cijelog Body-a od odgovora na POST zahtjev:



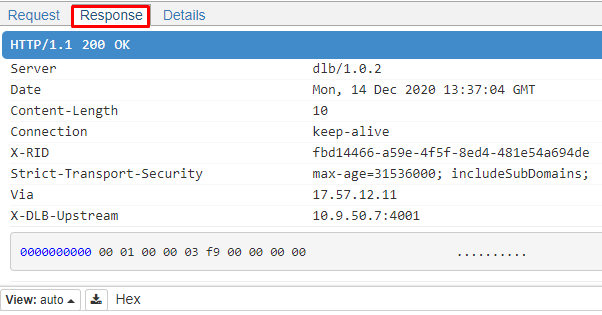
Slika 4.11 Prikazuje Requst od 2. zahtjeva (POST zahtjeva)



Slika 4.12 Prikazuje Response od 2. zahtjeva (POST zahtjev)



Slika 4.13 Prikazuje Requst od 3. zahtjeva (POST zahtjev)



Slika 4.14 Prikazuje Response od 3. zahtjeva (POST zahtjev)

Prvi GET zahtjev ima sljedeći URL: [*https://en.apis-it.hr/submission/diagnosis-key-file-urls?all=false*](https://en.apis-it.hr/submission/diagnosis-key-file-urls?all=false) te ukoliko pokušamo izbrisati sve argumente tj. maknemo u URLu ***?all=false*** tada dobivamo isti rezultat kao i da postoji dani argument stoga nije implementirano dobro usmjerenje jer se za drugi URL dobiva ista stvar. Ne samo to, nego ukoliko dodamo još bilo kakav argument npr. ***?all=false&first=true***također ćemo dobiti isti rezultat kao i da smo pozvali početni URL, tako da implementacijski gleda se samo jedan argument, postoji li taj argument (all=false) ili ih ima više uvijek se dobiva ista stvar, a to su exposure key-ovi za hrvatsku. Ukoliko je argument ***?all=true*** tada se dobiva zip datoteke koje predstavljaju exposure key-eve za cijelu Europu, no taj podataka nije trebao biti dostupan tako da smo uspijeli uhvatiti promet koji nismo smijeli. Tako smo manipulacijom GET Zahtjeva uspijeli dobiti resurse koji su nama trebali biti skriveni.

Kako bi smo lakše nalazili što se nalazi na URL-ovima od GET zahtjeva napisan jer jedan jednostavan JavaScript server koji šalje GET zahtjeve na određeni URL. Na slici 4.15 prikazan je izvorni kod servera te na kojim portovima se može naći rezultat GET zahtjeva. Konkretno u ovom slučaju to su **port** **3000** i **port 3001**:



Slika 4.15 Prikazuje kod servera koji se koristi za slanje GET zahtjeva

# Zaključci i preporuke

Reverznim inženjerstvom pokušali smo saznati koje sve podatke aplikacija šalje na poslužitelje. Možemo zaključiti da aplikacija komunicira s APIS serverima koji su zaduženi za normalan rad aplikacije jer bi se na njih trebali slati ključevi koje aplikacija razmjenjuje s drugim mobilnim uređajima preko bluetootha. Sasvim neočekivano, otkrili smo da aplikacija komunicira i s Googleovim serverima, a komunikacija s dodatnim serverima nas dodatno zabrinjava zbog očuvanja naše privatnosti. Analizom paketa koje aplikacija šalje na poslužitelje nismo uspjeli saznati koji se točno podaci razmjenju zato što je sva komunikacija kriptirana. Iako u nijednom koraku analize aplikacije nismo uspjeli pronaći nešto što bi dalo slutiti da aplikacija šalje neželjene podatke na udaljene poslužitelje, upravo radi kriptirane komunikacije ne možemo sa stopostotnom sigurnošću reći da je aplikacija sigurna za korištenje i da čuva našu privatnost.

Preporuka za što bolju zaštitu aplikacije je da ne vjeruje korisničkim certifikatima kojima mi možemo dati potpuno povjerenje u postavkama, tj. da se radi na principu WhatsApp i Snapchat, ako je aktiviran nepoznati korisnički certifikat, spajanje na server nije moguće. Korištenje GET zahtjeva smanjiti na minimum jer je vrlo lako pogoditi argumente te doći do određenih informacija koje ne bi trebale biti dostupne, ako se već i koriste GET zahtjevi omogućiti pristup serveru samo određenim osobama, a ostale preusmjeriti na status 403 FORBIDDEN. Korištenje POST zahtjeva je poželjnije jer je potrebno u JSON-u koji se šalje postaviti ispravne parametre kako bismo izvukli informacije. Obrana od ovakvog tipa napada se izvršava provjerom tko šalje paket ako je ne očekivani pošiljatelj POST zahtjeva tada bi se zahtjev trebao odbiti.

# Literatura

1. <https://en.wikipedia.org/wiki/COVID-19_pandemic>, 23.12.2020. [↑](#endnote-ref-1)
2. <https://www.google.com/covid19/exposurenotifications/>, 23.12.2020. [↑](#endnote-ref-2)
3. <https://www.bag.admin.ch/bag/en/home/krankheiten/ausbrueche-epidemien-pandemien/aktuelle-ausbrueche-epidemien/novel-cov/swisscovid-app-und-contact-tracing.html>, 23.12.2020. [↑](#endnote-ref-3)
4. <https://github.com/DP-3T/documents>, 23.12.2020. [↑](#endnote-ref-4)
5. <https://ncase.me/contact-tracing/>, 23.12.2020. [↑](#endnote-ref-5)
6. <https://github.com/Stop-COVID-19-Croatia>, 23.12.2020. [↑](#endnote-ref-6)
7. <https://developer.android.com/studio>, 23.12.2020. [↑](#endnote-ref-7)
8. <https://play.google.com/store/apps/details?id=hr.miz.evidencijakontakata>, 23.12.2020. [↑](#endnote-ref-8)
9. <https://developers.google.com/android/exposure-notifications/exposure-notifications-api#glossary>, 23.12.2020. [↑](#endnote-ref-9)
10. <https://github.com/skylot/jadx>, 23.12.2020. [↑](#endnote-ref-10)
11. <https://www.apis-it.hr/apisit/index.html#/>, 23.12.2020. [↑](#endnote-ref-11)
12. <https://play.google.com/store/apps/details?id=app.greyshirts.sslcapture>, 23.12.2020. [↑](#endnote-ref-12)
13. <https://www.wireshark.org/>, 23.12.2020. [↑](#endnote-ref-13)
14. <https://developers.google.com/android/exposure-notifications/verification-system#codes-tokens-certs> [↑](#endnote-ref-14)
15. <https://en.wikipedia.org/wiki/Man-in-the-middle_attack>, 23.12.2020. [↑](#endnote-ref-15)
16. <https://www.telerik.com/fiddler>, 23.12.2020. [↑](#endnote-ref-16)
17. <https://mitmproxy.org/>, 23.12.2020. [↑](#endnote-ref-17)