Sprječavanje prevara u sustavima mobilne naplate

Brođanac, Damir

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

SPRJEČAVANJE PREVARA U SUSTAVIMA MOBILNE NAPLATE

Diplomski rad

Damir Brođanac

Osijek, 2021.

SADRŽAJ

1. UVOD

Iako je prvi mobilni telefon nastao 1973. godine, tek tokom devedesetih godina prošlog stoljeća su mobilni uređaji počeli nalikovati na nešto što bi ljudi mogli nositi u džepu i svakodnevno koristiti. Mobilni operateri, uvijek u potrazi za novim izvorom prihoda, su uvidjeli da bi osim naplaćivanja SMS, MMS poruka i poziva, mogli naplaćivati i dodatne sadržaje koji su postali dostupni razvojem mobilnih uređaja poput melodija zvona ili pozadinskih slika. Danas korisnici širom svijeta na ovakvu vrstu sadržaja potroše 54,5 milijardi dolara na godišnjoj razini. [1] S obzirom da se radi o velikoj industriji u kojoj se vrti sve veća količina novca, bilo je samo pitanje vremena kada će se u njoj pojaviti kriminalne skupine koje će na neki ilegalan način pokušati doći do navedenog novca. U zadnjih dvadesetak godina zabilježeno je nebrojeno puno raznih pokušaja prevare metodama kao što su obmanjujuće reklame, maliciozne aplikacije, takozvana krađa klikova i slično. Prema istraživanju kompanije Evina koja se bavi zaštitom od prevara u svijetu mobilnog plaćanja, u 2021. godini, 19,8% svih transakcija plaćenih mobilnim putem bilo je neovlašteno.[2] Mobilni operateri postupno uvode razne zaštite u svoje naplatne sustave koji štite korisnike od raznih vrsta prevara poput krađe klikova ili skrivenih okvira unutar naplatnih web stranica, ali često su i operateri bespomoćni jer se prevara događa izravno na mobilnom uređaju kojim upravlja korisnik. Stoga je važno biti oprezan i svjestan svih mogućnosti i opasnosti koje dolaze uz korištenje mobilnih uređaja za plaćanje kao što se s oprezom pristupa plaćanju kreditnom karticom. Cilj ovog diplomskog rada je napraviti aplikaciju koja bi pružila korisnicima jednu vrstu zaštite od prevara u sustavima mobilne naplate, konkretno zaštitu od malicioznih aplikacija. Drugo poglavlje sadrži kratak opis povijesti sustava mobilne naplate te opisuje dvije glavne metode naplate pomoću mobilnih uređaja. Treće poglavlje sadrži pregled nekoliko metoda prevara koje su trenutno najprisutnije na globalnoj razini. Četvrto poglavlje sadrži opisane tehnologije korištene u izradi aplikacije i detaljan opis uz primjere koda. U posljednjem poglavlju je prikazan je rad same aplikacije.

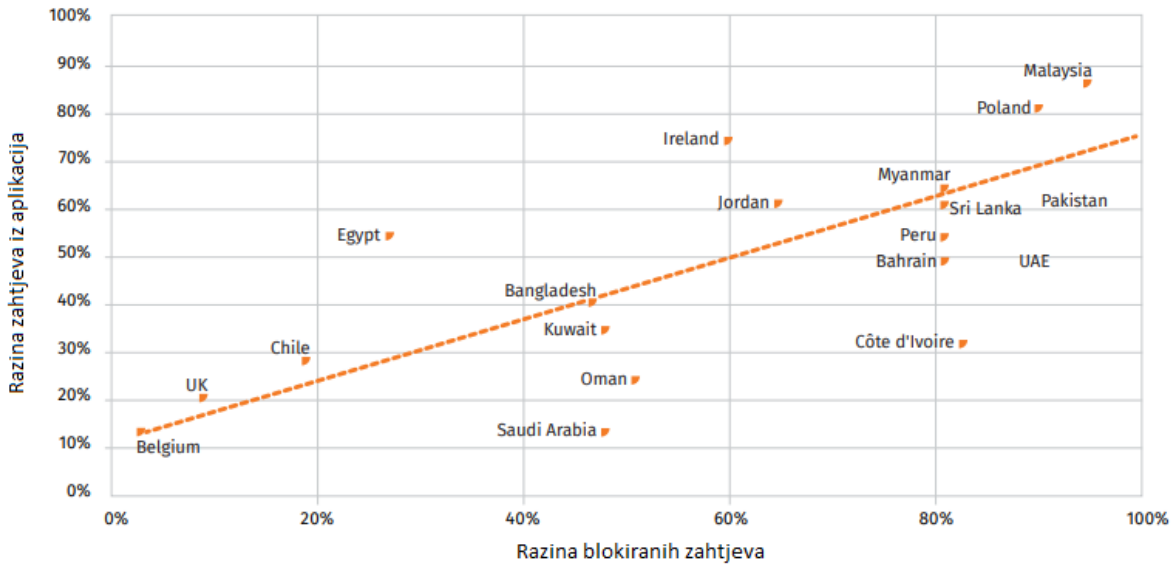
2. PREGLED PODRUČJA TEME – VRSTE I UTJECAJ PREVARA NA SUSTAVE MOBILNE NAPLATE

U ovom poglavlju opisani su godišnji izvještaji tri najveće kompanije koje se bore protiv prevara u sustavima mobilnih naplata – Evina, Empello i Opticks Security. Uz prikaz najčešćih metoda prevara prikazani su i korišteni načini zaštite te njihova učinkovitost.

2.1. Stanje tržišta mobilne naplate

Prema izvještaju kompanije Evina za prva dva kvartala 2021. godine, 19,8% svih transakcija mobilnog plaćanja na području Europe nastalo je kao posljedica neke vrste prevare. Ovaj postotak varira od zemlje do zemlje, tako je npr. Austrija u istom promatranom periodu bilježila 32.3% problematičnih transakcija, Grčka 26,4%, Portugal 20,4%, a Švicarska 13,2%. Također se navodi da je najčešći oblik obmane platitelja putem malicioznih aplikacija (37,7%), dok su odmah iza njega krađa klikova sa 26,3% i takozvana Bypass metoda (instaliranje aplikacija na mobilni uređaj bez prave privole korisnika) sa 22,8%. Najpopularnija metoda prevare je logično pomoću aplikacija jer one omogućuju napadaču da vrlo jednostavno preuzme potrebnu razinu kontrole nad uređajem na koji je instalirana. Naizgled potpuno bezazlena aplikacija sa službene Googleove trgovine može u sebi sadržavati dio koda koji će nakon instaliranja omogućiti aplikaciji slanje ili čitanje poruka, otvaranje internet preglednika u pozadini ili uspostavljanje poziva, sve to može dovesti do velikih troškova. Aplikacije koje su uzrokovale najviše problema u prvoj polovici 2021. godine su Fast Cleaner, preko 500,000 puta instaliran kroz službeni Google Play Store, uz ocjenu 4,5, Blood Pressure Tracker, instaliran 10,000 puta uz prosječnu ocjenu 3,5 te Blood Pressure Recorder, instaliran 50,000 puta uz ocjenu 3,5. Iako bi rijetko tko pomislio da će aplikacija koja je preuzeta pola milijuna puta, uz visoku ocjenu zadovoljstva njome od čak 4,5 stvarati ikakve probleme, ovo istraživanje je pokazalo da opreza nikad dovoljno. Evina koristi svoj produkt DCBprotect koji detaljno analizira sesije korisnika te na temelju ponašanja korisnika, uređaja i preglednika koji se koristi otkriva radi li se o prevari. Njihov sustav može otkriti prevare nastale putem malicioznih aplikacija, krađe klikova, ubacivanja koda, daljinski kontroliranih aplikacija i Bypass metode. [7]

Prema godišnjem izvještaju kompanije Empello za 2020. godinu, 70% transakcija koje su prošle kroz njihov sustav zaštite FraudStop su bile blokirane zbog sumnje na prevaru. Izvještaj je dokazao da razina visokorizičnih zahtjeva za plaćanjem korelira sa razinom zahtjeva za plaćanjem koji potiču iz aplikacija, kao što je prikazano na slici 2.1..



Sl. 2.1. Odnos zahtjeva iz aplikacija i blokiranih zahtjeva

Operateri širom svijeta pokušali su spriječiti ovakav način prevare implementacijom kompliciranijeg toka naplate, tj. ubacivanjem dodatnog koraka u kojem se mora unijeti PIN kod koji se korisniku šalje putem SMS poruke, međutim maliciozne aplikacije su u stanju pročitati kod čim je poruka primljena te ga upisati umjesto korisnika u za to predviđeni prostor. Empello nudi rješenje u vidu svog FraudStop produkta koji se ugrađuje unutar stranice za plaćanje te blokira zahtjeve iz potencijalno opasnih aplikacija, internet stranica na kojima je detektirana prisutnost i-okvira (engl. *iframe*) sumnjivih tokena ili ako sustav detektira čudnu aktivnost koja ne podsjeća na ljudsku (npr. prebrzi klik na gumb za potvrdu plaćanja). [14]

Opticks je demonstrirao razinu prevara pomoću analize podataka prikupljenih tokom 2020. godine tokom koje je zabilježeno preko 466 milijardi posjeta stranicama u preko 200 država. Za lakše razumijevanje prevare su podijelili u tri kategorije. Prva je „zli“ botovi (engl. *bad bots*) koja se odnosi na bilo koji automatizirani dio programa korišten za maliciozne aktivnosti dok se pritom pretvara da ih izvodi stvarni korisnik. Druga kategorija je ubacivanje malicioznog koda unutar aplikacija ili web stranica. Treća kategorija obuhvaća sve ostale vrste zahtjeva koji su se iz nekog razloga pokazali sumnjivima. Najpogođenije zemlje na europskom tržištu su tako bile Švicarska, Nizozemska, Njemačka, Bjelorusija i Ukrajina. Primijećeno je da se korištene metode prevara razlikuju od tržišta do tržišta što bi se moglo objasniti regulativama oglašavanja koje se definiraju na lokalnoj razini svake države. Tako primjerice u Italiji prevladavaju „zli“ botovi, dok ubačenog malicioznog koda gotovo da i nema. U isto vrijeme u Velikoj Britaniji zabilježeno je znatno više prevara i prvog i drugog tipa. Zanimljivo je da je razina sumnjivih transakcija, koje spadaju u treću kategoriju u ovom izvještaju, izuzetno visoka u svakom promatranom tržištu. Također je dan kratki

osvrt na najčešće metode prevara putem aplikacija. Navedeno je da se maliciozne aplikacije maskiraju izmjenom imena paketa aplikacije, pa se zahtjev iz takve aplikacije prikazuje pod lažnim imenom te ako se ne blokira, može izvršiti naplatu koje korisnik uopće nije svjestan. Drugi najveći problem su zahtjevi koje dolaze iz aplikacija koje se ne nalaze na službenoj Google trgovini, takve se aplikacije automatski smatraju opasnima jer nad njima nisu izvršene nikakve sigurnosne provjere i upitno je s koje stranice ih je korisnik preuzeo. Opticks Security nudi niz rješenja za detekciju krađe klikova, prepoznavanje ranjivosti internet preglednika i otkrivanje zahtjeva iz opasnih aplikacija te je uz Evinu i Empello, jedna od najpoznatijih kompanija koja se bavi sprječavanjem prevara u sustavima mobilne naplate. [12]

2.2. Pregled metoda za sprječavanje prevara pomoću aplikacija i odabir vlastitog rješenja

Pregledom prethodno navedenih istraživanja, jasno je vidljivo da su zloćudne aplikacije u posljednjih nekoliko godina postale izuzetno velika prijetnja za svaku osobu koja koristi pametne telefone. Aplikacije je vrlo jednostavno instalirati putem službenih trgovina aplikacija ili putem trećih stranica, tj. neslužbenih trgovina, na internetu. Dovoljno je svega nekoliko klikova kako bi se instalacijska datoteka preuzela, aplikacija instalirala i pritom dobila razna dopuštenja na koja korisnik mobilnog uređaja vjerojatno nije niti obratio pažnju. Zloćudno ponašanje aplikacije se može manifestirati odmah nakon instalacije, ali i u bilo koje kasnije vrijeme ako napadač ima mogućnost daljinske kontrole, što je vrlo čest slučaj. Trenutne metode zaštite koje su implementirali razni operateri i pružatelji plaćanja mobilnih usluga se temelje na blokiranju zahtjeva koje dolaze iz aplikacija. Ideja je blokirati zahtjev na temelju imena paketa koje svaka aplikacija sadrži i na taj način u startu prekinuti proces naplate. Ime paketa se uspoređuje sa imenima paketa ranije detektiranih zloćudnih aplikacija te u slučaju podudaranja dolazi do blokiranja tog zahtjeva, detaljnije o ovoj metodi u poglavlju 4.4. Problem je što je ime paketa vrlo lako izmijeniti pa će mnoge zloćudne aplikacije neometano slati zahtjeve koje ovakva vrsta zaštite neće detektirati.

Pretraživanje i uklanjanje malicioznih aplikacija na samom uređaju mogu odraditi i razni antivirusni programi, međutim oni svoju pretragu baziraju na raznim kriterijima koji ne ciljaju nužno na aplikacije koje uzrokuju prevare u sustavima mobilne naplate pa neće otkriti neke aplikacije koje izgledaju benigno i „samo“ imaju dopuštenja za slanje poruka ili uspostavljanje poziva.

Rješenje opisano i izrađeno u ovom radu je aplikacija koja se instalira na mobilni uređaj i vrši pretragu zloćudnih aplikacija prema dva kriterija. Prvi kriterij je usporedba imena paketa s popisom zloćudnih aplikacija koji je kreiran na temelju suradnje nekoliko mobilnih operatera i regulatora (belgijski Proximus, kenijski Safaricom, norveški Strex i južnoafrički regulator WASPA). Nakon toga, provjerava se imaju li otkrivene aplikacije neko od dopuštenja koje bi moglo uzrokovati neželjene dodatne naplate (slanje SMS poruka, pristup imeniku, uspostavljanje poziva itd.). Nakon što je pretraga završena korisniku se prikazuje popis svih otkrivenih potencijalno opasnih aplikacija. Klikom na svaku aplikaciju s liste korisnik dobiva uvid u sva potencijalno opasna dopuštenja koje aplikacija posjeduje. Tada je odluka na korisniku da odabere hoće li navedenu aplikaciju ukloniti sa svog uređaja ili ju ostaviti i riskirati da bude žrtva prevare. Veći dio tehnološki pismene populacije lako će se zaštiti ovim rješenjem, dovoljno je samo logički razmisliti imaju li dopuštenja pojedine aplikacije smisla s obzirom na njezinu namjenu, npr. ako aplikacija za uređivanje slika ima dopuštenje za uspostavljanje poziva, odmah je jasno da tu nešto nije u redu i da se aplikacija treba ukloniti.

3. SUSTAVI MOBILNE NAPLATE

U ovom poglavlju opisan je nastanak sustava mobilne naplate, dvije glavne metode naplate te njihova tehnička realizacija i primjena.

3.1. Kratka povijest mobilne naplate

Naplata usluga putem mobilnog računa pojavila se vrlo brzo nakon pojave mobilnih uređaja 90-ih godina prošlog stoljeća. Mobilni operateri su tražili novi način monetizacije svojih usluga pa su korisnicima nudili dodatne zabavne sadržaje poput posebnih melodija zvona, monokromatskih sličica ili mobilnih igrica. Korisnici bi prvo trebali poslati SMS sa zahtjevom za ovakvom vrstom sadržaja i tada bi povratnim SMS-om dobili pristup traženom sadržaju i bili naplaćeni za iznos koji je obično znatno veći od cijene „obične“ SMS poruke. Kako bi ih lakše razlikovali od standardnih SMS poruka, dodan im je prefiks „Premium“ koji odmah sugerira da se radi o posebnom sadržaju koji će biti dodano naplaćen. Prvi Premium SMS servis je pokrenut 1998. godine od strane operatera Saunalahti (danas naziva Elisa) u Finskoj, a omogućavao je kupovinu posebnih melodija zvona isključivo za Nokia mobitele.

Ubrzo su i ostali mobilni uređaji dobili podršku za ovakve sadržaje pa je došlo do ogromnog rasta prihoda, a procjenjuje se da se granica od jedne milijarde dolara prihoda premašila nakon četiri godine (na globalnoj razini). Do 2008. godine prihodi samo od prodaje melodija zvona su bili iznad 5 milijardi američkih dolara. Ovakva razina zarade ubrzo je pobudila zanimanje u raznim granama industrija koje su se na bilo koji način mogle povezati sa mobilnom i ponuditi svoje sadržaje putem nje. Tako su se pojavili Premium SMS servisi koji su omogućavali plaćanje parkinga, donacija, dobivanje astroloških savjeta, uvid u događanja u idućoj epizodi omiljene meksičke sapunice i slično. Praktički bilo kakav sadržaj koji se mogao isporučiti unutar SMS poruke je bio dozvoljen ako je zadovoljavao određen skup pravila koja su definirali operateri zajedno sa regulatorom u svakoj pojedinoj zemlji.

3.2. Premium SMS

3.2.1. Općenito o Premium SMS-u

Premium SMS (u daljnjem tekstu PSMS), označava metodu naplate mobilnih korisnika putem SMS (engl. *Short Message Service*) poruke. Ovaj oblik naplate datira kao jedan od najstarijih te je i danas najraširenija metoda naplate mobilnih korisnika. Osnovna razlika između standardnog SMS-a i PSMS-a je cijena koja je naplaćena krajnjem korisniku. Ukoliko uzmemo za primjer

jednog mobilnog operatera u Hrvatskoj - Telemach, cijena standardne SMS poruke za Telemach korisnike je 0,49 HRK (podatak preuzet s službenog cjenika usluga Telemach https://telemach.hr/upload/pdf/Cjenik_telekomunikacijskih_usluga.pdf (datum pristupa 30.06.2021.)), kod PSMS-a cijena jedne poruke može biti od 0,49 HRK do 75,00 HRK. U tom slučaju, korisnik za poruku koju je platio više od standardne cijene propisane cjenikom mobilnog operatera, mora primiti zauzvrat nekakav oblik usluge ili digitalnog dobra kako bi naplata bila opravdana i u skladu s regulativom.

Po učestalosti naplate razlikujemo tri vrste PSMS usluga:

- Jednokratne PSMS usluge
- Pretplatničke PSMS usluge
- *Chat* (usluge čavrljanja)

Jednokratne PSMS usluge naplaćuju se samo onda kada to korisnik eksplicitno inicira prvi, te će se naplatiti samo onda, bez periodičkog ponavljanja naplate osim ako to korisnik ponovno ne zatraži. Plaćanje parkiranja PSMS uslugom je primjer jednokratne PSMS usluge gdje korisnik inicira naplatu slanjem SMS poruke sa tekstom svoje registracije na već predviđeni kratki broj.

Pretplatničke PSMS usluge naplaćuju se periodično s tim da korisnik ne mora inicirati naplatu svaki put, već se korisnik pridružuje pretplatničkoj usluzi slanjem prve, te najčešće slijedi druga potvrdna poruka da korisnik zaista želi biti pretplaćen na uslugu po određenoj cijeni i određenom intervalu naplate. U sustavu se za tog korisnika otvara sesija kako bi se mogla izvršiti periodična naplata. Izlazak iz pretplatničkog servisa najčešće se obavlja tako da korisnik šalje poruku „STOP“ na kratki broj koja se registrira kao naredba za zaustavljanje pretplate. Najbolji primjer takve usluge je tjedni horoskop, gdje korisnik najčešće šalje poruku s ključnom riječi svojeg horoskopskog znaka, te mu svakih 7 dana stiže nova naplatna poruka po cijeni koja je komunicirana prilikom iniciranja usluge.

Chat tj. usluge čavrljanja kombinira elemente pretplatničkih i jednokatnih usluga. Korisnik inicira uslugu slanjem poruke na kratki broj predviđen za tu uslugu te na svoju poruku dobiva odgovor od moderatora koji je zadužen za čavrljanje. Ova se usluga ne naplaćuje periodički kao pretplatničke usluge, već se u većini slučajeva naplaćuje svaka poruka koju korisnik pošalje. Međutim, korisnik ne mora svakom porukom slati i ključnu riječ, jer se otvara sesija vezana za kratki broj kod pružatelja usluga, pa korisnik može slati poruke bez ključne riječi jer sustav prepoznaje da u tom trenutku korisnik ima interakciju s *chat* uslugom gdje je otvorena sesija.

3.2.2. Premium SMS i njegova uporaba u praksi

U svojim začecima, PSMS je bio korišten kao mobilni oblik naplate za usluge kao što su kupovina pozadina ili melodija zvona za mobilni telefon, a kasnije su došle i razne igre koje korisnik može kupiti i zatim skinuti na svoj mobilni telefon. Danas se PSMS oblik naplate također koristi u svrhu donacija, glasovanja u popularnim televizijskim emisijama, slanja poruka u obliku pozdrava na raznim televizijskim ili radijskim programima i slično.

Nerijetko čitamo u medijima o zloporabi PSMS usluga i pružateljima usluga koji koriste ljudsku neinformiranost ili koriste lažne promocije kako bi mobilni korisnici (ne)svjesno platili mobilnu uslugu, a pritom biti navođeni informacijama kako se radi ili o besplatnoj usluzi ili o usluzi koja zahtjeva naplatu kako bi se završio proces kojeg je korisnik započeo besplatno. Ovdje možda najbolji primjer predstavljaju IQ testovi koji su u svom punom zamahu poharali Republiku Hrvatsku. Mobilni korisnik bi riješio sva pitanja IQ testa, a kako bi saznao rezultat morao se pretplatiti na uslugu koja je koštala 36,00 – 40,00 HRK tjedno. U zadnje vrijeme popularni način za varanje korisnika je putem oglasa na internetu obavještavalo kako mu je mobilni telefon „zaražen virusom“ ili da je na njegovom uređaju pronađeno određen broj prijetnji te da moraju poslati poruku na određeni broj kako bi zaštitili svoj uređaj. Za cijeli slučaj bila je potrebna i intervencija hrvatskog tržišnog regulatora za telekomunikacije – HAKOM-a koja će kasnije u radu biti detaljnije opisana.

3.2.3. Tehnička realizacija Premium SMS-a

PSMS metoda naplate tehnički je realizirana na sljedeći način; operater, tehnički pružatelj usluge, pružatelj usluge koja se naplaćuje te sam korisnik su najbitnije četiri uloge u tehničkoj realizaciji. Mobilni operater je jedina figura koja može teretiti mobilni račun nekog korisnika za iznos koji je predviđen za neku uslugu. Kako se na mobilne operatere ne bi spajali svi pružatelji usluga posebno, tu postoje agregatori tj. tehnički pružatelji usluga koji služe kao čvorište te ih je u pravilu na tržištu manje od pet. Kada tehnički pružatelj usluge i operater uspostave konekciju između svojih sustava i SMS centra, tehnički pružatelj usluge može na svoj sustav spajati sve druge pružatelje usluga putem HTTP (engl. *Hypertext transfer protocol*) ili SMPP (engl. *Short Message Peer to Peer*) protokola.

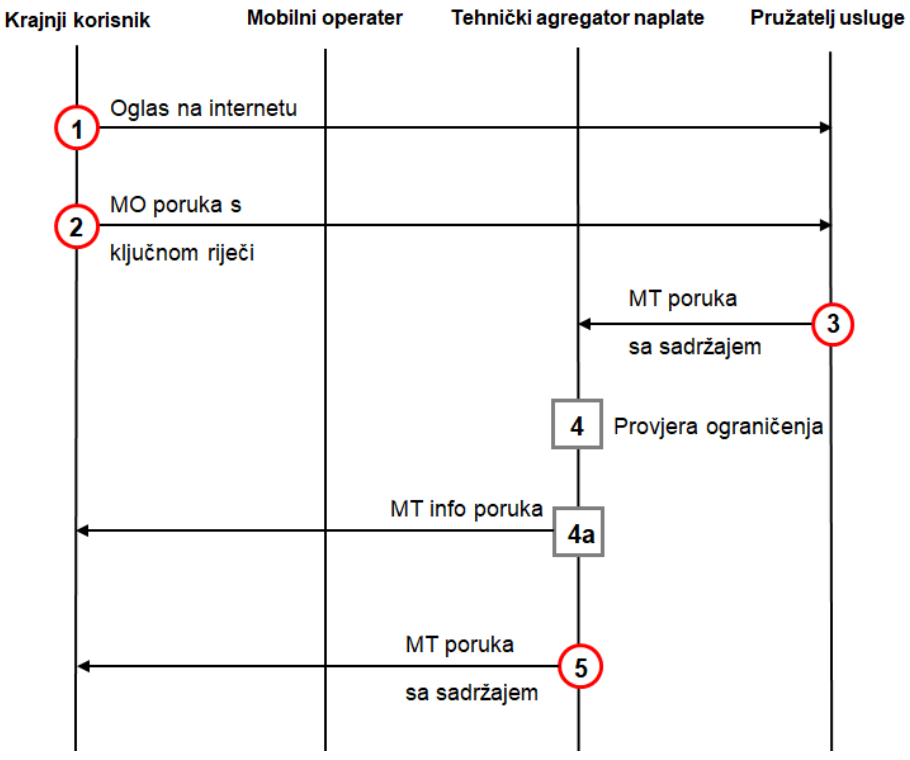
Za cijeli proces naplate izrazito su bitni i kratki brojevi prema kojima će mobilni korisnik poslati svoju poruku, te ključne riječi koje će u sustavu tehničkog pružatelja usluge „okinuti“ početak naplate. Kratki brojevi su definirani najčešće od strane tržišnog regulatora za telekomunikacije (npr. HAKOM) dok su ključne riječi definirane od samog pružatelja usluge ovisno o tome kakve

su mu preferencije. Kada korisnik pošalje na predefinirani kratki broj poruku koja na samom početku sadrži ključnu koja je spojena u sustavu tehničkog pružatelja naplate – sustav će dalje komunicirati sa sustavom operatera i pružatelja usluge kako bi se provjerila naplativost mobilnog korisnika te isporuka same usluge mobilnom korisniku.

U PSMS-u postoje dva različita oblika naplate:

- MO (engl. *mobile originated*) naplata i
- MT (engl. *mobile terminated*) naplata.

Osnovna razlika između ova dva oblika naplata je smjer iz kojeg stiže naplatna poruka. MO naplata označava da je sam korisnik poslao prvu poruku u želji da plati neku uslugu – u tom slučaju će ta prva poruka koju korisnik pošalje na kratki broj već biti naplatna, iako će on uslugu ili pristup usluzi dobiti u (besplatnoj) poruci koja će tek stići na njegov mobilni uređaj. MT naplata označava da je korisnik inicirao naplatu, međutim njegova prva poruka neće biti naplatna već ona koja mu stiže na mobilni uređaj.

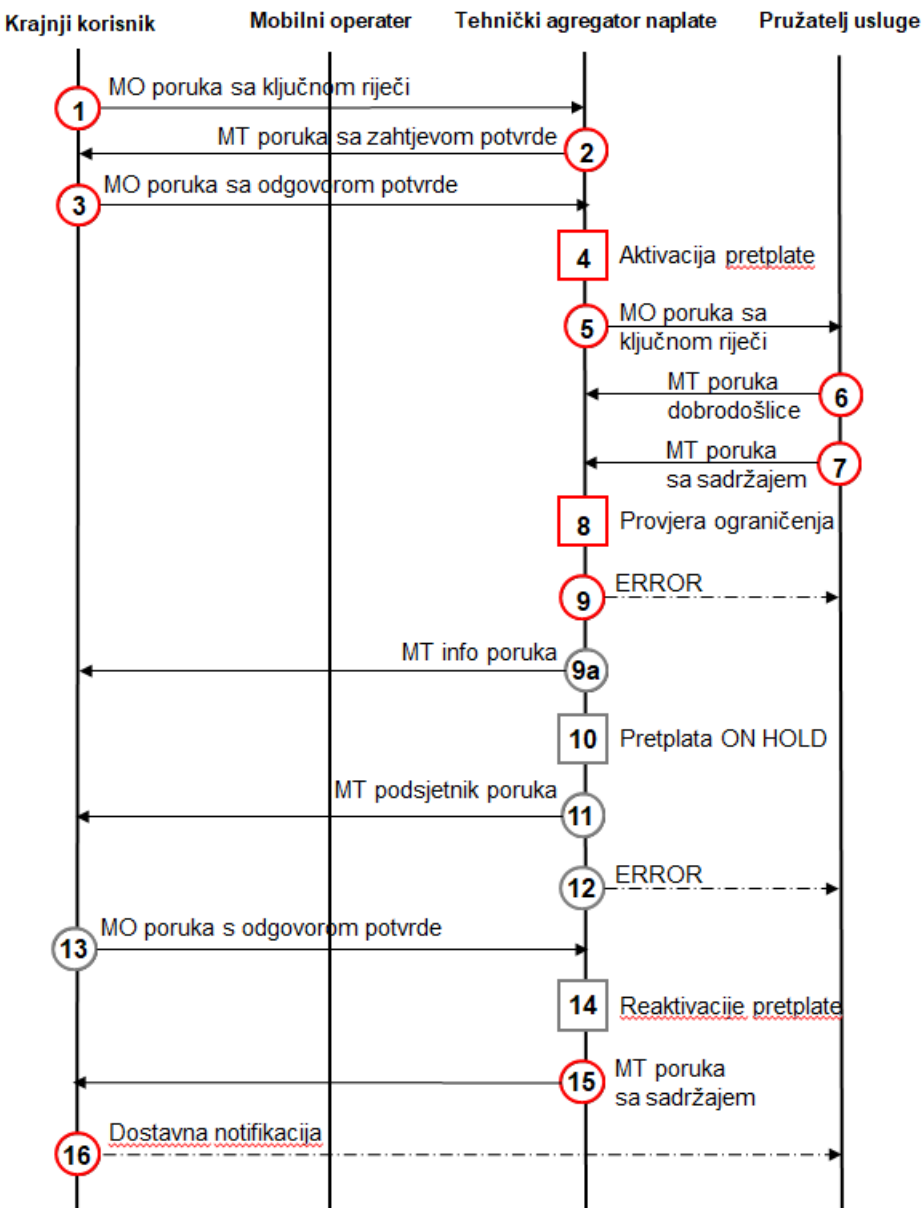


Sl. 3.1. Dijagram toka naplate jednokratne SMS usluge

Slika 3.1. prikazuje dijagram toka naplate putem jednostavne, jednokratne SMS usluge naplative putem SMS MO ili MT poruke. Ako se radi o MO naplati, biti će dovoljno izvesti samo korake 1

i 2, dok u slučaju MT naplate postoje dodatne provjere koje će uključivati cijeli prikazani proces. Krajnji korisnik će najprije slučajnim pretraživanjem naići na oglas na internetu koji mu primjerice nudi dnevni horoskop. Zatim će pročitati upute koje kažu kako je potrebno poslati SMS poruku s ključnom riječi njegovog horoskopskog znaka. Kada korisnik pošalje tu poruku, ona najprije stiže do mobilnog operatera koji ju automatski prosljeđuje tehničkom agregatoru naplate zbog toga što SMS centar mobilnog operatera prepoznaje da je kratki broj na koji se šalje poruka dodijeljen određenom tehničkom agregatoru. Tehnički agregator također pristiglu MO poruku prosljeđuje odgovarajućem pružatelju usluga s obzirom da je ključna riječ u MO poruci dodijeljena tom pružatelju usluge. U ovom slučaju, MO SMS poruka bila je naplatna, te se naplata izvršila u onom trenutku kada je poruka stigla do mobilnog operatera. Pružatelj usluge tada šalje MT poruku u kojoj se nalazi sadržaj.

Ako se radi o MT naplati, navedena MT poruka se zaustavlja kod tehničkog agregatora kako bi se provjerilo ukoliko postoje ograničenja u naplati za taj MSISDN (engl. *Mobile Station International Subscriber Directory Number)*. Primjerice ako je tržišni regulator postavio određenu svotu novca koju korisnik može potrošiti na usluge s dodanom vrijednošću. Tada sustav tehničkog agregatora naplate provjerava ukoliko je taj MSISDN bio prije naplaćivan, koliko puta te da li je dostigao ograničenje ukoliko ono postoji. Ukoliko je MSISDN došao do nekog ograničenja tada ide korak 4a, kada se šalje MT informacijska poruka o potrošnji na korisnikov MSISDN. Ako nema nikakvih ograničenja za ovog korisnika, tehnički agregator može proslijediti MT poruku sa sadržajem do krajnjeg korisnika koja će se pri dostavi naplatiti po posebnoj cijeni.



Sl. 3.2. Dijagram toka naplate kod SMS pretplatničkih usluga

Slika 3.2. prikazuje dijagram toka naplate kod SMS pretplatničkih usluga. Pretplatničke usluge kod PSMS-a mogu se ostvariti samo pomoću MT naplate. Korisnik pretraživanjem interneta nailazi na reklamu koja predstavlja pretplatnički sadržaj, primjerice tjedni horoskop. Takav reklamni sadržaj korisniku daje upute koju ključnu riječ poslati na koji kratki broj, koja je frekvencija naplate te po kojoj cijeni.

Korisnik tada na kratki broj šalje ključnu riječ koja se prosljeđuje do tehničkog agregatora naplate. U slučaju pretplatničkih servisa, radi prevencije prevara, operateri najčešće zahtijevaju da se pristupanje pretplatničkom servisu provjerava kroz dodatnu poruku potvrde. Tada tehnički agregator naplate šalje MT poruku zahtjeva potvrde na koju će korisnik ako se zaista želi pretplatiti

na uslugu odgovoriti s DA ili OK ovisno o tome kako je to tržišni regulator/operater ranije propisao.

Kada korisnik potvrdi da se zaista želi pretplatiti na uslugu, njegova se MO poruka prosljeđuje do tehničkog agregatora naplate u čijem se sustavu pokreće događaj aktivacije pretplate tj. otvara se sesija za MSISDN. MO poruka se prosljeđuje do pružatelja usluge.

Pružatelj usluge šalje dvije MT poruke od čega je jedna naplatna. Prva poruka, MT poruka dobrodošlice, korisnika obavještava o usluzi koju je kupio. Sljedeća poruka je naplatna i u njoj bi u pravilu trebao biti sadržaj kojeg je korisnik kupio, primjerice tjedni horoskop. Obje se te poruke zadržavaju kod tehničkog agregatora naplate kako bi se izvršile dodatne provjere o ograničenjima. Provjerava se ukoliko korisnik već ima aktivnu sesiju za uslugu koju pokušava aktivirati, jesu li parametri i frekvencija naplate postavljeni točno te je li korisnik dostigao dnevna/mjesečna ograničenja postavljena od stane operatera ili tržišnog regulatora. Ukoliko provjera ograničenja nije prošla za neki od tri kriterija, MT poruke će biti blokirane, a pružatelj usluge će biti obaviješten o razlogu blokade poruka. Isto tako, korisnik će dobiti besplatnu MT poruku ako je dostigao ograničenja o potrošnji postavljena od strane operatera.

Ukoliko je na osmom koraku, provjeri ograničenja provjera obustavljena jer je korisnik dostigao 150 HRK ograničenja postavljenog od strane regulatora, tada se pretplata privremeno zaustavlja, a korisniku se šalje MT poruka podsjetnik o potrošnji na koju mora odgovoriti s DA ako želi nastaviti koristiti uslugu, u određenom vremenskom roku. Kada korisnik potvrdi s porukom DA, pretplata se reaktivira te se napokon prosljeđuje naplatna MT poruka sa sadržajem, a ovisno o njenoj naplativosti i isporučivosti, pružatelju usluga se šalje dostavna notifikacija. Koraci od 7 do 16 će se ponavljati svaki period dok korisnik ne pošalje STOP poruku kako bi se odjavio s usluge.

3.3. Direct Carrier Billing – DCB

3.3.1. Općenito o Direct Carrier Billing-u

Direct Carrier Billing (također poznato kao i *Direct Operator Billing*), kraće DCB najnoviji je sustav naplate koji za razliku od ranije spomenutog PSMS-a, ne zahtjeva korištenje SMS poruka u procesu naplate, već se ona izvršava direktno na račun, a poruke koje eventualno stižu su informativnog karaktera. Jedni od prvih operatera koji su uveli DCB način naplate su oni iz Ujedinjenog Kraljevstva koji su sustav DCB naplate objedinili u jedinstveni sustav Payforit, koji je svoju prvu verziju objavio u 2007. godini.[2] Isto kao i kod PSMS sustava naplate, postoje 2 vrste usluga koje se mogu nuditi putem DCB naplatnog sustava, jednokratne i pretplatničke usluge.

Važno je napomenuti da je i Premium SMS u suštini *Carrier Billing*, tj. naplata putem mobilnog operatera, ali se zbog distinkcije koristi drugi naziv jer je odmah jasno da se naplata odvija putem SMS poruke, a kod DCB-a se naplaćuje „direktno“ na naplatnoj platformi operatera.

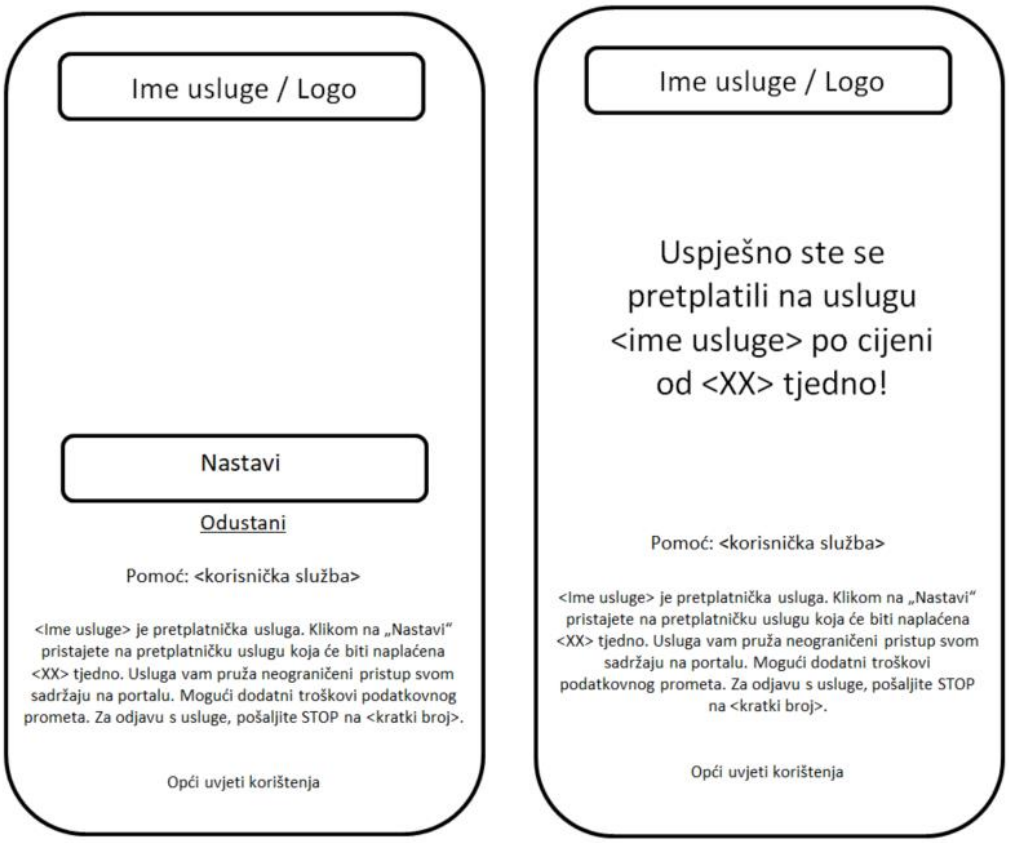
Prednost DCB naplate realizira se u minimalnoj uključenosti korisnika u proces dok kod PSMS-a korisnik mora poslati ključnu riječ na kratki broj kako bi započeo proces naplate, kod DCB-a korisnik je s reklame na internetu proslijeđen na stranicu za naplatu gdje se od njega traži samo pritisak na gumb (ukoliko koristi mobilni internet) ili unos svojeg mobilnog broja (MSISDN) te potvrda PIN kodom (ukoliko koristi WiFi Internet). Kada korisnik pretražuje internet koristeći mobilnu podatkovnu mrežu, njegov se MSISDN koji je u tom trenutku poznat samo operateru šalje u zaglavlju HTTP zahtjeva prema tehničkom pružatelju usluge koji ga nadalje prosljeđuje pružatelju usluge. Taj MSISDN može biti u punom ili šifriranom obliku.

3.3.2. Vrste tijeka DCB naplate

Tijek mobilne naplate DCB metodom može se razlikovati od operatera do operatera. Standardni (često i zlorabljeni) tijek DCB naplate je sljedeći:

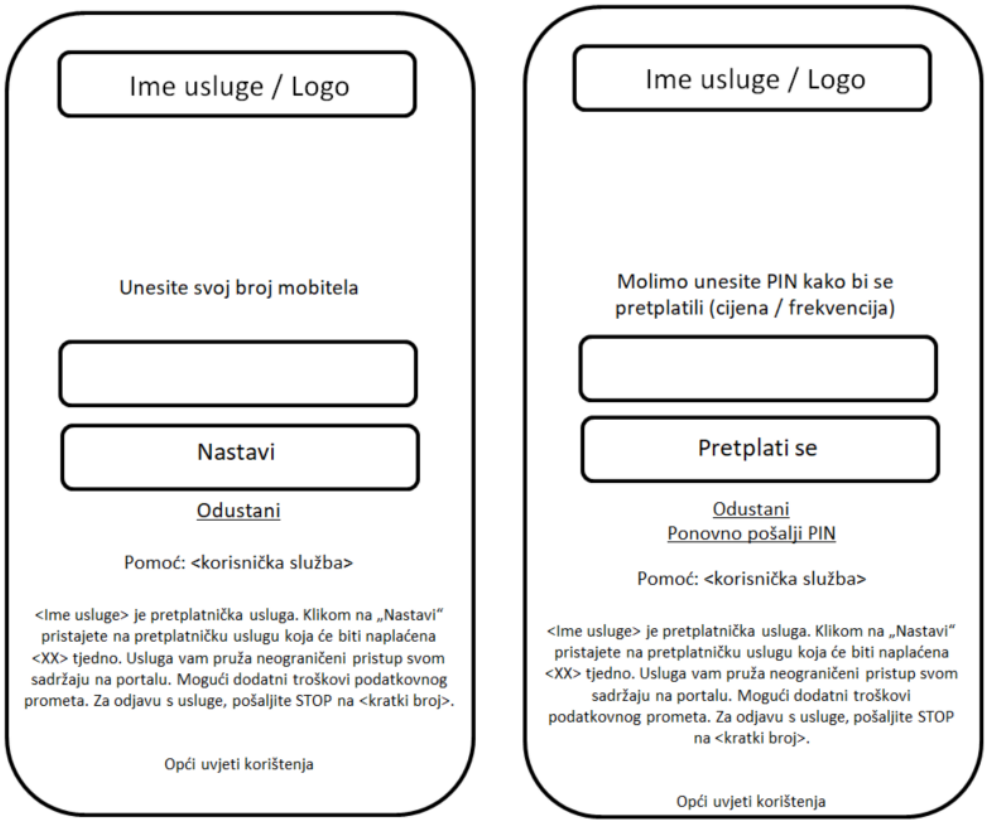
1. Korisnik pretražuje internet te naiđe na reklamu za mobilnu uslugu koja ga zanima i klikne na nju.
2. Reklama ga odvodi do pristupne stranice (engl. *landing page*) same usluge, gdje se nalaze detalji o vrsti usluge, cijeni i frekvenciji naplate. Korisnik klikne na gumb za kupovinu.
3. Gumb za kupovinu će poslati zahtjev prema tehničkom pružatelju usluge koji će vratiti odgovor u obliku preusmjeravanja korisnika s pristupne stranice na naplatnu stranicu operatera gdje je potrebno ponovno potvrditi kupnju.
4. Korisnik potvrđuje svoju kupnju te mu eventualno stiže poruka informativnog karaktera s informacijama o načinu konzumiranja usluge.

S obzirom na to da je interakcija korisnika minimalna u ovom klasičnom tijeku DCB naplate, mnogi pružatelji usluge zlorabit će taj položaj te namjerno navoditi korisnike na kupovinu. Verifikacija samog korisnika vrši se preko obogaćenog HTTP zaglavlja (engl. *Header enrichment, skraćeno HE)* što znači da kada korisnik pretražuje internet koristeći svoj podatkovni promet, operater ima njegov MSISDN i šalje ga u zaglavlju prema tehničkom pružatelju usluge.



Sl. 3.3. Izgled stranica za naplatu putem DCB metode

Slika 3.3. prikazuje stranice za naplatu koje su u većini slučajeva smještene kod operatera ili tehničkog agregatora. Ovo je najosnovniji i najjednostavniji tijek naplate, ali on je ujedno i najranjiviji na sve oblike prevare. U ovom slučaju korisnik klikne na gumb „Nastavi“ te ga na sljedećoj stranici čeka potvrda o kupnji.

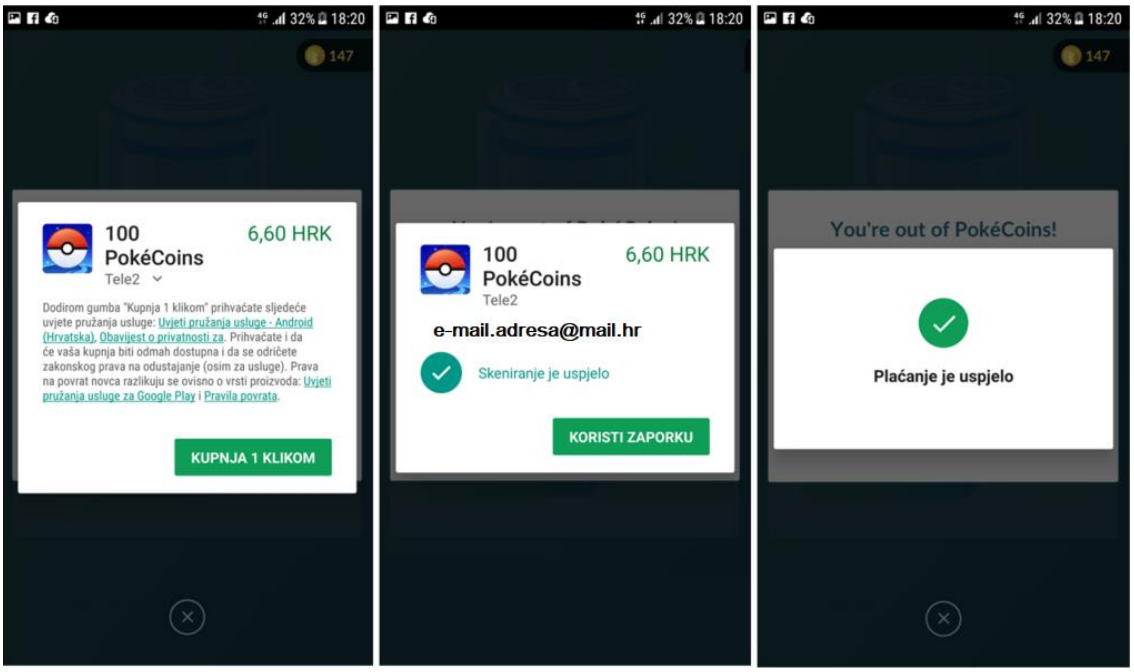


Sl. 3.4. Stranice za verifikaciju naplate, s dodatnom verifikacijom pomoću PIN koda

Na slici 3.4. prikazan je izgled stranice za verifikaciju naplate pomoću PIN koda. Ukoliko korisnik koristi WiFi Internet, tj. ne koristi podatkovnu mrežu putem koje bi se mogao dobiti njegov MSISDN, on će biti preusmjeren na stranicu za unos svojeg mobilnog broja. Kako bi se provjerila vjerodostojnost unesenog broja, na isti će se poslati PIN kod kojeg je korisnik obvezan upisati na sljedeću stranicu kako bi završio proces verifikacije naplate. Provjeru verifikacije PIN koda može provoditi operater ili tehnički agregator ovisno o tome kako je dogovoreno.

3.3.3. Primjena DCB naplate

Trenutno najraširenija primjena DCB-a je unutar Google Play trgovine aplikacija u velikom broju zemalja. Osim što se naplata može izvršiti kreditnom karticom, Google je svojim Android korisnicima omogućio plaćanje putem mobilnog računa, tj. DCB metodom naplate. Korisnik sam odabire način verifikacije (lozinka Google računa, otisak prsta, skeniranje rožnice ili lica, ukoliko je tehnologija dostupna) te može odabrati ako želi verifikaciju provoditi svaki put prilikom naplate ili samo pri prvom plaćanju.



Sl. 3.5. *Google Play naplata putem DCB metode*

Slika 3.5. prikazuje tijek naplate korisnika za virtualni „novac“ unutar popularne mobilne igre PokemonGO. Prema općim uvjetima poslovanja, sve aplikacije koje se nalaze unutar Google Play trgovine moraju plaćanja za svoje usluge vršiti putem metoda naplate koje su podržane od strane Google-a za određenu zemlju.

U ovom slučaju korisnik se odlučuje za jednokratnu kupovinu koja će ga koštati 6,60 HRK, te će svoju kupovinu potvrditi otiskom prsta (verifikacija u dva koraka). Na kraju se prikazuje poruka koja ga obavještava o uspješnoj kupovini.

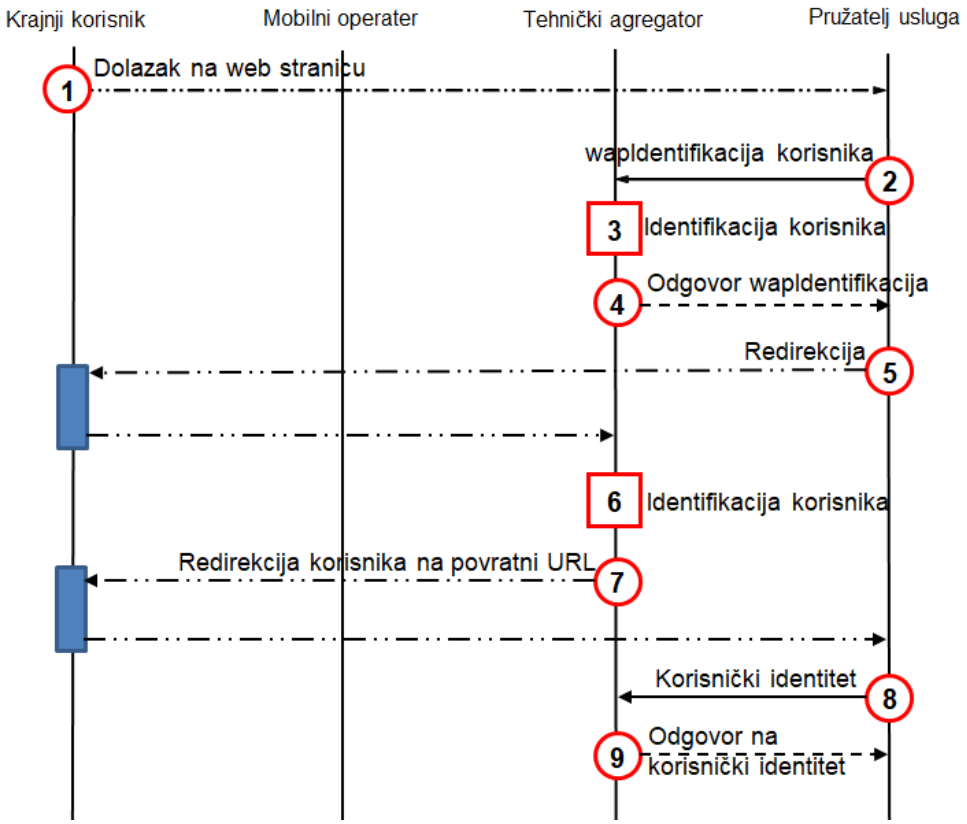
Drugi veliki pružatelji digitalnih dobara također se oslanjaju na DCB sustav naplate jer se takva metoda dokazala kao vrlo pristupačna i stoga ju korisnici radije odabiru ispred plaćanja kreditom karticom.

Međutim, upravo zbog te činjenice kako je tijek naplate zaista jednostavan, mnogi pružatelji usluga zlorabljuju DCB metodu kako bi korisnike putem raznih načina prevare spomenutih u narednim poglavljima, pretplatili na skupe usluge čija se naplata vrši na dnevnoj, višednevnoj, tjednoj ili mjesečnoj razini. Korisnicima, često nesvjesnim kako su aktivirali pretplatničku uslugu, budu naplaćeni pozamašni iznosi iako oni tu uslugu nisu htjeli ili nisu bili u potpunosti upoznati s detaljima naplate. Prema portalu The Global Economy, i zadnje dostupnim podacima iz 2017. godine, u Hrvatskoj je oko 35% građana posjedovalo kreditnu, a 68% debitnu karticu. U isto vrijeme prema GSMA (GSM Association) kojoj pripada preko 750 mobilnih operatera širom

svijeta, u Hrvatskoj je mobilna penetracija (odnos ukupnog broja aktivnih SIM kartica i ukupnog broja stanovnika) 126%. Odnosi se naravno razlikuju od države do države, ali u većini slučajeva vrijedi pravilo da znatno više korisnika posjeduje mobilni telefon nego kreditnu ili debitnu karticu, stoga je logično da mobilno plaćanje ima ogroman potencijal za još veći rast, ali i da nosi velike rizike ukoliko je regulativa manjkava.

3.3.4. Tehnička realizacija DCB metode naplate

DCB, slično kao i PSMS također zahtijeva 4 važne uloge (operater, krajnji korisnik, tehnički pružatelj usluge i pružatelj usluge). Osnovna je razlika u komunikaciji između sustava operatera, tehničkog i običnog pružatelja usluge. Ona je nešto kompliciranija i odvija se u više koraka.

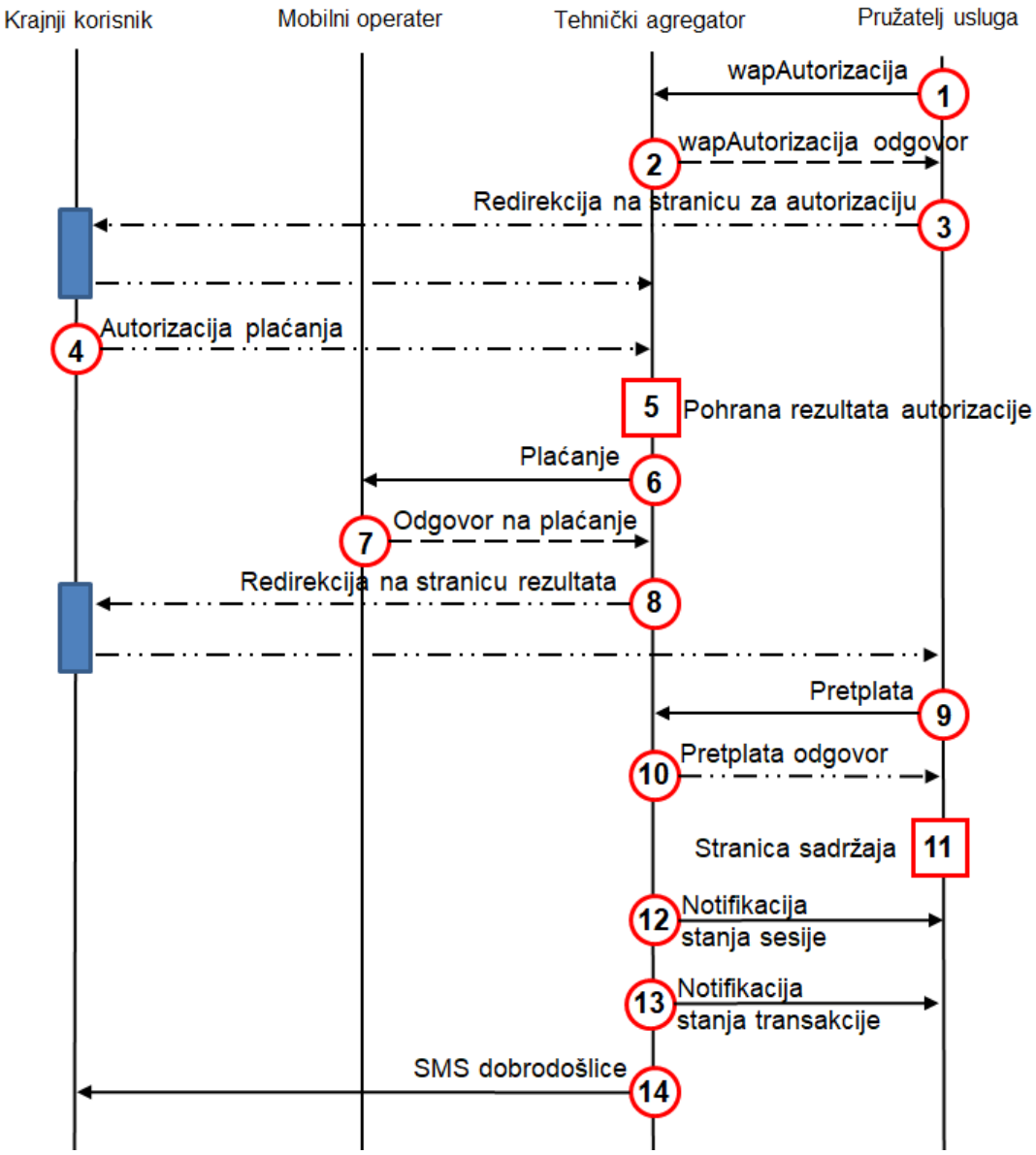


Sl. 3.6: Identifikacija korisnika u DCB sustavu

Slika 3.6. prikazuje proces identifikacije korisnika u DCB sustavu. Korisnik pretraživanjem interneta nailazi na web stranicu pružatelja usluga, te pružatelj usluge šalje zahtjev za identifikacijom korisnika. U tom se zahtjevu šalje nekoliko parametara poput IP adrese korisnika, dodijeljenih parametara za račun klijenta konfiguriran na strani tehničkog agregatora te povratnu poveznicu (engl. *Callback URL*) koja će se kasnije pozivati. Tada tehnički agregator započinje

proces identifikacije korisnika, te odmah šalje odgovor pružatelju usluga na metodu wapIdentifikacija.

Kada pružatelj usluge dobije odgovor na wapIdentifikaciju, on šalje zahtjev za preusmjeravanjem korisnika na povratnu poveznicu. Za to vrijeme tehnički agregator pokušava dobiti identitet (MSISDN) korisnika te kada je korisnik obrađen (dobiven je MSISDN), tehnički agregator će poslati zahtjev korisniku da ga se preusmjeri na povratnu poveznicu koja je dobivena u parametrima tokom wapIdentifikacije. Kada je korisnik preusmjeren na povratnu poveznicu, pružatelj usluga šalje zahtjev prema tehničkom agregatoru kako bi od njega dobio informaciju o identitetu korisnika, a nazad dobija odgovor u obliku MSISDN-a. Tada kreće proces naplate.



Sl. 3.7. *Naplata korisnika u DCB sustavu*

Slika 3.7. prikazuje tok naplate (autorizacije) unutar DCB sustava. Pružatelj usluge započinje sam proces slanjem wapAutorizacija zahtjeva prema tehničkom agregatoru. Tehnički agregator vraća odgovor na taj zahtjev ovisno da li korisnik već ima postojeću sesiju za tu uslugu ili ne. Ukoliko je sve u redu s korisnikom i sesijom, proces se nastavlja. Pružatelj usluga zatim korisnika preusmjerava na stranicu za autorizaciju koja je u ovom slučaju na strani tehničkog agregatora, no može se nalaziti i kod operatera ili samog pružatelja usluge.

Korisnik će ili odbiti ili prihvatiti naplatu klikom na gumb koji se nalazi na naplatnoj stranici, te se taj odgovor registrira kod tehničkog agregatora s obzirom da se stranica za autorizaciju naplate nalazi na toj strani. Tehnički agregator započinje proces same naplate šaljući zahtjev prema operateru, jer je operater jedini u ovom procesu koji može naplatiti korisnika. Operater vraća odgovor o statusu naplate ovisno o tome je li korisnik uspješno naplaćen ili ne (ukoliko korisnik nema dovoljno novčanih sredstava, operater će vratiti negativan odgovor).

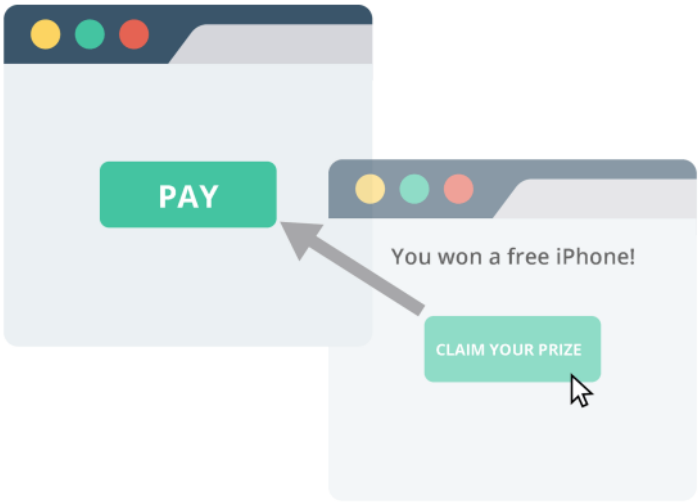
Nakon odgovora o uspješnoj naplati, tehnički agregator će preusmjeriti korisnika na povratnu poveznicu, a zatim će pružatelj usluge poslati zahtjev za otvaranjem pretplatničke sesije na strani tehničkog agregatora. Na taj će zahtjev tehnički agregator odgovoriti s pozitivnim ili negativnim odgovorom ovisno o tome da li je bilo moguće otvoriti pretplatničku sesiju za korisnika. Kada pružatelj usluge dobije odgovor o otvaranju pretplatničke sesije, korisniku će se ovisno o stanju pretplate pokazati stranica sa sadržajem ili stranica greške ukoliko otvaranje sesije nije bilo moguće. Tehnički agregator zatim šalje pružatelju usluge notifikaciju o stanju sesije i o stanju transakcije, te ukoliko je sve prošlo uspješno, tehnički agregator korisniku šalje besplatan SMS dobrodošlice u uslugu.

Nerijetko operateri uvode dodatne korake verifikacije poput verifikacije mobilnog korisnika PIN ili USSD kodom te u zadnje vrijeme koristeći *Captcha* metodu verifikacije (npr. korisnik mora označiti sve semafore na slici ili prepisati kod sa slike). Takve metode verifikacije svakako se moraju koristiti kada korisnik želi izvršiti naplatu putem DCB-a ali koristi WiFi internet, tj. internet koji nije mobilni pa se u zaglavlju ne šalje MSISDN. U tom slučaju, kada korisnički MSISDN nije dobiven iz zaglavlja zahtjeva koji se šalje, korisnik se preusmjerava na dodatnu stranicu gdje mora unijeti svoj MSISDN te će mu nakon toga stići poruka s PIN kodom kojeg je potrebno prepisati na stranicu za naplatu kako bi se izvela verifikacija da je korisnik zaista onaj kojim se predstavlja.

4. NAČINI PREVARE I SPRJEČAVANJE PREVARE U PROCESU MOBILNE NAPLATE

U ovom poglavlju su opisane najkorištenije metode prevara u sustavima mobilnog plaćanja prema istraživanju iz 2021. godine [7]

4.1. Prevara putem nevidljivih okvira



Sl.4.1. Ilustrativni prikaz *Clickjackinga*

Napadač koristi višestruke nevidljive ili djelomično prozirne slojeve na web stranici kako bi zavarao korisnika da klikne na drugu poveznicu unutar otvorenog okvira kao što je slikovito prikazano na slici 4.1.. Iako korisnik misli da je kliknuo unutar vidljivog okvira, internet preglednik je registrirao taj klik na (oku) nevidljivom okviru. Na taj način mogu se ukrasti klikovi i korisnika preusmjeriti prema drugoj neželjenoj stranici ili aplikaciji, zbog toga se ovaj način prevare često naziva i *clickjacking* (krađa klikova). U industriji mobilne naplate problem se događa ako nije uvedena zaštita kojom se blokiraju ovakve vrste okvira i međuokvira, jer u tom slučaju napadač vrlo lako može preusmjeriti klik krajnjeg korisnika kako bi u njegovo ime potvrdio plaćanje određene usluge bez da je korisnik toga svjestan; u praksi se pokazalo da se nakon ukradenog klika korisnika još nekoliko puta preusmjeri na razne stranice kako bi se pokušao prikriti trag do stranice na kojoj se ukradeni klik zapravo registrirao.

CSP (Content security policy) i LBE (Legacy Browser Exploit) zaštite u kombinaciji sa X-frame postavkama mogu u velikoj mjeri neutralizirati ovakve vrste prevara.

Glavni zadatak CSP je sprječavanje *Cross-site Scriptinga* koji je sada već stara metoda ubacivanja malicioznog koda ili skripti unutar kod same web aplikacije, koji jednom kad se aktivira može na razne načine ugroziti sigurnost korisnika. S obzirom da se u procesu naplate izmjenjuju važni osobni podaci korisnika pomoću kojih se može ostvariti novčana transakcija, *Cross-site* skripte predstavljaju velik sigurnosni problem, iako su svoj vrhunac dosegle prije desetak godina.

Jednostavna naredba „*frame-ancestors*“ se umeće unutar zaglavlja i određuje smije li se okvir ili i-okvir učitati na stranici. Ovime se direktno sprječava krađa klika pomoću korištenja skrivenih okvira u svakom pregledniku koji podržava CSP.

```
header("Content-Security-Policy: frame-ancestors 'none'");
frame-ancestors 'none'; - onemogućava otvaranje sadržaja unutar okvira bilo
kojoj domeni
frame-ancestors 'self'; - omogućava otvaranje sadržaja unutar okvira samo
trenutnoj domeni
frame-ancestors 'self' '*.drugastranica.com'
'https://moja.drugastranica.com'; - Omogućava otvaranje sadržaja unutar
okvira trenutnoj stranici, ali i svakoj podstranici na drugastranica.com
koristeći bilo koji protokol, i samo stranici moja.drugastranica.com,
koristeći HTTPS protokol na portu 443.
```

Pod starijim (engl. *legacy*) preglednicima se smatraju stariji preglednici poput Internet Explorera, ili npr. starije verzije Mozilla Firefoxa za koje je specifično da su kompatibilni sa većinom web aplikacija i da su sposobni pokrenuti praktički bilo kakav sadržaj, ali u isto vrijeme imaju velike sigurnosne propuste koji su kasnije ispravljeni. Obično isti podržavaju *JavaScript* ali ne i X-okvir (*eng. X-frame*) zaglavlja ili CSP. [11]

Metoda koja se ovdje koristi je javno poznata i lako se može pronaći na stranicama poput Stack overflow. U prvom koraku se stvori s*tyle* ID koji se u drugom koraku odmah obriše u slučaju da se ne otkrije potencijalna krađa klika i stranica se dalje normalno učitava sa svim okvirima.

```
<style id="antiClickjack">body{display:none !important;}</style>
<script type="text/javascript">
if (self === top) {
    var antiClickjack = document.getElementById("antiClickjack");
    antiClickjack.parentNode.removeChild(antiClickjack);
} else {
    top.location = self.location;
}
</script>
```

U suprotnom će umjesto otvaranja nove stranice preglednik vratiti HTTP 200 (OK) odgovor i spriječiti preusmjeravanje korisnika na stranicu podokvira.

X-Frame opcije HTTP zaglavlja se koriste za davanje dopuštenja pregledniku da otvori novu stranicu u okviru ili međuokviru. Uključivanjem ove opcije na sve HTTP zahtjeve koji sadrže HTML sadržaj se uvelike može spriječiti krađa klika i preusmjeravanje korisnika na neželjene stranice.

```
header('X-Frame-Options: DENY');
```

DENY – onemogućuje svakoj domeni da otvara nove okvire

SAMEORIGIN – omogućava samo trenutnoj stranici da otvara okvire

ALLOW-FROM url – omogućava određenoj stranici 'url' da otvara okvire na trenutnoj stranici

U nekim vrlo specifičnim slučajevima, tj. u specifičnim verzijama internet preglednika, s obzirom da metoda nije standardizirana za sve preglednike, može doći do sukobljavanja sa CSP metodom i na taj način omogućiti napadaču da uspješno zaobiđe sigurnosne postavke. U tom slučaju se može improvizirati sa isključivanjem jedne od metoda, ako se u prethodnoj interakciji sa korisnikom saznala verzija preglednika koji koristi.[13]

4.2. Prevara pomoću aplikacija s umetnutim kodom (Bypass metoda)

Slika 4.2. (ispod) prikazuje jednu od najčešćih metoda koje korisnika navode na instalaciju aplikacije koju korisnik originalno nije htio instalirati. Klikom na gumb „Remove virus!“ korisnik umjesto čišćenja svog mobilnog uređaja od virusa zapravo može dati privolu za plaćanje nekog pretplatničkog servisa koji je skriven u pozadini. To je moguće ako se na stranici nalazi dodatni okvir, iFrame, zbog kojeg korisnik ne vidi što se zapravo nalazi iza ove lažne slike koja mu poručuje da ima virus na mobilnom uređaju.

Sl. 4.2.: Prevara korisnika korištenjem lažne kampanje

Također, ovakve marketinške kampanje mogu „prisiliti“ korisnika na instalaciju neželjene aplikacije, koja u sebi može sadržavati dio koda koji će u pozadini, bez korisnikovog znanja i pristanka, naplatiti uslugu s kojom korisnik prethodno nije bio upoznat. S obzirom da se na mobilnim uređajima, pogonjenim Android operativnim sustavom, prilikom instalacije moraju dati dopuštenja za korištenje raznih resursa mobilnog uređaja od strane aplikacije koja se instalira, važno je posebno obratiti pažnju imaju li dopuštenja koja aplikacija zahtjeva smisla s obzirom na ono za što ona zapravo služi. Problem nastaje kada korisnici bez razmišljanja i čitanja samo klikaju gumb „dalje“ kako bi što prije instalirali aplikaciju. Na taj način mogu aplikaciji dozvoliti korištenje nativne SMS aplikaciju za slanje i primanje poruka, što može uzrokovati slanje i primanje naplatnih SMS poruka, ili mogu omogućiti uspostavljanje poziva koji se kasnije odvijaju bez korisnikova znanja.

U svjetskim medijima je posebno odjeknuo slučaj kada je tvrtka Upstream, koja se bavi istraživanjem i sprječavanjem prevara u mobilnim sustavima, otkrila kako je kineska kompanija Tecno prodala na afričkom tržištu preko 53,000 mobilnih uređaja koji su već imali predinstaliranu malicioznu aplikaciju. Nakon što bi se mobilni uređaj aktivirao, aplikacija je u pozadini pretraživala pretplatničke servise i slala zahtjeve za plaćanjem prema istima, bez da je korisnik i dotaknuo svoj mobilni uređaj. [4]

Slična situacija se dogodila i u Hrvatskoj 2018. godine kada je HAKOM morao reagirati zbog sličnog načina prevare. Na mobilnom uređaju bi se pojavila slično upozorenje kao na slici 4.2., te

bi se klikom na gumb instalirala aplikacija koja je kasnije slala zahtjeve za naplatu prema raznim pretplatničkim servisima. Stoga je HAKOM proveo inspekcijski nadzor nad tvrtkom DIMOCO Europe [6] koja je jedan od tehničkih agregatora na hrvatskom tržištu i zaključio „da operator usluga s posebnom tarifom DIMOCO Europe ... nije postupao sukladno članku 43. Pravilnika o načinu i uvjetima obavljanja elektroničkih komunikacijskih mreža i usluga i to na način da nije ispravno pružao uslugu s posebnom tarifom na kratkom SMS kodu 860086 u djelu koji se odnosi na osiguravanje da promidžbene aktivnosti ni na koji način ne budu zavaravajuće ili da na bilo koji način dovode u zabludu potencijalne korisnike.“ [5]

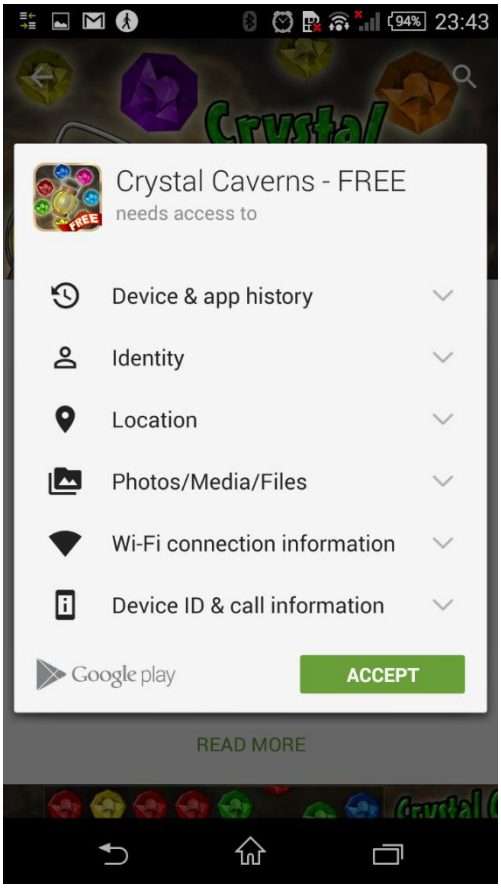
4.3. Obmanjujuće reklame

Obmanjujuće reklame kao oblik prevare su jedan od najraširenijih i ne zahtijevaju preveliku tehničku sposobnost napadača. Korisnika se u ovom slučaju navodi da misli da je usluga nešto što u svojoj pravoj prirodi zapravo nije. Jedan od najčešćih oblika je zasigurno „Test inteligencije“ koji korisnika vodi kroz niz pitanja kako bi se utvrdila njegova razina inteligencije, nakon što korisnik dođe do kraja rješavanja mora poslati poruku ili kliknuti na gumb kako bi na svoj mobilni uređaj primio rezultat testa. U mnogo slučajeva informacije o usluzi neće biti prikazane nigdje na stranici (cijena i učestalost naplate) ili će biti napisane sitnim slovima na dnu stranice. Ovdje korisnik misli kako će besplatno saznati razinu svoje inteligencije, jer mu tako sugerira obmanjujuća reklama, a zapravo se radi o prikrivenoj naplatnoj usluzi kao što se vidi na slici 4.3. [8,9]

Sl. 4.3.: Primjer reklame koja pokušava zainteresirati korisnika za IQ test

4.4. Prevare putem mobilnih aplikacija

Rastom popularnosti pametnih telefona i novim metodama naplate poput DCB-a, bilo je samo pitanje kada će se pojaviti nove metode kojima će se krajnje korisnike financijski oštetiti. Najviše prevara putem aplikacija se događa na Android sustavu, što je i logično ako se uzme u obzir Googleova blaga politika odobravanja prodaje aplikacija na službenoj Play Store trgovini bez detaljnih sigurnosnih provjera, za razliku od Appleove trgovine aplikacija (App store) koja ima puno rigoroznije metode.



Sl. 4.4.: Prikaz prekomjernih dopuštenja koje posjeduje jedna igra

Pri instalaciji aplikacije korisniku se nudi izbor hoće li omogućiti aplikaciji pristup određenim funkcijama telefona kao npr. pristup fotografijama, imeniku, dozvola za obavljanje poziva ili slanje poruka i slično, kao što je prikazano na slici 4.4.. U većini slučajeva korisnik samo želi što prije instalirati aplikaciju i bez čitanja tih upozorenja dozvoli aplikaciji sva dopuštenja koja zatraži.

Na taj način aplikacija može u pozadini umjesto korisnika slati poruke, presretati iste, brisati ih i na taj način prikriti prevaru, otvarati u pozadini internet preglednik i potvrđivati naplate umjesto korisnika i razne druge stvari koje su prepuštene mašti i kreativnosti napadača. [3]

Ovaj problem se vrlo lako može izbjeći sa malo pažnje pri instalaciji aplikacije i obraćanju pažnje na dozvole koje aplikacija traži, npr. ako aplikacija za uređivanje fotografija traži pristup imeniku ili slanju poruka, logično je prekinuti instalaciju jer aplikacija zasigurno ugrožava sigurnost korisnika. Naravno to nije dovoljno jer se ne može računati da će svaki korisnik paziti pri instaliranju aplikacija. Jedino sigurno je da su napadači svakim danom sve kreativniji i pronalaze razne načine za „ubacivanje“ potencijalno opasnih aplikacija na mobitele.

Kada aplikacija pokuša pokrenuti proces naplate u DCB sustavu, komunikacija se odvija putem HTTP protokola, a Android aplikacije u zahtjeve ubacuju zaglavlje koje izgleda ovako:

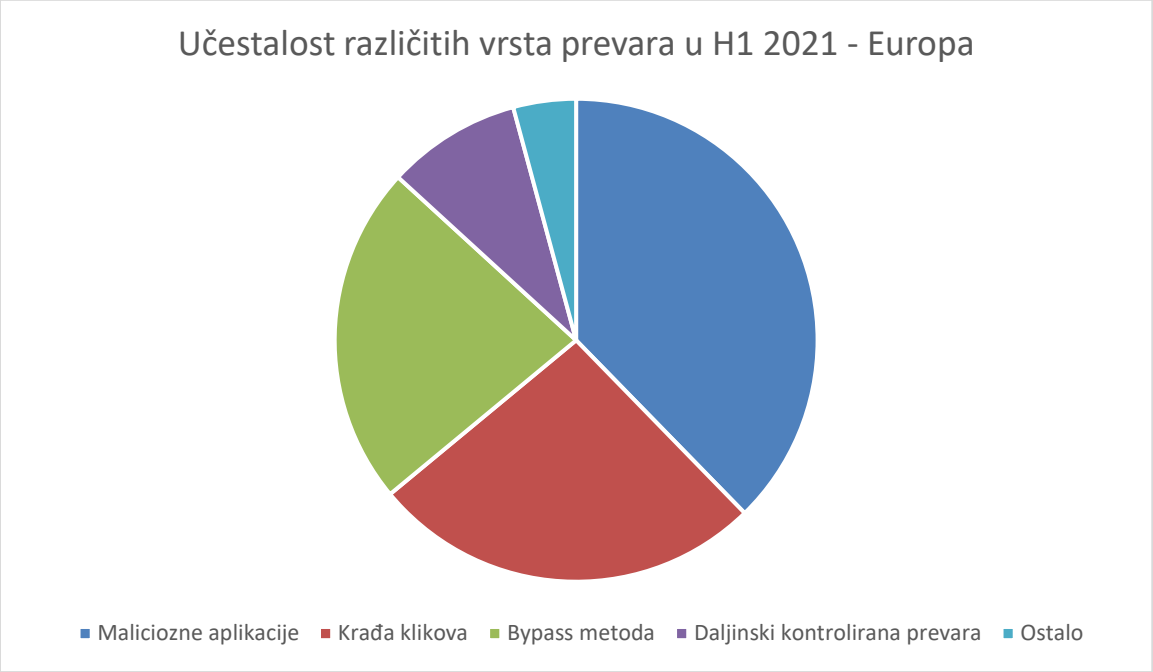
X-Requested-With: „ime paketa“

gdje je ime paketa u biti parametar prema kojem se može vidjeti iz koje se aplikacije šalje zahtjev, a pojavljuje se u formatu „com.xxx.yyy“, npr. com.keyboard.smile. Neke od popularnih aplikacija koje su omogućavale prevare su Color Camera (com.hdsj.hdey), File Manager (com.file.manager.gp) i X WALLPAPER (com.newac.wallpaper).[10]

U slučaju da se sumnja na potencijalne prevare iz aplikacija, može se ili blokirati određen set aplikacija kod kojih je primijećeno takvo ponašanje, ili se potpuno blokirati bilo kakvi zahtjevi iz aplikacija. Problem je što je potrebno postići ravnotežu između sigurnosti korisnika i mogućnosti oglašavanja usluga ili pokretanja naplata iz aplikacija. U slučaju detekcije zaglavlja koje sadrži X-Requested-With na temelju ranije postavljenih pravila će se taj zahtjev propustiti i obraditi kao regularan ili će se samo vratiti odgovor HTTP 200 (OK) i spriječiti daljnje preusmjeravanje korisnika ili samo idući korak u procesu naplate.

Ako je napadač dovoljno vješt programer postoji mogućnost da izmjeni ime paketa koje aplikacija šalje unutar zahtjeva pa se može prikazati kao da zahtjev dolazi iz druge aplikacije i na taj način zaobići zaštitu i opet uzrokovati probleme. U takvim slučajevima jedina zaštita je potpuno blokiranje zahtjeva iz aplikacija.

Metoda blokiranja zahtjeva na temelju imena paketa iz HTTP zaglavlja ima smisla samo kod DCB sustava mobilne naplate, ali ako se aplikacija koristi za slanje SMS poruka ili pozivanje brojeva koji naplaćuju pozive po posebnim tarifama, korisnik je prepušten sam sebi i u većini slučajeva mu je jedina opcija ulaganje žalbe svom mobilnom operateru nakon što je već oštećen za određenu sumu novca.



Sl 4.5. Učestalost tipova prevare u mobilnoj naplati

Evina kao jedna od glavnih kompanija u polju sprječavanja prevara u sustavima mobilne naplate, redovito provodi istraživanja o trendovima u svijetu prevara. Na slici 4.5. preuzetoj iz njihovog posljednjeg izvješća se vidi da su upravo ranije opisane metode one koje najčešće pogađaju korisnike mobilnog plaćanja na europskom tržištu. [7]

5. APLIKACIJA ZA OTKRIVANJE POTENCIJALNO OPASNIH APLIKACIJA NA MOBILNOM UREĐAJU

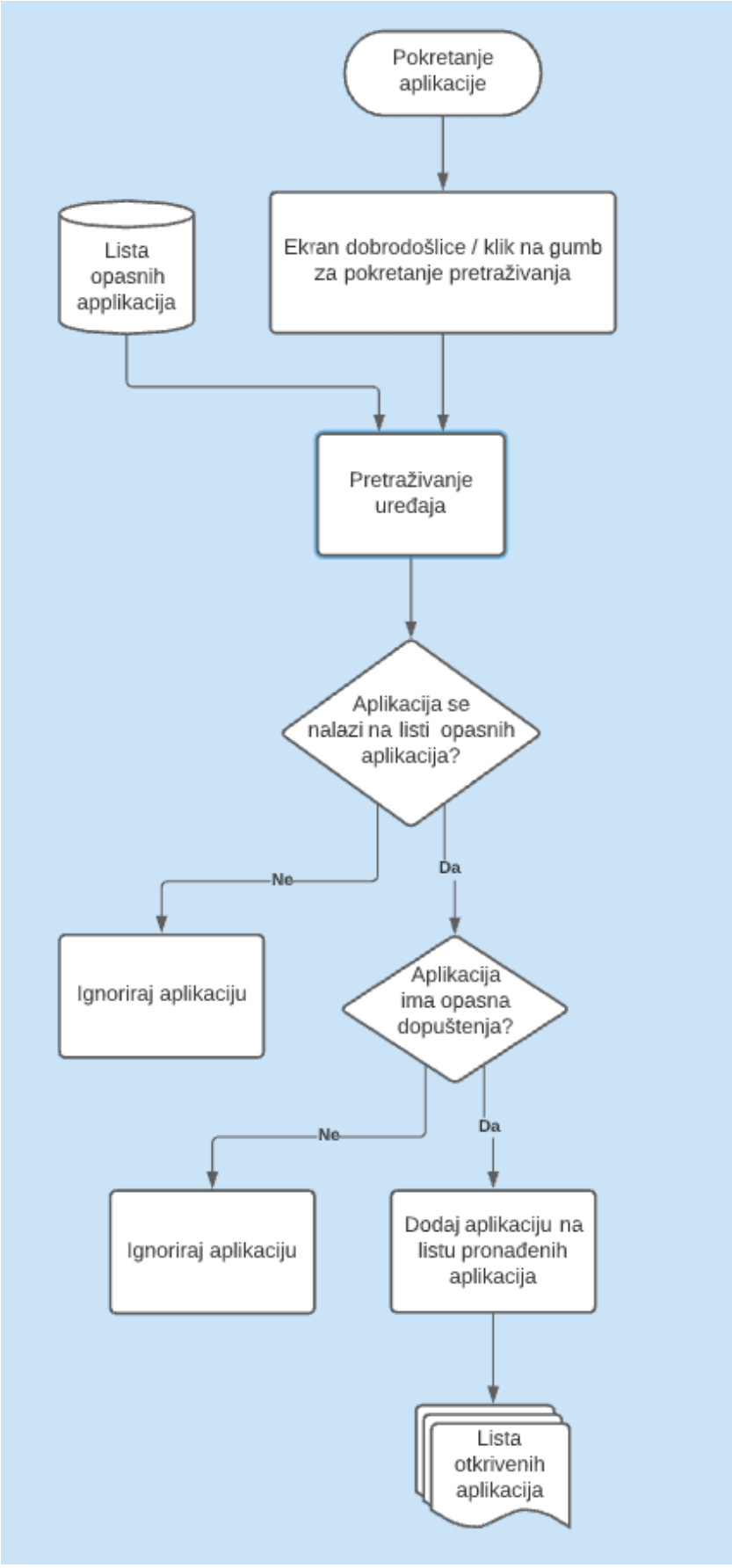
U ovom poglavlju biti će opisana izrada i funkcioniranje aplikacije koja će korisniku omogućiti otkrivanje potencijalno opasnih instaliranih aplikacija na njegovom mobilnom uređaju pogonjenom Android operativnim sustavom. Aplikacija podržava sve inačice operativnog sustava do aktualne verzije 11.

5.1. Korištene tehnologije

Aplikacija je pisana u programskom jeziku otvorenog koda Kotlin koji je izvrsna alternativa Javi. Kotlin je razvila ruska kompanija za razvoj programskih rješenja Jetbrains te je on prvi službeno podržani programski jezik u Android Studiu. Upravo je Android Studio korišten za kompajliranje koda i stvaranje instalacijske *apk* datoteke same aplikacije Antifrauder, kao i dodatne aplikacije „topple“ koja će simulirati malicioznu aplikaciju. Vizualni izgled aplikacije i svi prikazani tekstovi su definirani u XML dokumentima. Za razliku od HTML koda koji je sposoban prikazati podatke koji su definirani unutar njega, XML služi samo za prenošenje podataka, stoga je potrebno koristiti funkcije unutar programskog jezik, koje će prikazati podatke definirane unutar XML dokumenata.

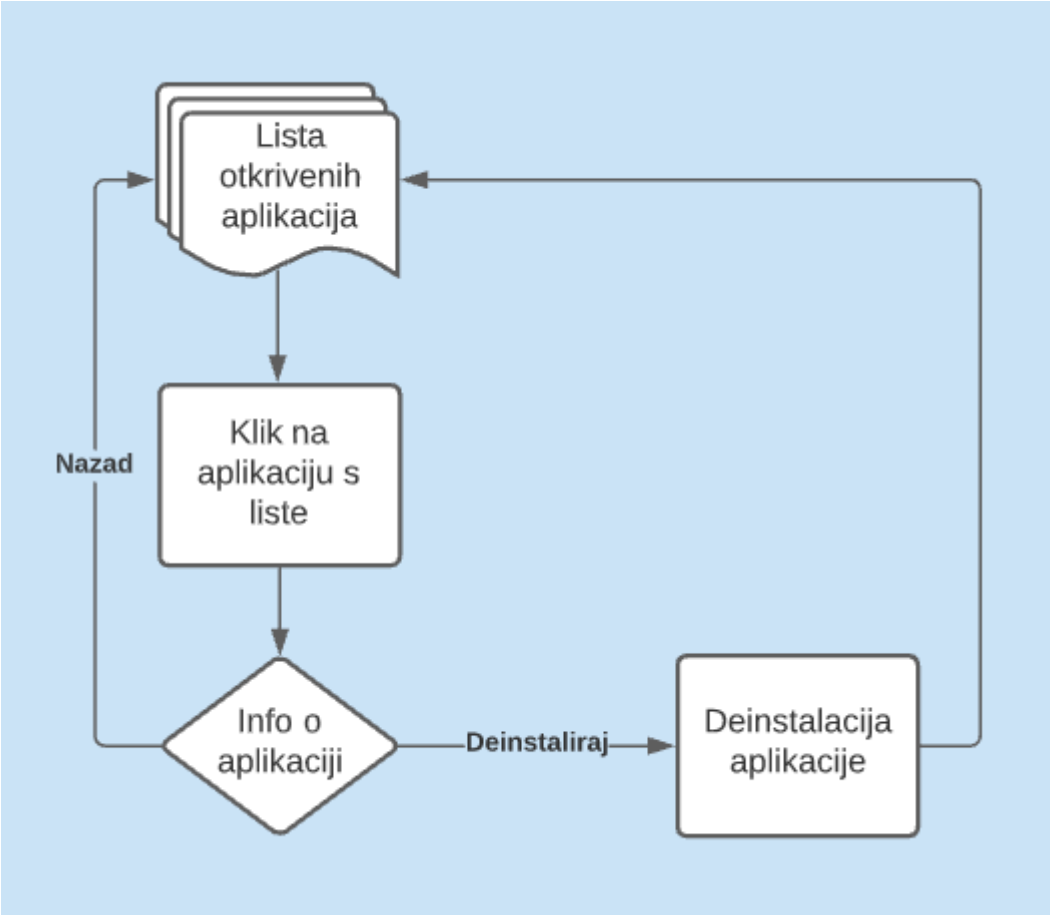
5.2. Arhitektura aplikacije

Aplikacija se sastoji od dvije glavne aktivnosti, *MainActivity* i *ApplicationInfoActivity*. Na slici 5.1. prikazan je dijagram toka koji opisuje radnje koje odrađuju razne funkcije uz pomoć klikova korisnika unutar sučelja. Pri pokretanju aplikacije prikazuje se ekran dobrodošlice s porukom „Pretražite potencijalno opasne aplikacije na Vašem uređaju“ i gumbom na koji se klikom pokreće pretraživanje. Prije pretraživanja učitava se lista opasnih aplikacija u .xls formatu prema kojoj se uspoređuju instalirane aplikacije na uređaju. Ako se instalirana aplikacija nalazi na listi opasnih aplikacija, dodaje se na listu pronađenih aplikacija, tada ide dodatna provjera u kojoj se promatra ima li otkrivena aplikacija neko od potencijalno opasnih dopuštenja poput slanja SMS poruka, uspostavljanja poziva ili pristupa imeniku. Ako aplikacija ispuni oba kriterija, dodaje se na listu otkrivenih aplikacija koja se po završetku pretrage prikaže korisniku.



Sl 5.1. Dijagram toka aktivnosti *MainActivity*

Druga aktivnost prikazana na slici 5.2., *ApplicationInfoActivity* omogućuje korisniku da klikom na aplikaciju sa liste dobije više informacija o dopuštenjima koja otkrivena aplikacija posjeduje. Na detaljnom prikazu nudi se opcija deinstaliranja aplikacije klikom na gumb „Deinstaliraj“ pri dnu ekrana. Nakon uspješne deinstalacije korisnik se vraća na prikaz preostalih otkrivenih aplikacija.



Sl 5.2. Dijagram toka aktivnosti *ApplicationInfoActivity*

5.3. Opis koda

Aplikacija se sastoji od ukupno četiri najbitnija dijela koda, dvije aktivnosti (engl. *Activity*) i dva prilagodnika (engl. *Adapter*). Aktivnosti su *MainActivity*, koja je zadužena za događanja u prvom prikazu kada se aplikacija pokrene i kada se izlistaju aplikacije nakon pritiska na gumb za pretraživanje i *ApplicationInfoActivity* kojom je određen drugi prikaz u kojem se vide sva dopuštenja odabrane aplikacije uz opciju za deinstalaciju. U prilagodniku *AppAdapter* je definiran prikaz pronađenih aplikacija, dok je u adapteru *PermissionAdapter* definiran prikaz svih sigurnosnih dopušenja koja se vide odabirom pojedine aplikacije. U oba slučaja se postavljaju pravila za općenito formatiranje prikaza, teksta, izgled ikonica i pozicija na zaslonu.

Aktivnost *MainActivity*, prikazana na slici 5.3., starta pri pokretanju aplikacije. Definirane funkcije omogućuju korisniku pretraživanje aplikacija koje bi mogle uzrokovati dodatne troškove na mobilnom uređaju. Nakon pretraživanja, prikazuje se popis aplikacija. Korisnik može kliknuti na svaku pojedinu aplikaciju i saznati više detalja o dopuštenjima koje aplikacija ima.

```
class MainActivity : AppCompatActivity(), AppAdapter.AppListener {

    private lateinit var bottomNavigationView: BottomNavigationView
    private lateinit var mRecyclerView: RecyclerView
    private lateinit var appAdapter: AppAdapter
    private lateinit var assetManager: AssetManager
    private lateinit var emptyView: RelativeLayout
    private lateinit var progressOverlay: View
    private var dangerousAppList: MutableList<String> = ArrayList()
    private var dangerousInstalledAppList: MutableList<ApplicationInfo> =
ArrayList()


    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        assetManager = assets

        initializeViews()
        readExcelFile()
        setBottomNavigationView()
        setRecyclerView()
        setEmptyView()

    }
```

Sl. 5.3. *Glavna aktivnost MainActivity*

Slika 5.4. prikazuje funkciju *initializeViews* koja inicijalizira prazni prikaz sa animacijom učitavanja.

```
    private fun initializeViews() {
        emptyView = find(R.id.emptyView)
        progressOverlay = find(R.id.progress_overlay)
    }
```

Sl. 5.4. *Funkcija za inicijalizaciju praznog prikaza i animacije učitavanja*

Slika 5.5. prikazuje funkciju *setBottonNavigationView* koja incijalizira donju navigacijsku traku

```
    private fun setBottomNavigationView() {
        bottomNavigationView =
find<BottomNavigationView>(R.id.navigation view scan).apply {
            setOnNavigationItemSelectedListener { item ->
                when (item.itemId) {
                    R.id.navigation scan -> {
                        scanDeviceWithLoadingAsync()
                        return@setOnNavigationItemSelectedListener true
                    }
                }
                false
            }
        }
    }
```

Sl. 5.5. *Funkcija za inicijalizaciju donje navigacijska trake*

Funkcija *setRecyclerView*, na slici 5.6., incijalizira *RecyclerView* prikaz sa modificiranim prilagodnikom. *RecyclerView* je vrlo fleksibilan i napredan *widget* koji omogućuje prikaz većih količina podataka na puno ljepši način od tradicionalne *ListView* opcije koja omogućuje ispis podataka uz minimalne modifikacije prikaza. RecyclerView također ima ugrađene funkcije koje omogućuju prikazivanje u obliku linearne liste (vertikalno ili horizontalno), mreže i mreže sa asimetričnim poljima. Za prikaz popisa aplikacija korišten je linearni vertikalni prikaz.

```
    private fun setRecyclerView() {
        mRecyclerView = find(R.id.recyclerView)
        mRecyclerView.layoutManager =
androidx.recyclerview.widget.LinearLayoutManager(this)
        appAdapter = AppAdapter(this, dangerousInstalledAppList, this)
        mRecyclerView.adapter = appAdapter
    }
```

Sl. 5.6. *Funkcija za inicijalizaciju RecyclerView prikaza*

Funkcija *setEmptyView*, na slici 5.7., stvara prikaz ako je popis potencijalno opasnih aplikacija prazan

```
    private fun setEmptyView() {
        if (dangerousInstalledAppList.isEmpty()) {
            emptyView.visibility = View.VISIBLE
        } else {
            emptyView.visibility = View.GONE
        }
    }
```

Sl. 5.7. *Funkcija setEmptyView za inicijalizaciju prikaza praznog popisa*

Funkcija *refreshRecyclerView, na slici 5.8.,* za osvježavanje prikaza potencijalno opasnih aplikacija

```
private fun refreshRecyclerView() {
    mRecyclerView.adapter?.notifyDataSetChanged()
}
```

Sl. 5.8. *Funkcija refreshRecyclerView*

Funkcija *readExcelFile*, na slici 5.9., učitava *app_block_list(27062021).xls* datoteku koja se nalazi u "assets" mapi i popunjava popis varijable *dangerousAppList*. Pri korištenju funkcije potrebno je definirati i list (engl. *Sheet*) iz kojeg se podaci povlače. Svi učitani podaci se zapisuju u *dangerousAppList* varijablu.

```
private fun readExcelFile() {

    val excelFile = assetManager.open("app_block_list(27062021).xls")
    val workbook = HSSFWorkbook(excelFile)
    val sheet = workbook.getSheet("app_block_list(27062021)")
    val rows = sheet.iterator()

    while (rows.hasNext()) {
        val currentRow = rows.next()

        dangerousAppList.add(currentRow.getCell(0).toString())
    }
    workbook.close()
    excelFile.close()
}
```

Sl. 5.9. *Funkcija readExcelFile*

Funkcija *getInstalledApps*, na slici 5.10., prikuplja instalirane aplikacije i popunjava popis varijable *dangerousInstalledAppList* sa informacijama o ne-sistemskim aplikacijama čije je ime paketa na popisu *dangerousAppList* i ima jedno od sljedećih android dopuštenja: *CALL_LOG, PHONE, SMS*. Aplikacije u *dangerousInstalledAppList* varijabli se sortiraju prema abecednom redu.

```
    private fun getInstalledApps() {
        val packageManager = packageManager
        val packages =
packageManager.getInstalledApplications(PackageManager.GET META DATA)

        for (applicationInfo in packages) {
            if (applicationInfo.packageName in dangerousAppList) {
                if (!isSystemApp(applicationInfo)) {
                    val packageInfo = packageManager.getPackageInfo(
                        applicationInfo.packageName,
                        PackageManager.GET PERMISSIONS
                    )

                    if (packageInfo.requestedPermissions != null) {
                        for (permission in
packageInfo.requestedPermissions) {
                            if (applicationInfo in
dangerousInstalledAppList) {
                                break
                            } else if (permission in
resources.getStringArray(R.array.dangerous permissions list)) {

dangerousInstalledAppList.add(applicationInfo)
                            }
                        }
                    }
                }
            }
        }
        dangerousInstalledAppList.sortBy {
it.loadLabel(this.packageManager).toString() }
    }
```

Sl. 5.10. *Funkcija getInstalledApps*

Funkcija *isSystemApp*, na slici 5.11., provjerava je li aplikacija instalirana na sistemskoj particiji

```
    private fun isSystemApp(applicationInfo: ApplicationInfo): Boolean {
        var isSystemApp = false
        if (applicationInfo.flags and ApplicationInfo.FLAG SYSTEM != 0) {
            if (applicationInfo.flags and
ApplicationInfo.FLAG UPDATED SYSTEM APP == 0) {
                isSystemApp = true
            }
        }
        return isSystemApp
    }
```

Sl. 5.11. *Funkcija isSystemApp*

Funkcija scanDeviceWithLoadingAsync, na slici 5.12., metoda se poziva kako bi osvježila prikaz ako korisnik deinstalira neku aplikaciju u prethodnoj aktivnosti. U slučaju da su ostale još neke potencijalno opasne aplikacije prikazat će novu listu bez prethodno uklonjene aplikacije, a ako je

upravo ta aplikacija bila posljednja i na mobilnom uređaju nema više niti jedna potencijalno opasna aplikacija, ispisati će se poruka „Vaš uređaj nema instaliranih potencijalno opasnih aplikacija :)“.

```
    private fun scanDeviceWithLoadingAsync() {

        mRecyclerView.stopScroll()
        dangerousInstalledAppList.clear()
        refreshRecyclerView()
        setEmptyView()
        AndroidUtils.animateView(progressOverlay, View.VISIBLE, 0.4f, 200)

        doAsync {
            getInstalledApps()
            onComplete {
                AndroidUtils.animateView(progressOverlay, View.GONE, 0f,
200)
                refreshRecyclerView()
                setEmptyView()
                if (dangerousInstalledAppList.isEmpty()) {
                    AndroidUtils.showToast(
                        this@MainActivity,
                        R.string.toast message no dangerous apps
                    )
                }
            }
        }
    }
```

Sl. 5.12. *Funkcija scanDeviceWithLoadingAsync*

Funkcija *onAppClicked*, na slici 5.13., se poziva kada se klikne na bilo koju od aplikacija u *RecyclerView* prikazu.

```
    override fun onAppClicked(applicationInfo: ApplicationInfo) {
        val intent = ApplicationInfoActivity.newIntent(this)
        intent.putExtra(ApplicationInfoActivity.APPLICATION_INFO_KEY,
applicationInfo)
        startActivityForResult(
            intent,
            ApplicationInfoActivity.APPLICATION_INFO_ACTIVITY_REQUEST_CODE
        )
    }
```

Sl. 5.13. *Funkcija onAppClicked*

Funkcija *onActivityResult*, na slici 5.14., se učitava pri povratku sa prikaza svih dopuštenja pojedine aplikacije na početnu listu sa pronađenim potencijalno opasnim aplikacijama te osvježava prikaz kako bi se ne bi prikazivala aplikacija koja je u prethodnom koraku uklonjena s uređaja.

```
    override fun onActivityResult(requestCode: Int, resultCode: Int, data:
Intent?) {
        super.onActivityResult(requestCode, resultCode, data)
        if (requestCode ==
ApplicationInfoActivity.APPLICATION_INFO_ACTIVITY_REQUEST_CODE) {
            scanDeviceWithLoadingAsync()
        }
    }

}
```

Sl. 5.14. *Funkcija onActivityResult*

Aktivnost *ApplicationInfoActity*, na slici 5.15., se pokreće iz aktivnosti *MainActivity* klikom na neku od aplikacija s liste. Prikazuje ima li aplikacija dopuštenja iz grupa: *CALL_LOG, PHONE, SMS* te detaljan opis svakog od dopuštenja. Dodatno omogućuje korisniku deinstalaciju svake pojedine aplikacije.

```
class ApplicationInfoActivity : AppCompatActivity() {

    private lateinit var titleTextView: TextView
    private lateinit var mRecyclerView: RecyclerView
    private lateinit var backImageButton: ImageButton
    private lateinit var bottomNavigationView: BottomNavigationView
    private lateinit var permissionAdapter: PermissionAdapter
    private var permissionList: MutableList<String> = ArrayList()

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_application_info)

        getPermissions()
        setTitle()
        setBackButton()
        setRecyclerView()
        setBottomNavigationView()
    }
```

Sl. 5.15. *Aktivnost ApplicationInfoActivity*

Funkcija *setTitle*, na slici 5.16., inicijalizira ime odabrane aplikacije

```
    private fun setTitle() {
        titleTextView = find(R.id.titleTextView)
        titleTextView.text = getAppInfo()?.loadLabel(packageManager)
    }
```

Sl. 5.16. *Funkcija setTitle*

Funkcija *setBackButton*, na slici 5.17, inicijalizira gumb za povratak na prethodni prikaz

```
    private fun setBackButton() {
        backImageButton = find<ImageButton>(R.id.backImageButton).apply {
            setOnClickListener {
                onBackPressed()
            }
        }
    }
```

Sl. 5.17. *Funkcija setBackButton*

Funkcija *setRecyclerView*, na slici 5.18., inicijalizira *RecyclerView* prikaz sa modificiranim prilagodnikom

```
    private fun setRecyclerView() {
        mRecyclerView = find(R.id.recyclerView)
        mRecyclerView.layoutManager =
androidx.recyclerview.widget.LinearLayoutManager(this)
        permissionAdapter = PermissionAdapter(this, permissionList)
        mRecyclerView.adapter = permissionAdapter
    }
```

Sl. 5.18. *Funkcija setRecyclerView*

Funkcija *setBottomNavigationView*, na slici 5.19., inicijalizira donju navigacijsku traku

```
    private fun setBottomNavigationView() {
        bottomNavigationView =
find<BottomNavigationView>(R.id.navigation view uninstall).apply {
            setOnNavigationItemSelectedListener { item ->
                when (item.itemId) {
                    R.id.navigation uninstall -> {
                        uninstallApplication(getAppInfo()?.packageName)
                        return@setOnNavigationItemSelectedListener true
                    }
                }
                false
            }
        }
    }
```

Sl. 5.19. *Funkcija setBottomNavigationView*

Funkcija *getAppInfo*, na slici 5.20., dohvaća informacije o aplikaciji odabranoj u prethodnoj aktivnosti

```
    private fun getAppInfo(): ApplicationInfo? =
        intent.getParcelableExtra(APPLICATION_INFO_KEY)
```

Sl. 5.20. *Funkcija getAppInfo*

Funkcija *getPermissions* , na slici 5.21., dohvaća dopuštenja koja odabrana aplikacija posjeduje

```
    private fun getPermissions() {
        val packageManager = packageManager
        val packageInfo =

packageManager.getPackageInfo(getAppInfo()?.packageName.toString(),
PackageManager.GET_PERMISSIONS)

        if (packageInfo.requestedPermissions != null) {
            for (permission in packageInfo.requestedPermissions) {
                if (permission in
resources.getStringArray(R.array.dangerous_permissions_list)) {
                    permissionList.add(permission)
                }
            }
        }
    }
```

Sl. 5.21. *Funkcija getPermissions*

Funkcija *uninstallApplication*, na slici 5.22., omogućuje deinstalaciju aplikacija s uređaja

```
    private fun uninstallApplication(packageName: String?) {
        val intent = Intent(Intent.ACTION_UNINSTALL_PACKAGE)
        intent.data = Uri.parse("package:$packageName")
        intent.putExtra(Intent.EXTRA_RETURN_RESULT, true)
        startActivityForResult(intent, UNINSTALL_REQUEST_CODE)
    }
```

Sl. 5.22. *Funkcija uninstallApplication*

Funkcija *onActivityResult*, na slici 5.23., se poziva nakon što sustav vrati poruku o uspješnoj ili neuspješnoj deinstalaciji aplikacije te ovisno o uspješnosti uklanjanja prikazuje prikladnu poruku. Ako korisnik stisne gumb za povratak prikazati će se poruka „Deinstalacija prekinuta“, a ako deinstalacija iz nekog razloga ne bude uspješna prikazati će se poruka „Nešto je pošlo po krivu, molim pokušajte ponovno.“.

```
    override fun onActivityResult(requestCode: Int, resultCode: Int, data:
Intent?) {
        super.onActivityResult(requestCode, resultCode, data)

        if (requestCode == UNINSTALL_REQUEST_CODE) {
            when (resultCode) {
                RESULT_OK -> onBackPressed()
                RESULT_CANCELED -> AndroidUtils.showToast(
                    this,
                    R.string.toast_message_action_cancelled
                )
                RESULT_FIRST_USER -> AndroidUtils.showToast(
                    this,
                    R.string.toast_message_something_went_wrong
                )
                else -> AndroidUtils.showToast(this,
R.string.toast_message_something_went_wrong)
            }
        }
    }
```

Sl. 5.23. *Funkcija onActivityResult*

Prilagodnik *AppAdapter*, na slici 5.24., korišten za modificiranje *RecyclerView* prikaza svih aplikacija. Definirano je povlačenje izgleda ikone i imena aplikacije za svaku stavku na listi.

```
class AppAdapter(
    private val context: Context,
    private var appList: List<ApplicationInfo>,
    private var appListener: AppListener
) :
androidx.recyclerview.widget.RecyclerView.Adapter<AppAdapter.ViewHolder>()
{

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
ViewHolder {
        val view = LayoutInflater.from(parent.context)
            .inflate(R.layout.adapter_dangerous_apps, parent, false)
        return ViewHolder(view)
    }

    override fun onBindViewHolder(holder: ViewHolder, position: Int) {
        val item: ApplicationInfo = appList[position]


holder.appIconImageView?.setImageDrawable(item.loadIcon(context.packageMana
ger))
        holder.appNameTextView?.text =
item.loadLabel(context.packageManager)

        holder.itemView.setOnClickListener {
            appListener.onAppClicked(item)
        }
    }

    override fun getItemCount(): Int {
        return appList.size
    }

    class ViewHolder(itemView: View) :
        androidx.recyclerview.widget.RecyclerView.ViewHolder(itemView) {
        val appIconImageView: ImageView? = itemView.app_icon_image_view
        val appNameTextView: TextView? = itemView.app_name_text_view
    }
```

Sl. 5.24. *Prilagodnik AppAdapter*

Sučelje *interface*, na slici 5.25., za svaku stavku sa liste aplikacija. Prilikom klika na aplikaciju na listi vraća informacije o aplikaciji.

```
    interface AppListener {



        fun onAppClicked(applicationInfo: ApplicationInfo)
    }
```

Sl. 5.25. *Sučelje AppListener*

Prilagodnik *PermissionAdapter*, na slici 5.26., korišten za modificiranje *RecyclerView* prikaza koji sadrži sva dopuštenja odabrane aplikacije.

```
class PermissionAdapter(
    private val context: Context,
    private var permissionList: List<String>
) :
androidx.recyclerview.widget.RecyclerView.Adapter<PermissionAdapter.ViewHol
der>() {

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
ViewHolder {
        val view = LayoutInflater.from(parent.context)
            .inflate(R.layout.adapter permission info, parent, false)
        return ViewHolder(view)
    }

    override fun onBindViewHolder(holder: ViewHolder, position: Int) {

        val item: String = permissionList[position]
        val packageManager = context.packageManager
        val permissionInfo = packageManager.getPermissionInfo(item,
PackageManager.GET META DATA)

        holder.appIconImageView?.setImageDrawable(
            ContextCompat.getDrawable(context,
getPermissionResourceId(item)))

        holder.appNameTextView?.text =

permissionInfo.loadLabel(packageManager).toString().capitalize()

        holder.appPermissionTextView?.text =

permissionInfo.loadDescription(packageManager).toString().capitalize()

    }

    override fun getItemCount(): Int {
        return permissionList.size
    }

    private fun getPermissionResourceId (item: String ): Int {
        var resourceId = 0

        when (item) {
            "android.permission.READ_CALL_LOG" -> {
                resourceId = R.drawable.ic outline phone 24
```

Sl. 5.26. *Prilagodnik PermissionAdapter*

Petlja, na slici 5.27., koja provjerava koje od zadanih dopuštenja aplikacija posjeduje. Sva tražena dopuštenja su iz grupa *CALL_LOG*, *PHONE i SMS.*

```
        when (item) {
            "android.permission.READ_CALL_LOG" -> {
                resourceId = R.drawable.ic_outline_phone_24
            }
            "android.permission.WRITE_CALL_LOG" -> {
                resourceId = R.drawable.ic_outline_phone_24
            }
            "android.permission.PROCESS_OUTGOING_CALLS" -> {
                resourceId = R.drawable.ic_outline_phone_24
            }
            "android.permission.READ_PHONE_STATE" -> {
                resourceId = R.drawable.ic_outline_phone_24
            }
            "android.permission.READ_PHONE_NUMBERS" -> {
                resourceId = R.drawable.ic_outline_contacts_24
            }
            "android.permission.CALL_PHONE" -> {
                resourceId = R.drawable.ic_outline_phone_24
            }
            "android.permission.ANSWER_PHONE_CALLS" -> {
                resourceId = R.drawable.ic_outline_phone_24
            }
            "android.permission.ADD_VOICEMAIL" -> {
                resourceId = R.drawable.ic_outline_voicemail_24
            }
            "android.permission.USE_SIP" -> {
                resourceId = R.drawable.ic_outline_dialer_sip_24
            }
            "android.permission.SEND_SMS" -> {
                resourceId = R.drawable.ic_outline_chat_bubble_outline_24
            }
            "android.permission.RECEIVE_SMS" -> {
                resourceId = R.drawable.ic_outline_chat_bubble_outline_24
            }
            "android.permission.READ_SMS" -> {
                resourceId = R.drawable.ic_outline_chat_bubble_outline_24
            }
            "android.permission.RECEIVE_WAP_PUSH" -> {
                resourceId = R.drawable.ic_outline_chat_bubble_outline_24
            }
            "android.permission.RECEIVE_MMS" -> {
                resourceId = R.drawable.ic_outline_mms_24
            }
        }

        return resourceId
    }
```

Sl. 5.27. *Petlja za provjeru dopuštenja aplikacije*

U klasi *ViewHolder*, na slici 5.28., je definirano prikazivanje ikone dopuštenja, naziva dopuštenja i detaljan opis dopuštenja prema Googleovim definicijama.

```
    class ViewHolder(itemView: View) :
        androidx.recyclerview.widget.RecyclerView.ViewHolder(itemView) {
        val appIconImageView: ImageView? = itemView.app_icon_image_view
        val appNameTextView: TextView? = itemView.app_name_text_view
        val appPermissionTextView: TextView? =
itemView.permission_description_text_view
    }
}
```

Sl. 5.28. *Petlja za provjeru dopuštenja aplikacije*

5.4. Korištenje aplikacije

Zadatak je napraviti aplikaciju koja bi otkrivala potencijalno opasne aplikacije instalirane na mobilnom uređaju te prikazala sigurnosna dopuštenja Android sustava zbog kojih se ta aplikacija smatra opasnom. Aplikacija je pisana u programskom jeziku otvorenog koda Kotlin, koji je izvrsna alternativa Javi. Kotlin je razvila kompanija za razvoj programskih rješenja Jetbrains, koja je također velikim dijelom zaslužna i za razvoj Android Studia, službenog integriranog razvojnog sučelja za Android.

Prevare pomoću aplikacija se najčešće događaju zbog nepažnje krajnjih korisnika. Često pri instalaciji umjesto čitanja upozorenja, u želji da se što prije završi proces, korisnik samo tapka na gumb „Dalje“ ili „Završi“ što rezultira u davanju raznih sigurnosnih dopuštenja samoj aplikaciji koja kasnije može samostalno slati SMS poruke, uspostavljati pozive, pristupati kontaktima u imeniku i slično. Otkrivanje takvih aplikacija izvedeno je prema dva kriterija.

Prvi kriterij je uspoređivanje naziva paketa aplikacije, oblika „com.android.myapplication“, koji sadrži svaka aplikacija sa posebnom listom ranije otkrivenih malicioznih aplikacija koja se učitava pri svakom pokretanju aplikacije. Ova lista je kreirana na temelju suradnje nekoliko mobilnih operatera i regulatora kao što su belgijski Proximus, kenijski Safaricom, norveški Strex i južnoafrički regulator WASPA (Udruženje pružatelja usluga bežičnih aplikacija) s ciljem da se spriječe prevare putem mobilnih aplikacija. Lista sadrži opsežan popis aplikacija koje su bile povezane sa problematičnim naplatama širom svijeta. Drugi kriterij je uspoređivanje svih dopuštenja koje pronađene aplikacije imaju sa službenom listom potencijalno opasnih dopuštenja koju pruža Google.

Bitno je napomenuti da će se kao rezultat pretrage pojaviti i određene aplikacije koje neće nužno uzrokovati štetu krajnjem korisniku nego su im samo omogućena određena dopuštenja, npr. popularna aplikacija Whatsapp će se prikazati na listi jer da bi ju korisnik mogao koristiti mora joj omogućiti uspostavljanje poziva i slanje poruka. Korisniku se uz imena otkrivenih aplikacija

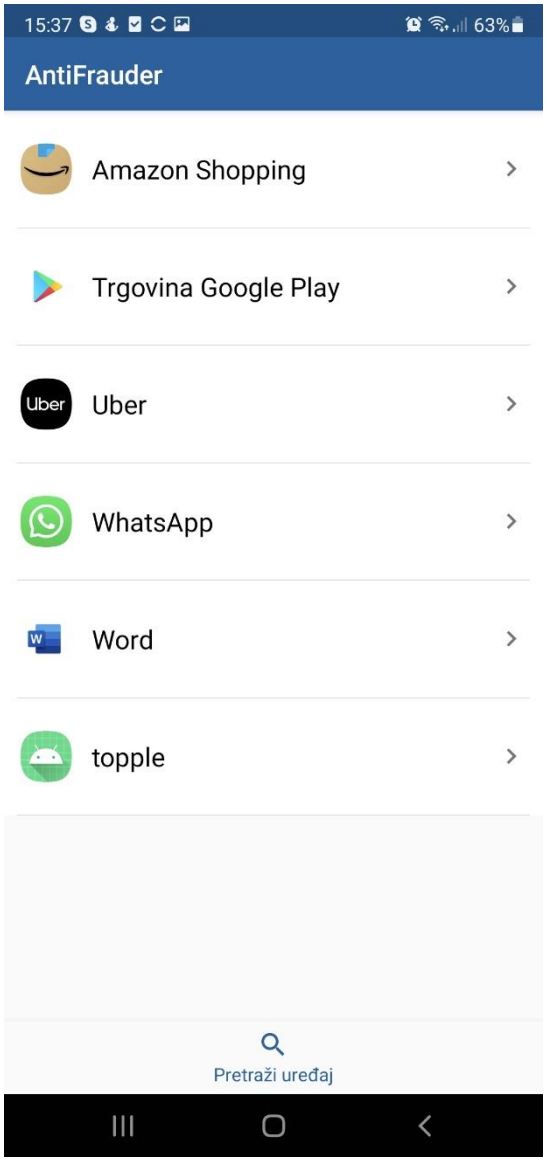
pokažu i sva dopuštenja koje pojedine aplikacije imaju te on sam odlučuje hoće li ih ukloniti ili ostaviti na mobilnom uređaju.

Aplikacija je nazvana AntiFrauder, a početni zaslon prikazan na slici 5.29. daje kratku instrukciju da se pokrene pretraživanje potencijalno opasnih aplikacija na mobilnom uređaju klikom na tipku „Pretraži uređaj“ pri dnu ekrana.

Sl. 5.29. Početni zaslon Antifrauder

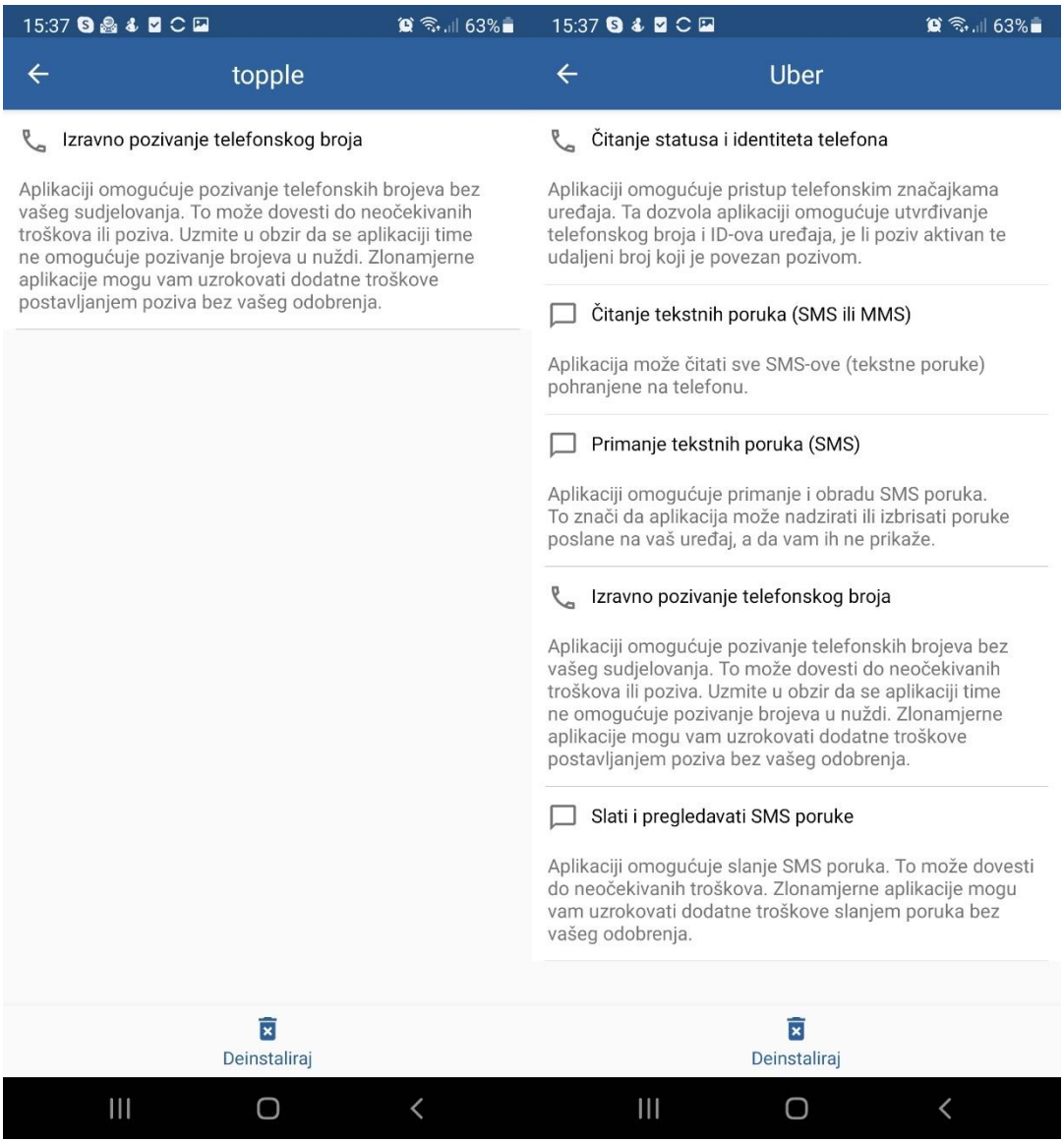
U pozadini se radi usporedba prema ranije navedenim kriterijima i prikazuju se sve aplikacije koje su pronađene. U svrhu dokazivanja da aplikacija ispravno detektira potencijalno opasne aplikacije prema zadanim kriterijima, napravljena je i aplikacija „topple“ s nazivom paketa „biz.pegboard.topple“ koji se nalazi na listi zloćudnih aplikacija i dopuštenjem da uspostavlja

pozive. Sama aplikacija „topple“ nema nikakvu funkciju, ali je modificirani naziv paketa i jedno dopuštenje dovoljno da imitira malicioznu aplikaciju.



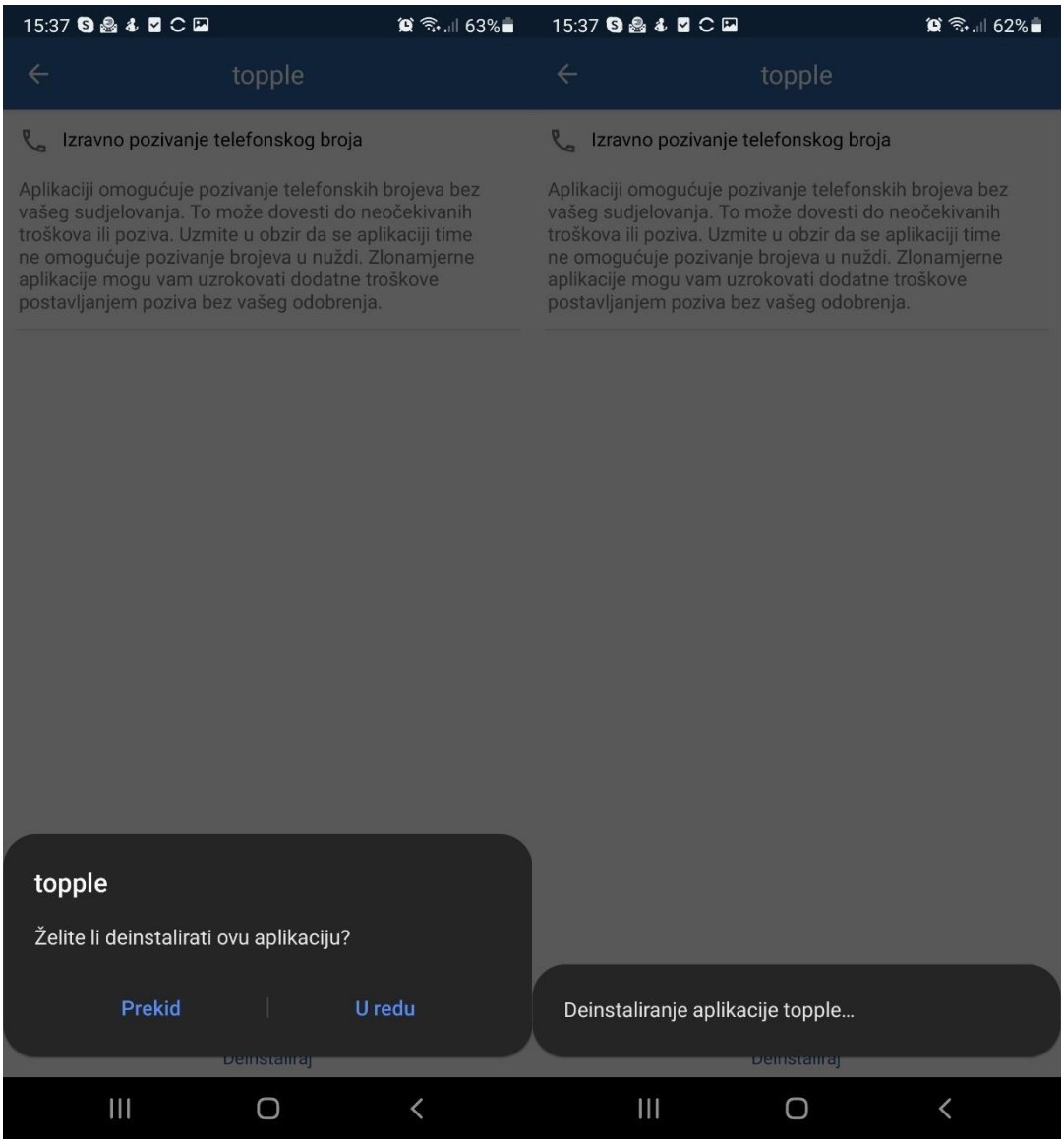
Sl. 5.30. Prikaz svih otkrivenih aplikacija

Klikom na bilo koju aplikaciju s liste prikazat će se popis svih dopuštenja koje aplikacija ima, vidljivo na slici 5.30., a koja bi mogla uzrokovati dodatne troškove. Također korisnik na ovom prikazu ima ponuđenu opciju deinstaliranja odabrane aplikacije.



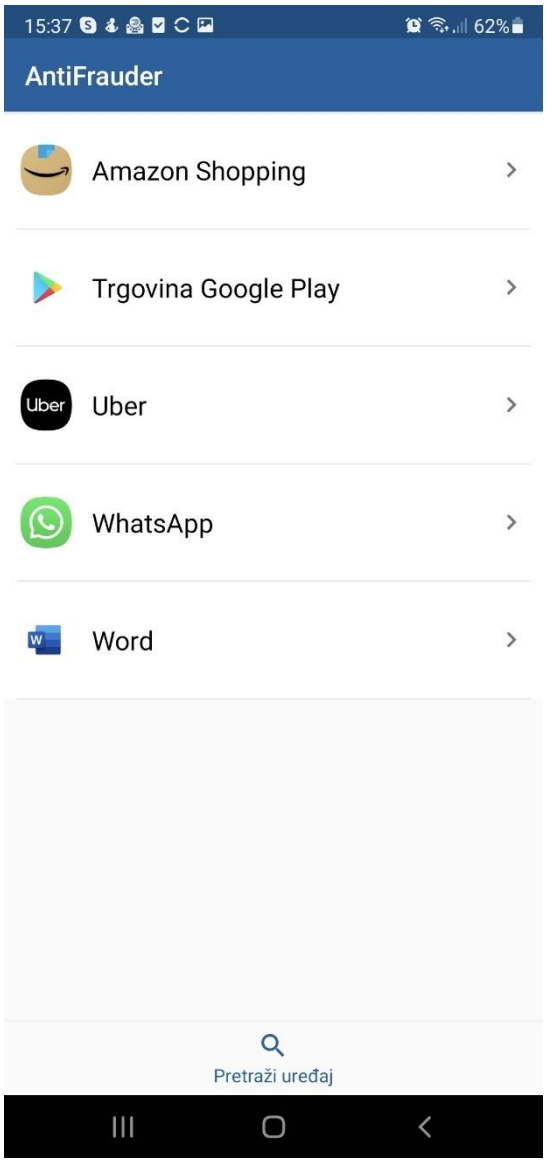
Sl. 5.31. Dopuštenja aplikacija „topple“ i „Uber“

Pri dnu ekrana se nalazi gumb koji pokreće deinstalaciju aplikacije za koju su trenutno prikazana dopuštenja, kao što je prikazano na slici 5.31.. Klikom na gumb „Deinstaliraj“ pojavi se upit od strane Android sustava koji zahtjeva dodatnu potvrdu za uklanjanje. U slučaju potvrdnog odabira, kroz nekoliko sekundi aplikacija je deinstalirana.



Sl. 5.32. Proces deinstalacije aplikacije „topple“

Nakon što je odabrana aplikacija deinstalirana, korisniku se prikazuje prethodni ekran sa otkrivenim aplikacijama, kao na slici 5.32., uz izuzetak uklonjene aplikacije.



Sl. 5.33. Prikaz otkrivenih aplikacija nakon deinstalacije aplikacije „topple“

Slika 5.33. prikazuje listu nakon deinstalacije aplikacije „topple“. Korisnik može nastaviti pregledavati otkrivene aplikacije i pokrenuti proces deinstalacije za svaku od njih ako to želi. U slučaju da na mobilnom uređaju nema više niti jedne otkrivene aplikacije, prikazati će se poruka „Vaš uređaj nema instaliranih potencijalno opasnih aplikacija :)“.

6. ZAKLJUČAK

Razvijena aplikacija omogućuje korisniku da provjeri postoje li potencijalno opasne aplikacije na njegovom mobilnom uređaju. Podržani su Android uređaji do verzije operativnog sustava 11. Potencijalno opasne aplikacije se detektiraju na temelju dva kriterija. Prvi kriterij je lista poznatih malicioznih aplikacija koju dijeli južnoafrički regulator WASPA, a nastala je kroz suradnju više mobilnih operatera i kompanija koje se bave sprječavanjem prevara u sustavima mobilne naplate. Drugi kriterij su dopuštenja koje je korisnik omogućio aplikaciji. U slučaju da aplikacija odgovara zadanim kriterijima, prikazat će se korisniku na listi potencijalno opasnih aplikacija uz dodatne informacije koje će korisniku dati uvid u dopuštenja koja aplikacija ima i što točno ta dopuštenja znače. Korisnik zatim ima mogućnost obrisati aplikaciju sa liste ukoliko smatra da je uistinu opasna za njegov mobilni uređaj. Manjkavost ovog rješenja je lista malicioznih aplikacija koja se mora redoviti obnavljati kako se pojavljuju nove opasne aplikacije. Druga manjkavost je što će se na listi potencijalno opasnih aplikacija pojaviti i neke aplikacije koje neće uzrokovati probleme korisniku, ali su tamo zbog dopuštenja koja imaju. Iz ovog razloga je bitno raditi na edukaciji krajnjih korisnika o sigurnosti u sustavima mobilne naplate kako bi mogli bolje razumjeti na koji su način ugroženi od strane aplikacija koje instaliraju na svoje uređaje. Na kraju, poboljšanje rješenja bi također bila i verzija aplikacije za Apple uređaje pogonjene iOS operativnim sustavom kako bi se obuhvatili gotovo svi mobilni uređaji na tržištu.

LITERATURA

[1] Večernji.hr, „Upozorenje HAKOMA: Ne šaljite SMS-ove dobijete li ovu informaciju“, https://www.vecernji.hr/biznis/hakom-upozorenje-prevara-korisnika-sms-aplikacija-1226318 (datum pristupa 04.07.2021.)

[2] TxtNation, „Increase converion rates with Payforit mobile payments in the UK.“, https://www.txtnation.com/mobile-billing/payforit/, (datum pristupa 04.07.2021.)

[3] Netzwelt, „Trenutna upozorenja na prevare“, https://www.netzwelt.de/betrugswarnungen/index.html, (datum pristupa 04.07.2021.)

[4] BBC, „Chinese phones with built-in malware sold in Africa“ https://www.bbc.com/news/technology-53903436, (datum pristupa 04.07.2021.)

[5] HAKOM, „Rješenje inspekcijskog nadzora: DIMOCO“ https://www.hakom.hr/UserDocsImages/2018/odluke_rjesenja_presude/Rjesenje-Inspekcijski%20DIMOCO-180319.pdf, (datum pristupa 04.07.2021.)

[6] Dnevnik.hr, „HAKOM poslao upozorenje: SMS poruka mogla bi vam donijeti neželjene troškove“, https://dnevnik.hr/vijesti/hrvatska/hakom-zbog-moguce-prevare-s-aplikacijom-za-brisanje-virusa-pokrece-nadzor-nad-tvrtkom-dimoco---506744.html, (datum pristupa 04.07.2021.)

[7] Evina, „Fraud Report Jan. – Jun. 2021 - Europe“ https://www.evina.com/wp-content/uploads/2021/06/EN-Europe-Q1-Q2-Evina-Master-Fraud-Report.pdf, (datum pristupa 04.07.2021)

[8] Net.hr, „Čuvajte se testova inteligencije!“, https://net.hr/danas/hrvatska/cuvajte-se-testova-inteligencije-7ea608ee-b1d1-11eb-8e74-0242ac140050, (datum pristupa 04.07.2021.)

[9] Vesti Online, „"IQ test srpskog naroda" - prevara i dalje neometano traje“, https://arhiva.vesti-online.com/Vesti/Hronika/19821/IQ-test-srpskog-naroda--prevara-i-dalje-neometano-traje (datum pristupa 04.07.2021.)

[10] Check Point Software Technologies, „ExpensiveWall: A Dangerous ‘Packed’ Malware On Google Play That Will Hit Your Wallet“ https://blog.checkpoint.com/2017/09/14/expensivewall-dangerous-packed-malware-google-play-will-hit-wallet/ (datum pristupa 04.07.2021.)

[11] Web Fundamentals, „Content Security Policy“, https://developers.google.com/web/fundamentals/security/csp/#policy_applies_to_a_wide_variety_of_resources, (datum pristupa 04.07.2021.)

[12] Opticks Security, „The 2020 annual ad fraud report“, https://resources.optickssecurity.com/the-2020-annual-ad-fraud-report, (datum pristupa 04.07.2021.)

[13] Portswigger, „Clickjacking (UI redressing)“, https://portswigger.net/web-security/clickjacking (datum pristupa 04.07.2021.)

[14] Empello, „2020 VAS/DCB Market Round-up“, https://www.empello.com/read-empellos-2020-vas-dcb-round-up-report/ (datum pristupa 26.07.2021.)

SAŽETAK

Zadatak rada je kreirati rješenje za sprječavanje jedne od opisanih metoda prevare u sustavima mobilne naplate. Napravljena je aplikacija za Android operativni sustav, u programskom jeziku Kotlin, koja otkriva potencijalno opasne instalirane aplikacije na mobilnom uređaju. Otkrivanje se vrši prema dva kriterija, prvi je pretraživanje postoji li instalirana aplikacija na listi poznatih malicioznih aplikacija, drugi provjerava ima li aplikacija neka od dopuštenja za radnje koje bi mogle uzrokovati dodatne troškove. Korisniku se klikom na pronađenu aplikaciju na listi otvara dodatni prikaz na kojem može vidjeti detaljan opis dopuštenja koje aplikacija ima te ako on smatra da je u opasnosti, aplikaciju može jednostavno deinstalirati klikom na gumb u dnu ekrana. Kada se deinstalacija završi, ponovno se vraća na prikaz svih potencijalno opasnih aplikacija, ili ako više nema niti jedne takve aplikacije na uređaju, prikaže se poruka „Vaš uređaj nema instaliranih potencijalno opasnih aplikacija :)“.

Ključne riječi: aplikacija, mobilna naplata, prevare, Kotlin

ABSTRACT

PREVENTING FRAUD IN MOBILE PAYMENT SYSTEMS

Objective of this thesis is to create a solution for preventing one of the described fraud methods in mobile payment systems. Application for the Android operating system was created, in Kotlin programming language, which is able to discover potentially dangerous installed applications on the mobile device. The discovery is based on two criteria, first is to check if the installed application exists within an imported list of known malicious applications, the second one checks if the application has some of the permissions which are able to cause additional expenses to the user. By clicking on the discovered listed application, the user is presented with an additional view where he can examine all the permissions this application has in detail, and in case he considers it potentially harmful, the application can be simply uninstalled by clicking on the button at the bottom of the screen. Once the uninstallation process is complete, the view of all potentially dangerous applications is presented once again, or in case that there are not more dangerous applications on the mobile device, the user will be shown a message "Your device has got no potentially dangerous applications installed :)"

Keywords: application, mobile payment, fraud, Kotlin

ŽIVOTOPIS

Damir Brođanac rođen je 12. veljače 1991. godine u Osijeku. Osnovnoškolsko školovanje je završio u Osnovnoj školi Dalj u Dalju nakon čega upisuje Elektrotehničku i prometnu školu u Osijeku gdje je i maturirao. Nakon završetka srednje škole upisuje redovni preddiplomski studij računarstva na Elektrotehničkom fakultetu u Osijeku, na drugoj godini se prebacuje na preddiplomski studij elektrotehnike. Nakon završetka preddiplomskog studija 2012. godine redovno upisuje diplomski studij elektrotehnike, smjer komunikacije i informatika, na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija. Trenutno je zaposlen u engleskoj tvrtki MessageCloud gdje radi na razvoju poslovanja u području mobilnog plaćanja.

PRILOZI (NA CD-U)

Prilog 1. Pisani oblik diplomskog rada

Prilog 2. Rješenje zadatka u obliku programskog koda

Prilog 3. Instalacijska datoteka aplikacije Antifrauder

Prilog 4. Instalacijska datoteka aplikacije topple