



UNIVERZITET U SARAJEVU
ELEKTROTEHNIČKI FAKULTET
ODSJEK ZA RAČUNARSTVO I INFORMATIKU

Aplikacija za evidentiranje prisustva

ZAVRŠNI RAD
- DRUGI CIKLUS STUDIJA -

Autor:
Malik Koljenović, BA Ing. IT

Mentor:
Vanr. prof. dr Saša Mrdović, dipl.ing.el.

Sarajevo,
februar 2019.

Abstract This thesis addresses the problem of large scale electronic attendance taking in university setting by presenting an Android based attendance taking application, based on immutable and non repudiable location proofs backed by RSA cryptography, utilizing NFC and HCE for ease of use, emulating NFC Forum Tag Type 4 it is also compatible with existing reader infrastructures. It also presents a general overview of the utilized technologies and select implementation details.

Apstrakt Ova teza tretira problem masovnog elektronskog bilježenja prisustva u univerzitetском okruženju izradom prijedloga aplikacija bazirane na Android platformi korištenjem neizmjenjivih i neporecivih vremensko-lokacijskih dokaza osiguranih korištenjem RSA kriptografije, te NFC i HCE tehnologija u cilju jednostavnosti upotrebe; emulirajući NFC Forum Tag Tip 4 kompatibilna je sa postojećim infrastrukturama čitača. Dat je i opšti pregled korištenih tehnologija i izdvojenih implementacijskih detalja.

MSC Primary 68P25; Secondary 94A60;

Keywords: NFC - near-field communication, HCE - host card emulation, security, Android, attendance, RSA, cryptography, NDEF, NTAG, geolocation, location proofs

Elektrotehnički fakultet, Univerzitet u Sarajevu
Odsjek za računarstvo i informatiku
Vanr. prof. dr Saša Mrdović, dipl.ing.el.
Sarajevo, decembar 2016.

Postavka zadatka završnog rada II ciklusa:

Aplikacija za evidentiranje prisustva

Potrebno je napraviti Android aplikaciju koja omogućava evidentiranje prisustva upotrebom NFC tehnologije. Aplikacija treba da bude jednostavna za upotrebu i zaštićena od zloupotreba.

U okruženju u kom se većina komunikacija odvija elektronski za očekivati je da se i evidentiranje prisustva može raditi na ovaj način. Međutim, postoje otvorena pitanja pogodnosti i sigurnosti elektronskog evidentiranja. NFC može biti osnova za siguran i jednostavan sistem. Kako savremeni mobilni uređaji uglavnom imaju NFC oni mogu biti iskorišteni kao sredstvo prijavljivanja i vođenja evidencije bez potrebe za dodatnim karticama i čitačima.

U radu je potrebno objasniti šta se podrazumjeva pod pojmom evidencija prisustva i koja su otvorena pitanja elektronskog vođenja ove evidencije. Potrebno je objasniti šta je NFC i kako radi. Potrebno je teoretski objasniti kako je moguće napraviti siguran elektronski sistem evidentiranja prisustva zasnovan na NFC koji je lak za upotrebu. U sklopu rada potrebno je napraviti praktičnu izvedbu sistema koji omogućava korisnicima koji imaju mobilne uređaje sa NFC da se pomoću njih registruju i potvrde prisustvo. Ovaj sistem treba biti zaštićen od zloupotreba. Prokomentarisati iskustva stečena tokom praktične realizacije i dati savjete za buduće izvedbe.

Polazna literatura:

- [1] V. Coskun, K. Ok, B. Ozdenizci, "Professional NFC Application Development for Android", Wrox, 2013
- [2] T. Igoe, D. Coleman, B. Jepson, "Beginning NFC: Near Field Communication with Arduino, Android, and PhoneGap", O'Reilly Media, 2014
- [3] J. Annuzzi Jr., L. Darcey, S. Conder, "Introduction to Android Application Development: Android Essentials", 4. izdanje, Addison-Wesley Professional, 2013.
- [4] Ross Anderson, "Security Engineering", 2nd edition, Wiley, 2008.
- [5] Bill Phillips, Brian Hardy, "Android Programming: The Big Nerd Ranch Guide", Big Nerd Ranch Guides, 2013.
- [6] V. Coskun, K. Ok, B. Ozdenizci, "Near Field Communication (NFC): From Theory to Practice", Wiley, 2012

Potpisi članova Komisije za ocjenu i odbranu završnog rada II ciklusa

Red. prof. dr Dženana Đonko, dipl.el.ing,
predsjednica komisije

Vanr. prof. dr Saša Mrdović, dipl.ing.el.,
mentor i član komisije

Vanr. prof. dr Samir Omanović, dipl.el.ing,
član komisije

Izjava o autentičnosti radova

Završni rad II ciklusa studija

Ime i prezime: Malik Koljenović

Naslov rada: Aplikacija za evidentiranje prisustva

Vrsta rada: Završni rad drugog ciklusa studija

Broj stranica: 39

Potvrđujem:

- da sam pročitao dokumente koji se odnose na plagijarizam, kako je to definirano Statutom Univerziteta u Sarajevu, Etičkim kodeksom Univerziteta u Sarajevu i pravilima studiranja koja se odnose na I i II ciklus studija, integrirani studijski program I i II ciklusa i III ciklus studija na Univerzitetu u Sarajevu, kao i uputama o plagijarizmu navedenim na web stranici Univerziteta u Sarajevu;
- da sam svjestan univerzitetskih disciplinskih pravila koja se tiču plagijarizma;
- da je rad koji predajem potpuno moj, samostalni rad, osim u dijelovima gdje je to naznačeno;
- da rad nije predat, u cjelini ili djelimično, za stjecanje zvanja na Univerzitetu u Sarajevu ili nekoj drugoj visokoškolskoj ustanovi;
- da sam jasno naznačio prisustvo citiranog ili parafraziranog materijala i da sam se referirao na sve izvore;
- da sam dosljedno naveo korištene i citirane izvore ili bibliografiju po nekom od preporučenih stilova citiranja, sa navođenjem potpune reference koja obuhvata potpuni bibliografski opis korištenog i citiranog izvora;
- da sam odgovarajuće naznačio svaku pomoć koju sam dobio pored pomoći mentora i akademskih tutora/ica.

Sarajevo, februar 2019.

Potpis:

Malik Koljenović, 984/2015

Sadržaj

Popis slika	vii
Indeks pojmova	viii
1 Uvod	1
2 Postavka problema	2
3 Kriptografske osnove rješenja	3
3.1 Hash funkcije	3
3.2 Kriptografija javnog ključa	5
3.2.1 Sigurnosni ciljevi	6
3.3 RSA kriptosistem	7
3.3.1 Ključevi i komunikacija	8
3.3.2 Digitalni potpis	9
3.4 PKI - infrastruktura javnog ključa	11
3.4.1 Životni ciklus ključeva	11
3.4.2 Hijerarhija povjerenja	12
3.4.3 Certifikati	13
4 Prijedlog rješenja	15
4.1 Logički model rješenja	15
4.2 Tehnički model rješenja	18
5 Pregled korištenih tehnologija	21
5.1 NFC (<i>en. near-field communication</i>)	21
5.1.1 NXP NTAG216	21
5.1.2 NDEF (<i>en. NFC Data Exchange Format</i>)	22
5.1.3 HCE (<i>en. Host card emulation</i>)	22
5.2 Ostalo	23
6 Izdvojeni detalji implementacije	24
6.1 Podatkovni i kriptografski primitivi	24
6.1.1 SPIM paket	24
6.1.2 SESS paket	26
6.2 Pregled implementacije	27
6.2.1 MainActivity	27
6.2.2 LogitAPDUService	27
6.2.3 processCommandApdu	30

6.2.4 AttendanceActivity	31
7 Korisničko uputstvo	33
7.1 Instruktorski način rada	34
8 Sigurnosna analiza	35
9 Zaključak	36
Literatura	37

DRAFT

Popis slika

3.1	Primjena hash funkcije na dva različita ulazna parametra daje različitu i jedinstvenu izlaznu hash vrijednost za dati ulazni parametar	4
3.2	Tajna komunikacija unutar javnog kriptosistema	5
3.3	Verifikacija integriteta i autentičnosti uz garancija neporecivosti u okviru javnog kriptosistema	9
3.4	Primjer hijerarhijske infrastrukture javnog ključa	12
3.5	Prikaz Logit PKI hijerarhije	13
3.6	Osobine certifikata	14
4.1	Dijagram interakcije - uspješna registracija i generisanje ključeva	16
4.2	Dijagram interakcije - bilježenje prisustva studenata (Master BUMP)	17
4.3	Dijagram interakcije - prijava prisustva studenta (Slave BUMP)	17
4.4	Dijagram interakcije - pohranjivanje potpisa na LAPI (SYNC)	18
4.5	Logit UI Android prikaz korisničkog interfejsa	20
5.1	NTAG216 organizacija memorije[1]	22
5.2	NFC HCE virtuelnog sigurnog elementa (SE)[2]	23
6.1	Dijagram klasa SPIM i SESS objekata	25
6.2	QR oblik SPIM objekta	26
7.1	Zaglavlje aplikacije prikazuje aktivnog korisnika	34
7.2	Glavni izbornik, opisi funkcionalnosti u nastavku	34
7.3	Trenutno zabilježena lokacija korisničkog uređaja	34
7.4	Ordinalno numerisan spisak prisutnih studenata	34

Indeks pojmova

ATTN repozitorij potpisanih prisustva spremljen na LAPI.

BUMP približavanje mobilnih uređaja, otvara jednosmjerni komunikacijski kanal u smjeru od slave (S) prema master (M) uređaju.

CERT javni dio korisničkog kriptografskog ključa.

DEVICE korisnički Android uređaj.

HCE (*en. Host card emulation*) softverska arhitektura koja omogućava virtualnu emulaciju elektronskog identiteta.

ISO (*en. International Organization for Standardization*) - Međunarodna organizacija za standardizaciju u Ženevi, Švicarska.

ISO/IEC 14443 Tip A standard fizičkog sloja NFC komunikacijskog protokola.

JSON (*en. Java Simple Object Notation*) - vrlo jednostavna šema serijalizacije objekata u tekstualni oblik.

KEYS jedinstveni set korisničkih RSA ključeva dužine 2048 bita.

LAPI Logit API, Python serverska aplikacija, komponenta LAPP platforme.

LAPP Logit višekomponentna aplikacijska platforma.

M (*en. master*) - Android UI komponenta pokrenuta na uređaju koji bilježi prisustvo.

NDEF vrsta standardizovanog paketa korištena za NFC komunikaciju između uređaja.

NDEFMSG NDEF poruka koja sadrži vremensko-lokacijski dokaz potpisan od strane korisnika.

NFC (*en. near-field communication*) - skup komunikacijskih protokola omogućava uspostavu komunikacijskog kanala između dva uređaja koji se nalaze u neposrednoj blizini.

NFC Forum Tag standardizovani format NFC taga.

S (*en. slave*) - Android komponenta koja se izvršava u pozadini na uređaju čije se prisustvo bilježi.

SPIM (*en. SPacetime*) - lokacijski dokaz (JSON objekat, struktura podatka).

SSO (*en. Single Sign On*) - politika autentifikacije korištenjem jedinstvenog repozitorija.

UI Android komponente LAPP platforme.

DRAFT

Poglavlje 1

Uvod

Prodor digitalnih računara i komunikacijskih tehnologija u sve sfere ljudskog života i djelovanja, te dramatično povećanje broja korisnika interneta u posljednjoj deceniji nametnulo je mnoštvo novih društvenih i tehničkih izazova. Društveni izazovi najbolje su uočljivi kroz višedecenijsku debatu o privatnosti i vlasništvu nad ličnim podacima, samim time zadiru duboko u diskusiju o ljudskim pravima i identitetu sa jedne i često suprotstavljenim komercijalnim interesima sa druge strane. Ukoliko se u tom kontekstu posmatra aktuelna EU uredba o zaštiti podataka[3] (*en. GDPR*) postaje jasno da su digitalna tehnologija i komunikacije postale integralni dio društvene i emocionalno-psihološke realnosti[4], do te mjere da se digitalni tragovi smatraju dijelom nepovredivog identiteta osobe. Iz navedenog je jasno da se radi o institucionalizaciji jedne potpuno nove društveno-tehnološke paradigme unutar pravnih okvira Europske unije.

Sa tehničke strane, dostignuća na poljima kriptografije, teorije mreža i novih komunikacijskih tehnologija, te njihova široka prihvaćenost otvorila su mogućnosti izrade računarskih sistema spremnih da odgovore na novonastale društvene izazove u okviru opisane nove paradigme. Pomenuti računarski sistemi kao dodatno izvršno okruženje imaju društveno-pravnu realnost te se u tim okvirima izvršavaju masovno, dobrovoljno, distribuirano i interaktivno[5] van centralizovanog računarskog izvršnog okruženja u smislu Von Neumannove arhitekture. Opisani sistemi mogu se okarakterisati kao sistemi potpomaganja (*en. assist*), npr. kriptografski računarski sistem u domenu autentifikacije i autorizacija u novoj paradigmi postmatra se kao sistem računarski-potpomognutog povjerenja, ekvivalentno višem nivou apstrakcije.

Registri u kontekstu društvenih institucija su elementarni mehanizam sistema povjerenja, sigurnosne karakteristike takvih institucionalnih registara stoga čine osnov istraživačkog interesa u domenu institucionalne sigurnosi. Napredni elektronski registri izrađeni korištenjem kriptografskih tehnika i savremenih komunikacijskih protokola za prikupljanje i obradu podataka omogućavaju poboljšanje njihovih sigurnosnih osobina, otvarajući nove načine primjene i stvarajući uslove za viši nivo društvenog razvoja i institucionalne efikasnosti, uz to pružaju i adekvatan odgovor na novonastale društvene izazove. Stoga, ukoliko se obezbijede i ispoštuju preduslovi izrade sigurnog sistema[6], evidenciju prisustva u kontekstu naprednog elektronskog registara treba posmatrati i kao vremensko-prostorni dokaz određenog događaja, ovaj rad usmjeren je na izradu jednog takvog sistema računarski-potpomognutog povjerenja u obliku institucionalnog registra elektronske evidencije prisustva.

Poglavlje 2

Postavka problema

Projektni zadatak ovog završnog rada je izrada aplikacije na Android platformi sa pripadajućom udaljenom komponentom, koje u cjelini treba da omoguće evidentiranje prisustva nastavnim aktivnostima na Elektrotehničkom fakultetu u Sarajevu. U skladu sa zadatim funkcionalnim zahtjevima, a iz razloga olakšanog korištenja i praktičnosti upotrebe neophodno je iskoristiti beskontaktnu komunikacijske mogućnosti savremenih mobilnih telefona u vidu NFC komunikacijskog protokola.

Također neophodno je osigurati korisnike aplikacije od mogućih zloupotreba korištenjem dostupnih kriptografskih metoda i tehnologija, te stvoriti neophodne uslove za sticanje povjerenja u širi sistem bilježenja prisustva putem neporecivosti i neizmjenjivosti prethodno unesenih podataka. Poželjna mogućnost je jednostavna integracija sa postojećim sistemima, prvenstveno onim autentifikacijskim i autorizacijskim, te planiranje arhitekture za buduća proširenja u vidu omogućavanja integracije sa infrastrukturnim hardverskim čitačima i TAG karticama.

Potrebno je dokumentovati proces izrade i opisati korištene tehnologije, sa posebnim osvrtom na korištene kriptografske metode i tehnologije, te identifikovati otvorena pitanja na polju elektronskih registara prisustva, mogućnosti i izazove koje oni predstavljaju uz rješenja koja navedena aplikacija nudi u datom kontekstu.

Poglavlje 3

Kriptografske osnove rješenja

Enkripcija, tj. skriveno pisanje, izvorni je cilj kriptografije[7], mogućnost tajne komunikacije intrigirala je čovječanstvo tokom poznate historije, te je razvijeno mnoštvo načina da se taj cilj i ostvari, od primitivnih supstitucijskih metoda, pa sve do moderne formalno utemeljene kriptografije[8]. Kako namjena rješenja u okviru ovog rada diktira, primarni fokus ovog poglavlja biti će međutim stavljen na dodatne mogućnosti moderne kriptografije koje su posebno došle do izražaja razvojem kriptografije javnog ključa i to autentifikaciju entiteta, te utvrđivanje autentičnosti i integriteta poruke.

3.1 Hash funkcije

Kriptografske hash funkcije igraju ključnu ulogu u savremenoj kriptografiji, njihova osnovna namjena je preslikavanje domena X u užu kodomen Y . Najčešće se koriste u sklopu utvrđivanja integriteta podataka i autentifikacije poruka. Kao ulaz hash funkcije primaju niz bita a kao izlaz vraćaju rezultirajuće preslikavanje koje nazivamo hash-kod, hash-vrijednost ili jednostavno **hash**[9], koji je i sam niz bita ali u praksi se gotovo uvijek ispisuje u heksadecimalnom zapisu, npr. B0E2B76996D3A488CEFDA9C4B83ECE1E3E49121A.

Preciznije hash funkcija h preslikava niz konačne dužine n bita, za domen D i kodomen R tako da $h : D \rightarrow R$ i $|D| > |R|$, ovakvo preslikavanje je *many-to-one* i implicira postojanje *kolizija*, tj. različitih parova ulaza sa identičnim izlazom, ovo je nepoželjna ali i neizbježna osobina hash funkcija te se u praktičnim implementacijama pokušava minimizirati njen efekat. Osnovna ideja je da hash vrijednost služi kao kraća reprezentativna slika ili digitalni otisak izvornog objekta i koristi se kao da je jedinstveno identifikabilna sa izvornim objektom.

U okviru ovog rada posebno je zanimljiv slučaj korištenja hash funkcija zajedno sa šemama digitalnog potpisivanja u cilju utvrđivanje integriteta podatkovne strukture, gdje se poruka obično prvo hashuje i rezultirajuća hash vrijednost kao reprezent originalne poruke potpisuje korisničkim privatnim ključem. Prema opisanoj definiciji hash funkcija može uzeti mnoštvo oblika, stoga je za bilo kakvu ozbiljniju diskusiju o hash funkcijama potrebno napraviti klasifikaciju njenih različitih oblika koje u zavisnosti od domena primjene uzimaju različite osobine i nameću dodatne zahtjeve i ograničenja, time dajući jasniju sliku o hash funkcijama uopšte. Dvije osnovne osobine svake hash funkcije su:

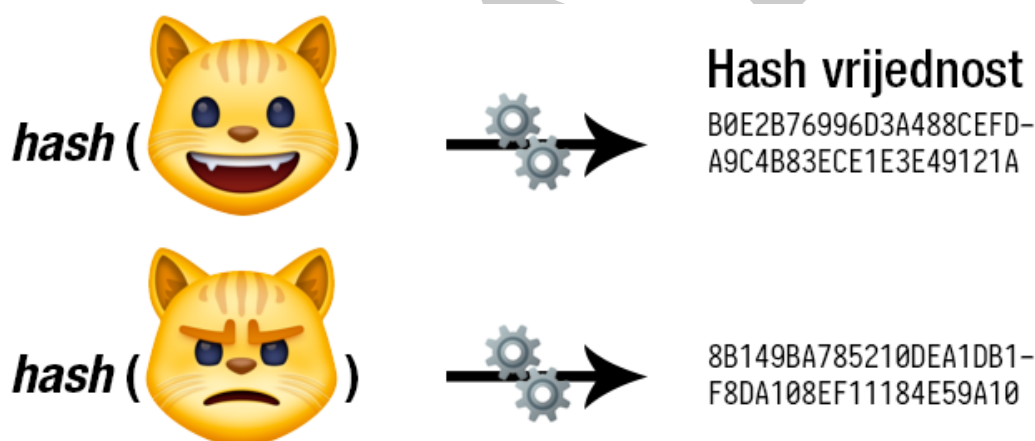
- *kompresija* - funkcija h preslikava ulaz x proizvoljne konačne dužine t niza bita u izlaz $h(x)$ fiksne dužine n bita

- *jednostavnost izračuna* - za datu funkciju h , i ulaz x , $h(x)$ je jednostavna i brza za izračun.

Sve hash funkcije zadovoljavaju dvije pobrojane osobine, ali to svojstvo nije dovoljno za njihovu kriptografsku primjenu za koju je neophodno osigurati dodatne garancije, stoga se nameće podjela na nekriptografske i kriptografske hash funkcije, koje dodatno moraju zadovoljiti i najmanje jedan od dva niženavedena uvjeta:

- *jednosmjernost* - osigurava da je izračunski teško za datu izlaznu hash vrijednost y pronaći izvornu vrijednost parametra x hash funkcije $h(x)$,
- *otpornost na kolizije* - pronalazak dvije iste izlazne hash vrijednosti za dva različita ulazna parametra x izračunski je teško i nepraktično, što obično znači da bi za pronalazak kolizije uzevši najbrže trenutno zamislive računare trebalo više vremena nego je proteklo od postanka univerzuma do danas, što se može smatrati razumnom garancijom, koju je kako se je do sada pokazalo praktično teško ispoštovati.

Uzevši u obzir sve opisane karakteristike jasno je da broj praktičnih implementacija kriptografskih hash funkcija koje pružaju neophodne garancije relativno mali, no postoje funkcije koje pružaju dovoljno dobre garancije za svakodnevnu praktičnu primjenu, od kojih su najpoznatije SHA, MD i BLAKE familije hash funkcija, u okviru ovog rade korištena je SHA-256 hash funkcija.



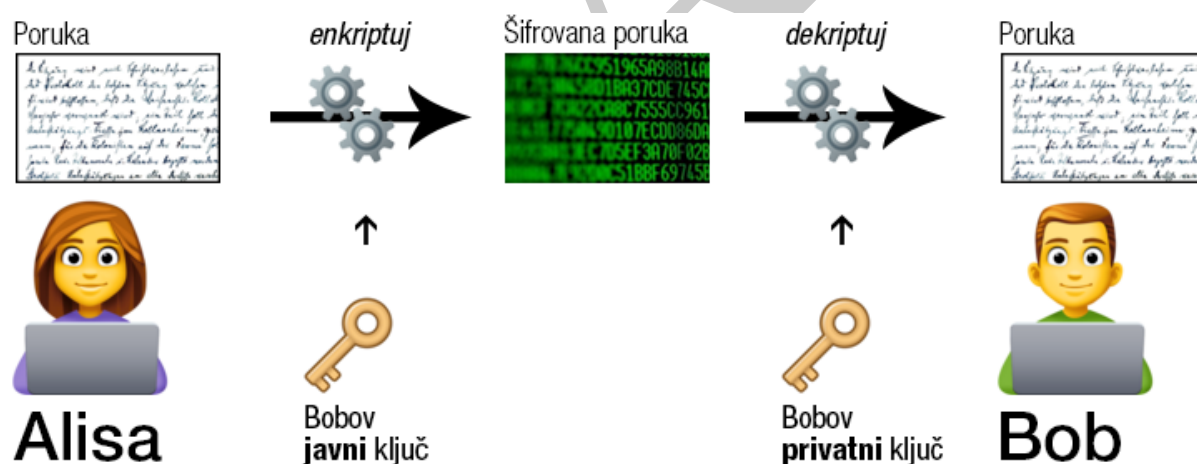
Slika 3.1: Primjena hash funkcije na dva različita ulazna parametra daje **različitu i jedinstvenu** izlaznu hash vrijednost za dati ulazni parametar

Ilustrativno na slici 3.1 dat je prikaz rada SHA-1 hash funkcije za vrijednosti "HAPPYCAT" i "ANGRYCAT", gde je jednostavno uočljiva različita izlazna hash vrijednost, dodatno navedene vrijednosti bi trebale biti jedinstvene za date ulaze, tj. ne bi smjela postojati neka druga vrijednost osim navedene koja bi dala istu hash vrijednost kao rezultat. SHA-1 funkcija uvijek daje rezultat fiksne dužine 40 bajta. Sa sigurnosnog aspekta relevantno je spomenuti još i to da postoje tabele koje sadrže prethodno izračunate hash vrijednosti za mnoge ulazne parametre i različite funkcije, kao i pretraživače po hash vrijednostima što djelimično narušava garanciju jednosmjernosti, takve tabele nazivaju se *rainbow* tabelama.

3.2 Kriptografija javnog ključa

Simetrična kriptografija zahtjeva razmjenu tajnog ključa prije ostvarivanja sigurnog komunikacijskog kanala između dva ili više učesnika, iako vrlo pouzdan metod, često je teško provodiv u praksi, pogotovo u slučajevima ad-hoc i masovne decentralizovane komunikacije nalik internetu, neophodno je da učesnici ili sami izvrše međusobnu razmjenu tajnih ključeva koje će naknadno koristiti ili da jedan autoritativni entitet u kojeg se ima povjerenje to uradi umjesto njih tako što će generisati i distribuirati ključeve svim učesnicima u komunikaciji, očite slabosti ovog modela su da zahtjeva povjerenje i postojanje funkcionalnog sigurnog kanala za razmjenu tajnih ključeva - što često nije slučaj. Asimetrična kriptografija ili kriptografija javnog ključa razvijena je sa ciljem prevazilaženja navedenih nedostataka od strane britanske službe GCHQ početkom sedamdesetih godina XX stoljeća, javnosti je naknadno predstavljena kroz radove Merklea[10], Diffiea i Hellmana[11], a nešto kasnije u obliku danas opštepoznate praktične implementacije RSA kriptosistema Rivesta, Shamira i Adelmana[12], koji dodatno uvodi i pojam elektronskog potpisa, kao i njegove komercijalne namjene.

Kriptosistemi javnog ključa uvode ideju dva različita ali povezana ključa, jedan - javni samo za enkripciju i jedan - privatni samo za dekripciju, a kako nije moguće saznati jedan ključ iz drugog korisnik je slobodan javno objaviti svoj ključ za enkripciju, tako da ukoliko npr. Alisa želi poslati tajnu poruku Bobu, dovoljno je da posjeduje Bobov sada *javni enkripcijski ključ* i da ga iskoristi da njime šifruje poruku. Kada Bob primi takvu poruku iskoristiti će svoj *tajni privatni ključ* i dešifrovati Alisinu poruku, kompletna interakcija i sastavni elementi prikazani su na slici 3.2.



Slika 3.2: Tajna komunikacija unutar javnog kriptosistema

Kriptografija javnog ključa također se naziva i *asimetričnom kriptografijom*, ona ne osigurava samo tajnost komunikacije kao u opisanom slučaju, nego se može koristiti i za potvrdu autentičnosti. Za takav scenario dovoljno je da Bob svojim privatnim ključem *potpiše* željenu poruku ili datotetu i Alisa će biti u mogućnosti da provjeri da je ta poruka autentično kreirana od strane Boba, za tu namjenu potrebno je da Alisa posjeduje Bobov javni ključ od ranije ili da ga dobavi iz nekog povjerljivog izvora jer mora biti sigurna da neko lažno ne podmetne svoj ključ kao Bobov, više o ovoj namjeni asimetričnih kriptosistema biti će dato u zasebnom poglavlju u nastavku, no za sada je bitno imati je na umu. Spomenuti proces potpisivanja sastoji se od par kriptografskih operacija, vrlo sličnih kriptovanju poruke, no kako u ovom slučaju nije potrebno prenijeti kompletan sadržaj nego samo omogućiti verifikaciju, proces je moguće učiniti

efikasnijim tako što će se poruka prije obrade privatnim ključem prethodno provući kroz neku od hash funkcija, koja će ga znatno smanjiti i ubrzati sam proces.

3.2.1 Sigurnosni ciljevi

Sigurnosni ciljevi kriptografije javnog ključa pored povjerljivosti, kao primarne funkcije, kako je već i pomenuto, uključuju i mogućnosti za provjeru integriteta i autentičnosti poruke, autentifikacija entiteta, te osiguravanje neporecivosti izvršene akcije[13], ovakav jedinstven i širok skup funkcionalnosti kriptografiji javnog ključa daje fundamentalni značaj u modernim digitalnim komunikacijama i internet poslovanju, stoga su osnovne karakteristike svakog od navedenih ciljeva ukratko opisane u nastavku.

Povjerljivost i privatnost

Povjerljivost je osnovni sigurnosni cilj kriptografije i odnosi se na osobinu da tajni podaci neće biti dostupni neautorizovanim osobama. Povjerljivost omogućava privatnost i međusobno su usko povezane. Privatnost označava sposobnost očuvanja vlastitih podataka tajnim za sve one koji nisu eksplicitno autorizovani za njihov pregled i predstavlja osnovno ljudsko pravo garantovano članom 12 Univerzalne deklaracije o ljudskim pravima[14]:

"Niko ne smije biti podvrgnut samovoljnom miješanju u njegovu privatnost, obitelj, dom ili dopisivanje, niti napadima na njegovu čast i ugled. Svako ima pravo na zakonsku zaštitu protiv takvog miješanja ili napada."

Mnoštvo je primjera važnosti povjerljivosti i privatnosti u savremenoj eri opšteprisutne digitalizacije i internet poslovanja, nažalost narušavanje ovih temeljnih vrijednosti postalo je gotovo normalna i svakodnevna pojava bilo da se radi o krađi osjetljivih podataka i digitalnih dobara od strane malicioznih agenata ili prisluškivanju od strane korporativnih i vladinih agencija koje vrše masovno špijuniranje na globalnom nivou prikupljanjem privatnih podataka putem raznorodnih programa[15].

Autentifikacija entiteta

Autentifikacija se odnosi na proces utvrđivanja stvarnog identiteta, navedeni identitet može se odnositi na osobu, kompaniju ili bilo koji drugi koncept koji se od drugog razlikuje po sebi svojstvenim osobinama. Za isti proces može se približno precizno koristiti i termin identifikacija. Poznavanje identiteta u okviru digitalnog okruženja često je neophodno za ispravno funkcionisanje programskog rješenja i pravilnu raspodjelu autorizacija, kao i za relacije povjerenja između različitih kategorija korisnika. Često se za ovu namjenu koriste korisnička imena i lozinke kao najprostiji vid implementacije, no mnogo su pouzdaniji namjenski izrađeni repozitoriji identiteta u vidu specifičnih rješenja ili infrastrukture javnih ključeva (*PKI - public key infrastructure*), koji omogućavaju mnogo sigurniju autentifikaciju, bolje provođenje dobrih sigurnosnih praksi, širi spektar primjene i bolju integraciju. Aplikacija izrađena u okviru ovog rada sadrži namjenski izrađen pokazni repozitorij identiteta i javnih ključeva u vidu Logit API implementacije.

Integritet i autentičnost poruke

Integritet se odnosi na garanciju da podaci nisu mijenjani nakon što ih je izvorni autor sačinio, mnoštvo je bitnih aktivnosti gdje je upravo ovakva garancija od iznimne važnosti. Koncept in-

tegriteta dodatno se proširuje kroz pojam autentičnosti poruke, gdje se zahtjeva i mogućnost utvrđivanja njenog izvorišta, te njegova autentifikacija. Ove aktivnosti izvode se putem digitalnog potpisivanja i vjerodostojnih repozitorija koji sadrže provjerene identitete entiteta koji se autoriziraju, najčešće u vidu već opisanih PKI.

Kao primjer možemo uzeti svakodnevno poznate korisničke scenarije, svaki računarski program ili nadogradnja kada se distribuira krajnjim korisnicima može biti naknadno izmjenjen u cilju izvršavanja određenih zlonamjernih aktivnosti koje mogu naštetiti korisniku, da bi se ovo izbjeglo većina savremenih operativnih sistema podržava određeni način provjere integriteta softvera prilikom instalacije na korisničkom računaru i autentičnost identiteta njenog izvorišta, ovakve provjere posebno su bitne u sigurnosno osjetljivim okruženjima i djelatnostima gdje bi pokretanje zlonamjernog koda moglo ugroziti živote ili uzrokovati veliku materijalnu štetu, primjeri takvih sistema su računari u zdravstvu i kontrolni sistemi javnih infrastruktura, poput aerodroma, električne ili vodovodne mreže, no nikako ne treba zanemariti i kućne korisnike koji su itekako izloženi raznim vrstama sigurnosnih prijetnji. Za namjene utvrđivanja integriteta i autentifikacije izvorišta programskih rješenja održavaju se repozitoriji sa identitetima softverskih razvojnih kuća i njihovim ključevima.

Dodatno utvrđivanje integriteta poruke i autentifikacija učesnika se koristi u okviru aplikacije za bilježenje prisustva studenata predložene u okviru ovog rada na način da se potpisano vrijeme i lokacija svih studentskih uređaja potpisuje predavačkim ključevima unutar jedinstvene sesije (npr. nastavnog časa) koju po zaključenju nije moguće naknadno mijenjati bez narušavanja integriteta navedene sesije. Identitet svih učesnika i autentičnost njihovih potpisa provjerava se korištenjem namjenskog repozitorija na Logit API.

Neporecivost

Neporecivost je osobina podataka koja sprječava poduzimača određene aktivnosti da istu porekne nakon njenog okončanja, npr. slanje poruke, novčana transakcija ili prisustvo predavanju. Kod primjera prisustva, radi se o nemogućnosti studenta da nakon što digitalno prijavi svoje prisustvo na predavanju unutar predložene Logit NFC aplikacije naknadno to prisustvo porekne jer postoji jedinstveni digitalno potpisan trag koji to dokazuje, za čvrst dokaz neophodno je osigurati i jaku povezanost korisnika i nosioca identiteta, u ovom slučaju mobilnog uređaja, za ovakve namjene posebno su pogodne višefaktore metode identifikacije koje uključuju i biometrijske osobine.

3.3 RSA kriptosistem

Rivest, Shamir i Adelman[12] predložili su kriptosistem koji može osigurati osobine privatnosti i potpisivanja poruka ekvivalentne ili bolje od onih kakve posjeduje papirna pošta. Njihov sistem baziran je na konceptu sistema javnog ključa kakav su ranije predložili Diffie i Hellman[11]. Sveukupna procedura sastoji se od procesa enkripcije E , procesa dekripcije D i poruke M , te za takav sistem vrijedi:

1. dešifrovanje kriptovane forme poruke M daje M ,

$$D(E(M)) = M$$

2. i E i D su jednostavne za izračun,
3. javno obznajući E korisnik ne otkriva jednostavan način za izračun D . Praktično ovo znači da samo on može dekriptovati poruke kriptovane pomoću E
4. ukoliko je poruka M prvo dešifrovana a onda šifrovana, rezultat je M ,

$$E(D(M)) = M$$

Enkripcijska (ili dekripcijska) procedura se sastoji od *opšte metode* i *enkripcijskog ključa*. Opšta metoda, pod kontrolom ključa, šifruje poruku M i rezultira šifrovanom porukom C (*enciphertext*). Svako može koristiti istu opštu metodu, sigurnost date procedure počiva na sigurnosti ključa. Otkrivanje enkripcijskog algoritma tada znači i otkrivanje (*javnog*) ključa.

Kada korisnik otkrije E , on otkriva vrlo neefikasan način izračuna $D(C)$ testiranjem svih mogućih poruka M sve dok se ne nađe ona koja zadovoljava $E(M) = C$. Ukoliko je osobina (3) zadovoljena broj takvih poruka nije praktičan za izračun.

Funkcija E ukoliko zadovoljava osobine (1)-(3) predstavlja tzv. "*jednosmjernu funkciju sa stupicom*", a ukoliko zadovoljava i (4) onda je "*jednosmjerna permutacija sa stupicom*". Diffie i Hellman[11] uveli su koncepte jednosmjernih funkcija sa stupicom ali nisu dali implementaciju. Ove funkcije nazivaju se jednosmjernim jer ih je jednostavno izračunati u jednom smjeru ali bi trebalo biti vrlo teško u suprotnom, dok su "sa stupicom" jer su njihovi inverzi jednostavni za izračun ukoliko je poznata određena informacija, u slučaju šifrovanja je to privatni ključ. Ovakva funkcija koja također zadovoljava i (4) mora biti permutacija: svaka poruka je ciphertekst neke druge poruke i svaki ciphertekst je dozvoljena poruka. Zadovoljenje osobine (4) neophodno je za implementaciju potpisivanja.

3.3.1 Ključevi i komunikacija

Da bi kreirali vlastite privatne i javne ključeve učesnici moraju svaki za sebe i nasumično odabrati dva velika prosta broja p i q takva da nije vjerovatno da računar može izvršiti cjelobrojnu faktORIZACIJU $n = p * q$, danas se minimalno preporučuju brojevi slične veličine koji daju proizvod reda 2048 bita kao sigurni do 2030. godine[16]. Proizvod n postaje dijelom javnog ključa, dok se pojedinačni faktori moraju čuvati u tajnosti zbog njihovog korištenja u derivaciji privatnog ključa.

Korisnici također moraju izabrati i cjelobrojnu vrijednost e takvu da,

$$1 < e < \phi(n) = (p-1)(q-1) \wedge \gcd(e, (p-1)(q-1)) = 1.$$

Obratite pažnju da je e uvijek neparno jer je $(p-1)(q-1)$ parno. Dalje korisnik računa cjelobrojnu vrijednost d ,

$$1 < d < (p-1)(q-1) \wedge de \equiv 1 \pmod{(p-1)(q-1)}.$$

Korisnikov javni ključ sada je par (n, e) , dok je njegov privatni ključ d . Broj n naziva se *RSA modulus*, e je *enkripcijski eksponent*, a d - *dekripcijski eksponent*[13].

Ponovno se može poslužiti primjerom Boba i Alise opisanim ranije, sada se može reći da kada je Bob korištenjem opisane procedure generisao neophodne vrijednosti ključeva i poslao Alisi svoj javni ključ, tj. vrijednosti (n, e) , koje će ona iskoristiti za šifrovanje proizvoljne poruke m , što tada možemo izraziti kao

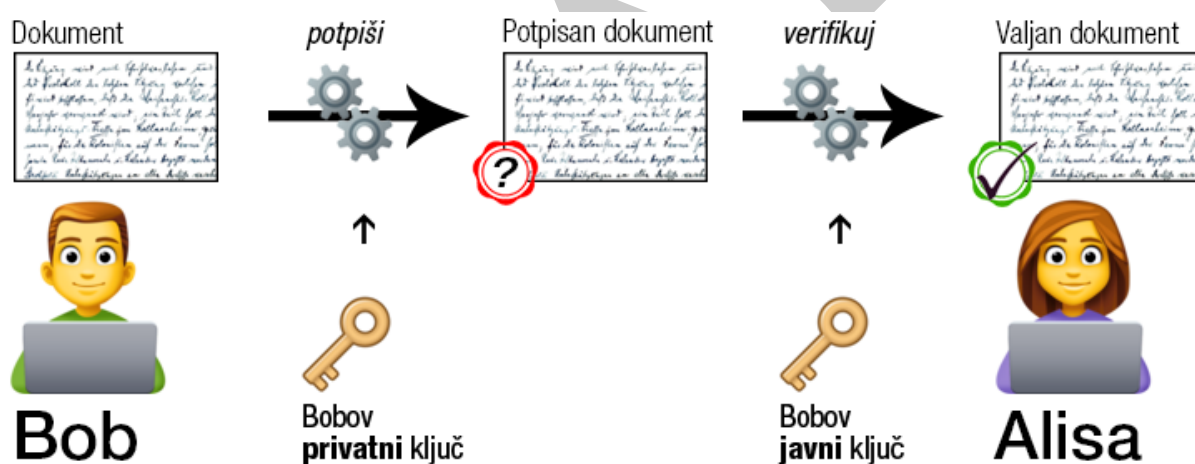
$$c = m^e \bmod n.$$

Kada Bob primi Alisinu šifrovanu poruku c iskoristiti će tajnu vrijednost d i dešifrovati poruku

$$m = c^d \bmod n.$$

3.3.2 Digitalni potpis

Digitalni potpis osigurava mehanizam provjere integriteta i u kombinaciji sa adekvantnom infrastrukturom - autentičnost izvorišta poruke. Ukoliko Bob želi potpisati određeni dokument, on korištenjem svog privatnog ključa izračunava jedinstveni niz bita, koji Alisi garantuje da će korištenjem Bobovog prethodno dobavljenog javnog ključa moći verifikovati da navedeni dokument nije izmjenjen u putu do nje, kao i da je Bob originalni potpisnik dokumenta, navedena interakcija prikazana je na slici 3.3, dodatno Bob u budućnosti ne može poreći da je potpisao navedeni dokument, što osigurava još jednu vrlo bitnu funkcionalnost cjelokupnog sigurnosnog sistema. Ovakav sistem predstavlja temelj na kojem je izgrađen siguran internet i digitalna ekonomija te je duboko utkan u osnovne protokole poput TLS, SSH, PGP etc.



Slika 3.3: Verifikacija integriteta i autentičnosti uz garancija neporecivosti u okviru javnog kriptosistema

RSA šema najčešće je korišten algoritam digitalnog potpisivanja. Generacija ključeva funkcioniše na istom principu kao i u primjeru šifrovanja poruke opisanom u poglavlju 3.3.1. Svaki digitalni potpis zavisen je od poruke kao i od potpisnika, u protivnom bilo bi moguće da isti potpisnik koristi jedan potpis za više dokumenata, što ovdje nije slučaj. Da bi se uspješno implementiralo digitalno potpisivanje, kriptosistem mora zadovoljavati osobine spomenute jednodsmjerne permutacije sa stupicom, budući da će dekripcijski algoritam biti primjenjivan na izvorni - nešifrovan dokument.

Opisani primjer razmjene potpisanog dokumenta ili poruke M između Alise i Boba može se unutar RSA kriptosistema preciznije prikazati kao:

$$S = D_B(M),$$

gdje je S digitalno potpisana poruka a D_B Bobova funkcija za dešifrovanje korištenjem privatnog ključa. Kako je već napomenuto primjena funkcije za dešifrovanje na nešifrovanu poruku ima smisla u slučaju da kriptosistem zadovoljava potrebne uvjete. Ukoliko je dodatno potrebno osigurati i tajnost potpisane poruke Bob je može naknadno šifrirati Alisivnim javnim ključem, u tom slučaju Alisa po prijemu prvo vrši dešifrovanje svojim privatnim ključem da bi dobila Bobovu izvornu potpisanu poruku S , korištenjem Bobovog javnog ključa Alisa sada "šifruje" potpisanu poruku

$$M = E_B(S)$$

time dolazeći u posjed uređenog para (M, S) sa osobinama sličnim onima koje ima originalni fizički potpisan dokument.

Sama praktična implementacija potpisivanja razlikuje se međutim od opisane procedure u tome da se u praksi ne vrši potpisivanje cjelokupne izvorn poruke, nego se nad njom prvobitno korištenjem hash funkcije otporne na kolizije izračunava jedinstvena hash vrijednost, pa Bob tako umjesto poruke potpisuje dobijenu hash vrijednost svojim privatnim ključem, dok na drugom kraju Alisa ponavlja proces izračunavanja hash vrijednosti koristeći Bobovu izvornu poruku, njegov javni ključ i istu hash funkciju, te dobijeni hash poredi sa Bobovim hashom, u slučaju poklapanja vrijednosti Alisa može biti sigurna u valjanost digitalnog potpisa. Jedina funkcionalna razlika ovakvog pristupa je da se koncept poruke ili dokumenta, odvaja od samog digitalnog potpisa, što olakšava rad i osigurava dodatne sigurnosne i upotrebne prednosti.

Praktična implementacija može se formalno opisati kao primjena hash funkcije h tako da potpis s niza karaktera poruke $m \in \{0, 1\}^n$ glasi

$$s = h(m)^d \bmod n.$$

d je u prikazanom slučaju Bobov *dekriptijski eksponent*. Da bi Alisa verificovala zaprimljeni potpis s koristi Bobov javni ključ (n, e) i izračunavanjem hash vrijednost $h(m)$ poruke m i vrši provjeru

$$h(m) = s^e \bmod n.$$

Potpis je valjan ako i samo ako važi data jednakost. Za potrebe Logit sistema korištena je SHA-256 hash funkcija u kombinaciji sa RSA potpisom.

3.4 PKI - infrastruktura javnog ključa

Iako kriptografija javnog ključa ne zahtjeva razmjenu tajnih ključeva da bi se ostvarila povjerljiva komunikacij, vrlo je bitan aspekt upravljanja ključevima, kako javnim tako i privatnim. Namjena infrastrukture javnog ključa *en. PKI - private key infrastructure* je upravo to, efikasno i sigurno upravljanje javnim i privatnim ključevima tokom njihovog životnog ciklusa.

3.4.1 Životni ciklus ključeva

Životni ciklus počinje od samog generisanja, koje je opisano u poglavlju 3.3.1, nakon čega slijedi upotrební vijek tokom kojeg se privatni ključevi koriste za potpisivanje ili dešifrovanje primljenih poruka. Korisnici također imaju pristup javnim ključevima drugih korisnika, te ključeve koriste za šifrovanje ili verifikaciju potpisanih dokumenata. U završnoj fazi ključevi izlaze iz upotrebe bilo kroz proces zastarjevanja ili neki drugi događaj. Svaki od navedenih faza u životnom ciklusu para ključeva mora imati određene procedure da bi se osiguralo provođenje dobrih praksi u njihovom upravljanju.

Generisanje i pohrana

Prvi korak je osiguravanje pouzdanog okruženja za generisanje sigurnog para ključeva, najbolje je da tu aktivnost izvode sami korisnici, jer u tom slučaju privatni ključ ne bi trebao biti dostupan nijednoj trećoj strani, no to često nije slučaj, budući da korisničko okruženje može biti kompromitovano ili na neki drugi način neadekvatno za tu namjenu, stoga se mora pribjeći generisanju ključeva u okruženju neke povjerljive treće strane za koju se može biti sigurno da ključeve neće zloupotrijebiti. Pored navedenog i samo skladištenje ključeva, pogotovo privatnih, pruža mnoge sigurnosne izazove. Uzimajući u obzir nabrojano jasno je da navedenim problemima treba pristupiti planski i krajnje ozbiljno jer u protivnom sigurnost kompletnog sistema može biti kompromitovana od samog početka. Logit aplikacija privatne ključeve pohranjuje isključivo unutar Android repozitorija ključeva na korisničkom uređaju, za koji postoje garancije da nije moguće ostvariti pristup korištenjem bilo koje druge aplikacije.

Upotrební vijek

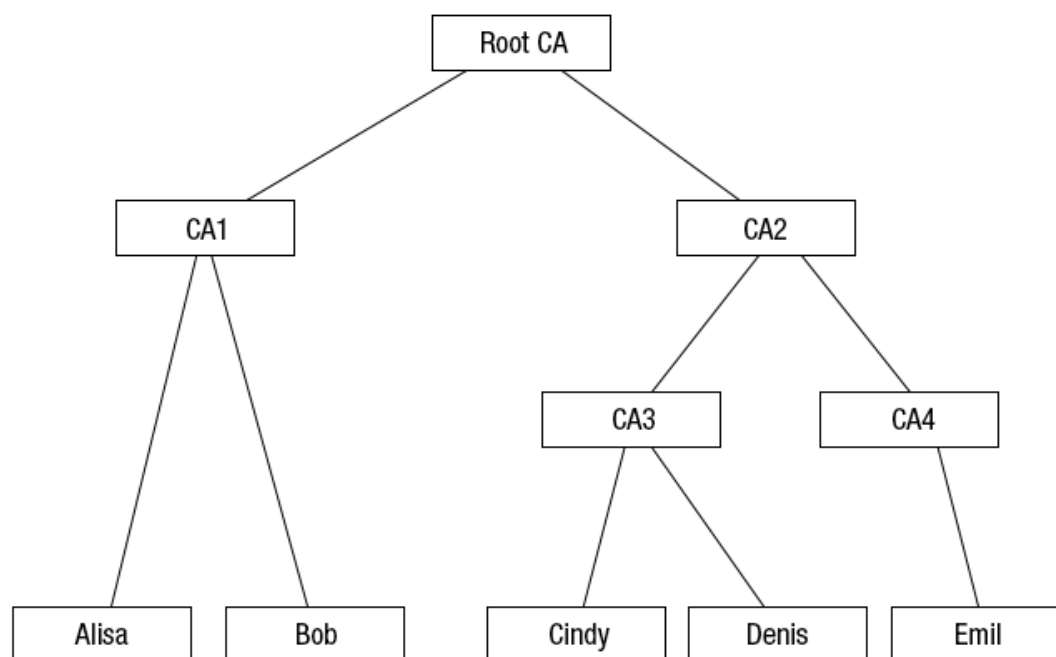
Tokom upotrebne faze životnog vijeka osnovna funkcionalnost je obezbjediti siguran pristup korisničkim javnim ključevima. Pored toga da bi se mogle pružiti adekvatne garancije autentifikacije i potvrde autentičnosti neophodno je utvrditi i provesti jasne procedure koje će osigurati jasnu povezanost entiteta/osoba sa ključevima, u protivnom može doći do krađe identiteta i lažnog predstavljanja. Logit API vršit funkciju repozitorija javnih ključeva za korisnike sistema i povezuje korisnike sa njihovim ključevima, osigurana je osnovna provjera identiteta putem korisničke prijave na fakultetski ZAMGER sistem.

Kraj životnog ciklusa

Dodatno potrebno je utvrditi jasne procedure za kraj životnog vijeka i pohranu starih ključeva, kod Logit API repozitorija ne postoji vremenski ograničen vijek trajanja para ključeva, no pri svakoj novoj instalaciji aplikacije ili zamjeni uređaja, stari par ključeva se arhivira a novogenerisani ključevi se koriste kao sigurnosno relevantni. Pored navedenih funkcionalnosti Logit API se koristi i kao repozitorij potpisanih dokumenata, u ovom slučaju, prisutava predavanjima, no to izlazi iz okvira domena PKI i može se smatrati dodatno funkcionalnošću.

3.4.2 Hijerarhija povjerenja

Za praktičnu upotrebu kriptografije javnog ključa neophodno je da korisnici vjeruju u autentičnost dostupnih javnih ključeva. Stoga su pored prostog direktnog modela povjerenja sa dva učesnika koji razmjenjuju sopstvene javne ključeve uspostavljene raznovrsne hijerarhije povjerljivih učesnika koje omogućavaju formiranje složene mreže koja i sama kao svoju osnovu koristi digitalni potpis i dostupne kriptografske metode.



Slika 3.4: Primjer hijerarhijske infrastrukture javnog ključa

Primjer jedne složene hijerarhije povjerenja prikazan je na slici 3.4, u ovakvom modelu javne ključeve svih nižih učesnika potvrđuje CA (*en. certification authority*), koji ujedno na adekvatan način vrši i verifikaciju autentičnosti. Većina ovakvih hijerarhijskih PKI arhitektura uređena je u skladu sa standardom X.509. U ovakvom okruženju može da učestvuje više nivoa posrednih CA (*en. intermediate CA*) i obično je uređeno u strukturu drveta sa listovima, gdje se u korjenu nalazi CA (*en. root CA*) kojeg svi posredni CA uzimaju kao pouzdanog i koji sam potpisuje svoj certifikat. Na dnu strukture drveta nalaze se krajnji korisnici kriptosistema. Da bi se ostvarila relacija povjerenja između dva korisnika, nije neophodno da oni dijele isti posredni CA, dovoljan uslov je da se može napraviti veza prema jednom CA u kojeg oba korisnika imaju povjerenje da bi pomenuta relacija bila zadovoljena.

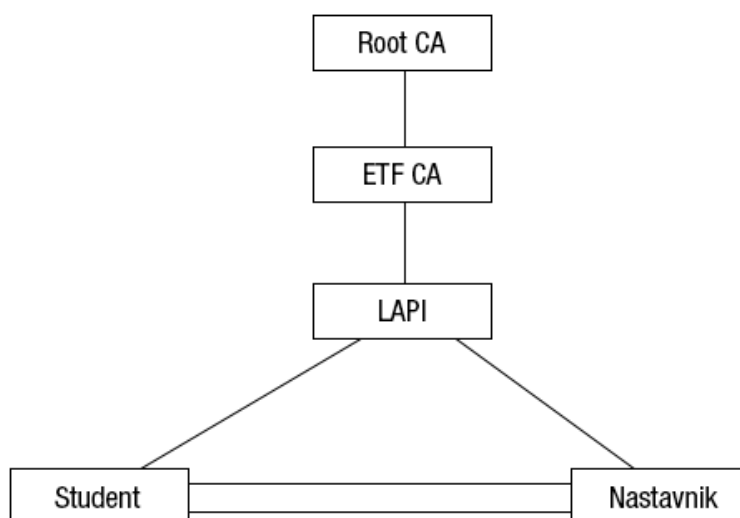
Primjer jedne relacije povjerenja iz priložene hijerarhije može se formirati između korisnika Alisa i Emil, gdje se da jasno utvrditi *certifikacijska putanja* koja za oba korisnika seže do korjenskog Root CA, ilustrativno za Alisu i Emila važi:

$$Alisa \rightarrow CA1 \rightarrow RootCA$$

$$Emil \rightarrow CA4 \rightarrow CA2 \rightarrow RootCA$$

dubina certifikacijske putanje također može da igra ulogu u zavisnosti od različitih zahtjeva, u nekim slučajevima duže putanje, poput Emilove se mogu smatrati nepouzdanim za određene namjene, u svakom slučaju poželjno je korištenje što bližih međusovnih putanja.

Logit API posjeduje svoj set ključeva kojim potpisuje sve zaprimljene korisničke sesije prisustva. Da bi se upotpunio model povjerenja neophodno je da se LAPI certifikat potvrdi od strane institucije koja implementira navedeni sistem, u ovom slučaju Elektrotehničkog fakulteta, koja je dalja poveznica na vanjski korjenski CA i omogućava izgradnju šireg sistema povjerenja gdje više institucija koje koriste isti sistem može ostvariti posredne relacije povjerenja. Važno je istaknuti također da korisnici registracijom bivaju uključeni u LAPI repozitorij, što se može smatrati ekvivalentom certifikata, dodatno korisnici međusobno potvrđuju prisustvo a samim tim ostvaruju vezu međusobnog povjerenja, jer aktivnost prikupljanja potpisa inherentno utvrđuje međusobnu identifikaciju korisnika. Prikaz takvog modela dat je na slici 3.5.



Slika 3.5: Prikaz Logit PKI hijerarhije

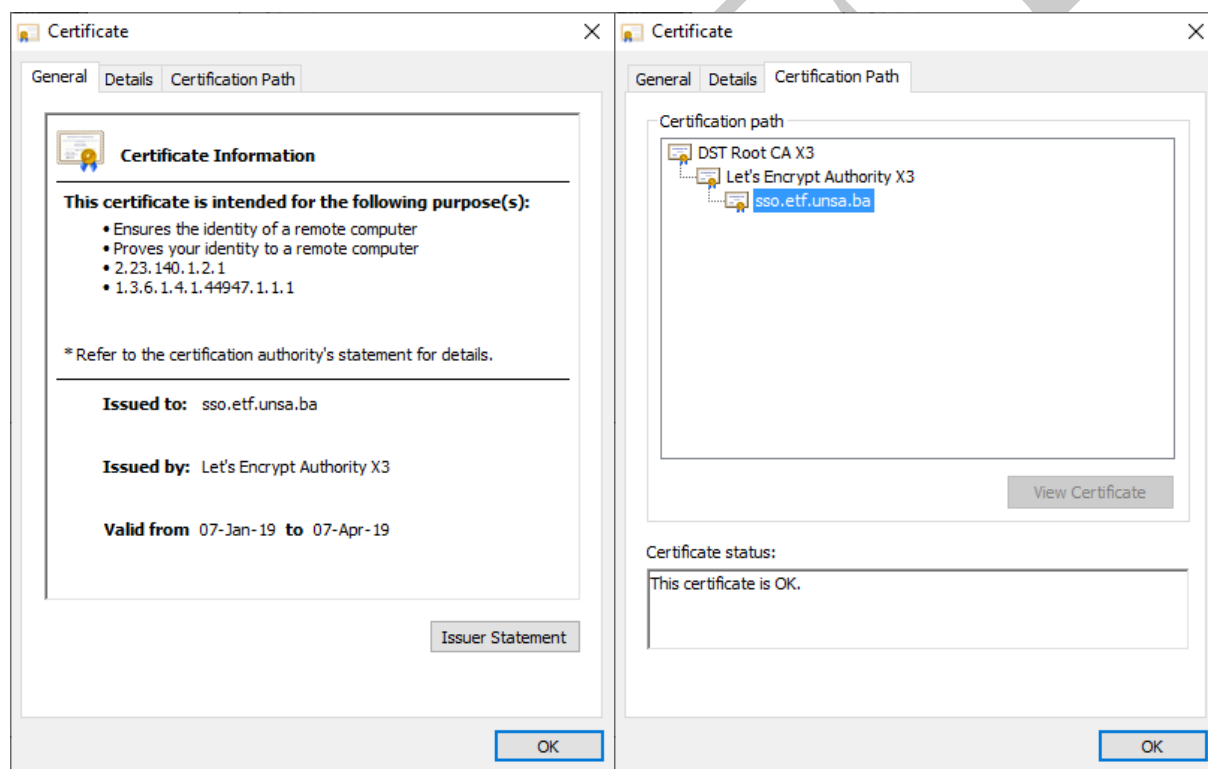
3.4.3 Certifikati

Bitan aspekt PKI je obezbjeđivanje potvrde autentičnosti za javne ključeve, jedan od načina da se to ostvari su svakako certifikati, tj. strukture koje povezuju javne ključeve sa entitetima ili osobama, najčešće su pohranjeni na PKI koji se brinu da osiguravaju što jaču garanciju relacije između entiteta i javnog ključa. U cilju iskoristivosti neophodno je da certifikati obezbjeđe određeni nivo tehničkih i domenski relevantnih podataka, u većini slučajeva to su:

- ime ili naziv entiteta za čiji javni ključ je vezan
- javni ključ entiteta
- korišteni kriptografski algoritam
- serijski broj
- period važenja
- naziv izdavača certifikata, koji je ujedno i potpisnik
- namjena i ograničenja korištenja javnog ključa

Sam sadržaj certifikata u većini slučajeva je standardiziran, danas najčešće standardom X.509, gdje se i izdati certifikat naziva prema standardu X.509 certifikat, primjer jednog takvog certifikata dat je na slici 3.6a, navedene su osnovne informacije o certifikatu i opšte informacije pobrojane iznad, dodatno na slici 3.6b prikazana je certifikacijska putanja datog certifikata. Pored navedenih informacija certifikat obično sadrži mnoštvo i drugih informacija, budući da standard dopušta proširivanje da bi obuhvatio širok domen primjene. Bitno je još napomenuti da zbog vrlo široke primjene u različitim programskim rješenjima postoji mnoštvo različitih formata zapisivanja i razmjene samih certifikata, te je često neophodno konvertovati certifikate u jedan od odgovarajućih formata.

X.509 certifikati koriste ASN.1 (*en. abstract syntax notation version 1*) kao jezik za izvornu specifikaciju osobina certifikata, pomoću navedenog jezika moguće je izraziti mnoštvo kompleksnih struktura za opis osobina certifikata, jedan od mnoštva načina da se navedena specifikacija zapiše je DER (*en. distinguished encoding rules*), koja je dalje bazirana na BER pravilima za zapisivanje (*en. basic encoding rules*), navedene strukture predstavljaju svojevrsnu hijerarhiju deskriptivnih jezika i meta-jezika sličnu relaciji SGML, HTML.



(a) opšte karakteristike

(b) lanac povjerenja

Slika 3.6: Osobine certifikata

Logit aplikacija interno koristi namjenski i nestandardni format certifikata, no sve LAPI i korisničke certifikate moguće je pretvoriti u bilo koji od standardnih formata i tako ostvariti interakciju sa ostalim sistemima ukoliko se za to ukaže potreba. Dovoljno je na Logit API strani implementirati novu URI lokaciju koja bi vraćala certifikate u standardizovanom formatu.

Poglavlje 4

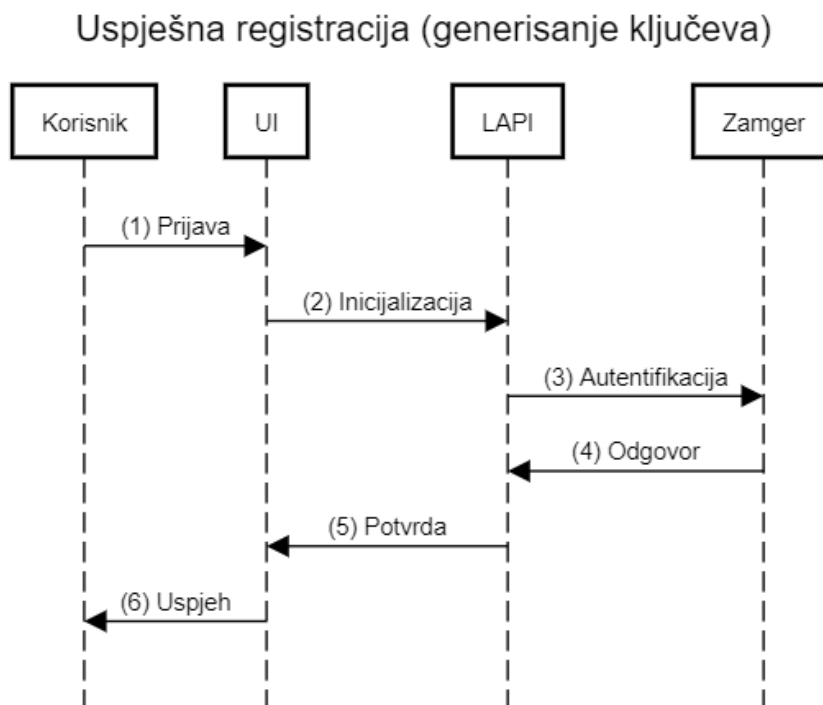
Prijedlog rješenja

U skladu sa datim zahtjevima predložena je izrada aplikacijske platforme pod nazivom Logit (LAPP), opisane u nastavku, sa detaljnim tehničkim detaljima u narednim poglavljima. Uzimajući u obzir data ograničenja, te funkcionalne i nefunkcionalne zahtjeve određeno je da se korisnička aplikacija izradi na Android platformi sa podrškom za Android API nivo počevši od nivoa 19 (4.4 KitKat), to je najniži nivo koji omogućava korištenja naprednih NFC i kriptografskih funkcionalnosti te osigurava dobru pokrivenost potencijalne korisničke baze sa ukupnom adopcijom od preko 90% za navedenu ili višu verziju[17]. Za uspješan rad aplikacije neophodno je da korisnički uređaj podržava i NFC funkcionalnosti, prema prognozama analitičke kuće IHS Technology, do 2020. godine svaki treći uređaj imati će podršku za NFC.[18]

4.1 Logički model rješenja

Priloženi dijagrami interakcije osnovnih funkcionalnosti Logit platforme i pripadajući opis imaju za cilj stvoriti opštu sliku sistema, te tako olakšati praćenje tehničkog modela rješenja datog u nastavku. Tehnički model opisuje dosta detaljniju sliku funkcioniranja sistema i može služiti kao svojevrsan uvod u kod platforme.

Registracija korisnika



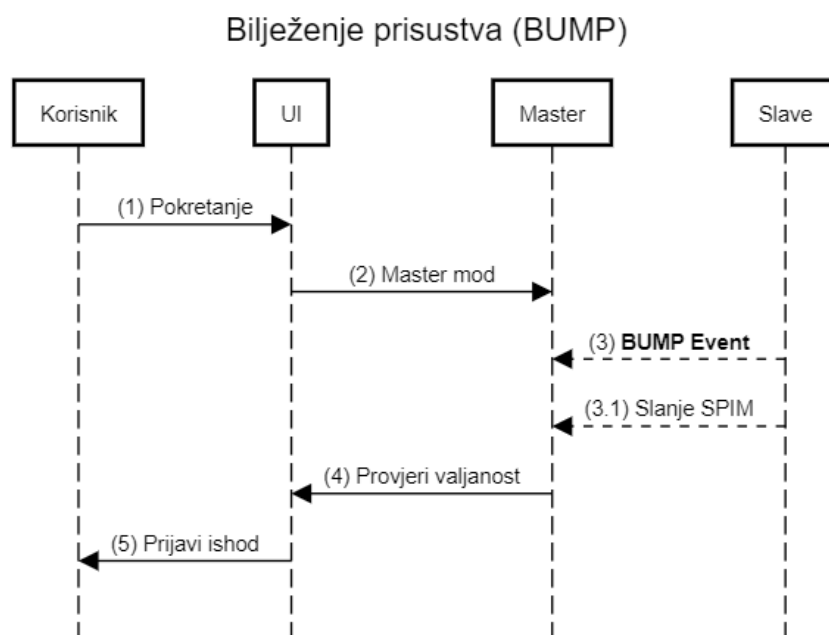
Slika 4.1: Dijagram interakcije - uspješna registracija i generisanje ključeva

Nakon uspješne instalacije aplikacije na korisnički Android uređaj (DEVICE) potrebno je obaviti proces registracije koji se izvršava u dva bitna koraka. Prvi korak sastoji se od unosa već postojećih autentifikacijskih podataka za ZAMGER sistem Elektrotehničkog fakulteta, korisnik se korištenjem datih podataka posredstvom LAPI servisa autentificira na ZAMGER sistemu, bitno je napomenuti da Logit platforma ne sprema korisničku lozinku ZAMGER sistema, navedeni podaci se koriste isključivo za povezivanje postojećeg identiteta i novogenerisanog para korisničkih RSA ključeva (KEYS), što je ujedno i drugi korak u procesu registracije na Logit platformu.

U slučaju uspješne autentifikacije, korisnika se obavještava o završenoj registraciji te se preusmjerava na glavni ekran za bilježenje prisustva. Generisani javni ključ (CERT) i identifikacioni podaci korisnika spremaju se u LAPI direktorij korisničkih certifikata.

Bilježenje prisustva studenta

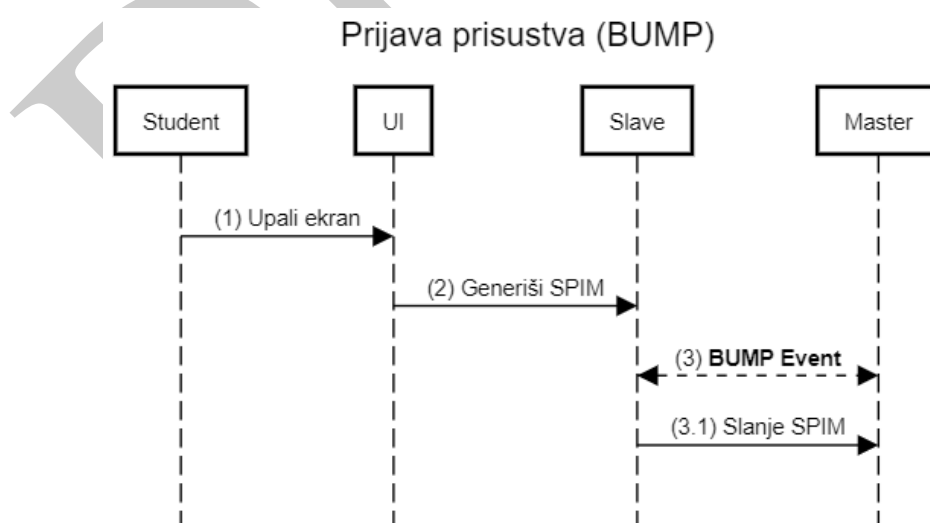
Bilježenje prisustva studenata od strane predmetnog nastavnika izvodi se u Master (M) modu funkcionisanja aplikacije, aplikacija se pri samom pokretanju i nakon uspješno obavljene registracije automatski stavlja u takav mod operacije i u njemu ostaje sve dok je upaljen ekran korisničkog uređaja (DEVICE) i Logit aplikacija (UI) se izvršava u prednjem planu (*en. foreground*), navedene zahtjeve diktira sama Android platforma.



Slika 4.2: Dijagram interakcije - bilježenje prisustva studenata (Master BUMP)

Ukoliko su ispunjeni prethodno pobrojani zahtjevi, dovoljno je da student sa podešenom Logit aplikacijom na svom uređaju prinese slave (S) uređaj master (M) uređaju i da njegovo prisustvo bude zabilježeno i prikazano na ekranu M uređaja. Samu interakciju (BUMP) inicira studentski S uređaj. Prilikom ovog BUMP događaja dolazi do razmjene kriptografski potpisanih podataka o vremenu i lokaciji (SPIM) sa S na M, gdje M vrši validaciju primljenih podataka poredeći studentsko vrijeme i lokaciju sa vremenom i lokacijom na M uređaju, gdje se prisustvo odbija ukoliko se ustanovi pokušaj lažiranja podataka.

Prijava prisustva



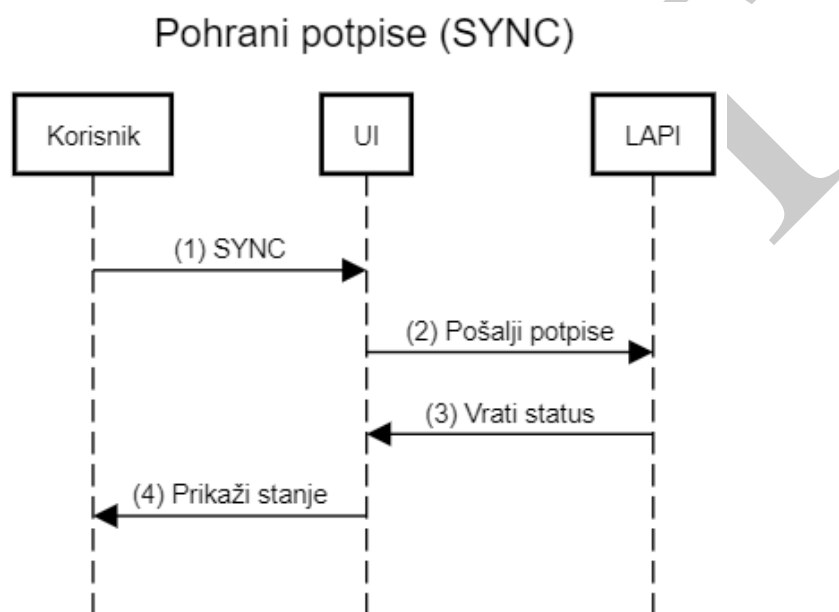
Slika 4.3: Dijagram interakcije - prijava prisustva studenta (Slave BUMP)

Studentski S uređaj i M uređaj nastavnog osoblja podešavaju se na isti način opisan iznad, jedina praktična razlika javlja se prilikom korištenja, gdje je za S uređaj čije se prisustvo bilježi

dovoljno upaliti ekran uređaja da bi se mogla ostvariti BUMP interakcija prislanjanjem S na M. Ovo je moguće jer se NFC HCE emulator Logit aplikacije izvršava u pozadini Android sistema.

Pohranjivanje potpisa sesije na LAPI

Svako bilježenje prisustva unutar Logit Android UI odvija se unutar sesije (SESS) koja se automatski započinje prilikom prvog uspješno zabilježenog prisustva i traje sve dok korisnik ne izvrši pohranu navedene sesije na LAPI servis. Klikom na SYNC dugme prikupljeni podaci šalju se LAPI servisu, provjeravaju se jedinstveni potpisi studenata te potpis ukupne sesije od strane M uređaja, ukoliko se ne pronađu nepravilnosti navedeni podaci se pohranjuju u LAPI repozitorij potpisa, takvi podaci kriptografski su osigurani od naknadne izmjene.



Slika 4.4: Dijagram interakcije - pohranjivanje potpisa na LAPI (SYNC)

Potpisi pohranjeni u LAPI repozitoriju mogu dalje biti korišteni u integrisanim aplikacijskim rješenjima koja zahtijevaju ovakvu vrstu podataka pomoću ponuđenog LAPI REST interfejsa, te se mogu smatrati relevantnim i sigurnim dokazom prisustva.

4.2 Tehnički model rješenja

Uvodi se dodatno pojam lokacijskog dokaza[19] koji u širem smislu u kontekstu podređenog korisnika (en. slave), obuhvata kriptografski potpisan korisnički identitet, korisnički uređaj, vrijeme i GPS lokacijske podatke korisničkog uređaja. Za svrhu osiguranja jedinstvenosti identiteta i vjerodostojnosti potvrde lokacijskih dokaza odabrano je korištenje RSA asimetrične enkripcije, gdje se pri uspješnoj autentifikaciji generiše jedinstveni set ključeva za korisnički uređaj, privatnom dijelu ključa nije moguće pristupiti izvan aplikacije (SEC1), niti je moguće eksportovati ključ (SEC2), a u određenom vremenskom period može postojati samo jedan valjan set ključeva za jednog korisnika jer se raniji ključevi ne uzimaju u obzir ukoliko postoji noviji set (SEC3), sprječavajući tako replikaciju identiteta na više uređaja.

Pored Android komponente aplikacije (UI) izrađena je i serverska aplikacija u programskom jeziku Python (LAPI), čija je namjena posredovanje u komunikaciji sa autentifikacijskim agentom (ZAMGER), te pohranjivanje i održavanje javnih korisničkih kriptografskih ključeva (CERT) i njihovo povezivanje sa autentifikacijskim podacima korisnika, pored toga služi i kao repozitorij za potpisana prisustva (ATTN). Na ovu komponentu se može gledati kao na integrisani namjenski repozitorij korisničkih certifikata i domenski repozitorij neporecivih i neizmjenjivih lokacijskih dokaza (SPIM).

Budući da na Elektrotehničkom fakultetu u Sarajevu postoji SSO (en. Single-Sign On) politika autentifikacije, u serverskoj komponenti (LAPI) je implementiran autentifikacijski posrednik koji prilikom prvog pokretanja aplikacije prijavljuje korisnika koristeći postojeće pristupne podatke, tom prilikom u slučaju uspješne prijave generiše se i jedinstveni set RSA ključeva dužine 2048 bita (KEYS), koji se pohranjuju na korisničkom uređaju (DEVICE), a javni dio, tj. certifikat (CERT) se pohranjuje i u repozitorij ključeva (LAPI) sa poveznicom na korisnički identitet, kasnije se ti certifikati koriste za provjeru valjanosti potpisa lokacijskih dokaza (SPIM).

Da bi se osigurala jednostavnost korištenja aplikacije odabrana je implementacija HCE emulacijskog načina rada NFC komunikatora koji omogućava korisniku da izvrši komunikaciju sa drugim uređajem bez potrebe da pokreće aplikacijski prozor na svom uređaju, dovoljno je da upali ekran svoj uređaja i prinese ga master (M) uređaju koji prikuplja potpise, u ovom slučaju drugoj instanci Logit aplikacije na kojoj je pokrenuta aktivnost za prikupljanje potpisa (LAPP).

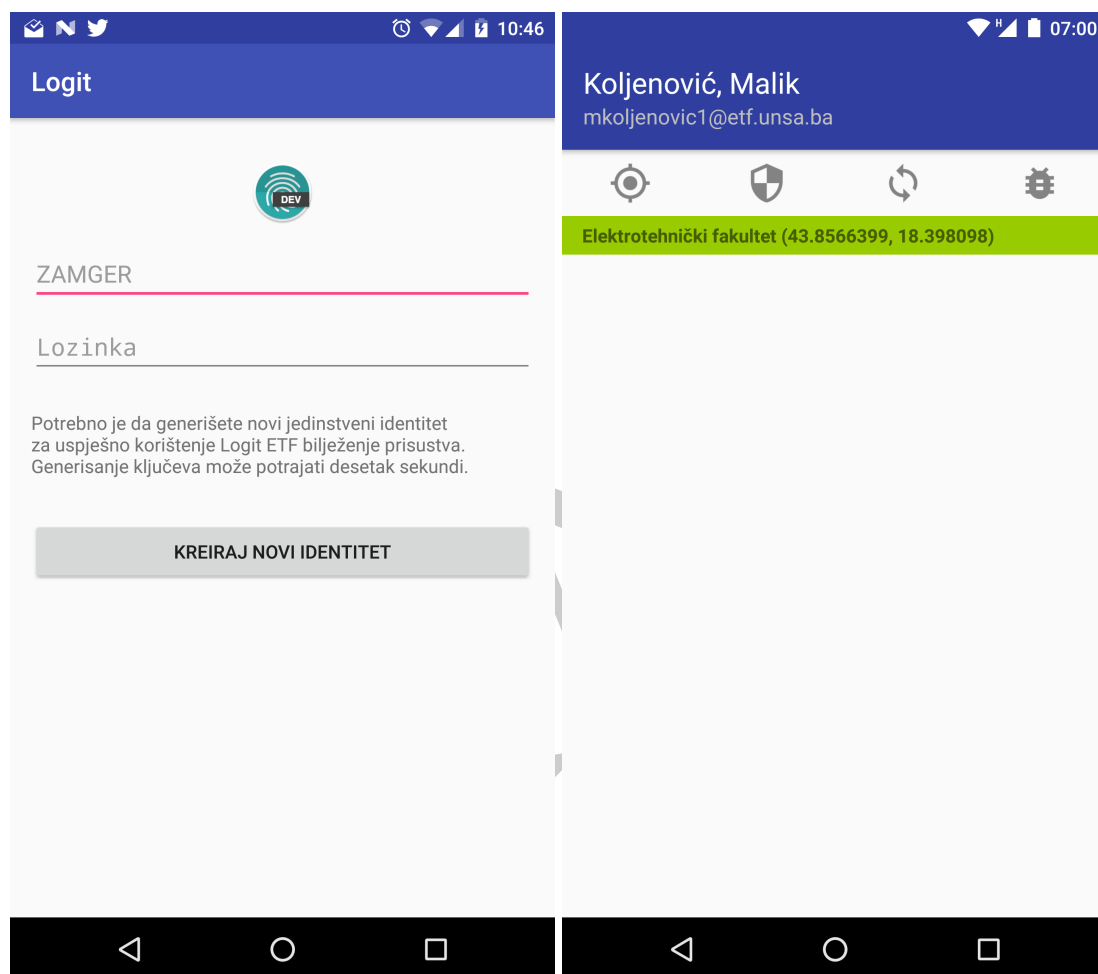
Približavanjem mobilnih uređaja (BUMP) otvara se jednosmjerni komunikacijski kanal u smjeru od slave (S) prema master (M) uređaju korištenjem ISO/IEC 14443 Tip A komunikacijskog protokola pri čemu se emulira NFC Forum Tag tipa 4 i putem NDEF Aplikacije prenosi jedna NDEF poruka (NDEFMSG) koja sadrži vremensko-lokacijski dokaz potpisan od strane korisnika, nadalje u tekstu označen kao SPIM (en. spime)[20].

Po primitku poruke nadređeni uređaj (en. master) koji osluškuje da mu se pridruže podređeni uređaji (en. slave) i ima pokrenutu Logit aplikaciju, tu poruku sprema u lokalni repozitorij potpisa ukoliko ona zadovolja uslove da očitana slave GPS lokacija nije udaljena više od 50 metara od očitane master GPS lokacije (VK1 - validacijski kriterij #1), te da podešena razlika satova master i slave uređaja nije veća od 300 sekundi (VK2), bez da nad SPIM objektom vrši ikakve izmjene, ukoliko SPIM objekat ne zadovoljava date validacijske kriterije odbija se i ispisuje se odgovarajuća poruka na master ekranu. Moguće je naknadno klikom na validacijsko dugme (ACTVAL) u korisničkom interfejsu izvršiti provjeru svih prikupljenih potpisa tokom jedne sesije (SESS), tom prilikom se, ukoliko postoji mrežna veza; svi potpisi pošalju Logit serveru (LAPI) na provjeru i vraća se stanje valjanosti potpisa za sve proslijeđene SPIM objekte.

Ukoliko master (M) želi da pohrani SPIM objekte iz jedne sesije (SESS) na Logit server (LAPI), klikom na sinhronizacijsko dugme u interfejsu (ACTSYNC), on vrši dodatno potpisivanje svakog SPIM objekta svojom komponentom privatnog ključa (MPRK), tako što potpiše hash (SHA256) vrijednost SPIM objekta (AID) sa dodatim svojim jedinstvenim master korisničkim imenom (MUSER) i jedinstvenim identifikatorom sesije (SID) i dodatno generiše SHA256 vrijednosti tih potvrda (CID), nakon čega objedinjuje sve CID vrijednosti i dodatno ih potpisuje svojim MPRK, sve te vrijednosti šalje Logit server (LAPI) na pohranjivanje, ovakvom procedurom se obezbjeđuje neporecivost i neizmjenjivost SPIM i SESS objekata, jer onemogućava

izmjene pojedinačnih SPIM objekata, te brisanje ili dodavanje objekata u finaliziranoj sesiji (SESS) od strane malicioznih aktera bez da naruši integritet SHA256 vrijednosti.

Uzimajući u obzir bitnost rješenja i visoku vjerovatnoću svakodnevne primjene kod ciljane korisničke grupe, te izazove koje takav slučaj korištenja predstavlja omogućena je i direktna e-mail komunikacija za prijavu grešaka ili slanje prijedloga sa glavnog korisničkog interfejsa (ACTBUG). Kako se radi o slojevitom i kompleksnom softverskom rješenju za više detalja referirati se na izvorni kod priložen u dodatku.



Slika 4.5: Logit UI Android prikaz korisničkog interfejsa

Poglavlje 5

Pregled korištenih tehnologija

5.1 NFC (*en. near-field communication*)

NFC skup komunikacijskih protokola omogućava uspostavu komunikacijskog kanala između dva uređaja koji se nalaze u neposrednoj blizini jedan drugog (1-4 cm) i razmjenu podataka između njih[21]. Komunikacija se odvija na način da MASTER uređaj osluškuje signal na prijemniku i u slučaju detektovanje SLAVE signala uređaja pošiljaoca, propisanog istim standardom zaprima podatke i vrši obradu nad njima, komunikacija se u većini slučajeva odvija jednosmjerno kratkim standardiziranim porukama (NDEF), no moguće je ostvariti i half-duplex komunikaciju između uređaja, kao i razmjenu nestandardnih poruka, u kojem slučaju se sam korisnik mora pobrinuti za implementaciju kompletnog komunikacijskog protokola. Potpuni detalji implementacija dati su u referencama relevantnih standarda u nastavku tehničkog pregleda, odličnu sintezu detalja i implementacije daju Igoe, Coleman i Jepson[22].

5.1.1 NXP NTAG216

U cilju zadovoljenja postavljenih funkcionalnih zahtjeva bilo je neophodno odabrati NFC Tag platformu koja će odrediti relevantne standarde pohrane binarnih podataka na uređajima kao i pripadajuće komunikacijske protokole, također dodatno su postavljeni zahtjevi ekonomičnosti implementacije i kompatibilnosti sa postojećim čitačima. Uzimajući u obzir nabrojane kriterije odabrana je platforma NTAG216 proizvođača NXP Semiconductors[1] bazirana na NFC Forum Tag tipu 2 i ISO/IEC14443 Tip A specifikaciji[23][24].

Mogućnosti navedene platforme dostatne su za ispunjenje navedenih funkcionalnih uslova, a pružaju i neke dodatne sigurnosne mehanizme - poput neizmjenjivog jedinstvenog serijskog broja svakog taga (Tag UID) potpisanog kriptografskim ključem proizvođača, navedena funkcionalnost nije implementirana u predstavljenom rješenju jer se bazira na zaštićenoj NXP tehnologiji i nije kompatibilna sa HCE emulacijom, no umnogome može doprinijeti ukupnoj sigurnosti fizičkih Tag čipova u slučaju produkcijske implementacije rješenja. U nastavku je data proizvođačka lista izdvojenih funkcionalnosti NTAG216:

- 7-bajtni UID programiran od strane proizvođača za svaki tag
- mogućnost jednokratnog programiranja i zaključavanja taga za dalje izmjene
- mogućnost read-only zaključavanja taga
- potpis originalnosti baziran na kriptografiji eliptičnih krivih
- zaštita memorijskih operacija 32-bitnom lozinkom

Page Addr		Byte number within a page				Description
Dec	Hex	0	1	2	3	
0	0h	serial number				Manufacturer data and static lock bytes
1	1h	serial number				
2	2h	serial number	internal	lock bytes	lock bytes	
3	3h	Capability Container (CC)				Capability Container
4	4h	user memory				User memory pages
5	5h					
...	...					
224	E0h					
225	E1h					
226	E2h	dynamic lock bytes			RFUI	Dynamic lock bytes
227	E3h	CFG 0				Configuration pages
228	E4h	CFG 1				
229	E5h	PWD				
230	E6h	PACK		RFUI		

aaa-008089

Slika 5.1: NTAG216 organizacija memorije[1]

5.1.2 NDEF (*en. NFC Data Exchange Format*)

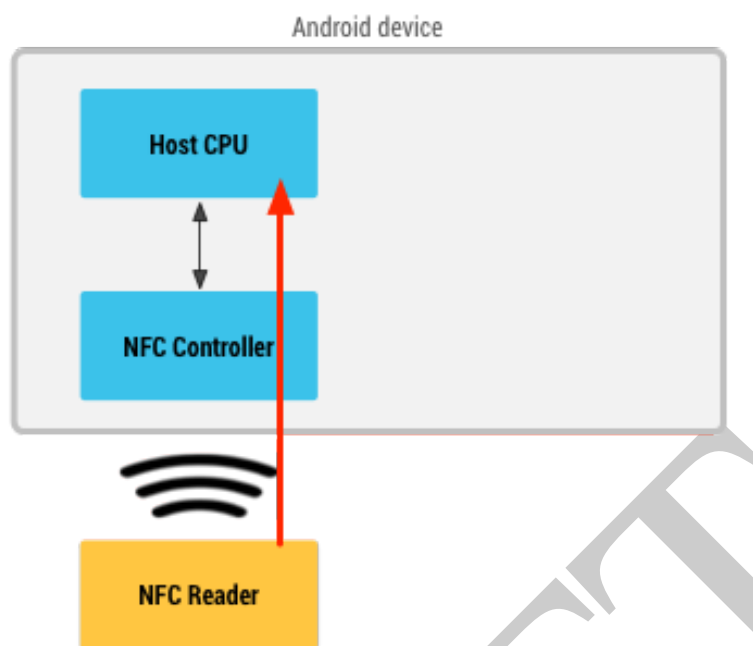
NDEF specifikacija definiše **format enkapsulacije poruke** za razmjenu informacija između dva NFC uređaja. NDEF je lagan binarni format poruke i može se koristiti za enkapsulaciju jednog ili više aplikacijski-definisanih tereta (*en. payload*) raznih vrsta i veličina unutar jedne NDEF poruke. Svaki teret opisan je od strane tipa, dužine i opcionalnog identifikatora. Identifikatori tipa mogu biti URI, MIME media tipovi, ili NFC-specifični tipovi. NDEF je striktno **format** poruke i ne poznaje pojam konekcije ili logičkog kola.[25]

Neki od ciljeva koje NDEF nastoji da ispuni:

- enkapsulacija dokumenata i binarnih objekata, slika etc.
- enkapsulacija podataka nepoznate dužine, npr. stream-a podataka
- agregacija srodnih sadržaja unutar jedne poruke
- kompaktna enkapsulacija malih datagrama

5.1.3 HCE (*en. Host card emulation*)

HCE je metod emuliranja virtuelnog identifikacijskog modula korisnika, u osnovi to je način zaobilaska hardverskih ograničenja (*en. hack, workaround*) koja onemogućavaju direktan pristup SIM (*en. Subscriber Identification Module*) modulu kod mobilnih telefonskih uređaja[26], ovakvo rješenje vuče korijene iz ekonomske realnosti sektora mobilnih komunikacija i kartičnog plaćanja, koja se najpreciznije može okarakterisati kao oligopol, naime Google je nastojao integrisati SIM karticu unutar Android operativnog sistema u vidu eSE (*en. embedded Secure Element*) korištenjem već postojeće SIM kartice operatera a u svrhu razvoja Google Wallet rješenja, no to nije odgovaralo operaterima i odbili su suradnju, nakon toga Google iznalazi alternativne načine rješenja problema poput HCE[27].



Slika 5.2: NFC HCE virtuelnog sigurnog elementa (SE)[2]

HCE na Android OS radi, kako i naziv govori u CE (*en. Card Emulation*) SLAVE modu, gdje se na svaki BUMP sa NFC čitačem odašilju pripremljeni podaci. Podaci koji se pri tome šalju moraju pratiti standard kartice koju žele emulirati i najčešće se vrši prijenos dokumenta ili datagrama unutar jednog ili više NDEF paketa. Logit koristi pristup prijenosa JSON formatiranog SPIM objekta `plain/text` unutar jednog NDEF paketa. Radi se o konceptu sa mnoštvom implementacijskih detalja, te zbog sažetosti za više detalja konsultujte oficijelnu dokumentaciju <https://developer.android.com/guide/topics/connectivity/nfc/hce>, dok je izvrstan logički prikaz sa primjerima dao Elenkov[28].

5.2 Ostalo

Dodatno koristi se Android arhitektura za dobavljanje geolokacije[29] uz reverzno geokodiranje od strane OpenStreetMap Nominatim projekta[30]. Za potpisivanje i enkripciju korisničkih podataka koristi se RSA[12] kriptografija sa 2048 bit ključem. LAPI je Python flask API i koristi UPnP, kompletan listing koda dostupan je na GitHub u repozitorijima `koljenovic/logit` i `koljenovic/logit-node`.

Poglavlje 6

Izdvojeni detalji implementacije

Detalji izdvojeni u ovom poglavlju ključni su za razumijevanje sigurnosnog modela aplikacije, sa tog aspekta posebno su zanimljiva dva objekta, *Attendance* i *Session*, koji u osnovi predstavljaju proširene kriptografski potpisane SPIM i SESS objekte.

6.1 Podatkovni i kriptografski primitivi

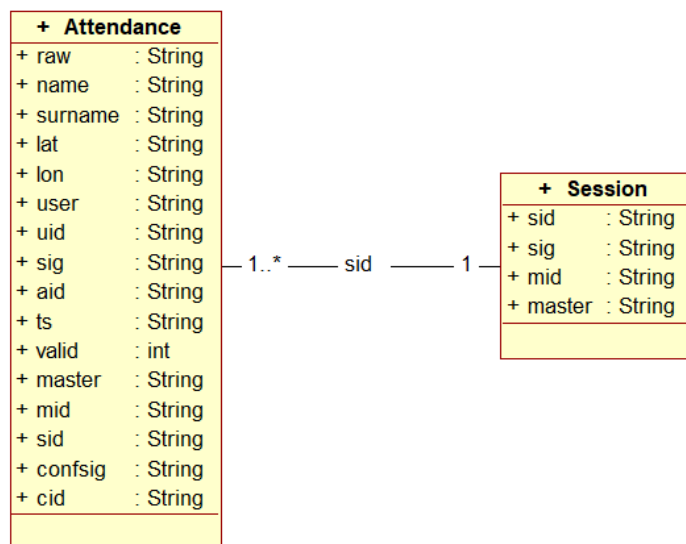
6.1.1 SPIM paket

Spim u širem smislu predstavlja vremensko-lokacijski objekat (*en. SPace-tIME*), koji korištenjem kriptografske obrade poprima karakteristike lokacijskog dokaza. Izvor za formiranje SPIM objekta sastoji se iz korisničkog imena studenta, geografske širine, geografske dužine i trenutnog vremena na studentovom mobilnom uređaju. Ovako komponovan objekat predstavlja implementaciju lokacijskog dokaza u užem smislu i koristi se dalje kao osnovni podatkovni primitiv za dalju kriptografsku obradu.

```
1 type: 'application/octet-stream'
2 Payload length: ~735 bytes
3
4 Payload data:
5
6 {
7   'lat': '43.9895212', 'lon': '18.180737', 'ts': '1511308824', 'sig':
8   '12947c41451df8ac4586ae9b211dec70087d7db0b1e25202a2b3ec07f267804
9   -d526ac641c20e282fe8bd12f7bdc9bcb2a5afcc65ef9b550e80b901c812c9a9f
10  -d87bb29828ddcd1bfd1b1d176dd963e336f4b5d5caf9ade5a3b9b60367cceb7d
11  -e807dc3dfb80d2b373993725b44e43d0116d72142cb2bf8164c4bcc4a690d949
12  -2513a72328181da2c4a0f05e57792322b7b1404ac6c21c8b71596a0cf262e044
13  -001151d90d7d9e986d50f2ec930bcaa515c04b645cea27866956f24f285abfd7
14  -6e0eb8130f67997ee7a4ded50937f1abe33ebee82da86b4c9bfeebf89d3026a
15  -423d5c09dd6ec65bd675e2ee6ea873dcc849b5f687c91a825ec010f86f238c5b',
16  'uid': '244825e54b1850b2d781bfd7cec37a302d7cf4c9ecac90e8f5e64fa09c-
17  66320a', 'name': '416a64696e', 'surname': '4d756a657a696e6f7669c487',
18  'user': 'am***'
19 }
```

Navedene vrijednosti stringova korisničkog imena studenta, geografske širine, geografske dužine i trenutnog vremena se lančaju u jedan string izloženim redoslijedom i takav string se potpisuje korištenjem RSA kriptografije, tako potpisan paket u obliku JSON objekta (prikazan

u listingu iznad) šalje se na profesorski master uređaj, gdje se dodaju podaci sesije, u vidu jedinstvenog identifikatora sesije (SID), te se potpisano studentsko prisustvo obilježava jedinstvenim heksadecimalnim identifikatorom AID izvedenim iz potpisa prisustva putem SHA256 hash funkcije, navedene vrijednosti, SID i AID se lančaju u jedan binarni string i potpisuju od strane profesora (CONFSIG), naknadno se na iz SHA256 hash vrijednosti CONFSIG profesorskog potpisa formira finalni identifikator potvrde prisustva CID, time se završava kriptografsko osiguravanje valjanosti prisustva u smislu SPIM objekta.

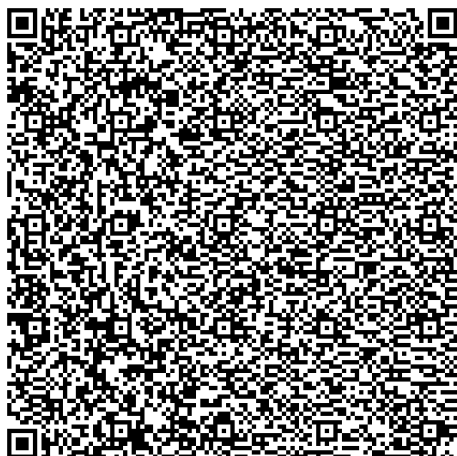


Slika 6.1: Dijagram klasa SPIM i SESS objekata

raw serijalizirana string JSON verzija objekta
name ime studenta
surname prezime studenta
lat geografska širina student
lon geografska dužina student
user ZAMGER korisničko ime studenta
uid User ID - SHA2 hex hash javnog ključa studenta
sig hex potpis SPIM-a (user:lat:lon:ts)
aid Attendance ID - hex SHA2 hash sig potpisa
ts vrijeme na studentovom uređaju
valid validation cache
master ZAMGER korisničko ime profesora
mid Master ID - SHA2 hex hash javnog ključa profesora
sid Session ID - identifikator profesorove sesije
confsig Master Conf. profesorov hex potpis (sid:aid)
cid Confirmation ID - SHA2 hex hash confsig-a

Fizički predstavljen opisani SPIM objekat manji je od 1 KB te je pored brzog NFC isl. elektronskog transfera moguće izvršiti prenos alternativnim metodama, kao posebno pogodna čini se QR kod reprezentacija[31] i prenos, koja može biti vrlo korisna u slučaju da nijedan od uređaja ne posjeduje NFC modem. Navedeni modus nije implementiran u aplikaciji i dat je kao sugestija zaobilazjenja hardverskih ograničenja, primjer QR oblika ranije datog SPIM objekta prikazan je na slici 6.2.

Dati QR prikaz je čitljiv ali je vidno uočljiva gustina zapisa koja može predstavljati problem u slučaju lošije kvalitete medija prikaza, u tom slučaju, kompletan SPIM paket moguće je značajno smanjiti zamjenom korištenog RSA kriptosistema za kriptosistem baziran na eliptičnim krivim, budući da su ključevi korišteni u tom slučaju znatno kraći[32], dužina navedenog potpisa bila bi smanjena sa 512 na minimalno 71 bajt[33] navedeni pristup nije prihvaćen u okviru ovog rada zbog povećanja kompleksnosti pokaznog sistema, no praktična implementacija moguća je bez većih programskih izmjena.



(a) oblik korištenjem RSA potpisa



(b) oblik korištenjem ECC potpisa

Slika 6.2: QR oblik SPIM objekta

6.1.2 SESS paket

Dodatno se za SESS objekat prilikom finaliziranja sesije na LAPI serveru vrši prikupljanje svih CID potpisa koji pripadaju datoj sesiji, te se CID vrijednosti ulančane hronološkim redoslijedom potpisuju LAPI ključem koji se nalazi samo na LAPI hardverskom uređaju, stoga je sigurnost LAPI servera od ključne važnosti za sigurnost ukupnog sistema. Ovako potvrđena sesija ne može biti naknadno mijenjana, lažirana ili porećena izvan LAPI izvršnog okruženja.

6.2 Pregled implementacije

6.2.1 MainActivity

Nakon prvobitnog pokretanja aplikacije a za daljnje uspješno korištenje neophodno je izvršiti autentifikaciju korisnika putem nekog već postojećeg korisničkog repozitorija, te generisati pripadajući virtualizirani sigurni element. Navedene aktivnosti izvršavaju se unutar MainActivity glavnog početnog prozora Logit aplikacije, prikaz relevantnog dijela koda za generisanje virtualiziranog sigurnog elementa dat je u listingu ispod.

```

1      KeyPairGenerator kpg = KeyPairGenerator.getInstance (
2          "RSA", "AndroidKeyStore");
3      Calendar start = Calendar.getInstance();
4      Calendar end = Calendar.getInstance();
5      end.add(Calendar.YEAR, 1);
6
7      KeyPairGeneratorSpec spec =
8          new KeyPairGeneratorSpec.Builder(this).setAlias("etf_logit_" +
9              ts)
10              .setKeySize(2048)
11              .setSubject(new X500Principal("CN=users.etf.ba"))
12              .setSerialNumber(BigInteger.valueOf(tsLong))
13              .setStartDate(start.getTime()).setEndDate(end.getTime()).build();
14
15      kpg.initialize(spec);
16
17      KeyPair kp = kpg.generateKeyPair();
18
19      KeyStore ks = KeyStore.getInstance("AndroidKeyStore");
20      ks.load(null);

```

Sigurnosni element generisan kao u primjeru iznad dalje se pohranjuje na korisničkom uređaju, gdje privatni dio nikada ne napušta uređaj i dostupan je isključivo Logit aplikaciji putem Android KeyStore providera. Javni dio se koristi kao dio identifikatora korisnika, te se dodatno pohranjuje i na Logit API korisnički repozitorij za potrebe identifikacije i verifikacije potpisa lokacijskog paketa.

6.2.2 LogitAPDUService

Ekstenzija Androidovog native interfejsa HostApduService koji za instaliranu aplikaciju sa android.permission.BIND_NFC_SERVICE permisijom vrši pokretanje HCE emulatora prilikom svakog starta operativnog sistema, emulator se u ovom slučaju ponaša kao generički NFC NTAG sa korisnički programiranom memorijom u obliku NDEF poruke koja prenosi jedinstveni potpisan studentski lokacijski dokaz. Navedena servisna komponenta aplikacije aktivna je svaki put dok je i ekran uređaja aktivan ili dok korisnik sam ne zaustavi pripadajući servis. Navedene funkcionalnosti postižu se uključivanjem dijela koda datog u nastavku unutar <application> direktive manifest fajla.

```

1      <service
2          android:name=".LogitApduService"

```

```

3         android:permission="android.permission.BIND_NFC_SERVICE">
4         <intent-filter>
5             <action
6             ↪ android:name="android.nfc.cardemulation.action.HOST_APDU_SERVICE" />
7             <category android:name="android.intent.category.DEFAULT" />
8         </intent-filter>
9
10        <meta-data
11            android:name="android.nfc.cardemulation.host_apdu_service"
12            android:resource="@xml/apduservice" />
13    </service>

```

Nadalje LogitAPDUService sadrži logiku za ispravno konstruisanje i formatiranje NDEF paketa[34] lokacijskog dokaza i njegovo potpisivanje, te ostalu neophodnu kriptografsku obradu. U nastavku će biti dat prikaz kompletnog servisa sa komentarima relevantnih dijelova.

```

1    final static int APDU_INS = 1;
2    final static int APDU_P1 = 2;
3    final static int APDU_P2 = 3;
4    final static int APDU_SELECT_LC = 4;
5    final static int APDU_READ_LE = 4;
6    final static int FILEID_CC = 0xe103;
7    final static int FILEID_NDEF = 0xe104;
8    final static byte INS_SELECT = (byte) 0xa4;
9    final static byte INS_READ = (byte) 0xb0;
10   final static byte INS_UPDATE = (byte) 0xd6;
11   final static byte P1_SELECT_BY_NAME = (byte) 0x04;
12   final static byte P1_SELECT_BY_ID = (byte) 0x00;
13   final static int DATA_OFFSET = 5;
14
15   final static byte[] DATA_SELECT_NDEF = {(byte) 0xd2, (byte) 0x76,
16   ↪ (byte) 0x00, (byte) 0x00, (byte) 0x85, (byte) 0x01, (byte) 0x01};
17   final static byte[] RET_COMPLETE = {(byte) 0x90, (byte) 0x00};
18   final static byte[] RET_NONDEF = {(byte) 0x6a, (byte) 0x82};
19   final static byte[] FILE_CC = {
20       (byte) 0x00, (byte) 0x0f,           // CCLen - CC container size
21       (byte) 0x20,           // Mapping version
22       (byte) 0x04, (byte) 0xff,       // MLe - max. read size
23       (byte) 0x08, (byte) 0xff,       // MLc - max. update size
24
25       // TLV Block (NDEF File Control)
26       (byte) 0x04,           // Tag - Block type
27       (byte) 0x06,           // Length
28       (byte) 0xe1, (byte) 0x04,       // File identifier
29       (byte) 0x04, (byte) 0xff,       // Max. NDEF file size
30       (byte) 0x00,           // R permission
31       (byte) 0x00,           // W permission
32   };

```

Deklariše konstantne vrijednosti brojnih dijelova neophodnih za konstrukciju standardne NDEF poruke, između ostalog capability fajl koji predstavlja svojevrsno zaglavlje NDEF paketa.

Metoda `generateSignature` instancira lokacijske servise i vrši konstrukciju paketa lokacijskog dokaza kao i kriptografskih primitiva neophodnih za potpisivanje istog. Za konstrukciju lokacijskog dokaza neophodno je pribaviti trenutno vrijeme uređaja, to je prikazano u narednom dijelu koda.

```
1 Long tsLong = System.currentTimeMillis() / 1000;
2 String ts = tsLong.toString();
3 byte[] signature;
```

Nakon toga vrši se instanciranje i učitavanja Android KeyStore objekta koji sadrži korisnički par ključeva neophodnih za potpisivanje lokacijskog paketa.

```
1 KeyStore ks = KeyStore.getInstance("AndroidKeyStore");
2 ks.load(null);
3 KeyStore.ProtectionParameter pp = new
↳ KeyStore.PasswordProtection(null);
```

Nadalje iz KeyStore objekta učitava se najažurniji virtualizirani sigurnosni element.

```
1 Enumeration<String> aliases = ks.aliases();
2 String alias = aliases.nextElement();
3 Entry entry = ks.getEntry(alias, pp);
```

Zbog potrebe za što kompaktnijim prenosom podataka za sve vrijednosti gdje je to moguće generišu se hash preslikavanja koja se kasnije koriste za verifikaciju i pohranjivanje. Za potrebe generisanja hash vrijednosti instancirase SHA-256 MessageDigest objekat.

```
1 MessageDigest md = MessageDigest.getInstance("SHA-256");
```

Iz virtualiziranog sigurnog elementa dalje učitavamo korisnički certifikat sa javnim ključem, te za potrebe verifikacije identiteta heksadecimalnu reprezentaciju njegove hash vrijednosti pripremamo za uključenje u paket lokacijskog dokaza.

```
1 Certificate c = ks.getCertificate(alias);
2 byte [] pubKey = c.getPublicKey().getEncoded();
3 md.update(pubKey, 0, pubKey.length);
4 byte [] pubKeyHash = md.digest();
5 String pubKeyHashString = LogitApplication.toHext(pubKeyHash);
```

Zatim iz virtualiziranog sigurnog elementa korištenjem korisničkog privatnog ključa pripremamo objekat koji vrši potpisivanje lokacijskog paketa.

```
1 Signature s = Signature.getInstance("SHA256withRSA");
2 s.initSign(((PrivateKeyEntry) entry).getPrivateKey());
3 SharedPreferences userData = getSharedPreferences("UserData", 0);
```

Dio lokacijskog paketa koji je pokriven korisničkim digitalnim potpisom sadrži korisničko ime, lokacijske parametre geografske dužine i širine, te vrijeme uređaja u trenutku kreiranja digitalnog potpisa.

```
1 String sigPkg = userData.getString("user", "unknown") +
2     ":" + location.getLatitude() +
3     ":" + location.getLongitude() +
4     ":" + ts;
```

Takav paket se potpisuje i bilježi se njegova SHA-256 preslikana vrijednost za potrebe verifikacije.

```
1 s.update(sigPkg.getBytes("UTF-8"));
2 signature = s.sign();
```

Osnovni potpisani lokacijski paket se proširuje vrijednostima generisanih SHA-256 preslikavanja i osnovnim korisničkim podacima, te se proslijeđuje metodi `createMessage` koja formira standardizovan NDEF paket, takav spreman NDEF paket stavlja se na raspolaganje komponenti za prenos putem NFC protokola.

```
1 msg = "{\\"lat\\":\\" + location.getLatitude() +
2     \\", \\"lon\\":\\" + location.getLongitude() +
3     \\", \\"ts\\":\\" + ts +
4     \\", \\"sig\\":\\" + LogitApplication.toHext(signature) +
5     \\", \\"uid\\":\\" + pubKeyHashString +
6     \\", \\"name\\":\\" +
7     LogitApplication.toHext(userData.getString("name",
8     "unknown").getBytes("UTF-8")) +
9     \\", \\"surname\\":\\" +
10    LogitApplication.toHext(userData.getString("surname",
11    "unknown").getBytes("UTF-8")) +
12    \\", \\"user\\":\\" + userData.getString("user", "unknown") +
13    \\"}";
14
15 NdefMessage ndef = createMessage(msg.getBytes("UTF-8"));
16 byte[] ndefarray = ndef.toByteArray();
17
18 mNdefFile = new byte[ndefarray.length + 2];
19
20 mNdefFile[0] = (byte) ((ndefarray.length & 0xff00) >> 8);
21 mNdefFile[1] = (byte) (ndefarray.length & 0x00ff);
22
23 System.arraycopy(ndefarray, 0, mNdefFile, 2, ndefarray.length);
24
25 logitApp.setMessage(mNdefFile);
```

6.2.3 processCommandApdu

Budući da se u prikazanom slučaju vrši emulacija pasivnog NTAG uređaja, APDU servis se izvršava u slave modu i zaprima instrukcije od strane master uređaja, neophodno je implementirati

parser petlju i logiku za komunikaciju sa master uređajem, unutar koje se vrši prepoznavanje zadatih instrukcija i priprema adekvatan odgovor, navedena logika implementirana je unutar `processCommandApu` metode. Kompletan logika emulacije tag uređaja svodi se na proslijeđivanje adekvatnog zaglavlja taga i READ FROM TO binarnog protokola za čitanje memorije koju šalje master, stoga glavnu navedene metode čini jedna switch petlja koja u skladu sa zadatom komandom vraća zaglavlje ili raspon bita emuliranog taga, u ovom slučaju sadržaj koji se emulira je prošireni lokacijski paket iznad. Detalje opisane metode možete pogledati u prilogu koda u dodatku.

6.2.4 AttendanceActivity

Za ostvarivanje pune funkcionalnosti aplikacije neophodna je bila implementacija master moda za prikupljanje i obradu NDEF paketa, u ovom slučaju korisničkih potpisa, enkapsuliranih u obliku potpisanog lokacijskog paketa. `AttendanceActivity` vrši navedenu funkcionalnost te dodatno vrši provjeru valjanosti potpisa i podataka korisničkih lokacijskih paketa. Verifikacija korisničkih lokacijskih paketa vrši se tako što se korisnički slave podaci o vremenu i lokaciji porede sa podacima o vremenu i lokaciji master uređaja, time se osigurava zaštita od napada lažiranja podataka, te bi za takvu vrstu prevare bila neophodna koluzija dva aktera suprostavljenih interesa, što znatno umanjuje vjerovatnoću takvih napada. Moguće je podešiti vrijednosti dozvoljenih odstupanja verifikacijskih parametara izmjenom koda za tu namjenu datog u nastavku.

```

1      final long timediff = System.currentTimeMillis() / 1000 -
↳ Long.parseLong(tmpAttn.getTs());
2      final Location userLocation = new Location("MOCK");
3      userLocation.setLatitude(Double.valueOf(tmpAttn.getLat()));
4      userLocation.setLongitude(Double.valueOf(tmpAttn.getLon()));
5      if (Build.VERSION.SDK_INT >= 23
6          && ContextCompat.checkSelfPermission(that,
↳ android.Manifest.permission.ACCESS_FINE_LOCATION) ==
↳ PackageManager.PERMISSION_GRANTED
7          && ContextCompat.checkSelfPermission(that,
↳ android.Manifest.permission.ACCESS_COARSE_LOCATION) ==
↳ PackageManager.PERMISSION_GRANTED
8          || Build.VERSION.SDK_INT < 23) {
9          FusedLocationProviderClient mFusedLocationClient =
↳ LocationServices.getFusedLocationProviderClient(that);
10         mFusedLocationClient.getLastLocation().addOnSuccessListener(new
↳ OnSuccessListener<Location>() {
11             @Override
12             public void onSuccess(Location location) {
13                 Float locdiff = userLocation.distanceTo(location);
14                 if (Math.abs(timediff) < 300) {
15                     if (locdiff < 100) {
16                         // Lokacija i vrijeme SLAVE uređaja su VALIDNI
17                     } else {
18                         Toast.makeText(that, "Greška: lokacije udaljene " +
↳ locdiff.intValue() + " metara.", Toast.LENGTH_LONG).show();
19                     }
20                 } else {
21                     Toast.makeText(that, "Greška: vrijeme nije tačno ili je
↳ TAG zastario.", Toast.LENGTH_LONG).show();
22                 }

```

Da bi se izbjegla obaveza reimplementiranja NDEF protokola za podatke primljene putem NFC podatkovnog interfejsa Android nudi predefinisani intent filter za direktnu manipulaciju NDEF poruka, te je za njegovo korištenje potrebno dodati ispod prikazani kod unutar manifest fajla Android aplikacije. Korištenjem ovog filtera programer kao rezultat uspješnog NFC prenosa dobija standardizovanu NDEF poruku spremnu za obradu. Ovaj interfejs se koristi unutar AttendanceActivity.

```
1 <intent-filter>
2     <action android:name="android.nfc.action.NDEF_DISCOVERED" />
3
4     <category android:name="android.intent.category.DEFAULT" />
5
6     <data android:mimeType="application/octet-stream" />
7 </intent-filter>
```

Ostatak koda unutar AttendanceActivity klase koristi se za prikaz i obradu elemenata korisničkog interfejsa master moda za prikupljanje korisničkih potpisa.

Poglavlje 7

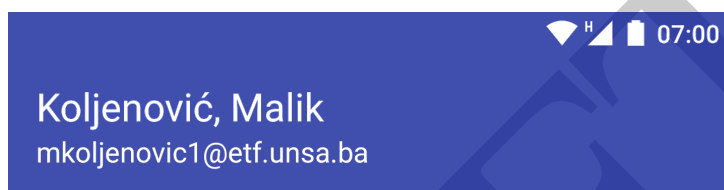
Korisničko uputstvo

1. **Uspostavite internet konekciju** prema uputama vašeg instruktora.
 - (a) potrebno je da na mreži bude dostupna Logit serverska aplikacija i certifikacijski repozitorij za uspješnu prijavu i korištenje, dostupnost možete provjeriti posjetom na <https://logit.mine.nu:5000>
2. **Uključite lokacijske usluge** vašeg Android mobilnog uređaja.
 - (a) detaljno uputstvo možete pronaći na <https://support.google.com/accounts/answer/3467281?hl=hr>
3. **Omogućite NFC komunikaciju** na vašem Android mobilnom uređaju i **isključite Android Beam** uslugu za optimalan rad aplikacije.
 - (a) Settings > NFC > Enable
 - (b) Settings > NFC > Android Beam > Disable
 - (c) više detalja za navedene postavke pročitajte na <https://support.google.com/nexus/answer/2781895?hl=hr>
4. Ukoliko niste, **omogućite sigurnosnu funkcionalnost zaključavanja vašeg ekrana**
 - (a) Android OS nudi usluge integrisane sigurnosti mobilnih uređaja, te je Keystore funkcionalnost sigurnog pohranjivanja privatnih ključeva usko vezana za ostale sigurnosne postavke, stoga omogućite zaključavanje ekrana slijedeći uputstvo na <https://support.google.com/nexus/answer/2819522?hl=hr>
5. Prihvatite poziv za alpha testiranje posjetom na <https://play.google.com/apps/testing/ba.unsa.etf.logit> i nastavite na Play Store te **instalirajte aplikaciju**
 - (a) ukoliko vaš mobilni uređaj nije izlistan kao podržan obratite se vašem instrukturu i biti će vam izdat jedinstveni NFC Certifikat, koji ćete koristiti za bilježenje prisustva
6. **Pokrenite Logit aplikaciju**
7. **Unesite vaše ZAMGER korisničke podatke**, ovi podatci koriste se jednokratno za provjeru valjanosti identiteta prije generisanja vašeg para ključeva, vaša lozinka ne ostaje pohranjena na Logit sistem i prenosi se https kanalom prema ZAMGER servisu

8. Aplikacija je spremna za korištenje i ne mora biti pokrenuta u prednjem planu za prijavu prisustva, **za optimalne rezultate** dovoljno je da upalite ekran vašeg Android uređaja na "lock screen" i prislonite na Android uređaj instruktora.
9. **Ukoliko želite koristiti aplikaciju u instruktorskom modu** i prikupljati prisustvo, dovoljno je da pokrenete Logit aplikaciju u prednjem planu te prislonite vaš uređaju studentskom uređaju u skladu sa korakom 8.

7.1 Instruktorski način rada

Pokretanjem glavnog prozora Logit aplikacije ulazite u mod za prikupljanje studentskih potpisa prisustva. Ovaj zaslon podijeljen je na četiri komponente, opisi kako slijedi u nastavku.



Slika 7.1: Zaglavlje aplikacije prikazuje aktivnog korisnika

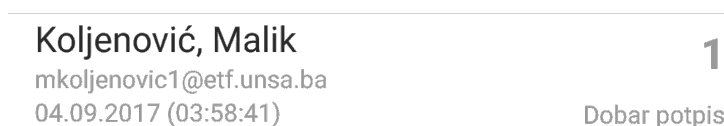


Slika 7.2: Glavni izbornik, opisi funkcionalnosti u nastavku

- Dugme 1** služi za ručno osvježavanje trenutne lokacije
- Dugme 2** koristite za provjeru valjanosti ključeva korištenih pri potpisu
- Dugme 3** sinhronizacija trenutne sesije na Logit server, svi potpisi se pohranjuju u jednu sesijsku cjelinu i brišu sa mobilnog uređaja (kreira se nova sesija)
- Dugme 4** otvara e-mail klijent po izboru korisnika u cilju lakše prijave grešaka



Slika 7.3: Trenutno zabilježena lokacija korisničkog uređaja



Slika 7.4: Ordinalno numerisan spisak prisutnih studenata

Poglavlje 8

Sigurnosna analiza

Sa aspekta teorije igara[35] u okruženju dva igrača unutar kooperativne igre[36] pronalazak optimalne strategije je uvijek prost u slučaju da međusobna saradnja maksimizira profit oba igrača, stoga ukoliko se prikupljanje obostrano korisnih dokaza u vidu prisustva predavanju svede u navedene okvire tada se radi o kolaborativnoj igri studenta i predavača te ne bi trebao da se javlja problem međusobnog povjerenja, dovoljan uvjet za sigurnost takvog sistema je ispravna tehnička implementacija kriptografskih rješenja koja sprječavaju napade jednostranim falsifikovanjem dokaza. Aplikacija izrađena u okviru ovog rada zadovoljava preduvjete kolaborativne igre **dva igrača**, budući da je prisustvo studenata dokaz prisustva profesora i obratno, njihov odnos je kriptografski osiguran strukturom nalik na hijerarhiju uređenog hash skupa, takva struktura osigurava osobine neizmjenjivosti i neporecivosti.

Sa tehničkog aspekta jedan vektor napada predstavlja lažiranje lokacije ili vremena na korisničkim uređajima, stim što je preduvjet za uspješnost takvog napada koluzija predavača i studenta da priskrbe korist na štetu treće strane, tj. institucije korisnice sistema i neučesnika u koluziji. Moguće je otežati izvodivost i osigurati detekciju ovog napada programskim mjerama zabrane i bilježenja korištenja ručno podešene (*en. mock*) lokacije uređaja i provjerom vremenskih podataka potpisa na LAPI strani koja osigurava pouzdane vremenske podatke. Dodatno treba napomenuti da cijena opisanog napada raste proporcionalno broju studenata učesnika jer svi studenti koji žele priskrbiti neostvarenu korist moraju učestvovati u koluziji, također svi neučesnici imaju štetu (npr. predavanje nije održano u predviđenom terminu zbog odsustva predavača, naknadno falsifikovan dokaz o održavanju u koluziji sa dva studenta, svi ostali studenti gube bodove za prisustvo), stoga i direktnu korist od razotkrivanja napada.

Sigurnost izloženog rješenja polazi i počiva na pretpostavci povjerljivosti i jedinstvenosti **korisničkog privatnog ključa** koji nikada ne napušta okruženje emuliranog sigurnosnog elementa korisničkog Android uređaja, ukoliko se naruši data pretpostavka ne postoje garancije sigurnosti sistema. Postoji jedan-na-jedan asocijacija između korisnika i sigurnosnog uređaja, gdje se zbog prirode problema i niskog nivoa prijetnje kao dovoljan uvjet asocijacije uzima posjedovanje uređaja, no takav uvjet ne može se smatrati dovoljnim za čvrst dokaz identiteta te se za namjene gdje je to neophodno preporučuje implementacija dodatnog faktora biometrijske identifikacije unutar pouzdanog okruženja (ne na korisničkom uređaju).

Poglavlje 9

Zaključak

Aplikacija izrađena u okviru ovog rada zadovoljava zahtjeve navedene u postavci zadatka i pripadajućem opisu. Korištene su savremene kriptografske metode za implementaciju sigurnosno osjetljivih funkcionalnosti i osigurano je stabilno okruženje za neometano funkcionisanje aplikacije, dodatno je prema zahtjevima uspješno realiziran NFC komunikacijski interfejs između studentskih i instruktorskih mobilnih uređaja. U cilju lakšeg skaliranja težilo se je što više koristiti standardizovane tehnologije, posebno kada je u pitanju NFC, gdje je dodatno implementirana emulacija NTAG vrste taga kao NDEF medija, time je omogućeno da se sistem u budućnosti prilagodi stacionarnim NFC čitačima i korištenju samostalnih NFC tagova.

Pokušana je pilot primjena sistema u saradnji sa nastavnim osobljem na predmetu "Tehnologije sigurnosti" u školskoj godini 2017/18. kojom prilikom je sačinjen spisak studenata i izvršene pripreme sistema. Navedena pilot okončan je neuspješno zbog sigurnosnih problema u integraciji sa postojećim sistemima, nedostatka resursa i nepostojanja pokusnog sistema pogodnog za projekte u ranoj fazi testiranja, u cilju povećanja inovativnosti i razvoja novih usluga preporučuje se izrada paralelnih pokusnih sistema odvojenih od produkcijskog u okviru Elektrotehničkog fakulteta u Sarajevu.

Tokom pripreme pilot primjene identifikovano je da značajan broj studenata ne posjeduje NFC omogućene mobilne uređaje, te su za njihove potrebe izrađene NTAG216 NFC token naljepnice, no primjena navedenih tokena uvjetovana je dodatnim istraživanjem i doradom Logit sistema za rad sa NTAG216 da bi osigurao isti ili viši nivo sigurnosnih garancija od onog koje pruža Android izvršno okruženje. Kao dodatna smjernica u istraživanju dat je prijedlog korištenja QR kodova za namjenu supstitucije u slučajevima nepostojanja tehničkih predispozicija za upotrebu sistema na strani korisnika, navedena tehnologija može dati dobre rezultate u praktičnoj primjeni i zavređuje dalji istraživački tretman.

Krajnja težnja Logit rješenja je obuhvatanje cjelokupnog sistema autentifikacije i modeliranja relacija povjerenja u materijalnopravnom okruženju, kroz izradu proširive bazne platforme koja može obuhvatiti digitalizaciju mnoštva svakodnevnih administrativnih zadataka jedne institucije, sa tim ciljem daljnje istraživačke napore zavređuje usmjeriti ka razvoju stabilne PKI infrastrukture, kao i digitalizaciji vjerodostojnih institucionalnih registara poput registra ispita sa ciljem digitalizacije studentskog indeksa i srodnih dokumenata.

Literatura

- [1] NXP Semiconductors, "NTAG213/215/216 NFC Forum Type 2 Tag compliant IC with 144/504/888 bytes user memory", NXP Semiconductors, Tech. Rep. October, 2011, dostupno na: http://www.jp.nxp.com/documents/short{_}data{_}sheet/NTAG203{_}SDS.pdf
- [2] Google, - <https://developer.android.com/guide/topics/connectivity/nfc/hce>, dostupno na: <https://developer.android.com/guide/topics/connectivity/nfc/hce> Pristupano: 2018-09-14. 2018.
- [3] EU, "Uredba (EU) 2016/679 Europskog parlamenta i Vijeća od 27. travnja 2016. o zaštiti pojedinaca u vezi s obradom osobnih podataka i o slobodnom kretanju takvih podataka te o stavljanju izvan snage Direktive 95/46/EZ (Opća uredba o zaštiti podataka)", str. 88, dostupno na: <http://data.europa.eu/eli/reg/2016/679/oj> 2016.
- [4] Searle, J. R., Willis, S. *et al.*, The construction of social reality. Simon and Schuster, 1995.
- [5] Cahill, V., Gray, E., Seigneur, J. M., Jensen, C. D., Chen, Y., Shand, B., Dimmock, N., Twigg, A., Bacon, J., English, C., Wagealla, W., Terzis, S., Nixon, P., Di Marzo Seru- gendo, G., Bryce, C., Carbone, M., Krukow, K., Nielsen, M., "Using trust for secure col- laboration in uncertain environments", IEEE Pervasive Computing, Vol. 2, No. 3, 2003, str. 52–61.
- [6] ISO/IEC, "27001: 2013", International Organization for Standardization, Standard, 2013.
- [7] Ferguson, N., Schneier, B., Kohno, T., Cryptography engineering: design principles and practical applications. John Wiley & Sons, 2011.
- [8] Singh, S., The code book: the science of secrecy from ancient Egypt to quantum crypto- graphy. Anchor, 2000.
- [9] Katz, J., Menezes, A. J., Van Oorschot, P. C., Vanstone, S. A., Handbook of applied cryptography. CRC press, 1996.
- [10] Merkle, R. C., "Secure communications over insecure channels", Communications of the ACM, Vol. 21, No. 4, 1978, str. 294–299.
- [11] Diffie, W., Hellman, M., "New directions in cryptography", IEEE transactions on Infor- mation Theory, Vol. 22, No. 6, 1976, str. 644–654.
- [12] Rivest, R. L., Shamir, A., Adleman, L., "A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM, Vol. 21, No. 2, 1978, str. 120– 126.

- [13] Buchmann, J. A., Karatsiolis, E., Wiesmaier, A., Introduction to public key infrastructures. Springer Science & Business Media, 2013.
- [14] Assembly, U. G., "Universal declaration of human rights", UN General Assembly, 1948.
- [15] , - <https://wikileaks.org/>, dostupno na: <https://wikileaks.org/> Pristupano: 2019-02-08. 2016.
- [16] Kaliski, B., "Twirl and rsa key size", 2003.
- [17] "Android statistics dashboard", - <https://developer.android.com/about/dashboards/>, pristupano: 2017-09-01.
- [18] "Nfc world - adoption statistics", - <https://nfcworld.com/2014/02/12/327790/two-three-phones-come-nfc-2018/>, pristupano: 2017-09-11. 2014.
- [19] Khan, R., Zawoad, S., Haque, M. M., Hasan, R., "Who, when, and where? location proof assertion for mobile devices", in IFIP Annual Conference on Data and Applications Security and Privacy. Springer, 2014, str. 146-162.
- [20] Sterling, B., Wild, L., Shaping things. The MIT Press, 2005.
- [21] NFC Forum, "NFC Digital Protocol Technical Specification", NFC Forum, Standard, 2010, dostupno na: <http://www.nfc-forum.org/specs/>
- [22] Igoe, T., Coleman, D., Jepson, B., Beginning NFC. O'Reilly, 2014.
- [23] NFC Forum, "Type 2 Tag Operation Specification", NFC Forum, Standard, 2011, dostupno na: http://apps4android.org/nfc-specifications/NFCForum-TS-Type-4-Tag{_}2.0.pdf
- [24] ISO/IEC, "ISO/IEC 14443-1:2018(en) 1-4 - Cards and security devices for personal identification", International Organization for Standardization, Geneva, CH, Standard, 2018.
- [25] NFC Forum, "NFC Data Exchange Format (NDEF)", NFC Forum, Standard, 2006.
- [26] Elenkov, N., "Accessing the embedded secure element in android 4.x", - <https://nelenkov.blogspot.com/2012/08/accessing-embedded-secure-element-in.html>, dostupno na: <https://nelenkov.blogspot.com/2012/08/accessing-embedded-secure-element-in.html> Pristupano: 2018-09-14. 2012.
- [27] C. Paya, - <https://randomoracle.wordpress.com/2014/08/12/fakeid-android-nfc-stack-and-google-wallet-part-i/>, dostupno na: <https://randomoracle.wordpress.com/2014/08/12/fakeid-android-nfc-stack-and-google-wallet-part-i/> Pristupano: 2018-09-14. 2014.
- [28] Elenkov, N., Android Security Internals. no starch press, 2015.
- [29] Google, - <https://developer.android.com/guide/topics/location/strategies>, dostupno na: <https://developer.android.com/guide/topics/location/strategies> Pristupano: 2018-09-14. 2018.
- [30] OpenStreetMaps, - <https://wiki.openstreetmap.org/wiki/Nominatim>, dostupno na: <https://wiki.openstreetmap.org/wiki/Nominatim> Pristupano: 2018-09-14. 2018.

- [31] Soon, T. J., "Qr code", Synthesis Journal, Vol. 2008, 2008, str. 59–78.
- [32] Maletsky, K., "RSA vs ECC Comparison for Embedded Systems", Atmel Corporation, Tech. Rep., 2015.
- [33] Cheneau, T., Laurent, M., Shen, S., Vanderveen, M., "Ecc public key and signature support in cryptographically generated addresses (cga) and in the secure neighbor discovery (send)", 2009.
- [34] Youssef Ojeil, - <https://e2e.ti.com/support/wireless-connectivity/other-wireless/f/667/p/486317/1778299?pi320995=3>, dostupno na: <https://e2e.ti.com/support/wireless-connectivity/other-wireless/f/667/p/486317/1778299?pi320995=3> Pristupano: 2018-11-14. 2016.
- [35] Davis, M. D., Game theory: A nontechnical introduction. Courier Corporation, 2012.
- [36] Nash, J., "Two-person cooperative games", Econometrica: Journal of the Econometric Society, 1953, str. 128–140.