Android aplikacija za steganografiju

Capan, Ana

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

ELEKTROTEHNIČKI FAKULTET

Diplomski studij

ANDROID APLIKACIJA ZA STEGANOGRAFIJU

Diplomski rad

Ana Capan

Osijek, 2016.

ETFOS
ELEKTROTEHNIČKI FAKULTET OSIJEK
Sveučilište Josipa Jurja Strossmayera u Osijeku

Obrazac D1: Obrazac za imenovanje Povjerenstva za obranu diplomskog rada

Osijek, 11.07.2016.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu diplomskog rada

Ime i prezime studenta:	Ana Capan
Studij, smjer:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D-606R, 23.10.2013.
Mentor:	Doc.dr.sc. Krešimir Nenadić
Sumentor:	
Predsjednik Povjerenstva:	Doc.dr.sc. Alfonzo Baumgartner
Član Povjerenstva:	Hrvoje Leventić
Naslov diplomskog rada:	Android aplikacija za steganografiju
Znanstvena grana rada:	**Obradba informacija (zn. polje računarstvo)**
Zadatak diplomskog rada:	Objasniti pojam steganografije. Izraditi aplikaciju za Android platformu pomoću koje će korisnik moći sakriti zadani tekst u fotografiju metodom najmanje značajnog bita. Aplikacija treba imati mogućnost čitanja poruke koja je skrivena u slici i prikaz poruke korisniku.
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 Postignuti rezultati u odnosu na složenost zadatka: 2 Jasnoća pismenog izražavanja: 3 Razina samostalnosti: 2
Datum prijedloga ocjene mentora:	11.07.2016.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis: Nenadić
	Datum: 10.7.2016.

ETFOS
ELEKTROTEHNIČKI FAKULTET OSIJEK
Sveučilište Josipa Jurja Strossmayera u Osijeku

IZJAVA O ORIGINALNOSTI RADA

Osijek, 14.07.2016.

Ime i prezime studenta:	Ana Capan
Studij:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D-606R, 23.10.2013.
Ephorus podudaranje [%]:	1%

Ovom izjavom izjavljujem da je rad pod nazivom: **Android aplikacija za steganografiju**

izrađen pod vodstvom mentora Doc.dr.sc. Krešimir Nenadić

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Capan

SADRŽAJ

1. UVOD

Razvojem informacijskih i komunikacijskih tehnologija omogućena je obrada i prijenos različitih vrsta digitalnih sadržaja kao što su fotografije, zvukovni i video zapisi. Digitalni oblik zapisa sadržaja omogućuje jednostavniji pristup i neovlašteno kopiranje sadržaja. U svrhu zaštite i kontrole korištenja digitalnog sadržaja, razvijene su naprednije steganografske metode. Steganografija omogućuje prikrivanje informacija unutar prijenosnog medija, te osigurava očuvanje tajnosti privatnih informacija.

Cilj ovog diplomskog rada je prikaz metoda koje se primjenjuju u steganografiji te prikaz primjene izradom aplikacije za Android operativni sustav.

U teorijskom će dijelu rada biti opisan povijesni razvoj i primjena steganografije kroz navedene primjere. Nadalje, u radu će biti opisane vrste steganografskih postupaka te objašnjene metode skrivanja fotografije u druge fotografije (*Downgrading* metoda), skrivanja poruke u fotografiju izmijenom najmanje značajnog bita (LSB metoda) te skrivanje u frekvencijskoj domeni (DCT metoda).

U praktičnom dijelu rada prikazana je izrada aplikacije za Android operativni sustav te objašnjena primjena LSB metode u izradi aplikacije.

U krajnjem dijelu rada prikazani su rezultati dobiveni korištenjem aplikacije za steganografiju te dani prijedlozi mogućih riješenja i poboljšanja za daljni rad.

1.1. Zadatak rada

Objasniti pojam steganografije. Izraditi aplikaciju za Android platformu pomoću koje će korisnik moći skriti zadani tekst u fotografiju metodom najmanje značajnog bita. Aplikacija treba imati mogućnost čitanja poruke koja je skrivena u slici i prikaz poruke korisniku.

2. STEGANOGRAFIJA

Steganografija je znanost i umjetnost skrivanja informacije na način koji prikriva samo postojanje informacije koja se prenosi unutar medija.

2.1. Povijesni razvoj steganografije

Naziv steganografija izveden je od grčkih riječi *steganos* (στεγανός) što znači „pokriveno“ i *graphein* (γράφειν) što znači „pisanje“. [1]
Prvi zapisi korištenja steganografije datiraju iz vremena antičke Grčke, oko 440. godine prije Krista, a navedeni su u djelu „Povijest“ grčkog povjesničara Herodota. U djelu je navedeno da su Grci tetovirali poruke na obrijane glave robova. Poruka je bila uspješno skrivena kada je kosa narasla, a primatelj ju je mogao pročitati ponovnim brijanjem glave. Drugi način skrivanja poruka bio je korištenjem voštanih ploča koje su se koristile kao površine za pisanje. Poruke su pisane izravno na drvenu podlogu koja je zatim prelivena voskom.[2]

Tijekom drugog svjetskog rata pojavili su se brojni primjeri steganografskih postupaka. Jedan od primjera je korištenje nevidljive tinte kojom se poruka skrivala između teksta na papiru. Tinta je sadržavala prirodne supstance poput voćnih sokova ili urina, a njihovim zagrijavanjem je postignuto prikazivanje skrivene poruke. Osim nevidljive tinte korištene su i mikrofotografije. Mikrofotografije (engl. *microdots*) su fotografije veličine i oblika točke odnosno interpunkcijskog znaka „ . “ koje su korištene za skrivanje podataka.[3]
Nulte šifre su još jedan primjer steganografije. Nulta šifra predstavlja poruku skrivenu unutar druge poruke čije je postojanje teško otkriti, a može ju otkriti primatelj koji zna na koji način je poruka skrivena (na primjer čitanjem trećeg slova u svakoj riječi). Ovakav način slanja skrivene poruke se često koristi i danas. [4]
Razvoj digitalne tehnologije omogućio je razvijanje naprednijih metoda steganografije. Primjeri steganografije koji se koriste u današnje vrijeme su posebne tinte za označavanje novčanica te vodeni pečat i digitalni potpis u zaštiti multimedijalnih sadržaja.

2.2. Primjena steganografije

Svrha steganografije je očuvanje tajnosti informacija. Prema tome, najčešća primjena steganografije je u označavanju autorskih prava te njihovoj zaštiti od potencijalnih napada i neovlaštenog pristupa.

Steganografija se može koristiti za skrivanje poruke u svrhu kasnijeg otkrivanja ili za označavanje autorskih prava kada je poruka koja se unosi korištena za označavanje dokumenta. Prema tome, primjena steganografije se može podijeliti na dvije skupine [3]:

- Zaštita protiv otkrivanja (skrivanje poruke)
- Zaštita protiv uklanjanja (označavanje dokumenata):
    - Digitalni vodeni pečat (engl. *Digital watermark*)
    - Digitalni potpis (engl. *Digital signature*)

2.2.1. Digitalni vodeni pečat

Digitalni vodeni pečat je skup informacija ugrađenih u multimedijsku datoteku. Vodeni pečati su najčešće skriveni kako bi se spriječilo njihovo otkrivanje i uklanjanje. Vidljivi vodeni pečati se koriste za ograničavanje uporabe kao što je detektiranje neovlaštenog kopiranja multimedijskog sadržaja.

Prema [4] vodeni pečat se koristi u svrhu:

- zaštite od neovlaštenog kopiranja - sprječavanje kopiranja multimedijskih datoteka
- zaštite autorskih prava - onemogućavanje krađe vlasništva nad multimedijskom datotekom
- provjere vjerodostojsnosti – provjera vjerodostojnosti multimedijske datoteke detekcijom lokacije vodenog pečata
- pohrane dodatnih informacija - dodavanje podataka koji mogu služiti kao bilješke o multimedijskoj datoteci

Vidljivi vodeni pečati se najčešće koriste na fotografijama kao što je prikazano na slici 2.1.

Sl. 2.1. Prikaz vidljivog digitalnog vodenog pečata

Kako bi se spriječilo otkrivanje neovlaštenog kopiranja, pokušava se ukloniti postojanje vodenog pečata obradom multimedijske datoteke. Različitim steganografskim postupcima omogućeno je skrivanje postojanja vodenog pečata unutar datoteke.

Prijenos multimedijskih datoteka najčešće zahtjeva obradu sadržaja kao što su sažimanje ili promjena veličine. Pojedini vodeni pečati nisu otporni na takvu obradu sadržaja stoga je prikladnije koristiti zaštitu digitalnim potpisom.

2.2.2. Digitalni potpis

Elektronički potpis je skup podataka u elektroničkom obliku koji služi za identifikaciju korisnika i provjeru vjerodostojnosti podataka koji se prenose elektroničkim putem. Digitalni potpisi predstavljaju podskupinu elektroničkih potpisa koji koriste kriptografske algoritme s javnim ključem za autentifikaciju.

Stvaranje digitalnog potpisa započinje izračunavanjem sažetka poruke (engl. *message digest*) korištenjem *hash*[1] funkcije. Dobivenom nizu znakova dodaje se informacija o vlasniku digitalnog potpisa te se šifrira koristeći privatni ključ pošiljatelja. Primatelj vrši identifikaciju poruke dešifriranjem koristeći javni ključ .

Steganografija se koristi kod generiranja sažetka poruke korištenjem *hash* funkcije. Dobiveni niz znakova je određene veličine te ga je moguće utvrditi ukoliko dođe do promjene izvornih podataka.

[1] *Hash* funkcije su jednosmjerni kriptografski algoritami koji stvaraju sažetak poruke određene veličine.

Hash funkcije se primjenjuju za digitalni otisak prsta (engl. *fingerprint*) na sadržaj podataka u svrhu dokazivanja da je sadržaj nepromijenjen. Napad na sadržaj datoteke može se otkriti zbog nepodudaranja *hash* vrijednosti tijekom provjere vjerodostojnosti.

Za stvaranje digitalnog potpisa može se koristiti i DSA (engl. *Digital Signature Algorithm*) kojeg je razvila Nacionalna agencija za sigurnost (NSA) [5].

Sadržaj web stranica moguće je zaštititi XML (engl. *eXtensible Markup Language*) digitalnim potpisom. Moguće je na web stranici potpisati bilo koji programski element (kao što su dijelovi HTML i XML programskog koda, polja formulara te njihovi sadržaji) [6].

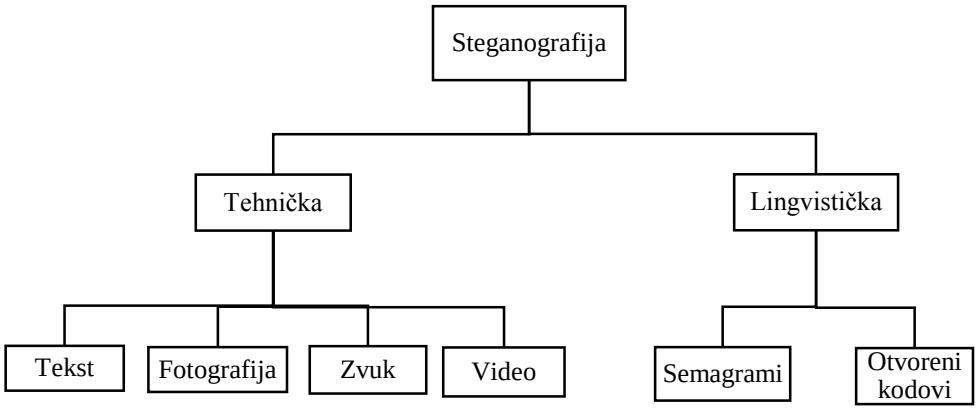
Steganografija ima i negativnu primjenu budući da omogućava skrivanje postojanja informacija. Primjer negativne primjene je korištenje steganografije u komunikaciji između terorističkih organizacija. Najčešće se steganografija koristi u negativne svrhe kao što su: krađa identiteta, neovlašteno kopiranje i distribucija zabranjenog sadržaja, krađa bankovnih računa ili privatnih informacija.

Postoje različiti programski alati za korištenje steganografije kao što su *OpenStego, MP3Stego, Steghide, S-Tools* te mnogi drugi [7].

3. PREGLED VRSTA STEGANOGRAFSKIH POSTUPAKA

Postoje različite tehnike koje omogućuju skrivanje poruka unutar medija. Nositelj poruke (prijenosni medij) treba se sastojati od takvog skupa podataka čija će promjena biti neprimjetna.

Slika 3.1. prikazuje pregled vrsta steganografskih postupaka.



Sl. 3.1. Pregled steganografskih postupaka

Prema [8] podjela steganografije je na lingvističku i tehničku steganografiju.

Lingvistička steganografija obuhvaća postupke skrivanja poruke unutar teksta pri čemu je postojanje poruke prikriveno. Dijeli se na semagrame i otvorene kodove.
Semagrami skrivaju poruku korištenjem simbola i znakova. Mogu biti:

- Vizualni semagrami - koriste svakodnevne fizičke predmete za skrivanje poruke (npr. postavljanje zastave)
- Tekstualni semagrami - skrivaju poruku izmjenom teksta (npr. promjena fonta)

Otvoreni kodovi skrivaju poruku unutar nositelja poruke koji se ponekad naziva i otvorena komunikacija. Dijele se na:

- Žargonski kod - korištenje jezičnog govora koji razumije određena skupina ljudi
- Skrivene šifre – skrivanje poruke koju je moguće otkriti ukoliko je poznata metoda kojom je poruka skrivena (npr. čitanje svakog trećeg znaka u riječi)

Tehnička steganografija obuhvaća znanstvene metode i posebne alate za skrivanje poruka.
Prijenosni medij može biti tekst, grafički, zvukovni ili video digitalni podaci. Najčešće korištene steganografske tehnike koriste fotografije za skrivanje poruka.

3.1. Tehnike prostorne domene

Tehnike prostorne domene su steganografske tehnike koje se temelje na zamjeni dijelova fotografije sa skrivenim podacima u prostornoj domeni (engl. *spatial domain*).
Ove steganografske tehnike poznate su kao tehnike supstitucije, a u nastavku su opisane najpoznatije i najčešće korištene metode [9].

3.1.1. Metoda zamjene bita najmanje važnosti

Metoda zamjene bita najmanje važnosti ili LSB (engl. Least Significat Bit) zamjena je najkorištenija tehnika skrivanja poruke u digitalnoj fotografiji.
U binarnom nizu važan je redoslijed bitova. Bit najmanjeg značaja (LSB) je krajnji desni bit koji ima najmanju aritmetičku vrijednost (2^0 =1) dok je bit najvećeg značaja (MSB) krajnji lijevi bit koji ima najveću aritmetičku vrijednost (2^7=128). Na slici 3.2. prikazan je redoslijed bitova binarnog niza [10].

Sl. 3.2. Redoslijed bitova

LSB metodom se svakom bajtu u skupu podataka najmanje značajan bit zamijeni bitom poruke. Neka se poruka sastoji od jednog slova A. Slovo A je prema ASCII (engl. *American Standard Code for Information Interchange*) standardu prikazan kao binarni zapis 01000001 kao što prikazuje tablica 3.1.

Tab. 3.1. Binarni zapis slova A

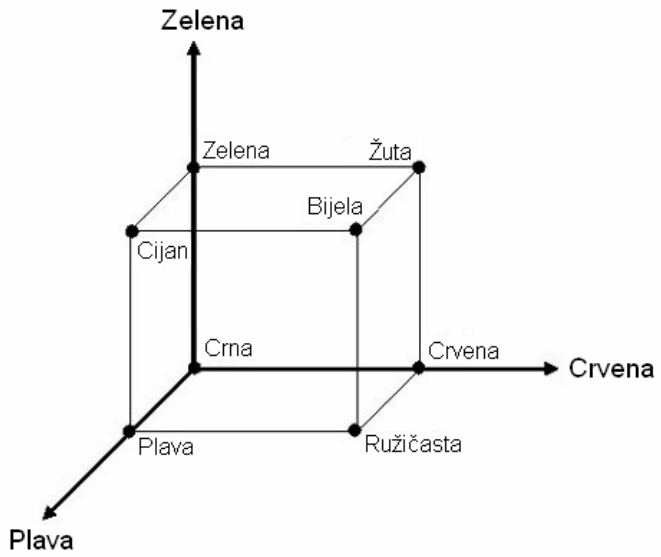
A= 65_{10}							
0	1	0	0	0	0	0	1

Tablica 3.2. prikazuje skrivanje bita poruke u najmanje značajan bit jednog bajta. Postupak se dalje ponavlja za preostale bitove poruke. Ukoliko vrijednost prvog bajta podatka iznosi 149_{10} , zamijena najmanje značajnog bita neće znatno promijeniti vrijednost.

Tab. 3.2. Zamjena najmanje značajnog bita

$A= 65_{10}$							
0	1	0	0	0	0	0	1
149_{10}							
1	0	0	1	0	1	0	**1**
148_{10}							
1	0	0	1	0	1	0	**0**

LSB metoda se primjenjuje pri zamjeni najmanjeg elementa fotografije s bitom poruke.
Svaki element digitalne fotografije sadrži određenu vrijednost boje. Dubina boje se određuje brojem bitova po pojedinoj boji. Sustav RGB (engl. *Red-Green-Blue*) je model boja u kojem svaka od tri komponente (crvena, zelena, plava) ima kanal od 8 bita. Svaki najmanji element unutar fotografije određen je sa 24 bita [9].



Sl. 3.3. RGB sustav [4]

LSB metoda može se koristiti za spremanje bitova poruke u svaki element fotografije odnosno po jedan bit u svaki bajt pojedine RGB komponente. Promjene vrijednosti boje pojedine RGB komponente gotovo je nemoguće uočiti te postojanje poruke ostaje skriveno.

3.1.2. *Downgrading* tehnika

Downgrading je postupak skrivanja fotografije u drugu fotografiju. Ovaj postupak se ostvaruje korištenjem prethodno navedene LSB metode.

Fotografija koja se skriva mora biti jednakih dimenzija kao fotografija koja ju prikriva. Postupak se provodi zamijenom četiri najmanje značajna bita pojedine RGB komponente fotografije sa četiri bita najvećeg značaja pojedine RGB komponente fotografije koja se skriva.

Kako bi se iz fotografije izdvojila skrivena fotografija, potrebno je izdvojiti četiri najmanje značajna bita iz svakog bajta RGB komponente te njima pridodati preostale bitove [9].

Tablica 3.3. prikazuje primjer zamjene bitova na RGB komponentama jednog elementa svake fotografije.

Tab. 3.3. Prikaz zamjene bitova

Element fotografije u koju se skriva fotografija																							
R								G								B							
1	0	0	1	**0**	**1**	**0**	**1**	0	0	1	1	**1**	**0**	**1**	**1**	0	1	0	1	**1**	**0**	**0**	**0**
Element fotografije koja se skriva																							
R								G								B							
1	**1**	**0**	**0**	0	0	1	0	**1**	**0**	**1**	**0**	0	1	1	0	**1**	**1**	**0**	**0**	1	1	0	1
Element stego fotografije																							
R								G								B							
1	0	0	1	**1**	**1**	**0**	**0**	0	0	1	1	**1**	**0**	**1**	**0**	0	1	0	1	**1**	**1**	**0**	**0**
Element fotografije nakon izdvajanja																							
R								G								B							
1	**1**	**0**	**0**	0	0	0	0	**1**	**0**	**1**	**0**	0	0	0	0	**1**	**1**	**0**	**0**	0	0	0	0

Iz navedenog primjera vidljivo je kako se nadopunom bitova prilikom izdvajanja fotografije gubi dio podataka.

Nedostatak korištenja LSB tehnike je gubitak informacije prilikom korištenja JPEG (engl. *Joint Photographic Experts Group*) kompresije ili obradom fotografije. Stoga je najprikladnije koristiti fotografije u GIF (engl. *Graphic Interchange Format*), BMP (engl. *Bitmap file format*) ili PNG (engl. *Portable Network Graphics*) formatu kompresije bez gubitaka.

3.2. Tehnike trasformacije domena

Steganografske tehnike transformacije domene koriste matematičke funkcije za skrivanje podataka unutar fotografije kako bi se spriječio njihov gubitak prilikom kompresije.
DCT (engl. *Discrete Cosine Transformation*) je diskretna kosinusna transformacija koja se koristi kod JPEG format kompresije fotografije. Ovom transformacijom fotografija se dijeli na blokove 8×8 sve dok se ne pronađu blokovi u kojima je promjena vrijednosti RGB komponente elementa niska. Kada se takvi blokovi pronađu, zamijenjuju se diskretnim koeficijentom kosinusne transformacije. Skrivanje poruke postiže se zamjenom bitova poruke s DCT koeficijentima, a skrivena poruka ostaje sačuvana i nakon kompresije [9].

4. REALIZACIJA ANDROID APLIKACIJE

U svrhu implementacije steganografije izrađena je mobina aplikacija za Android operativni sustav. Android je otvoreni operativni sustav za mobilne uređaje zasnovan na jezgri Linux. Za razvoj funkcionalnosti aplikacije korišten je programski jezik Java, a za izradu i prikaz korisničkog sučelja XML(engl. *eXtensible Markup Language*) programski jezik. Pohrana podataka omogućena je korištenjem SQLite sustava za upravljanje relacijskim bazama napisanim u C/C++ programskom jeziku.

4.1. Izrada steganografske fotografije

Android aplikacija za steganografiju pruža korisniku mogućnost skrivanja poruke unutar odabrane fotografije, prikaz rezultata (stego fotografije) te otkrivanje skrivene poruke.
Za izradu steganografske fotografije potrebno je najprije odabrati željenu fotografiju (izradom nove korištenjem kamere uređaja ili odabirom iz galerije uređaja). Zatim je potrebno upisati poruku koje se dalje prikriva unutar fotografije korištenjem LSB metode.

4.1.1. Kamera

Android radno okruženje uključuje podršku za kameru i različite mogućnosti kamere dostupne na mobilnom uređaju. Klasa koja sadrži sve metapodatke vezane uz multimedijske datoteke na uređaju i vanjskoj memoriji naziva se *MediaStore*.
Za izradu fotografije aplikacija mora imati pristup korištenju kamere uređaja. Deklaracije kojima aplikacija zatjeva određena dopuštenja definiraju se u manifest datoteci pod nazivom *AndroidManifest.xml*. Ova datoteka sadrži metapodatke o aplikaciji, definira sve aktivnosti, dopuštenja, ikonu, naziv aplikacije i sl.
Za spremanje izrađenih fotografija, aplikacija mora imati dopuštenje za spremanje na vanjsku memoriju mobilnog uređaja. Na slici 4.1. dan je prikaz zahtjeva za dopuštenje korištenja kamere i spremanja na vanjsku memoriju.

```
<uses-feature android:name="android.hardware.camera2" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

Sl. 4.1. Definiranje zahtjeva za dopuštenje

Kamera se može koristiti izravno ukoliko se žele koristiti dodatne mogućnosti kamere. Drugi način je neizravnim korištenjem pomoću *Intent* klase. Klasa *Intent* predstavlja klasu kojom se provodi neka namjera ili radnja. Najčešća namjera je kretanje između pojedinih aktivnosti aplikacije. Korištenjem *Intent* objekta navedena je radnja kojom se pokreće aplikacija za fotografiranje. Osim za fotografiranje, *Intent* objekt korišten je za pokretanje radnje za prikaz fotografije i radnje za odabir iz galerije slika. Na slici 4.2. prikazano je definiranje Intent objekta za navedene radnje.

```
Intent intent = new Intent(Intent.ACTION_PICK,
                MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);

Intent intent = new Intent(Intent.ACTION_VIEW);
```

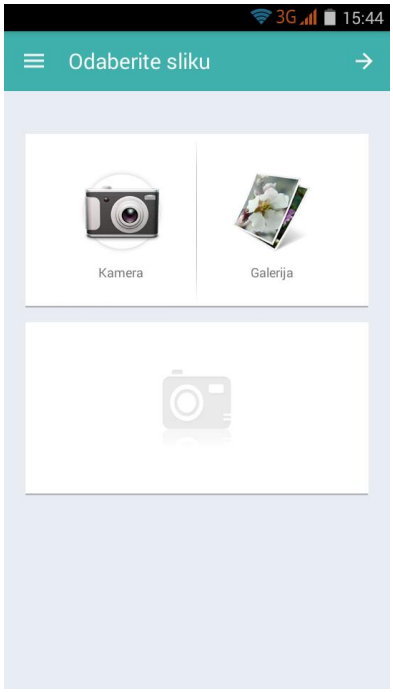
Sl. 4.2. Definiranje *Intent* objekata

Nakon izrade fotografije korištenjem kamere uređaja ili odabirom postojeće iz galerije, korisniku je pružen prikaz odabrane fotografije. Na slikama 4.4.- 4.7. prikazan je izgled aktivnosti za odabir te mogućnosti odabira fotografije. Ukoliko fotografija nije odabrana, korisniku je onemogućen nastavak izrade steganografske fotografije odnosno unos teksta za skrivanje u fotografiju.
Za proslijeđivanje odabrane fotografije na sljedeću aktivnost za unost teksta, korištena je metoda *putExtra()* kojom se u *Intent* objekt dodaje podatak o putanji odabrane fotografije kao što je prikazano na slici 4.3.

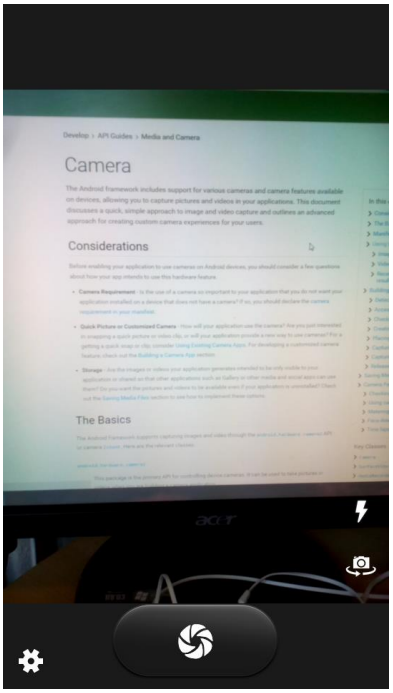
```
Intent intent = new Intent(NewImgActivity.this, HideTextActivity.class);
intent.putExtra("imgpath", path);
startActivity(intent);
```

Sl. 4.3. Proslijeđivanje podataka u drugu aktivnost

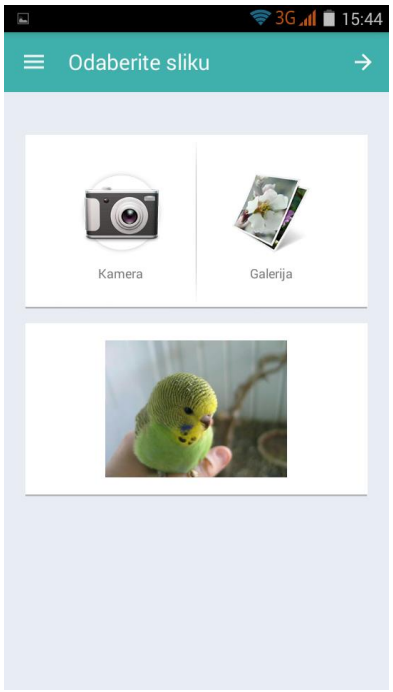
Za dohvaćanje podataka u drugoj aktivnosti, korištena je metoda *getIntent()* koja vraća *Intent* objekt koji je pokrenuo aktivnost, te metoda *getStringExtra()* za dohvaćanje proslijeđenog podatka.



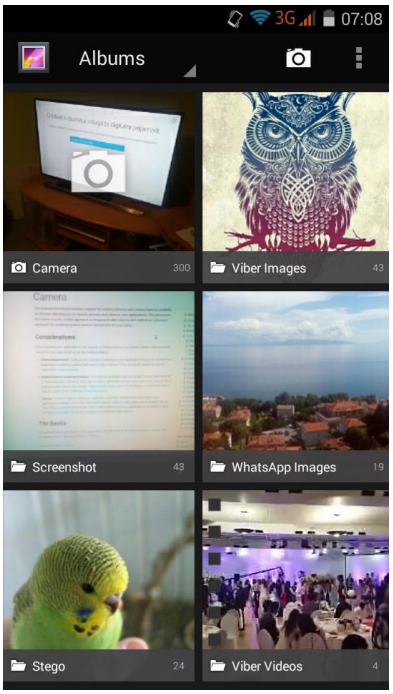
Sl. 4.4. Aktivnost za odabir fotografije



Sl. 4.5. Izrada korištenjem kamere uređaja



Sl. 4.6. Prikaz odabrane fotografije



Sl. 4.7. Odabir iz galerije uređaja

4.1.2. Primjena LSB metode

Bitmap grafika smatra se temeljem svake upotrebe Android grafike. *Bitmap* slikovna datoteka se može predstaviti kao pravokutna mreža sastavljena od najmanjih elemenata fotografije pri čemu svaki element može biti određen sa 8, 16, 24 ili 32 bita. Fotografije u 24-bitnom BMP formatu sadrže 8 bita po pojedinom RGB kanalu svakog elementa fotografije.

Bitmap je objekt korišten za rad sa fotografijama. Klasa *BitmapFactory* omogućuje stvaranje *Bitmap* objekta iz različitih izvora korištenjem metoda *decodeByteArray(), decodeStream(), decodeResource()* i drugih. Za stvaranje *Bitmap* objekta korištena je metoda *decodeFile()* kojoj je kao parametar predana putanja fotografije.
Pomoću metoda *setPixel()* i *getPixel()* moguće je pristupiti svakom elementu fotografije te obaviti željenu zamjenu. Korištenjem *getPixel()* metode u jednodimenzionalno polje je spremljena kopija sadržaja fotografije. Veličina polja jednaka je broju elemenata fotografije odnosno umnošku širine i visine fotografije.

```
Bitmap bitmap = BitmapFactory.decodeFile(path);
int w = bitmap.getWidth();
int h = bitmap.getHeight();
int[] pixels = new int[w * h];
bitmap.getPixels(pixels, 0, w, 0, 0, w, h);
```

Sl. 4.8. Kopiranje elemenata fotografije

U polju su spremljene vrijednosti svakog elementa fotografije čime je omogućena zamjena najmanje značajnih bitova s bitovima poruke.
Kako bi se moglo pristupiti pojedinom bitu, poruka koju korisnik želi skriti pretvorena je u polje bajtova metodom *getBytes()*. Sadržaju poruke dodana je duljina poruke kako bi se čitanjem poruke iz fotografije znalo kada je pročitana cijela skrivena poruka. Duljina poruke spremljena je u polje veličine 4 bajta. Cjelokupni sadržaj poruke predstavlja novo polje koje sadrži duljinu poruke i poruku koja se skriva. Na slici 4.9. prikazan je navedeni postupak pretvaranja.

```
byte[] poruka = tekst.getBytes();
ByteBuffer buffer = ByteBuffer.allocate(4);
buffer.putInt(poruka.length);
byte[] duljina_poruke=buffer.array();

int uk_duljina = duljina_poruke.length + poruka.length;
byte[] uk_poruka = new byte[uk_duljina];
System.arraycopy(duljina_poruke, 0, uk_poruka, 0, duljina_poruke.length);
System.arraycopy(poruka,0, uk_poruka,duljina_poruke.length,poruka.length);
```

Sl. 4.9. Pretvaranje poruke u polje bajtova

Korištenjem LSB metode, svaki najmanje značajan bit pojedinog elementa fotografije zamjenjuje se bitom poruke. Kako bi se pristupilo pojedinom bitu i izvršila zamjena, korišteni su operatori nad bitovima.

Operatori pomaka (engl. *shift*) pomiču binarni zapis broja u lijevu ili u desnu stranu. Operatori pomaka koriste dva operanda pri čemu je prvi operand cjelobrojnog tipa nad kojim se obavlja operacija, a drugi operand broj bitova za koji se treba izvršiti pomak. Operator „<<“ pomiče bitove u lijevu stranu, dok operator „>>“ pomiče bitove u desnu stranu. Ukoliko bajt poruke sadrži bitove 01010111, pomicanjem bitova za 5 mjesta u desnu stranu, binarni niz ima zapis 00000010.

Logički operatori nad bitovima(engl. *bitwise*) obavljaju operacije na bitovima koji se nalaze na odgovarajućim mjestima. Logički „I“ (&) služi za postavljanje određenih bitova na 0, dok logički „ILI“ (|) služi za postavljanje određenih bitova na 1.

Na slici 4.10. prikazano je pomicanje bitova vrijednosti elementa fotografije u svrhu određivanja bajta pojedine RGB komponente.

```
int R = ((pixels[i] >> 16) & 255);
int G = ((pixels[i] >> 8) & 255);
int B = (pixels[i] & 255);
```

Sl. 4.10. Određivanje RGB komponente

Pomicanjem bitova te korištenjem logičkog operatora „I“ određeno je 8 bitova pojedine RGB komponente elementa fotografije. Prvi bit poruke određen je pomicanjem bitova za 7 mjesta u desnu stranu te korištenjem logičkog operatora kako bi najmanje značajan bit ostao nepromijenjen, a ostali bitovi postavili na 0. Najmanje značajan bit R komponente postavljen je na 0 te se korištenjem logičkog operatora „ILI“ zamjenjuje bitom poruke.
Na slici 4.11. prikazan je primjer zamjene najmanje značajnog bita za R komponentu.

```
int bit = (uk_poruka[0] >> 7) & 1;
int R = ((((pixels[i] >> 16) & 255) & 254) | bit);
```

Sl. 4.11. Zamjena najmanje značajnog bita

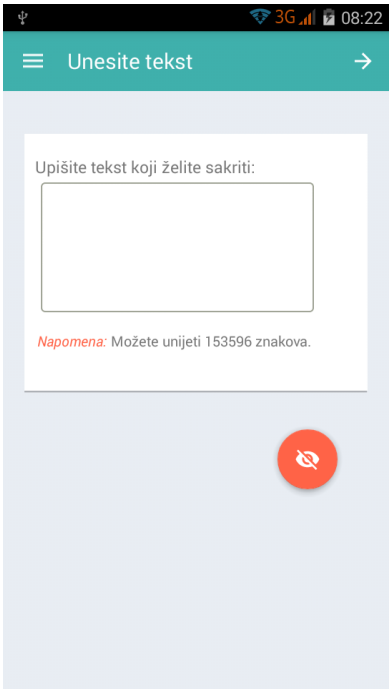
Nakon skrivanja svih bitova poruke, stvoren je *Bitmap* objekt kojem se *setPixel()* metodom vrijednosti boja elemenata zamjenjuju vrijednostima boja iz predanog joj polja. Dobivena fotografija se sprema u vanjsku memoriju uređaja korištenjem PNG kompresije bez gubitaka.

Prilikom otkrivanja poruke iz fotografije, najprije se izdvajaju prva četiri skrivena bajta poruke te se spremaju u polje koje određuje duljinu skrivene poruke. Najmanje značajan bit RGB komponente elementa dodan je na kraj polja i pomaknut za jedno mjesto u lijevu stranu korištenjem operatora „<<“. Postupak se ponavlja dok se ne izdvoje sva 32 bita koja određuju duljinu poruke.
Na slici 4.12. prikazan je primjer spremanja najmanje značajnog bita R komponente u polje.

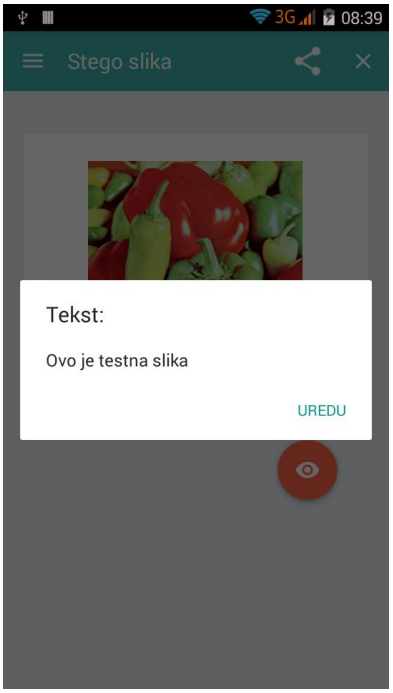
```
duljina[j] = (byte) ((duljina[j] << 1) | (((pixels[i] >> 16) & 255) & 1));
```

Sl. 4.12. Određivanje duljine poruke

Nakon što je poznata duljina poruke, stvoreno je polje bajtova za spremanje bitova poruke. Nakon što je polje popunjeno, pretvara se u znakovni niz (*string*) te se korisniku prikazuje skrivena poruka. Na slici 4.13. prikazana je aktivnost za unost teksta, a na slici 4.14. prikazano je otkrivanje teksta iz fotografije.



Sl. 4.13. Prikaz odabrane fotografije



Sl. 4.14. Odabir iz galerije uređaja

4.2. Očuvanje korisničkih podataka

Spremanje korisničkih i aplikacijskih podataka je nužno ukoliko se korisniku želi omogućiti ponovno korištenje podataka. Za očuvanje podataka, Android pruža tri opća načina:

- Korištenje standardnog datotečnog sustava
- Korištenje SQLite sustava za upravljanje bazama podataka
- Korištenje dijeljenih postavki (engl. *Shared prefrences*)

4.2.1. Datotečni sustav

Klasa *File* predstavlja apstraktnu reprezentaciju imena putanji datoteka i direktorija. Objekt *File* klase pogodan je za čitanje i zapisivanje velike količine podataka.

Aplikacija može spremati podatke u vanjsku i unutarnju memoriju uređaja. Za spremanje u unutranju memoriju uređaja nisu potrebna dopuštenja iz razloga što aplikacija uvijek može čitati iz unutarnje memorije te zapisivati u istu. Za spremanje podataka u vanjsku memoriju potrebno je zatražiti dopuštenja korištenjem deklaracija u manifest datoteci kao što je prikazano na slici 4.15.

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
```

Sl. 4.15. Definiranje zahtjeva za dopuštenje

Ukoliko aplikacija zahtjeva čitanje iz vanjske memorije, tada je potrebno navesti deklaraciju, inače navođenjem zahtjeva za pisanje u vanjsku memoriju, nije potrebno navoditi i zahtjev za čitanje.
Steganografske fotografije izrađene korištenjem aplikacije, spremljene su u zasebni direktorij pod nazivom „*Stego*“ u vanjskoj memoriji uređaja. Ukoliko navedeni direktorij ne postoji, izrađen je korištenjem metode *mkdirs()*. Korištena je klasa *FileOutputStream* kojom se izlazni tok podataka sprema u datoteku čiji je naziv predan kao parametar.
Aplikacija pruža mogućnost dijeljenja fotografija sa drugim aplikacijama, no prijenosom se gube informacije skrivene unutar fotografije. Prijenos fotografija elektroničkom poštom ne mijenja njihov sadržaj te se iz primljene fotografije mogu ponovno pročitati skriveni podaci.

4.2.2. Android *SQLite* baza podataka

SQLite baza jedna je od najčešće korištenih baza podataka na mobilnim uređajima. Podržava značajke standardnih relacijskih baza, a zahtjeva malo memorije prilikom izvođenja. Kada je programski stvorena, baza se sprema kao jedna datoteka i dostupna je samo aplikaciji koja ju koristi.
U ovom radu registracija i prijava korisnika te spremanje podataka realizirani su korištenjem baze podataka. Stvorena je klasa *LoginDBHelper* koja nasljeđuje klasu *SQLiteOpenHelper*. U ovoj su klasi definirani podaci (naziv baze, naziv tablice kao i nazivi stupaca) , SQL naredbe za stvaranje baze te unos i čitanje podataka iz baze podataka. Za rezultat upita na bazu koristi se klasa *Cursor*. Objekt klase *Cursor* se može promatrati kao pokazivač retka u tablici. Ovaj objekt pokazuje na redak iznad prvog unosa u tablici te ga je potrebno pomaknuti korištenjem metode *moveToFirst()*. Na slici 4.16. prikazan je izgled aktivnosti za prijavu korisnika.

Sl. 4.16. Prijava korisnika

Sl. 4.17. Registracija korisnika

Nakon pokretanja aktivnosti za prijavu, korisnik mora unijeti podatke. Ukoliko je polje prazno, pritiskom na gumb za prijavu pojavit će se obavijest da je potrebno popuniti sva polja. Za stvaranje obavijesti koristi se klasa *Toast*. Ukoliko su polja popunjena, pritiskom na gumb poziva se metoda za dohvaćanje korisnika prema nazivu elektroničke pošte. Ako se podaci ne nalaze u bazi podataka ili baza još nije stvorena, potrebna je registracija korisnika kao što je prikazano na slici 4.17. Pritiskom gumba za registraciju, poziva se metoda kojom je omogućeno spajanje na bazu i unos korisnika. Na slici 4.18. prikazano je spajanje na bazu te unos korisnika u bazu podataka.

```
SQLiteDatabase db = this.getWritableDatabase();
ContentValues values = new ContentValues();
values.put(KEY_NAME, name);
values.put(KEY_EMAIL, email);
values.put(KEY_PASS, password);
db.insert(TABLE_USER, null, values);
db.close();
```

Sl. 4.18. Unos korisnika u bazu podataka

Kako se prilikom svakog pokretanja aplikacije ne bi ponovno pokrenula aktivnost prijave korisnika, potrebno je spremiti informaciju o prijavi korisnika. U tu svrhu korištene su dijeljene postavke.

4.2.3. Dijeljene postavke

Korisničke postavke omogućuju spremanje manje količine podataka u obliku para ključ-vrijednost. Te su vrijednosti jednostavni tipovi podataka kao *integer*, *boolean* ili *string*. Postavke mogu biti namijenjene samo jednoj aktivnosti ili dijeljene među aktivnostima.
Za pristup postavkama koriste se metode:

- *getPreferences ()* – za pojedinu aktivnost
- *getSharedPreferences ()* – za dijeljenje među aktivnostima
- *getDefaultSharedPreferences()* – za rad s gotovim okruženjem za izradu postavki

Navedene metode vraćaju *SharedPreferences* objekt koji se koristi za postavljanje i dohvaćanje postavki. Za izmjenu ili spremanje novih vrijednosti postavki koristi se klasa *Editor* čiji se objekt stvara pozivom metode *edit()*. Stvorene postavke se spremaju u XML datoteku, a svaku postavku je moguće dohvatiti preko odgovarajućeg ključa.
Na slici 4.19. prikazan je primjer spremanja i dohvaćanja dijeljenih postavki.

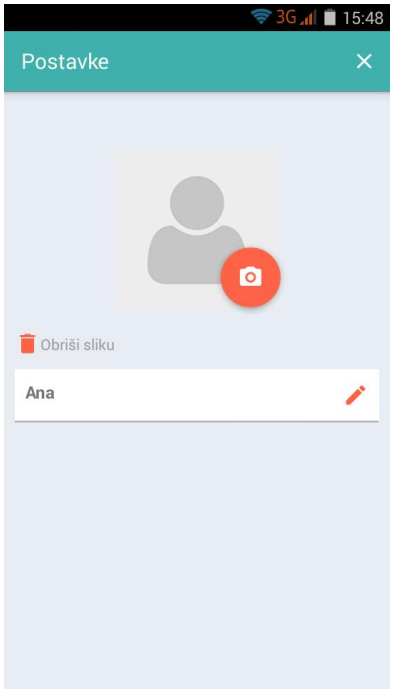
```
SharedPreferences prefs;
prefs =getApplicationContext().getSharedPreferences("prefs",MODE_PRIVATE);
prefs.getString("Ime", null);

SharedPreferences.Editor editor = prefs.edit();
editor.putString("Ime", "Pero");
editor.apply();
editor.clear();
```

Sl. 4.19. Dijeljene postavke

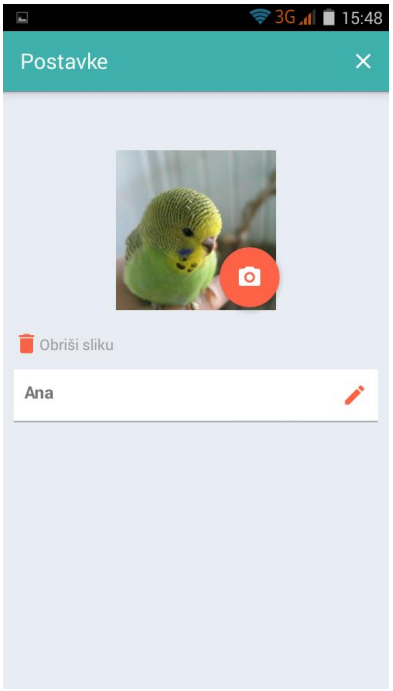
Metoda *getSharedPreferences()* kao parametar prima naziv datoteke koja se stvara i način izrade (MODE_PRIVATE je pretpostavljeni način koji označava da izrađenoj datoteci može pristupiti pozivajuća aplikacija).

Svaka postavka se dohvaća korištenjem oznake ključa, a ukoliko vrijednost nije prisutna kao rezultat je vraćena pretpostavljena (*null*) vrijednost. Metodom *commit()* spremaju se izvršene promjene postavki, a *apply()* se razlikuje po tome što spremanje izvodi u pozadini. Metodom *clear()* brišu se svi podaci iz dijeljenih postavki, dok se metodom *remove()* briše pojedina vrijednost čiji je ključ predan kao parametar.

U dijeljene postavke spremljena je vrijednost ukoliko je korisnik već prijavljen te se provjerom prilikom pokretanja aplikacije preskače pokretanje aktivnosti prijave korisnika. Osim toga, prilikom prijave korisnika u dijeljene postavke sprema se naziv korisnika i elektroničke pošte. Naziv korisnika se može promijeniti pokretanjem aktivnosti za postavke koja je prikazana na slici 4.20. Unutar te aktivnosti korisnik može postaviti ili obrisati fotografiju profila.

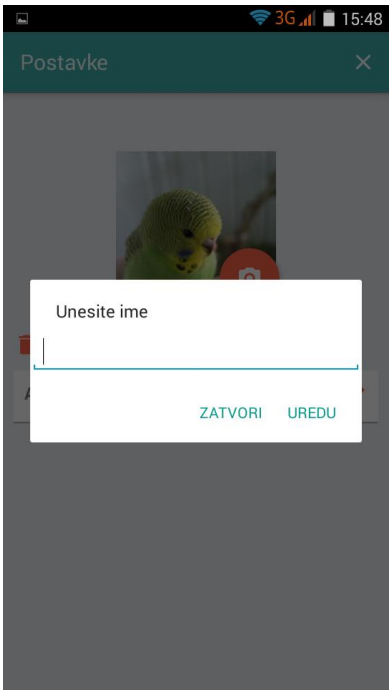


Sl. 4.20. Prikaz aktivnosti za postavke



Sl. 4.21. Odabir profilne fotografije

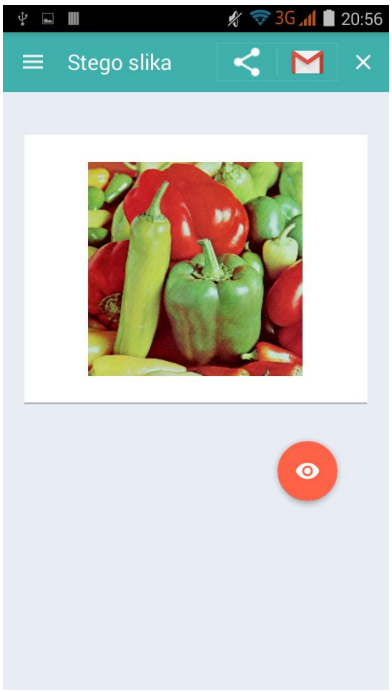
Na slici 4.22. prikazan je primjer izmjene korisničkog imena. Svaka izmjena korisničkog imena ili fotografije spremljena je korištenjem dijeljenih postavki te je prikazana u daljnjem korištenju aplikacije.



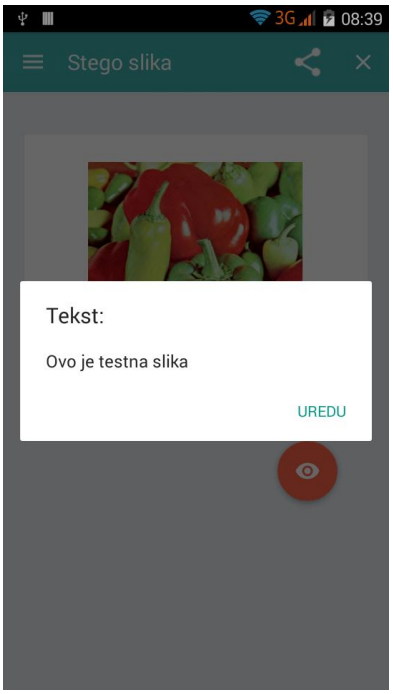
Sl. 4.22. Izmjena naziva korisnika

5. OPIS REZULTATA

Pokretanje aktivnosti u svrhu prikaza steganografske fotografije prikazuje da je neprimjetna razlika između izvorne fotografije i fotografije u kojoj je poruka skrivena. Otkrivanjem poruke je vidljivo da se informacija koju poruka sadrži nije izgubila prilikom skrivanja.
Na slikama 5.1. i 5.2. prikazan je rezultat korištenja steganografske metode u aplikaciji za Android.



Sl. 5.1. Aktivnost za prikaz fotografije



Sl. 5.2. Prikaz skrivene poruke

Testiranjem rada aplikacije utvrđena je funkcionalnost aplikacije. Rezultati pokazuju da se unutar fotografije može prikriti velika količina podataka. Unosom veće količine podataka od ograničene, aplikacija obavještava korisnika o pogrešci.

Sljedeće slike prikazuju rezultate korištenjem fotografija različitih formata i različitih dimenzija za skrivanje poruka. Fotografije većih dimenzija mogu sadržavati veću količinu skrivenog sadržaja.

Sl. 5.3. Izvorna fotografija

Sl. 5.4. Steganografska fotografija

Na slici 5.3. prikazana je korištena fotografija za skrivanje poruke. Fotografija je BMP formata dimenzija 512×512. Unutar fotografije je skriveno 23 bajta podatka odnosno 184 bita poruke. Na slici 5.4. prikazan je rezultat skrivanja poruke. Promjenu na fotografiji je nemoguće uočiti.

Sl. 5.5. Izvorna fotografije

Sl. 5.6. Steganografska fotografija

Na slici 5.5. prikazana je korištena fotografija za skrivanje poruke. Fotografija je PNG formata dimenzija 640×640. Unutar fotografije je skriveno 36 bajta podatka odnosno 288 bita poruke. Na slici 5.6. prikazan je rezultat skrivanja poruke. Promjenu na fotografiji je nemoguće uočiti.

Sl. 5.7. Izvorna fotografija

Sl. 5.8. Steganografska fotografija

Na slici 5.7. prikazana je korištena fotografija za skrivanje poruke. Fotografija je JPEG formata dimenzija 640×480. Unutar fotografije je skriveno 38 bajta podatka odnosno 304 bita poruke. Na slici 5.6. prikazan je rezultat skrivanja poruke. Promjenu na fotografiji je nemoguće uočiti.

6. ZAKLJUČAK

Steganografija pruža mogućnost skrivanja informacije na način da se prikriva postojanje informacije. Za skrivanje informacije potrebno je korištenje medija unutar kojeg se informacija prenosi. U ovom radu prikazan je primjer skrivanja poruke unutar fotografije. Pomoću steganografije moguće je informacije skriti unutar fotografija različitih formata (BMP, JPEG, PNG te ostalih). Korištenjem metode zamjene najmanje značajnog bita (LSB) omogućeno je skrivanje velike količine podataka unutar fotografija. Rezultati pokazuju da se povećala veličina fotografija unutar kojih je skrivena poruka u odnosu na izvorne fotografije. Zamjena najmanje značajnog bita svakog elementa fotografije pokazala se pogodnim rješenjem za skrivanje podataka iz razloga što je promjenu na fotografiji nemoguće uočiti. Međutim, zamjenom većeg broja najmanje značajnih bitova sa bitovima poruke, promjena je vidljiva te je tajnost poruke narušena.

Ograničenje LSB metode je da se prilikom sažimanja ili kompresije fotografije gube skriveni podaci. Kao rješenja ovog problema postoje steganografske tehnike transformacije domena pomoću kojih se ne gube podaci sadržani u fotografiji prilikom njene obrade ili kompresije.

Steganografija ima široku primjenu, a najčešće je korištena u zaštiti autorskih prava korištenjem digitalnog vodenog pečata ili zaštiti privatnih informacija. Budući da omogućuje skrivanje informacija, steganografija je korištena i za razmjenu nedopuštenih informacija putem različitih medija. Postoje različiti programski alati za korištenje steganografije kao što su *OpenStego*, *MP3Stego*, *Steghide* te mnogi drugi. Zbog razvijenih metoda za otkrivanje i uklanjanje skrivenih informacije, nastavlja se daljnji razvoj novih steganografskih metoda.

LITERATURA

[1] Wikipedia, s Interneta, https://bs.wikipedia.org/wiki/Steganografija , 6. ožujka 2016.

[2] B. Dunbar, „Steganographic Techniques and their use in an Open-Systems Environment“, SANS Institute InfoSec Reading Room, str. 3, 18. siječnja 2002.

[3] A. Kumar, K. Pooja, „Steganography-A Data Hiding Technique“, International Journal of Computer Applications (0975 – 8887) , br.7, Vol. 9, str. 20, studeni 2010.

[4] CARNet (CARNet CERT i LS&S), „Steganografija“, CCERT-PUBDOC-2006-04-154

[5] Mixideja, s Interneta, http://www.mixideja.hr/blog.html , 27. svibnja 2016.

[6] CARNet (CARNet CERT i LS&S), „Digitalni potpis“, CCERT-PUBDOC-2007-02-182

[7] Wikipedia, s Interneta, https://en.wikipedia.org/wiki/Steganography_tools , 27. svibnja 2016.

[8] K. Navneet, B. Sunny, „A Survey on various types of Steganography and Analysis of Hiding Techniques“, Internationa Journal of Engineering Trends and Technology, br. 8, Vol. 11, str. 389, svibanj 2014.

[9] A. Purohit, P.S. V.S. Sridhar, „Image Steganography: A Review“, International Journal of Computer Science and Information Technologies, br.4, Vol. 5, str. 4891-4893, 2014.

[10] Wikipedia, s Interneta, http://www.ips.org.uk/faq/index.php?title=File:MSB-LSB.png , 4. lipnja 2016.

SAŽETAK

U radu je opisan povijesni razvoj i primjena steganografije. Nadalje, opisane su različite metode koje se koriste u steganografiji te pružaju mogućnost zaštite informacija unutar različitih medija. Obrađena je najčešće korištena metoda LSB te njena primjena u skrivanju informacija unutar fotografija. Realizacija metode je prikazana na aplikaciji za Android operativni sustav. Sadržaj aplikacije obuhvaća odabir fotografije, skrivanje poruke te prikaz skrivenog sadržaja.

KLJUČNE RIJEČI: steganografija, LSB, android, informacija,poruka, skrivanje, fotografija, elektronička pošta, bit

ABSTRACT

Android application for steganography

This paper describes the history and use of steganography. Furthermore, different methods used in steganography are described, which provide the ability to protect information within different media. The most commonly used method LSB and it's application in hiding information within digital image is described in this paper. The implementation of the method is presented in an Android application. The content of the application includes image selection, hiding the message and displaying the hidden content.

KEY WORDS: steganography, LSB, android, information, message, hiding, image, email, bit

ŽIVOTOPIS

Ana Capan rođena je 29. lipnja 1991. godine u Osijeku. Nakon završetka osnovne škole u Tenji pohađala je II. gimnaziju u Osijeku. Završetkom srednjoškolskog obrazovanja, godine 2010. upisala je Preddiplomski studij računarstva na Elektrotehničkom fakultetu u Osijeku. Preddiplomski studij računarstva završila je 2013. godine te iste godine upisala Diplomski studij procesnog računarstva.

Ana Capan, univ.bacc.ing.comp.