

Secure Data Transfer Over Internet Using Image Steganography

Вовед

Стеганографија со слики се однесува на процесот на сокривање на податоци во слика. Сликата избрана за оваа цел се нарекува *cover image* (покривна слика), а сликата добиена по стеганографијата се нарекува *стего слика*.

Сликата е претставена како матрица $N \times M$ (во случај на црно-бели слики) или $N \times M \times 3$ (во случај на слики во боја) во меморијата, при што секоја влезна точка претставува вредност на интензитет на пикселот. Во стеганографија со слики, пораката се вметнува во слика со изменување на вредностите на некои пиксели, кои се избираат со помош на алгоритам за шифрирање. Примачот на сликата мора да биде запознаен со истиот алгоритам за да знае кои пиксели да ги избере за да ја извлече пораката.

Како работи стеганографијата

Чекор 1: Првиот чекор во стеганографијата е изборот на *cover medium*, што е фајлот или пораката која ќе го носи скриениот податок. Најчесто користени медиуми се:

- Слики (JPEG, PNG, BMP итн.)
- Аудио фајлови (MP3, WAV итн.)
- Виеа (MP4, AVI итн.)
- Текстуални фајлови или документи

Чекор 2: Понекогаш, пред да се вметне, скриената порака се криптира за да се додаде дополнителен слој на безбедност. Ова гарантира дека дури и ако некој ја детектира скриената порака, не може да ја прочита без клучот за дешифрирање.

Чекор 3: Потоа скриената порака се вметнува користејќи една од неколкуте техники:

- **Најмалку значаен бит (LSB):** Најмалку значајниот бит на еден бајт се менува за да се скрие порака. Оваа метода најчесто се користи во слики и аудио фајлови.

- **Домен на фреквенции:** Наместо да се менуваат суровите податоци (како пиксели или аудио примероци), скриената порака може да се вметне во фреквентните компоненти на слика или аудио.
- **Битни рамки (Bit Planes):** Во оваа метода, податоците се скриваат во повисоките битни рамки на сликата. Ова може да биде попрισταпно бидејќи се користат битови кои потешко се забележуваат.

Најчесто се користи **LSB енкодирање**.

Чекор 4: Модифицираните податоци потоа се вметнуваат во покривниот медиум. Резултатот е фајл кој сега ги содржи и оригиналната содржина и скриената порака, наречен **stego-објект**, кој може да се пренесува или чува без да предизвика сомнеж.

Чекор 5: Примачот на stego-објектот треба да знае која метода е користена за вметнување на скриената порака. Понекогаш е потребен и таен клуч за да се извлечат податоците, ако стеганографијата е комбинирана со криптирање.

Предности и недостатоци

Предности	Недостатоци
Безбедност	Откривање
Капацитет	Комплексност
Робустност	Долг процес на пренос
Тајна комуникација	Можност за губење на податоци
Отпорност на криптоанализа	Злоупотреба

Референци

<https://www.baeldung.com/java-aes-encryption-decryption>

<https://www.geeksforgeeks.org/computer-networks/image-steganography-in-cryptography/>

<https://www.geeksforgeeks.org/computer-networks/what-is-steganography/>

За апликацијата

Класа: Pixel

Оваа класа претставува еден пиксел од слика.

Полиња:

- **int x** → координата по X
- **int y** → координата по Y
- **Color color** → боја на пикселот
- **Излез:** Pixel објект.

Класа: LSBEncoder

Оваа класа е задолжена за вметнување порака во слика користејќи LSB (Least Significant Bit) техника.

public static BufferedImage embedToImage(File imageFile, String message)

- **Влез:**
 - **imageFile** → оригинална слика
 - **message** → порака за сокривање
- **Опис:** Ја чита сликата, прави копија, ја претвора пораката во бинарна форма и ја енкодира во пикселите преку LSB.
- **Излез:** Нова BufferedImage со сокриена порака.

private static BufferedImage deepCopy(BufferedImage image)

- **Влез: BufferedImage** → оригинална слика
- **Опис:** Прави независна копија од сликата за да не се менува оригиналната.
- **Излез:** нова BufferedImage копија.

private static Pixel[] toPixelArray(BufferedImage image)

- **Влез: BufferedImage** → слика
- **Опис:** Ја претвора сликата во низа од Pixel објекти со координати и боја.
- **Излез:** низа од Pixel објекти.

private static void encode(String[] binaryMessage, Pixel[] pixels)

- **Влез:**
 - **binaryMessage** → пораката во бинарна форма
 - **pixels** → пикселите од сликата
- **Опис:** Ја енкодира бинарната порака во пикселите групирани по 3.
- **Излез:** нема (се менуваат пикселите).

private static void writeBits(String bits, Pixel[] group, boolean lastChar)

- **Влез:**
 - **bits** → 8-битна репрезентација на еден карактер
 - **group** → група од 3 пиксели
 - **lastChar** → дали е последниот карактер
- **Опис:** Ги заменува LSB вредностите од пикселите со битови од пораката. Последниот пиксел содржи ознака за крај на порака.
- **Излез:** нема (менува пиксели).

private static int setLSB(int value, char bit)

- **Влез:**
 - **value** → вредност од RGB канал
 - **bit** → бит од порака
- **Опис:** Го заменува последниот бит од бинарната репрезентација со нов бит.
- **Излез:** нова целобројна вредност.

private static void applyPixels(Pixel[] pixels, BufferedImage image)

- **Влез:**
 - **pixels** → пиксели со нови бои
 - **image** → сликата за промена
- **Опис:** Ја ажурира сликата со новите пиксели.
- **Излез:** нема.

Класа: LSBDecoder

Оваа класа е задолжена за извлекување порака скриена со LSB техника.

public static void extract(String imagePath, SecretKey secretKey, Cipher cipher)

- **Влез:**
 - **imagePath** → патека до сликата
 - **secretKey** → таен клуч
 - **cipher** → AES Cipher
- **Опис:** Ја чита сликата, ја извлекува скриената порака и ја дешифрира со AES.
- **Излез:** печати дешифрирана порака.

private static Pixel[] toPixelArray(BufferedImage image)

- **Влез: BufferedImage** → слика
- **Опис:** Ја претвора сликата во низа од Pixel објекти со координати и боја.
- **Излез:** низа од Pixel објекти.

private static String extractMessage(Pixel[] pixels)

- **Влез: pixels** → пиксели од сликата
- **Опис:** Ја реконструира бинарната порака од LSB вредностите.
- **Излез:** скриена порака (String).

private static char toChar(Pixel[] group)

- **Влез: group** → група од 3 пиксели
- **Опис:** Ги чита LSB битовите и ги претвора во ASCII карактер.
- **Излез:** карактер.

private static char lastBit(int value)

- **Влез: value** → RGB вредност
- **Опис:** Го враќа последниот бит од бројот.
- **Излез:** '0' или '1'.

private static boolean isEnd(Pixel pixel)

- **Влез:** `pixel` → последен пиксел во групата
- **Опис:** Проверува дали сината компонента има бит за крај на порака.
- **Излез:** `true/false`.

Класа: **AESCryptoUtil**

Оваа класа нуди алатки за AES енкрипција/декрипција.

public static SecretKey generateKey(int keySize)

- **Влез:**
 - **keySize** → големина на клуч (128, 192 или 256 бита)
- **Опис:** Генерира нов AES таен клуч со дадената големина.
- **Излез:** `SecretKey` објект (AES клуч).

public static SecretKey fromBytes(byte[] keyBytes, String algorithm)

- **Влез:**
 - **keyBytes** → бајти кои ја претставуваат вредноста на клучот
 - **algorithm** → алгоритам (на пр. "AES")
- **Опис:** Го реконструира AES клучот од дадените бајти.
- **Излез:** `SecretKey` објект.

public static Cipher getCipher(String algorithm)

- **Влез:**
 - **algorithm** → име на алгоритам (на пр. "AES")
- **Опис:** Враќа `Cipher` објект иницијализиран за конкретниот алгоритам.
- **Излез:** `Cipher` објект.

public static String encrypt(String plainText, SecretKey secretKey, Cipher cipher)

- **Влез:**
 - **plainText** → оригиналната порака која треба да се енкриптира
 - **secretKey** → таен AES клуч

- **cipher** → AES Cipher објект
- **Опис:** Ја енкриптира пораката користејќи AES и го претвора резултатот во Base64 формат.
- **Излез:** String (енкриптиран Base64 текст).

public static String decrypt(String base64CipherText, SecretKey secretKey, Cipher cipher)

- **Влез:**
 - **base64CipherText** → порака енкриптирана во Base64 формат
 - **secretKey** → таен AES клуч
 - **cipher** → AES Cipher објект
- **Опис:** Ја декриптира пораката од Base64 назад во оригинален текст.
- **Излез:** String (оригиналниот декриптиран текст).

Класа: EmailService

Оваа класа е задолжена за испраќање email пораки со прикачена слика.

public void sendImage(String to, String subject, String text, BufferedImage image, String filename)

- **Влез:**
 - **to** → примач
 - **subject** → наслов на email
 - **text** → текст во email
 - **image** → слика што ќе се прикачи
 - **filename** → име на прикачената датотека
- **Опис:** Ја конвертира сликата во PNG бајти и ја праќа како email attachment.
- **Излез:** нема (праќа email).

Класа: ProektIbApplication

Главна класа со Spring Boot main метод, управува со интеракцијата на корисникот.

public static void main(String[] args)

- **Влез:**
 - **args** → аргументи од командна линија.
- **Опис:**
 - Ја стартува Spring Boot апликацијата.
 - Прикажува мени за избор:
 - **Опција А:** енкриптира порака и ја крие во слика.
 - **Опција В:** извлекува скриена порака од слика и ја декриптира.
- **Излез:**
 - Нема директен повратен резултат. (Прикажува пораки на конзола, извршува методи generateImage и extractText според изборот).

private static void generateImage(String message, String inputPath)

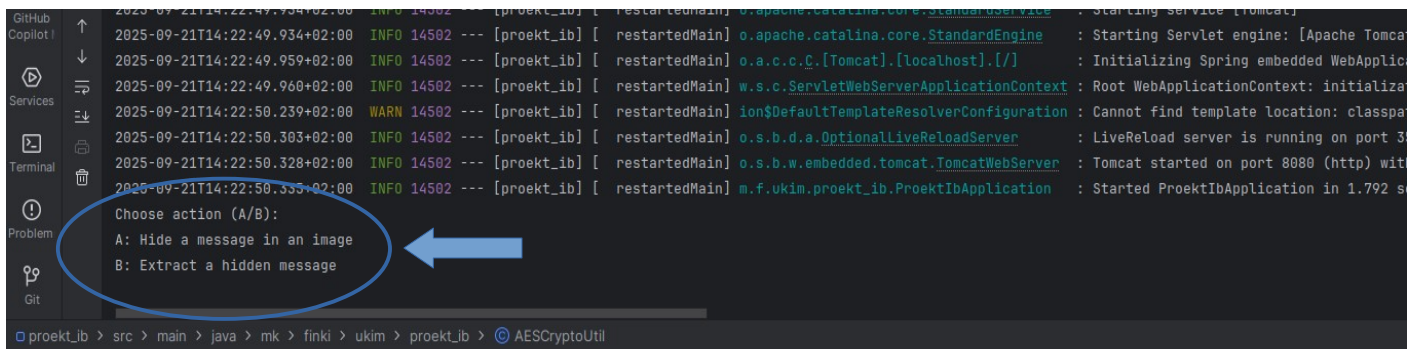
- **Влез:**
 - **message** → порака која треба да се скрие
 - **inputPath** → патека до оригиналната слика
- **Опис:**
 - Генерира AES таен клуч (256-битен).
 - Ја енкриптира пораката со AES.
 - Ја вметнува енкриптираната порака во сликата со помош на LSBEncoder.
 - Дава можност на корисникот да испрати email со:
 - модифицираната слика (со скриената порака),
 - AES клучот во форма на бајти.
- **Излез:**
 - Нема директен резултат. (Прикажува пораки на конзола, креира нова слика и може да испрати email).

private static void extractText(byte[] keyBytes, String path, String algo)

- **Влез:**
 - **keyBytes** → бајти на AES тајниот клуч (користени за енкрипција)
 - **path** → патека до сликата со скриена порака
 - **algo** → алгоритам кој се користи (пр. "AES")
- **Опис:**
 - Ја реконструира AES тајната вредност од бајтите (SecretKeySpec).
 - Иницијализира Cipher објект со AES.
 - Ја повикува LSBDecoder.extract за извлекување и декрипција на скриената порака.
- **Излез:**
 - Нема директен резултат. (Прикажува декриптирана порака или известување ако не може да се пронајде/декриптира).

Начин на работа(demo) на апликацијата

1. Се стартува апликацијата, каде во конзолата се прашува која акција сака да ја изврши
 - А – да вметне порака во слика
 - В – да извлече порака од слика

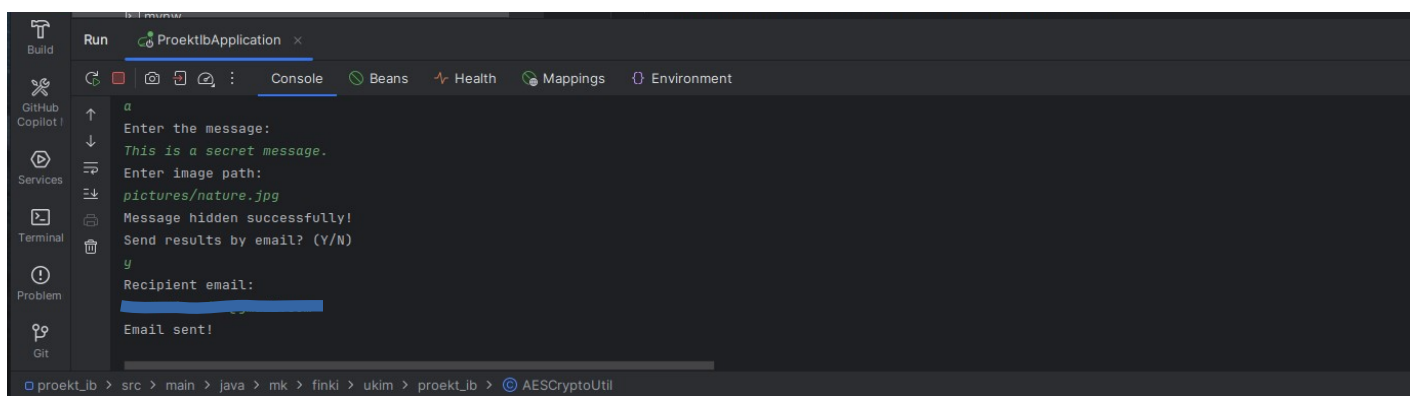


```
2025-09-21T14:22:49.794+02:00 INFO 14502 --- [proekt_ib] [ restartedMain] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2025-09-21T14:22:49.934+02:00 INFO 14502 --- [proekt_ib] [ restartedMain] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat]
2025-09-21T14:22:49.959+02:00 INFO 14502 --- [proekt_ib] [ restartedMain] o.a.c.c.c.f.[Tomcat].[/] : Initializing Spring embedded WebApplication
2025-09-21T14:22:49.960+02:00 INFO 14502 --- [proekt_ib] [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization started
2025-09-21T14:22:50.239+02:00 WARN 14502 --- [proekt_ib] [ restartedMain] ion$DefaultTemplateResolverConfiguration : Cannot find template location: classpath:/templates/
2025-09-21T14:22:50.303+02:00 INFO 14502 --- [proekt_ib] [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 3456
2025-09-21T14:22:50.328+02:00 INFO 14502 --- [proekt_ib] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8080 (http) with context path /
2025-09-21T14:22:50.335+02:00 INFO 14502 --- [proekt_ib] [ restartedMain] m.f.ukim.proekt_ib.ProektIbApplication : Started ProektIbApplication in 1.792 s

Choose action (A/B):
A: Hide a message in an image
B: Extract a hidden message
```

2. Две можни сценарија

2.1. **Корисникот избрал А** – најпрво ја пишува пораката која сака да ја скрие, потоа ќе му биде побарано да внесе локација на сликата во формат *folder/filename.extension*. Откако ќе ја внесе локацијата на сликата, ќе му се прикаже порака дека пораката е успешно скриена доколку патеката е валидна, како и порака дали сака да ја испрати сликата и информациите за клучот и алгоритмот преку мејл. Доколку одбере да(у), ќе треба да ја напише е-поштата на корисникот на кој сака да ги испрати информациите. Доколку мејлот е валиден и успешно е испратена сликата ќе му се прикаже соодветна порака дека мејлот е успешно испратен и дека е потребно да ја симне и зачува сликата доколку сака да ја извлече пораката од неа.



```
Run ProektIbApplication
Enter the message:
This is a secret message.
Enter image path:
pictures/nature.jpg
Message hidden successfully!
Send results by email? (Y/N)
y
Recipient email:
Email sent!
```

Добиениот мејл изгледа вака.



Потоа сликата се симнува и се зачувува во фолдерот во кој што се наоѓа оригиналната слика.



Оригинал



Слика со скриена порака

2.2. **Корисникот избрал В** – Се бара од корисникот да го внесе тајниот клуч кој му бил испратен на неговата е-пошта заедно со сликата. Откако ќе го внесе клучот, треба да ја внесе локацијата на симнатата слика со скриената порака во истиот формат како што се бара доколку избере опција А. Доколку локацијата на сликата е валидна, на конзолата се испишува пораката која била скриена во сликата претходно.

```
2025-09-21T14:50:50.981+02:00 INFO 15459 --- [proekt_ib] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : tomcat started on port 8080 (http) with context path '/'
2025-09-21T14:50:50.990+02:00 INFO 15459 --- [proekt_ib] [ restartedMain] m.f.ukim.proekt_ib.ProektIbApplication : Started ProektIbApplication in 1.138 seconds (process running for 1.712)
Choose action (A/B):
A: Hide a message in an image
B: Extract a hidden message
b
Enter secret key bytes (format: [1,2,3,...]):
[115, 96, -87, -67, 95, -34, -115, 34, 47, 16, 82, -28, -41, -119, -98, 97, 76, 57, 62, -78, -67, -11, -50, 105, 8, 52, -115, 42, 13, -4, 5, -8]
Enter image path:
pictures/hidden.png
Message: This is a secret message.
```