



UNIVERZITET U BEOGRADU
ELEKTROTEHNIČKI FAKULTET

Zaštita podataka – 2020/2021

Bulevar kralja Aleksandara 73, PF 35-54, 11120 Beograd, Srbija

OPENPGP CRYPTO GUI

Autori:

Vasilije Stambolić 0061/2018

Nemanja Maksimović 0355/2016

Sadržaj

1. Uvod	3
2. UI.....	4
2.1. Glavni meni.....	4
2.2. Enkripcija/potpisivanje	4
2.3. Manipulacija ključevima.....	5
2.4. Dekripcija.....	6
3. Opis korišćenih algoritma	7
3.1. Asimetrični algoritmi	7
3.2. Simetrični algoritmi	7
4. Opis klasa	8
4.1. Key managment	8
4.2. Transfer	10
4.3. Exceptions.....	11

1. Uvod

PGP (*Pretty Good Privacy*) je kriptografski protokol koji pruža privatnost (*enkripciju*) i autentikaciju (*potpisivanje*) u komunikaciji. PGP se koristi za potpisivanje, enkripciju i dekripciju mejlova, fajlova, tekstova da bi se povećala sigurnost u e-mail komunikaciji.

Cilj projekta je prikazati jednu implementaciju PGP protokola (OpenPGP) koristeći algoritme koji će biti opisani u nastavku dokumenta.

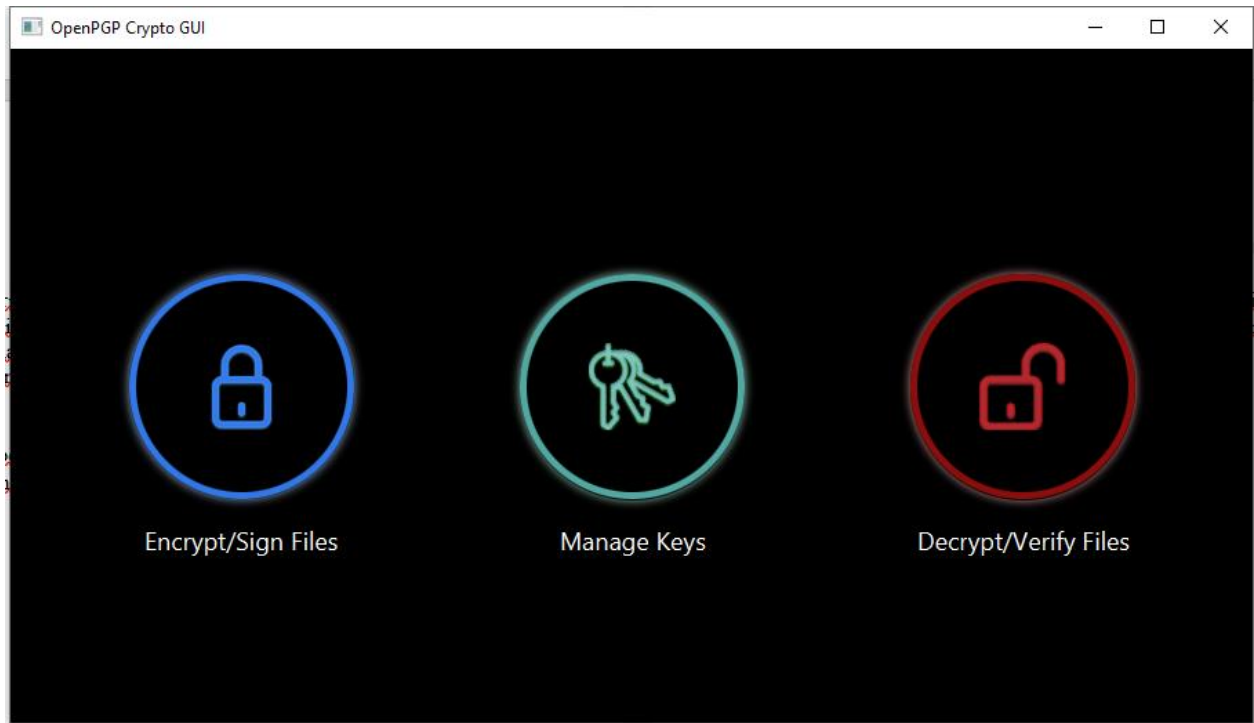
Glavne funkcionalnosti sistema su:

- Generisanje novog i brisanje postojećeg para ključeva
- Uvoz i izvoz javnih i privatnih ključeva
- Slanje poruke
- Primanje poruke

2. UI

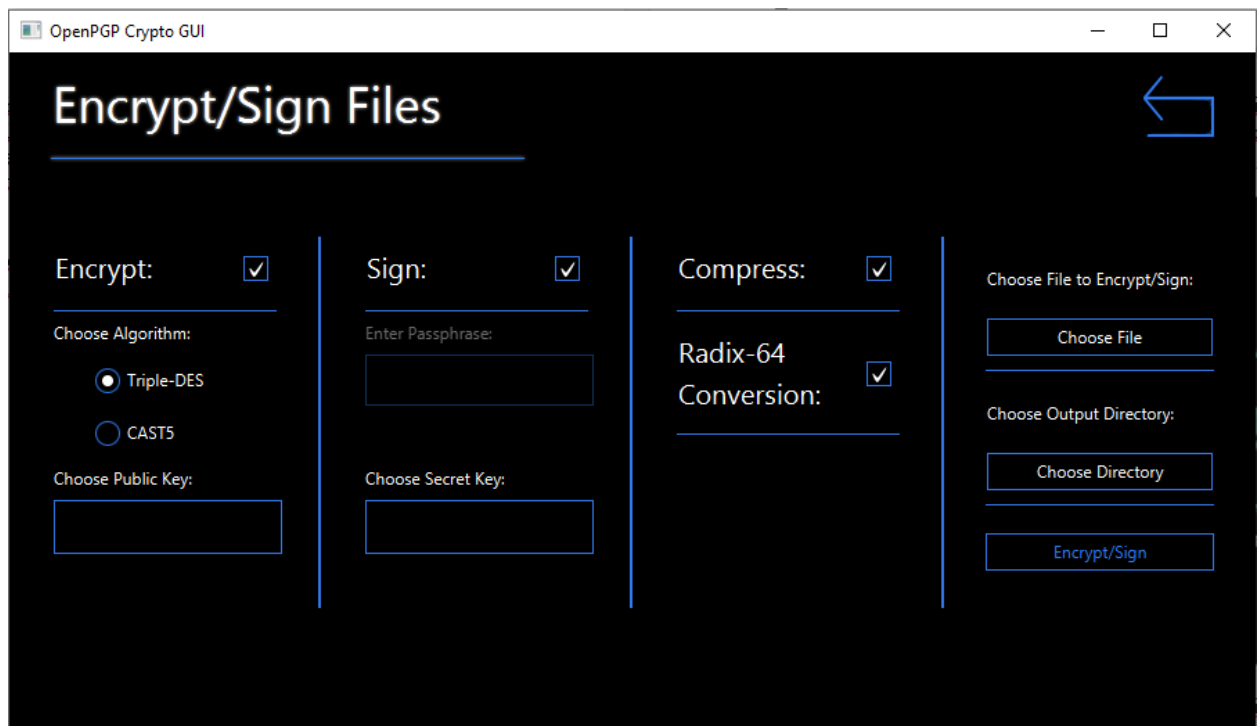
2.1. Glavni meni

Korisnički interfejs se sastoji iz glavnog menija koji dalje vodi u tri posebne stranice: enkripcija, kreiranje i manipulisanje ključevima i dekripcija.



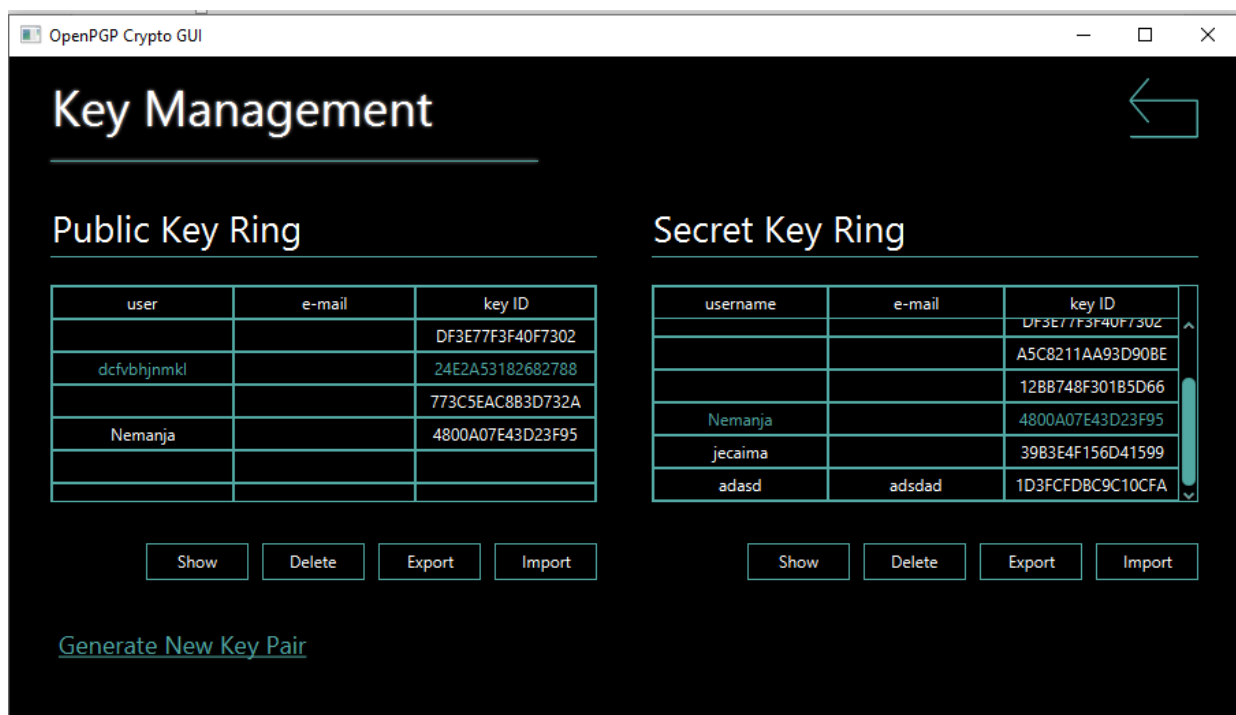
2.2. Enkripcija/potpisivanje

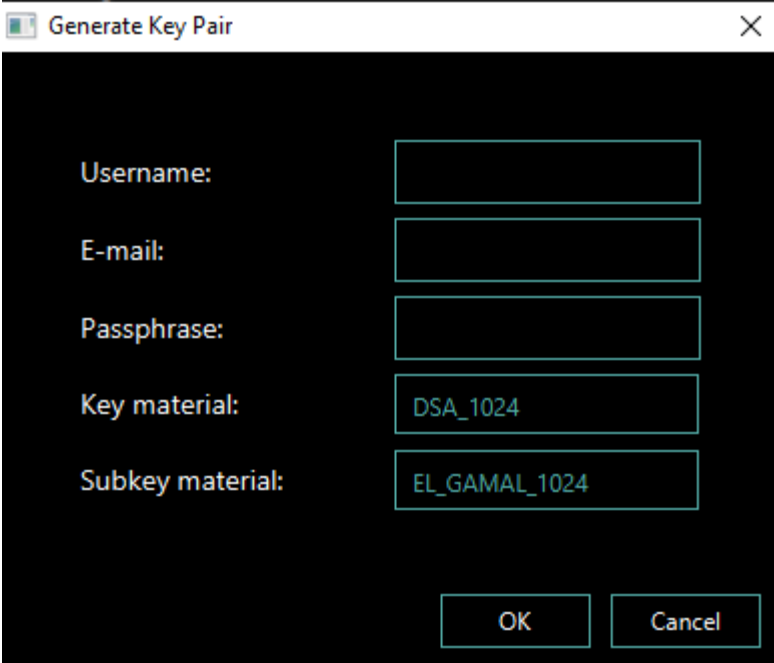
Stranica za enkripciju služi da se izabare fajl koji će da se enkriptuje i lokacija gde će on biti snimljen. Takođe na toj stranici se biraju svi potrebni parametric koji su potrebni za enkripciu/potpisivanje kao što su: Algoritam koji se koristi za enkripciju, javni i tajni ključ kao i opcije da li korisnik želi da odradi enkripciju, potpisivanje, kompresiju i konvertovanje u Radix64 format.



2.3. Manipulacija ključevima

Na stranici za manipulaciju ključeva (Key Management) može da se doda novi par ključeva kao i da se obriše već postojeći. Pored ovih funkcionalnosti ključevi mogu da se eksportuju i importuju.





Generate Key Pair

Username:

E-mail:

Passphrase:

Key material: DSA_1024

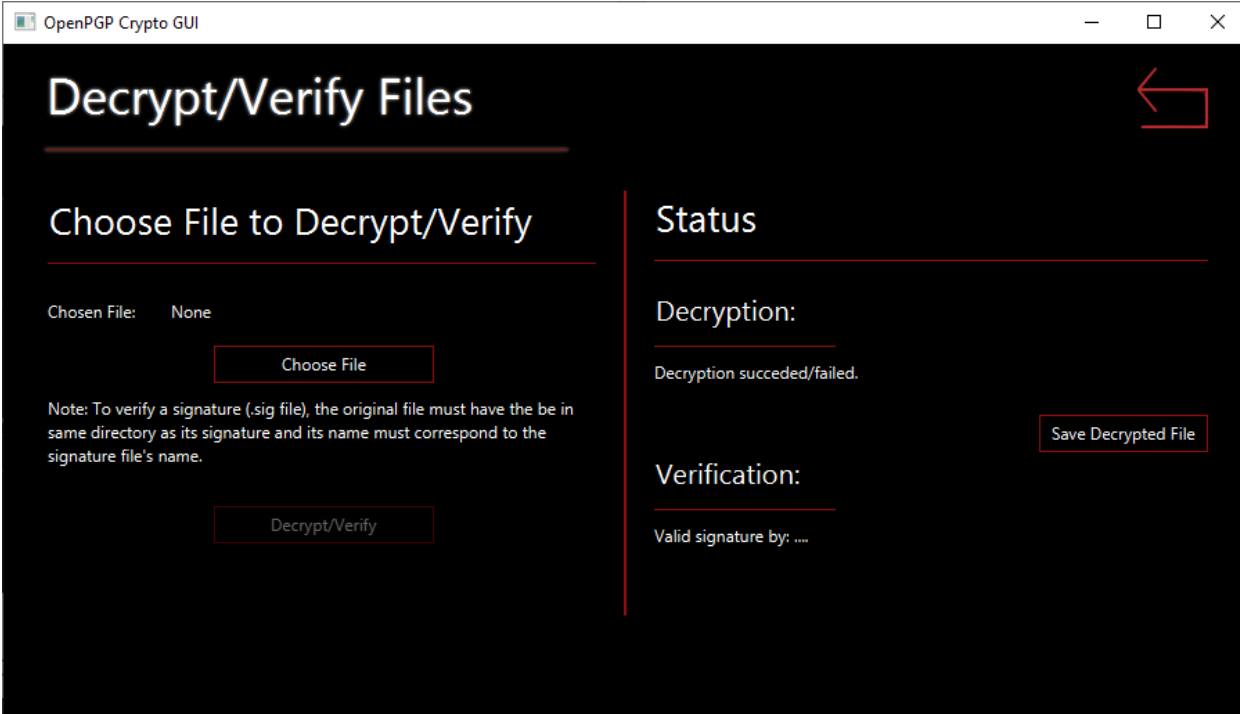
Subkey material: EL_GAMAL_1024

OK Cancel

This is a dialog box titled "Generate Key Pair". It contains five input fields: "Username:", "E-mail:", "Passphrase:", "Key material:", and "Subkey material:". The "Key material" field is pre-filled with "DSA_1024" and the "Subkey material" field is pre-filled with "EL_GAMAL_1024". At the bottom right, there are two buttons: "OK" and "Cancel".

2.4. Dekripcija

Na stranici za dekripciju i verifikaciju bira se fajl koji želimo da dekriptujemo ili verifikujemo kao i lokaciju gde fajl treba da se snimi.



OpenPGP Crypto GUI

Decrypt/Verify Files

Choose File to Decrypt/Verify

Chosen File: None

Choose File

Note: To verify a signature (.sig file), the original file must have the be in same directory as its signature and its name must correspond to the signature file's name.

Decrypt/Verify

Status

Decryption:

Decryption succeded/failed.

Save Decrypted File

Verification:

Valid signature by:

This is a screenshot of the "OpenPGP Crypto GUI" window. The window has a title bar with the text "OpenPGP Crypto GUI" and standard window controls. The main content area is dark-themed. At the top, there's a section titled "Decrypt/Verify Files". Below this, there's a sub-section "Choose File to Decrypt/Verify". It shows "Chosen File: None" and a "Choose File" button. A note below states: "Note: To verify a signature (.sig file), the original file must have the be in same directory as its signature and its name must correspond to the signature file's name." Below the note is a "Decrypt/Verify" button. To the right of the "Choose File" section is a "Status" section. It has a "Decryption:" sub-section with the text "Decryption succeded/failed." and a "Save Decrypted File" button. Below that is a "Verification:" sub-section with the text "Valid signature by:".

3. Opis korišćenih algoritma

Od algoritma za asimetrične ključeve aplikacija koristi DSA za potpisivanje i ElGamal za enkripciju.

Korišćeni algoritmi za simetrične ključeve su 3DES sa EDE konfiguracijom i CAST5.

3.1. Asimetrični algoritmi

DSA (*Digital Signature Algorithm*) je public-key enkripcioni algoritam koji se koristi za generisanje elektronskih potpisa. Kod DSA kreira se public-private par ključeva koji se koriste za potpisivanje i autentikaciju. Pošiljaoc potpisuje podatke koje šalje svojim privatnim ključem što obezbeđuje jedinstveni potpis za korisnika dok primaoci javnim ključem pošiljaoca verifikuju od koga je poruka došla.

ElGamal algoritam je algoritam za enkripciju sa asimetričnim ključevima. Poruka se šifrjuje javnim ključem primaoca dok primaoc dekriptuje poruku svojim privatnim ključem. Time se obezbeđuje da samo primalac može da dekriptuje poruku. U PGP algoritmu ElGamal se koristi za enkripciju simetričnog ključa koji se koristi za enkripciju same poruke.

3.2. Simetrični algoritmi

3DES algoritam je algoritam sa simetričnim ključem koji se koristi za enkripciju blokovskih podataka. 3DES je varijacija DES algoritma gde se sam algoritam ponovi tri puta. U ovoj konkretnoj implementaciji koristi se EDE konfiguracija (Encrypt-Decrypt-Encrypt) sa tri ključa.

CAST5 algoritam je algoritam sa simetričnim ključem koji se koristi kao podrazumevani algoritam u nekim verzijama PGPa. CAST5 koristi Feistel strukturu sa 12-16 rundi i ključevima od 40-128 bita (u konkretnoj implementaciji korišćeni su ključevi od 128 bita).

4. Opis klasa

4.1. Key management

Klasa **KeyInfo**: abstraktna klasa koja čuva podatke potrebne za generisanje ključeva kao što su keyID username i email korisnika.

Potpis metode	Povratna vrednost	Opis
KeyInfo(long keyId, String userId)		Konstruktor koji na osnovu keyId i userId kreira nov objekat
getUsername()	String	Geter za atribut username
getEmail()	String	Geter za atribut email
getKeyIdLong()	long	Geter za atribut keyId
setKeyId(long keyId)	void	Setter za za atribut keyId
setUsername(String username)	void	Setter za atribut username
setEmail(String email)	void	Setter za atribut email
setUserInfo(String userId)	void	Na osnovu id korisnika postavlja podatke o korisniku (username, email)
formatKeyId(long keyId)	String	Konvertuje keyId u string

Klasa **PublicKeyInfo**: nasleđuje **KeyInfo**. Koristi se za čuvanje podataka o javnom ključu

Potpis metode	Povratna vrednost	Opis
PublicKeyInfo(PGPPublicKey pgpPublicKey)		Konstruktor koji na osnovu PGPPublicKey objekta kreira novi objekat PublicKeyInfo

Klasa **SecretKeyInfo**: nasleđuje **KeyInfo**. Koristi se za čuvanje podataka o tajnom ključu

Potpis metode	Povratna vrednost	Opis
SecretKeyInfo(PGPPrivateKey pgpPrivateKey)		Konstruktor koji na osnovu PGPPrivateKey objekta kreira novi objekat SecretKeyInfo

Klasa **User**: čuva podatke o korisniku

Potpis metode	Povratna vrednost	Opis
User(String username, String email, String passphrase)		Konstruktor koji inicijalizuje attribute novog objekta
getUsername()	String	Geter za atribut username
getEmail()	String	Geter za atribut email
getPassphrase()	String	Geter za atribut passphrase
getId()	String	Kreira id korisnika na osnovu usernam-a i emaila

Klasa KeyManager: sadrži logiku za kreiranje, menjanje, brisanje, importovanje, eksportovanje ključeva unutar prstenova ključeva.

Potpis metode	Povratna vrednost	Opis
loadKeyRings()	void	Loads key rings from file
generateKeySubpacketVector()	PGPSignatureSubpacketVector	Generiše PGPSignatureSubpacketVector koji sadrži podatke o sertifikatu koji se koristi za potpisivanje
generateSubkeySubpacketVector()	PGPSignatureSubpacketVector	Generiše PGPSignatureSubpacketVector koji sadrži podatke o enkripciji asimetričnim ključem
generateKeys(User user, KeyMaterial keyMaterial, SubkeyMaterial subkeyMaterial)	Pair<PublicKeyInfo, SecretKeyInfo>	Generiše par ključeva koristeći prosledjene podatke
generateSubkeyPair(SubkeyMaterial subkeyMaterial)	KeyPair	Generiše par podključeva koristeći prosledjene parametre (za ElGamal)
generateKeyPair(KeyMaterial keyMaterial)	KeyPair	Generiše par ključeva koristeći prosledjene parametre (za DSA)
createKeyRingGenerator(User user, KeyPair keyPair, KeyPair subkeyPair)	PGPKeyRingGenerator	Kreira PGPKeyRingGenerator Za korisnika koristeći glavni i sporedni par ključeva
getPublicKeyInfoCollection()	Collection<PublicKeyInfo>	Vraća kolekciju javnih ključeva
getSecretKeyInfoCollection()	Collection<SecretKeyInfo>	Vraća kolekciju tajnih ključeva
exportSecretKeyRings()	void	Eksportuje kolekciju tajnih ključeva u fajl
exportPublicKeyRings()	void	Eksportuje kolekciju javnih ključeva u fajl
exportKey(KeyInfo keyInfo, File file)	void	Eksportuje konkretan ključ u fajl
importKeyRings(File file)	List<KeyInfo>	Importuje prsten ključeva iz prosleđenog fajla
getPublicKeyRing(KeyInfo keyInfo)	PGPPublicKeyRing	Dohvata prsten javnih ključeva kome pripada prosleđeni ključ

getSecretKeyRing(KeyInfo keyInfo)	PGPSecretKeyRing	Dohvata prsten tajnih ključeva kome pripada prosleđeni ključ
deletePublicKey(KeyInfo keyInfo)	void	Uklanja javni ključ
deleteSecretKey(KeyInfo keyInfo)	void	Uklanja tajni ključ
isEncrypted(KeyInfo keyInfo)	boolean	Proverava da li je prosleđeni tajni ključ zaštićen šifrom (passphrase)

4.2. Transfer

Klasa Sender: sadrži logiku za enkripciju i potpisivanje fajlova

Potpis metode	Povratna vrednost	Opis
Sender(File file, File outputDirectory, boolean compressionEnabled, boolean radix64Enabled, boolean encryptEnabled, int symmetricAlgorithmId, PublicKeyInfo publicKeyInfo, boolean signEnabled, String passphrase, SecretKeyInfo secretKeyInfo)		Konstruktor
send()	void	Sadrži svu potrebnu logiku za enkripciju, potpisivanje, kompresiju u zavisnosti od atributa klase

Klasa Reciever: Sadrži logiku za dekriptovanje i verifikaciju fajlova

Potpis metode	Povratna vrednost	Opis
Receiver(File file)		Konstruktor
receive()	void	Sadrži logiku za dekripciju i verifikaciju u odnosu na sadržaj fajla (da li je fajl enkriptovan i/ili potpisan)
decrypt()	void	Dekriptuje fajl koristeći odgovarajući algoritam
decompress()	void	Vrši dekompresiju fajla ukoliko je bio kompresovan
read()	void	Čita sadržaj fajla
verify()	void	Verifikuje potpis fajla

Klasa RecieverStatus: Klasa u koju se upisuje rezultat dekripcije/verifikacije. Sadrži dekriptovanu poruku kao i podatke o potpisivaču. Klasa ne sadrži metode sa posebnom logikom već samo getere i setere za attribute koji čuvaju potrebne podatke.

4.3. Exceptions

Postoje nekoliko klasa koje nasleđuju Exception i služe za hendlovanje posebnih grešaka.

- InvalidFormatException : greška koja se prijavljuje ukoliko fajl koji je prosleđen na dekripciju nije u odgovarajućem formatu
- RecieverException : tip greške koji se javlja tokom dekripcije/verifikacije.

Klase koje nasleđuju RecieverException:

- InvalidPassphraseException : javlja se ukoliko passphrase koji je unet ne odgovara odabranom ključu
- KeyNotFoundException : greška kada ključ korišćen za enkripciju ne postoji u odgovarajućem prstenu
- PassphraseRequiredException : greška kada nije unet passphrase sa određenim ključem
- SignerKeyNotFoundException : greška kada ključ korišćen za potpisivanje ne postoji u prstenu ključeva