Elektrotehnički fakultet

Univerzitet u Beogradu

Projekat iz predmeta Zaštita podataka

Implementacija PGP protokola

Grupa 2.

Algoritami za asimetrične ključeve: DSA za potpisivanje sa ključevima veličine 1024 i 2048 i ElGamal za enkripciju sa kljucevima veličine 1024, 2048 i 4096 bita

Algoritmi za simetrične ključeve: 3DES sa EDE konfiguracijom i tri ključa i IDEA

Cilj projektnog zadatka je bolje razumevanje PGP protokola, mogućnosti koje on pruža i načina čnjegovog korišćenja. U tu svrhu projektovana je i implementirana aplikacija sa grafičkim korisničkim interfejsom u programskom jeziku Java.

Aplikacija omogućava:

- Generisanje i brisanje postojećeg para ključeva
- Uvoz i izvoz javnog ili privatnog ključa u .asc format
- Prikaz prstena javnih i privatnih ključeva sa svim potrebnim informacijama
- Slanje poruke (uz obezbeđivanje enkripcije i potpisivanja)
- Primanje poruke (uz obezbeđivanje dekripcije i verifikacije)

Rešenje je implementirano po standardu RFC4480 koji opisuje OpenPGP protokol. Testirano je tako da bude kompatibilno sa aplikacijom Kleopatra. U implementaciji su iskorišćene strukture i algoritmi iz org.bouncycastle.openpgp biblioteke.

Pregled glavnih klasa, funkcija i implementiranih algoritama

Klasa Gui

```
/**
    * Metoda koja kreira generator za kljuceve
* @param KeyPair
* @param dsaKeyPair
* @param KeyPair
* @param elGamalKeyPair
* @param identity
* @param passphrase
* @return
* @throws Exception
    */

public PGPKeyRingGenerator createPGPKeyRingGenerator(KeyPair dsaKeyPair, KeyPair
elGamalKeyPair, String identity, char[] passphrase)
```

```
/**
    * Metoda koja generise elgamal par kljuca
* @param keySize
    */
public static final KeyPair generateElGamalKeyPair(int keySize)

/**
    * Metoda koja generise dsa par kljuca
* @param keySize
    */
public static final KeyPair generateDsaKeyPair(int keySize)

/** Metoda koja napesta panel za generisanje kljuceva
    *
* @param 0
    */
private void setUpPanelForGeneratingKeys()

/**
    * Metoda koja vraca koja je velicina elgamala oznacena
* @param el1024
* @param el2048
* @param el4096
* @return int
*
    */
public int getElGamalSize(JRadioButton el1024, JRadioButton el2048, JRadioButton el4096)


    * Metoda koja vraca koja je velicina dsa oznacena
* @param dsa1024
* @param dsa2048
*  @return int
*
    */
public int getDSASize(JRadioButton dsa1024, JRadioButton dsa2048)

/**
    * Metoda koja namesta dugmice za generisanje
* @return 0
    */
public void setUpGenericKeyAction()

/**
```

```
 * Metoda za namestanje dugmica
* @return 0
    */
private void setUpActionButtons()

/**
    * Metoda koja namesta da se vide kljucevi
* @return 0
    */

public void setUpActionsForKeyViewing()

/**
    * Metoda koja napesta panel za dekricpiju
* @return 0
    */

public void setUpPanelForDekripcija()

/**
    * Metoda koja namesta panel za prikaz kljuceva
* @return 0
    */

public void setUpPanelForShowPrivateKeys()

/**
    * Metoda koja uvozi javni kljuc
* @return 0
    */

void uveziJavni()

/**
    * Metoda koja uvozi privatni kljuc
* @return 0
    */

void uveziPrivatni()

/**
    * Metoda koja izvozi privatni kljuc
* @param id
    */
```

```
void izveziPrivatni(String id)

/**
    * Metoda koja izvozi javni kljuc
* @param id
    */

void izveziJavni(String id)
 /**
    * Metoda koja brise odredjeni privatni kljuc
* @param id
    */
void obrisiPrivatni(String id, String sifra)

/**
    * Metoda koja brise neki javni kljuc
* @param id
    */

void obrisiJavni(String id)

/**
    * Metoda koja iskljuci panele
* @return 0
    */

void setPanelsFalse()

/**
    * Metoda koja namesta panel za enkripciju
* @return 0
    */
public void setUpPanelForEnkripcija()

/**
    * Metoda koja verifikuje potpis
* @param in
* @return
* @throws Exception
    */


private static void verifikujFajl(InputStream in)
/**
    * Metoda koja potpisuje fajl
* @param fileName
```

```
* @param keyIn
* @param pass
* @param armor
* @return
* @throws Exception
    */

private static void potpisiFajl(String fileName, PGPSecretKey keyIn, OutputStream out, char[]
pass, boolean armor)


/**
    * Metoda koja dekriptuje poruku
* @param encrypted
* @param passPhrase
* @return
* @throws Exception
    */

public static byte[] dekriptovanje(byte[] encrypted)
/**
    * Metoda koja kriptuje fajl
* @param clearData
* @param passPhrase
* @param fileName
* @param algorithm
* @param armor
* @return
* @throws Exception
    */

public static byte[] kriptovanje(byte[] clearData, PGPPublicKey passPhrase, String fileName, int
algorithm, boolean armor)
/**
    * Metoda koja se bavi kompresijom fajla
* @param clearData
* @param algorithm
* @param fileName
* @return
* @throws Exception
    */

private static byte[] compress(byte[] clearData, String fileName, int algorithm) /**
```

```
    * Konstruktor odakle sve pocinje
* @return 0
    */

public Gui()
```