УНИВЕРЗИТЕТ У БЕОГРАДУ
ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ
Заштита података – 2020/2021.

Булевар краља Александра 73, ПФ 35-54, 11120 Београд, Србија
телефон: 011/3218-321, dekanat@etf.bg.ac.rs

OPENPGP PROTOCOL

Grupa 1

Никола Ристић 2017/0661

Лука Симић 2017/0353

Београд, 13.06.2021.

- **etf.openpgp.rn170661sl170353.gui.MainView.java**

```
public class MainView{

    private JFrame frame;
    private JTable publicKeyRingTable;
    private JTable secretKeyRingTable;
    private JPopupMenu deletePublicKeyPopupMenu;
    private JPopupMenu deleteSecretKeyPopupMenu;

    public JFrame getFrame() {...}

    public static void main(String[] args) {...}

    public MainView() {...}

    private void initialize() {...}

    public void initPublicKeyTable(PGPPublicKeyRingCollection pgpPublicKeyRingCollection) {...}

    public void initSecretKeyTable(PGPSecretKeyRingCollection pgpSecretKeyRingCollection) {...}

    public void addPGPPublicKeyRow(PGPPublicKey publicKey) {...}

    public void addPGPSecretKeyRow(PGPSecretKey secretKey) {...}

    private void importPublicKey() {...}

    private void importSecretKey() {...}

    private void exportPublicKey() {...}

    private void exportSecretKey() {...}

    private void deletePublicKey() {...}

    private void deleteSecretKey() {...}

}
```

Klasa sa main metodom i glavnim prozorom za prikaz prstena javnih i privatnih kljuceva i prikaz svih opcija koje korisnik ima na raspolaganju.

public void initPublicKeyTable(PGPPublicKeyRingCollection pgpPublicKeyRingCollection)
public void initSecretKeyTable(PGPSecretKeyRingCollection pgpSecretKeyRingCollection)

- Sluze da inicijalizuju tabele za prikaz prstena javnih I privatnih kljuceva. Ima mogucnost da se desnim klikom izabere samo jedan red I obrise, ili selektovani red izveze, klikom na dugme Export Public Key ili Expor Private Key.

```
public void addPGPPublicKeyRow(PGPPublicKey publicKey)
public void addPGPSecretKeyRow(PGPSecretKey secretKey)
```

- Prethodne dve metode koriste ove metode kako bi iz prosledjenog objekta (publicKey ili secretKey) izvukli informacije od znacaja I prikazali u nasu tablu prstena javnih I privatnih kljuceva.

```
private void importPublicKey()
private void importSecretKey()
```

- Metode pomocu kojih uvozimo javni ili privatni kljuc I prikazujemo ga u tabeli. Korisniku ce se pojaviti meni pomocu kojeg bira fajl sa .asc ekstenzijom.

```
private void exportPublicKey()
private void exportSecretKey()
```

- Izvozi izabrani javni ili privatni kljuc koji je selektovan u tabeli. Za javni kljuc zahteva unos lozinke, ako je lozinka ispravna, kljuc ce se uspesno izvesti.

```
private void deletePublicKey()
private void deleteSecretKey()
```

- Desnim klikom na tabelu se izabrani javni ili privatni kljuc trajno birse. Ukoliko je izabran privatni kljuc, trazi se unos lozinke, pa ukoliko je ispravna kljuc se uspesno brise.

- **etf.openpgp.rn170661sl170353.keylogic.KeyManager.java**

```
public class KeyManager {

    private PGPPublicKeyRingCollection publicKeyRingCollection;

    private PGPSecretKeyRingCollection secretKeyRingCollection;

    public PGPPublicKeyRingCollection getPublicKeyRingCollection() {...}

    public PGPSecretKeyRingCollection getSecretKeyRingCollection() {...}

    private static KeyManager keyManager;

    private KeyManager() {...}

    public static KeyManager getInstance() {...}

    public void generateDSAElgamalKeyPair(
            String name,
            String email,
            char[] passPhrase,
            int dsaKeySize,
            int elgamalKeySize
    ){...}

    public void initPublicKeyRingCollection() {...}

    public void initSecretKeyRingCollection() {...}

    public PGPSecretKey readSecretKeyFromFile(String filename) {...}

    public PGPPublicKey readPublicKeyFromFile(String filename) {...}

    public void importPublicKeyFromFile(File publicKeyFile) throws Exception {...}

    public void importSecretKeyFromFile(File secretKeyFile) throws Exception {...}

    public void removePublicKey(String publicKeyId) throws Exception {...}

    public void removeSecretKey(String secretKeyId) throws Exception {...}
}
```

Singleton klasa koja sluzi za generisanje PGPPublicKeyRingCollection i PGPSecretKeyRingCollection, koji predstavljaju kolekciju svih PGPPublicKeyRign I PGPSecretKeyRing iz kojih cemo kasnije PGPPrivatetKey i PGPPublicKey, a njih cemo kasnije koristiti kasnije za siforvanje i desiforovanje.

```
public void generateDSAElgamalKeyPair(String name, String email, char[] passPhrase, int dsaKeySize, int elgamalKeySize)
```

- Metoda za generisanje novog para kljuceva sa zadatim identifikatorom, sto je konkatenacija name I email, sifrom koju je korisnik uneo, kao i sa izabranom velicinom kljuceva za DSA I ElGamal. Novi parovi kljuceva se ubacuju u kolekcije I prikazuju se u tabeli.

```
public void initPublicKeyRingCollection()
public void initSecretKeyRingCollection()
```

- Metode koje inicijalizuju PGPPublicKeyRingCollection i PGPSecretKeyRingCollection, tako sto ucitaju sve .asc fajlove iz foldera public-keys I secret-keys gde se lokalno cuvaju nasi kljucevi.

```
public PGPSecretKey readSecretKeyFromFile(String filename)
public PGPPublicKey readPublicKeyFromFile(String filename)
```

- Metode koje nam citaju PGPSecreyKey I PGPPublicKey iz .asc fajlova sa zadatim imenom iz direktorijuma public-keys i secret-key, a te objekte smo koristili u nastavku kod enkripcije I dekripcije.

```
public void importPublicKeyFromFile(File publicKeyFile) throws Exception
public void importSecretKeyFromFile(File secretKeyFile) throws Exception
```

- Metode koje se pozivaju iz klase MainView kada korisnik izabere kljuc koji zeli da uveze. Otvori se ulazni tok podataka ka njemu I ucita se u PGPPublicKeyRingCollection ili PGPSecretKeyRingCollection.

```
public void removePublicKey(String publicKeyId) throws Exception
public void removeSecretKey(String secretKeyId) throws Exception
```

- Metode koje se pozivaju iz MainView kada korisnik izabere I potvrdi da zeli da izbrise odgovarajuci kljuc. Kljuc se brise iz tabele kao I iz odgovarajuce kolekcije.

- **etf.openpgp.rn170661sl170353.keylogic.Encrypt.java**

```
public class Encrypt {

    private static Encrypt instance;

    private Encrypt() {...}

    public static Encrypt getInstance() {...}

    public void ecnryptData(
        List<String> publicKeys,
        boolean radix64,
        boolean isZipped,
        String encryptAlg,
        File selectedFile,
        String privateKeyID,
        char[] passphrase
    ) {...}
}
```

Singleton klasa koja sadrzi logiku za enkriptovanje podataka (slanje poruke) tj. jednu glavnu metodu encryptData(...).

`public void ecnryptData(List<String> publicKeys, boolean radix64, boolean isZipped, String encryptAlg, File selectedFile, String privateKeyID, char[] passphrase, boolean integrity)`

- Metoda sadrzi logiku za enkriptovanje i slanje poruke. Korisniku se otvara forma gde bira da li ce se potpisati sa svojim privatnim kljucem, za koga ce da enkriptuje poruku, da li ce biti provera integrity, algoritomom za enkripciju I fajl koji zeli da posalje. Nakon toga, se od korisnika trazi lozinka, ako zeli da potpies poruku koja se salje. Metoda pocinje tako sto proveravamo da li korisnik zeli da se potpies i trazi se njegov privatKey na osnovu sifre, ako se kljuc ne nadje metoda je gotova i salnje je neuspesno. Zatim orvaramo ulazne i izlazne tokove ka fajlu koji zelimo da posaljemo I ka fajlu koji je rezultat enkripcije. Zatim pravimo objekat klase PGPEncryptedDataGenerator, kome dodamo javne kljuce korisnika Kojima

saljemo I opciono otvaramo PGPCompressedDataGenerator sa ZIP algoritmom, ako je korsnik tako zahtevao. Na kraju otvaramo I PGPSIgnatureGenerator sa OnePassVersion, ako je korsnik izabrao da zeli potpis. Na kraju podatke se svi podacai upakuju u fajl sa .pgp ekstenzijom kao niz PGPObject-a.

- **etf.openpgp.rn170661sl170353.keylogic.Decrypt.java**

```
public class Decrypt {
    private static Decrypt instance;
    private Decrypt() {...}
    public static Decrypt getInstance() {...}
    public void decryptFile(File fileToBeDecrypted) throws Exception {...}
    public void noEncrypt(Object o, PGPObjectFactory fac) throws PGPException, IOException {...}
    public PGPPrivateKey getPrivateKey(long keyID , char[] passphrase) throws PGPException , NoSuchProviderException {...}
}
```

Singleton klasa koja sadrzi logiku za dekripciju enkriptovanog fajla (prijem poruke).

`public void decryptFile(File fileToBeDecrypted) throws Exception`

- Glavna metoda ove klase, kao argument prima objekat tipa File, koji je korisnik prethodno izabrao kao fajl koji zeli da dekriptuje. Metoda pocinje tako sto se otvori ulazni tok ka njemu a zatim se na osnovu tog objekta kreira objekat PGPObjectFactory koji nas fajl na nekin nacin “deserijalizuje” tj. zna kako da niz bajtova u njemu tretira kao niz pgp objekata razlicitih tipova. Nacin na koji ce niz bajtova prpoznavati kao pgp objekte zavisi od toga kako je fajl poslat (spakovan od strane PGP-a). Na primer, ako prethodno nismo imali enkripciju, a imali smo kompresiju, prvo ce fajl prepoznati kao jedan veliki PGPCOmpressedData objekat, a zatim posle njegove “dekompresije” cemo dobiti ostatak objekata kao PGPLiteralData, PGPOnePassSignatureList I druge. Zatim moramo da dobijene objekte na neki nacin da procesiramo. Objekat PGPOnePassSignatureList, procesiaramo tako dobijamo PGPOnePassSiganture I utaj potpis proveravamo sa javnim kljucem potpisivaca. Ako se ispostavi da je verifikacija tog potpisa ispravna, onda se ispisuje poruka o pozitivnom ishodu I identitet tog potpisivaca (u obliku name<email>). PGPLiteralData su podaci od znacaja, tj desifrovani podaci I njih jednostavno preko izlaznog toka na lokicju na koju je korsnik prethodno iskazao zelju da se primljena poruka sacuva, taj sadrzaj upise u fajl.
  Ovim postpukom poruka je upotpunosti dekriptovana.

- **etf.openpgp.rn170661sl170353.keylogic.EncryptionWizard.java**
- **etf.openpgp.rn170661sl170353.keylogic.KeyPairWizardDialog.java**

Sledece klase predstavljaju klase koje prosiriju klasu JDialog, tj. predstavljaju prozore preko kojih korisnik unosi informacije neophodne za generisanje novog para kljuca I neophodne podatke za enkripciju fajla odnosno slanja poruke. Korisnik bira zeljeni nacin enkripcije I parametre za genereisanje para kljuca, a zatim ActionListenero-om se hvata ActionPreformed dogadjaj i poziva se odgovarajuce metode iz singleton klasa iz paketa keylogic, sa prosledjenim parametrima prikupljenim iz generisanih dialoga(JDialog).