**БЕЗБЕДНОСТ НА КОМПЈУТЕРСКИ СИСТЕМИ**

**ЛАБОРАТОРИСКА ВЕЖБА 2**

Фисник Лимани, 151027

**ДОКУМЕНТАЦИЈА**

1. **Main.java**

* Го креираме објектот од Kerberos класата и ја повикуваме функцијата
  + startKerberosDemonstration()

на Kerberos класата.

1. **Kerberos.java**

* Во оваа класа ја имплементираме поедноставната верзија на Kerberos протоколот.
* Функциите:
  1. sendKeysToKDC()
     + Имплементираме испраќањето на клучевите од Алис и Боб до KDC
  2. sendRequestFromAliceToKDC()
     + Имплементираме испраќање на барање од Алис до KDC
  3. sendResponseFromKDCToAlice()
     + Имплементираме враќање на одговорот од страна на KDC
  4. sendKDCDataFromAliceToBob()
     + Имплементираме испраќање на податоците, вратени од страна на KDC, од Алис до Боб
  5. sendMessageFromAliceToBob()
     + Имплементираме испраќање порака од Алис до Боб искористувајќи го сесискиот клуч добиен од страна на KDC

1. **Alice.java**

* Во оваа класа го имплементираме една од засегнатите страни – Алис
* Функциите:
  1. sendKeyToKDC(KDC kdc)
     + го испраќа клучот од Алис до KDC
  2. sendRequestToKDC(KDC kdc, Bob bob)
     + имплементација на барањето од Алис до KDC
     + bob објектот се користи за земање на неговиот ID
  3. generateNonce()
     + изгенерира noone (низа од бајтови кои не се повторуваат)
  4. acceptResponse(ResponseFromKDC response)
     + имплементација на прифаќање на одговорот на KDC што го враќа до Алис.
  5. sendKDCDataToBob(Bob bob)
     + имплементација на испраќање до Боб на податоците добиени од KDC
  6. sendMessageToBob(Bob bob, String message)
     + испраќање на енкриптирана порака искористувајќи го сесискиот клуч добиен од KDC

1. **Bob.java**

* Во оваа класа ја имплементираме вториот од засегнатите страни – Боб
* Функциите:
  1. sendKeyToKDC(KDC kdc)
     + исто како кај Алис
  2. acceptDataFromAlice(DataToBob dataToBob)
     + имплементација на прифаќање на податоците од Алис (која ги има добиени од страна на KDC)
  3. acceptMessageFromAlice(String message)
     + имплементација на прифаќање на пораката испратена од страна на Алис.

**5. KDC.java**

* Во оваа класа ја имплементираме третата од засегнатите страни – KDC

(Key Distribution Center)

* Функциите:
  + addKey(String id, SecretKey key)
    - имплементација на додавање на нов клуч во листата на клучевите зачувани од страна на KDC
  + acceptRequest(RequestToKDC request)
    - имплементација на прифаќање на барање за добивање на сесиски клуч од страна на Алис
  + respondToAlice(Alice alice)
    - имплементација на враќање на одговор за добиеното барање од страна на Алис

1. **RequestToKDC.java**

* Во оваа класа се имплементира барањето што се испраќа до KDC
* Се чува:
  + IDA – idто на Алис
  + IDB – idто на Боб
  + rA – nonce

1. **ResponseFromKDC.java**

* Во оваа класа се имплементира одговорот што се враќа од страна на KDC
* Се чува:
  + yA – содржи енкриптиран (со клучот на Алис):
    - сесиски клуч
    - nonce
    - timelife
    - idто на Боб
  + yB – содржи енкриптиран (со клучот на Боб):
    - сесиски клуч
    - idто на Алис
    - timelife

1. **DataToBob.java**

* Во оваа класа се имплементира класата што ги содржи податоците што се испраќаат од страна на Алис до Боб, а Алис ги има добиени од страна на KDC
* Се чува:
  + yAB – содржи енкриптиран (со сесискиот клуч):
    - idто на Алис
    - timestampот
  + yB – како што беше дефиниран претходно (погоре).

1. **CustomKeyGenerator.java**

* Во оваа класа се имплементира изгенерирање на random клуч
* Функции:
  + getKey()
    - го добиваме изгенерираниот random клуч