Лабораториска Вежба 1

Доверливост и автентичност на пораки

Во самиот код имаме имплементација на АЕЅ (Advanced Encryption Standard) алгоритам за шифрирање и декодирање на податоци, заедно со механизам за проверување на интегритет со MIC (Message Integrity Check).

Опис на класите:

**ClearTextFrame** – Оваа класа ни претставува наведени информации што се составени од

Header, pn и data

Header ни претставува самото заглавје на една рамка (frame) , pn ни е Packet Number или даден број што се користи како id за дадената рамка, и data што ни е самата содржина т.е самите информации што се складираат и што понатаму ќе ги шифрираме.

**EncryptedFrame** – Како што може да претпоставиме од погоре, ова ќе ни биде енкриптираната верзија од пораката наведена во минатата класа

Во неа се состои encrypted data што ќе бидат самите информации што ќе се шифрираат со алгоритмот и mic, нашиот код што ќе го проверува интегритетот

Опис на функциите:

**Calculate\_mic(key, frame)**

Оваа функција го пресметува дадениот mic код наведен во втората класа. Овде користиме AES ECB мод за генерирање на кодот.

Нашиот вектор iv го иницијализираме со некоја фиксна вредност, уникатна за него.

Со cipher создаваме нов AES објект со претходно дадениот клуч во функцијата.

Го пресметуваме нашиот MIC код со тоа што го шифрираме претходно иницијализираниот cipher.

Наведуваме for циклус кој ќе ни извршува XOR со тековниот MIC во циклусот и повторно се шифрира.

На крај на функцијата, MIC наместо во 128 бита го правиме да биде 64 бита т.е 8 бајти за да биде поефикасен

**Encrypt\_frame (key, frame)**

Оваа функција ја шифрира рамката во CTR мод за разлика од претходниот AES ECB мод и исто така генерира MIC код

Го генерираме нашиот counter (CТR) со користење на функција Counter и со претходно наведената frame и нејзиниот соодветен packet number.

Овде повторно креираме AES објект, само што во случајот за креирање го користиме нашиот генериран counter.

Повторно како и претходно генерираме cipher што ќе го шифрираме и со веќе напишаната функција од погоре, ќе го искалкулираме нашиот MIC. На крај ги враќаме енкриптираните податоци со дадениот MIC.

**Decrypt\_frame (key, encrypted\_frame, pn)**

Оваа функција ќе го дешифрира некоја веќе шифрирана рамка. Овде го земаме packet number како главна компонента која ќе ни врши работа при декриптирање, бидејќи веќе спомнавме дека самиот се користи за енкрипција и декрипција на една рамка.

Како и претходно, го генерираме нашиот counter (CТR) со користење на функција Counter и со претходно наведената frame и нејзиниот соодветен packet number.

Овде повторно креираме AES објект, и повторно за креирање го користиме нашиот претходно генериран counter.

Креираме for циклус каде нашата енкриптирана рамка ќе се декриптира претходно и ќе се споредува секој дел од нејзе со нашата Clear Text рамка дали се е идентично.

Понатаму ќе се споредат и двата MIC кодови и ако еден од нив е различен, ќе се фрли error бидејќи нема веќе интегритет на дадените податоци.

Ако не завршила програмата во error, се враќаат декриптираните податоци.

Споредни функции:

Bytes\_xor (b1, b2) – изведува општ XOR операција на две наведени низи

Frame\_blocks(frame) – ги поделува податоците на рамката на 128 битни блокови

Main функција:

Дефинираме општ AES клуч и објект од класата ClearTextFrame што ќе ни биде наречен frame.

Се повикува функцијата encrypt\_frame за да ја шифрираме рамката, и ќе добиеме шифрирана рамка со даден MIC.

Понатаму се прикажуваат дадената шифрирана рамка и MIC на екран и исто така се извршува декрипција во наредната линија.

Со функцијата decrypt\_frame ја декриптираме нашата рамка и ако се е во ред, се прикажува на екран самата декриптирана порака.