PROJECT REPORT

18CSE381T - CRYPTOGRAPHY

(2018 Regulation)

III Year/ V Semester

Academic Year: 2022 -2023

By

UTKARSH AJMANI (RA2011030010130)

NITIN DESAI (RA2011030010125)

Under the guidance of

Dr. Mary Subaja Christo

Associate Professor

Department of Networking and Communications

COLLEGE OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Kancheepuram

NOVEMBER 2022

BONAFIDE

This is to certify that **18CSE381T - CRYPTOGRAPHY**
mini project report is the bonafide work of **UTKARSH AJMANI**
(RA2011030010130) and **NITIN DESAI (RA2011030010125)**

who undertook the task of completing the project within the allotted time.

Signature of the Faculty

Dr Mary Subaja Christo

Associate Professor

Department of NWC

SRM Institute of Science and Technology
Technology

Signature of the II Year Academic Advisor

Dr. Annapurani Panaiyappan .K

Professor and Head

Department of NWC

SRM Institute of Science and

What is Cryptography

Cryptography is technique of securing information and communications through use of codes so that only those person for whom the information is intended can understand it and process it. Thus preventing unauthorized access to information. The prefix “crypt” means “hidden” and suffix graphy means “writing”.
In Cryptography the techniques which are use to protect information are obtained from mathematical concepts and a set of rule based calculations known as algorithms to convert messages in ways that make it hard to decode it. These algorithms are used for cryptographic key generation, digital signing, verification to protect data privacy, web browsing on internet and to protect confidential transactions such as credit card and debit card transactions.

Techniques used For Cryptography:

In today’s age of computers cryptography is often associated with the process where an ordinary plain text is converted to cipher text which is the text made such that intended receiver of the text can only decode it and hence this process is known as encryption. The process of conversion of cipher text to plain text this is known as decryption.
Features Of Cryptography are as follows:

1. **Confidentiality:**
   Information can only be accessed by the person for whom it is intended and no other person except him can access it.
2. **Integrity:**
   Information cannot be modified in storage or transition between sender and intended receiver without any addition to information being detected.
3. **Non-repudiation:**
   The creator/sender of information cannot deny his intention to send information at later stage.
4. **Authentication:**
   The identities of sender and receiver are confirmed. As well as destination/origin of information is confirmed.

Types Of Cryptography:
In general there are three types of cryptography:

1. **Symmetric Key Cryptography:**
   It is an encryption system where the sender and receiver of message use a single common key to encrypt and decrypt messages. Symmetric Key Systems are faster and simpler but the problem is that sender and receiver have to somehow exchange key in a secure manner. The most popular symmetric key cryptography system is Data Encryption System(DES).

2. **Hash Functions:**
   There is no usage of any key in this algorithm. A hash value with fixed length is calculated as per the plain text which makes it impossible for contents of plain text to be recovered. Many operating systems use hash functions to encrypt passwords.
3. **Asymmetric Key Cryptography:**
   Under this system a pair of keys is used to encrypt and decrypt information. A public key is used for encryption and a private key is used for decryption. Public key and Private Key are different. Even if the public key is known by everyone the intended receiver can only decode it because he alone knows the private key.

Data Encryption Standard (DES)

What is DES encryption?

DES is a symmetric block cipher (shared secret key), with a key length of 56-bits. Published as the Federal Information Processing Standards (FIPS) 46 standard in 1977, DES was officially withdrawn in 2005.

The federal government originally developed DES encryption over 35 years ago to provide cryptographic security for all government communications. The idea was to ensure government systems all used the same, secure standard to facilitate interconnectivity.

Why DES is no longer effective

To show that the DES was inadequate and should not be used in important systems anymore, a series of challenges were sponsored to see how long it would take to decrypt a message. Two organizations played key roles in breaking DES: distributed.net and the Electronic Frontier Foundation (EFF).

- The DES I contest (1997) took 84 days to break the encrypted message using a brute force attack.
- In 1998, there were two DES II challenges issued. The first challenge took just over a month and the decrypted text was *"The*

unknown message is: Many hands make light work". The second challenge took less than three days, with the plaintext message *"It's time for those 128-, 192-, and 256-bit keys".*

- The final DES III challenge in early 1999 only took 22 hours and 15 minutes. Electronic Frontier Foundation's Deep Crack computer (built for less than $250,000) and distributed.net's computing network found the 56-bit DES key, deciphered the message, and they (EFF & distributed.net) won the contest. The decrypted message read *"See you in Rome (Second AES Candidate Conference, March 22-23, 1999)",* and was found after checking about 30 percent of the key space – finally proving that DES belonged to the past.

Even Triple DES is not enough protection

Triple DES (3DES) – also known as Triple Data Encryption Algorithm (TDEA) – is a way of using DES encryption three times. But even Triple DES was proven ineffective against brute force attacks (in addition to slowing down the process substantially).

According to draft guidance published by NIST on July 19, 2018, TDEA/3DES is officially being retired. The guidelines propose that Triple DES be deprecated for all new applications and disallowed after 2023.

Advanced Encryption Standard

The more popular and widely adopted symmetric encryption algorithm likely to be encountered nowadays is the Advanced Encryption Standard (AES). It is found at least six time faster than triple DES.

A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback but it was found slow.

The features of AES are as follows −

- Symmetric key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger and faster than Triple-DES
- Provide full specification and design details
- Software implementable in C and Java

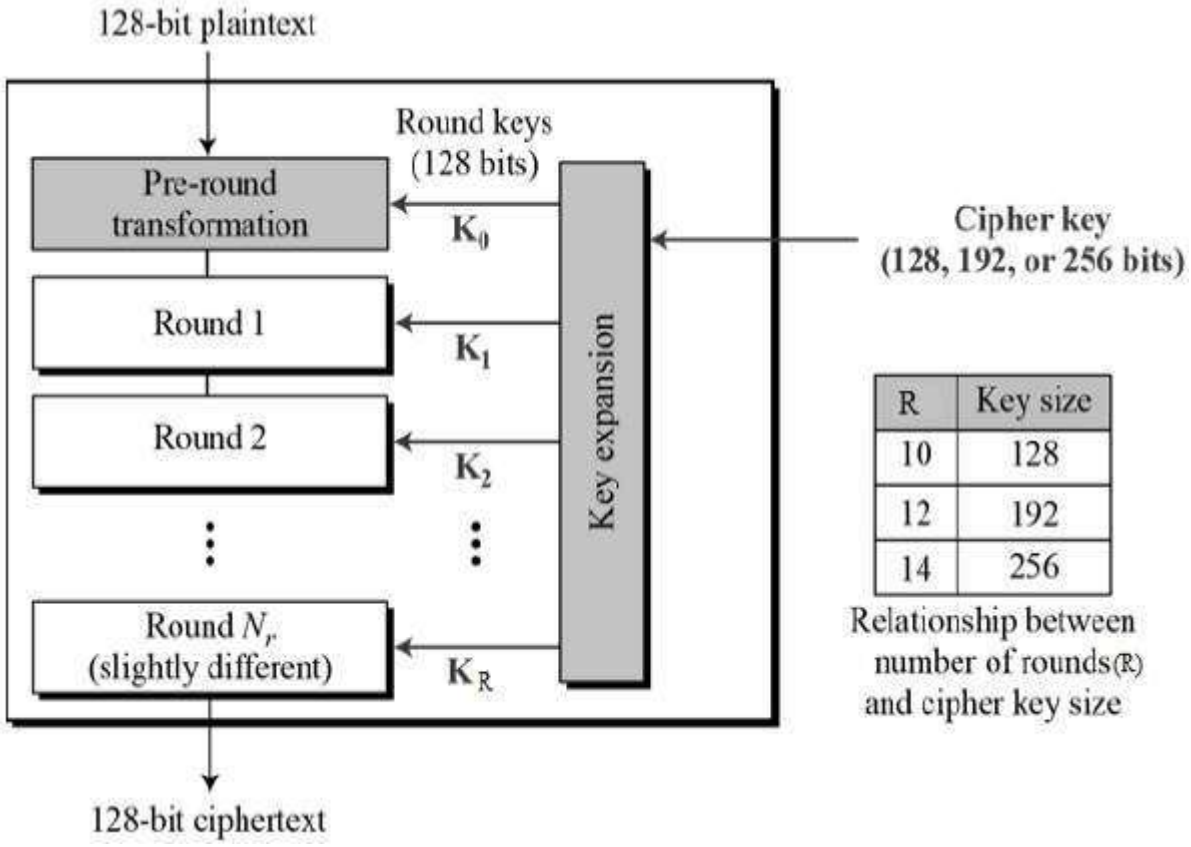
Operation of AES

AES is an iterative rather than Feistel cipher. It is based on 'substitution–permutation network'. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).

Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix −

Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

The schematic of AES structure is given in the following illustration −

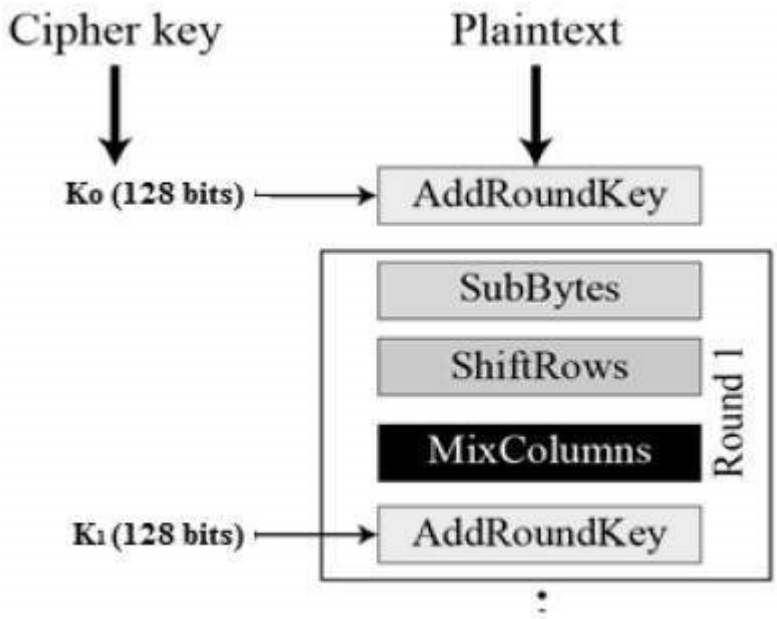


R	Key size
10	128
12	192
14	256

Relationship between number of rounds (R) and cipher key size

Encryption Process

Here, we restrict to description of a typical round of AES encryption. Each round comprise of four sub-processes. The first round process is depicted below −



Byte Substitution (SubBytes)

The 16 input bytes are substituted by looking up a fixed table (S-box) given in design. The result is in a matrix of four rows and four columns.

Shiftrows

Each of the four rows of the matrix is shifted to the left. Any entries that 'fall off' are re-inserted on the right side of row. Shift is carried out as follows −

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- Fourth row is shifted three positions to the left.
- The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

MixColumns

Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

Addroundkey

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

Decryption Process

The process of decryption of an AES ciphertext is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order −

- Add round key
- Mix columns
- Shift rows
- Byte substitution

Since sub-processes in each round are in reverse manner, unlike for a Feistel Cipher, the encryption and decryption algorithms needs to be separately implemented, although they are very closely related.

AES Analysis

In present day cryptography, AES is widely adopted and supported in both hardware and software. Till date, no practical cryptanalytic attacks against AES has been discovered. Additionally, AES has built-in flexibility of key length, which allows a degree of 'future-proofing' against progress in the ability to perform exhaustive key searches.

However, just as for DES, the AES security is assured only if it is correctly implemented and good key management is employed.

Code

```
import java.nio.charset.StandardCharsets;

import java.security.spec.KeySpec;

import java.util.Base64;

import java.util.*;

import javax.crypto.Cipher;

import javax.crypto.SecretKey;

import javax.crypto.SecretKeyFactory;

import javax.crypto.spec.IvParameterSpec;

import javax.crypto.spec.PBEKeySpec;

import javax.crypto.spec.SecretKeySpec;


class AES {
```

```
// Class private variables
private static final String SECRET_KEY
   = "my_super_secret_key_ho_ho_ho";

private static final String SALT = "ssshhhhhhhhhhh!!!!";

// This method use to encrypt to string
public static String encrypt(String strToEncrypt)
{
   try {

      // Create default byte array
      byte[] iv = { 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0, 0, 0, 0 };
      IvParameterSpec ivspec
         = new IvParameterSpec(iv);

      // Create SecretKeyFactory object
      SecretKeyFactory factory
         = SecretKeyFactory.getInstance(
            "PBKDF2WithHmacSHA256");

      // Create KeySpec object and assign with
      // constructor
      KeySpec spec = new PBEKeySpec(
         SECRET_KEY.toCharArray(), SALT.getBytes(),
         65536, 256);
```

```
            SecretKey tmp = factory.generateSecret(spec);
            SecretKeySpec secretKey = new SecretKeySpec(
                tmp.getEncoded(), "AES");

            Cipher cipher = Cipher.getInstance(
                "AES/CBC/PKCS5Padding");
            cipher.init(Cipher.ENCRYPT_MODE, secretKey,
                        ivspec);
            // Return encrypted string
            return Base64.getEncoder().encodeToString(
                cipher.doFinal(strToEncrypt.getBytes(
                    StandardCharsets.UTF_8)));
        }
        catch (Exception e) {
            System.out.println("Error while encrypting: "
                               + e.toString());
        }
        return null;
    }

    // This method use to decrypt to string
    public static String decrypt(String strToDecrypt)
    {
        try {

            // Default byte array
            byte[] iv = { 0, 0, 0, 0, 0, 0, 0, 0,
```

```
            0, 0, 0, 0, 0, 0, 0, 0 };
// Create IvParameterSpec object and assign with
// constructor
IvParameterSpec ivspec
   = new IvParameterSpec(iv);

// Create SecretKeyFactory Object
SecretKeyFactory factory
   = SecretKeyFactory.getInstance(
      "PBKDF2WithHmacSHA256");

// Create KeySpec object and assign with
// constructor
KeySpec spec = new PBEKeySpec(
   SECRET_KEY.toCharArray(), SALT.getBytes(),
   65536, 256);
SecretKey tmp = factory.generateSecret(spec);
SecretKeySpec secretKey = new SecretKeySpec(
   tmp.getEncoded(), "AES");

Cipher cipher = Cipher.getInstance(
   "AES/CBC/PKCS5PADDING");
cipher.init(Cipher.DECRYPT_MODE, secretKey,
          ivspec);
// Return decrypted string
return new String(cipher.doFinal(
   Base64.getDecoder().decode(strToDecrypt)));
```

```
        }
        catch (Exception e) {
            System.out.println("Error while decrypting: "
                               + e.toString());
        }
        return null;
    }
}

// driver code
public class Main {
    public static void main(String[] args)
    {
        // Create String variables
        Scanner sc= new Scanner(System.in);
        String originalString =sc.nextLine();

        // Call encryption method
        String encryptedString
            = AES.encrypt(originalString);

        // Call decryption method
        String decryptedString
            = AES.decrypt(encryptedString);



        System.out.println(encryptedString);
```

System.out.println(decryptedString);}}

Main.java

```
109 }
110
111 // driver code
112 public class Main {
113     public static void main(String[] args)
114     {
115         // Create String variables
116         Scanner sc= new Scanner(System.in);
117         String originalString =sc.nextLine();
118
119         // Call encryption method
120         String encryptedString
121             = AES.encrypt(originalString);
122
123         // Call decryption method
124         String decryptedString
125             = AES.decrypt(encryptedString);
126
127
128         System.out.println(encryptedString);
129         System.out.println(decryptedString);
130     }
131 }
```

input

```
his is cryptography project
XS4/FIxlK75oMtKLwS3gqyTePKJww17mOQuXEsXMsM=
his is cryptography project


..Program finished with exit code 0
ress ENTER to exit console.
```

```
104                 System.out.println("Error while decrypting: "
105                                    + e.toString());
106             }
107             return null;
108         }
109 }
110
111 // driver code
112 public class Main {
113     public static void main(String[] args)
114     {
115         // Create String variables
116         Scanner sc= new Scanner(System.in);
117         String originalString =sc.nextLine();
118
119         // Call encryption method
120         String encryptedString
121             = AES.encrypt(originalString);
122
123         // Call decryption method
124         String decryptedString
125             = AES.decrypt(encryptedString);
126
```

input

```
I am implementing aes algorithm
lAR+y+YSLm1ABIFQbhI4ObkkUtNymoYER7J5Xt2Zu0U=
I am implementing aes algorithm


...Program finished with exit code 0
Press ENTER to exit console.
```