Name: E Varun sai

Reg.no: 20BCD7179

Cryptography Analysis and Implementation

**Objective:** The objective of this assignment is to analyse cryptographic algorithms and implement them in a practical scenario.

**Introduction**

A cryptographic algorithm is the mathematical equation used to scramble the plain text and make it unreadable. They are used for data encryption, authentication and digital signatures.

To do this, we have three different classes of algorithms namely

* Symmetric Key Algorithms
* Asymmetric Key Algorithms
* Hash functions

Let’s have a brief discussion about each type of Cryptographic Algorithm

1. **Symmetric Key Algorithms:**

Both sender and receiver share a single key and the sender uses this key to encrypt plaintext. The cipher text is sent to the receiver, and the receiver can apply this same key to decrypt the message and recover the plain text from the sender.

1. **Asymmetric Key Algorithms(Public Key Cryptography):**

In public key cryptography (PKI), also known as asymmetric cryptography, there are two related keys called the public and private key. While the public key may be freely distributed, the paired private key must remain confidential. The public key is used for encryption and the private key is used for decryption.

1. **Hash functions:**

No key is used in this algorithm, but a fixed-length value is calculated from the plaintext, which makes it impossible for the contents of the plaintext to be recovered. Hash functions are often used by computer systems to encrypt passwords. A popular hash algorithm is 256-SHA.

3 cryptographic algorithms that I have chosen to explain for this assignment are

* Advanced Encryption Standard (AES) – from Symmetric key Algorithms
* RSA Algorithm – from Asymmetric Algorithms
* MD5 Algorithm – from Hash functions

Ok let’s start with AES Algorithm.

**AES Algorithm**

* AES is a block cipher.
* The key size can be 128/192/256 bits.
* Encrypts data in blocks of 128 bits each.

Working of the cipher:

AES performs operations on bytes of data rather than in bits. Since the block size is 128 bits, the cipher processes 128 bits (or 16 bytes) of the input data at a time.

The number of rounds depends on the key length as follows:

* 128 bit key – 10 rounds
* 192 bit key – 12 rounds
* 256 bit key – 14 rounds

Creation of Round keys:

A Key Schedule algorithm is used to calculate all the round keys from the key. So the initial key is used to create many different round keys which will be used in the corresponding round of the encryption.

Encryption:

AES considers each block as a 16 byte (4 byte x 4 byte = 128) grid in a column major arrangement.

[b0 | b4 | b8 | b12 |

| b1 | b5 | b9 | b13 |

| b2 | b6 | b10| b14 |

| b3 | b7 | b11| b15]

Each round comprises of 4 steps:

* SubBytes
* ShiftRows
* MixColumns
* Add Round Key

The last round doesn’t have the MixColumns round.

The SubBytes does the substitution and ShiftRows and MixColumns performs the permutation in the algorithm.

SubBytes:

This step implements the substitution.

In this step each byte is substituted by another byte. Its performed using a lookup table also called the S-box. This substitution is done in a way that a byte is never substituted by itself and also not substituted by another byte which is a compliment of the current byte. The result of this step is a 16 byte (4 x 4) matrix like before.

The next two steps implement the permutation.

ShiftRows:

This step is just as it sounds. Each row is shifted a particular number of times.

The first row is not shifted

The second row is shifted once to the left.

The third row is shifted twice to the left.

The fourth row is shifted thrice to the left.

(A left circular shift is performed.)

[b0 | b1 | b2 | b3 ] [ b0 | b1 | b2 | b3 ]

| b4 | b5 | b6 | b7 | -> | b5 | b6 | b7 | b4 |

| b8 | b9 | b10 | b11 | | b10 | b11 | b8 | b9 |

[ b12 | b13 | b14 | b15 ] [ b15 | b12 | b13 | b14 ]

MixColumns:

This step is basically a matrix multiplication. Each column is multiplied with a specific matrix and thus the position of each byte in the column is changed as a result.

This step is skipped in the last round.

[ c0 ] [ 2 3 1 1 ] [ b0 ]

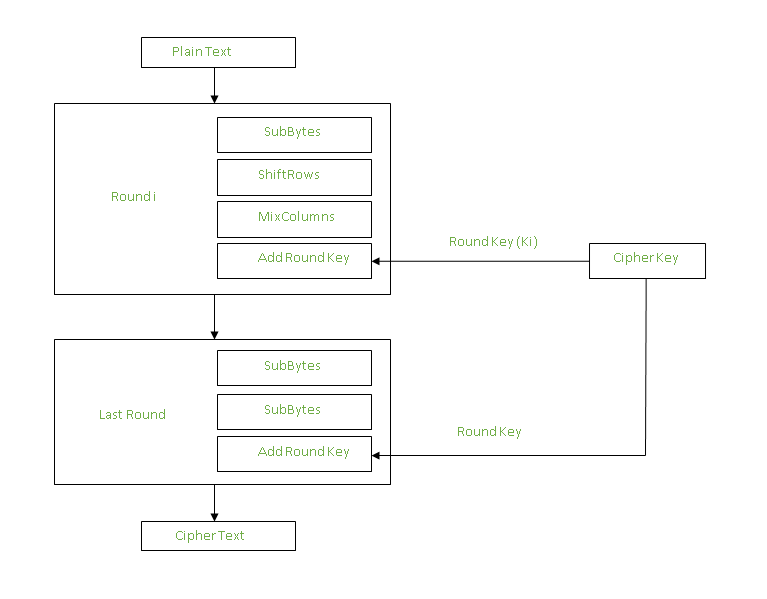
| c1 | | 1 2 3 1 | | b1 |

| c2 | | 1 1 2 3 | | b2 |

[ c3 ] [ 3 1 1 2 ] [ b3 ]

Add Round Keys:

Now the resultant output of the previous stage is XOR-ed with the corresponding round key. Here, the 16 bytes is not considered as a grid but just as 128 bits of data.



After all these rounds 128 bits of encrypted data is given back as output. This process is repeated until all the data to be encrypted undergoes this process.

**Decryption:**

The stages in the rounds can be easily undone as these stages have an opposite to it which when performed reverts the changes.Each 128 blocks goes through the 10,12 or 14 rounds depending on the key size.

The stages of each round in decryption is as follows :

* Add round key
* Inverse MixColumns
* ShiftRows
* Inverse SubByte

The decryption process is the encryption process done in reverse so i will explain the steps with notable differences.

Inverse MixColumns:

This step is similar to the MixColumns step in encryption, but differs in the matrix used to carry out the operation.

[ b0 ] [ 14 11 13 9 ] [ c0 ]

| b1 | = | 9 14 11 13 | | c1 |

| b2 | | 13 9 14 11 | | c2 |

[ b3 ] [ 11 13 9 14 ] [ c3 ]

Inverse SubBytes:

Inverse S-box is used as a lookup table and using which the bytes are substituted during decryption.

**Advantages of AES algorithm**

➨As it is implemented in both hardware and software, it is most robust security protocol.

➨It uses higher length key sizes such as 128, 192 and 256 bits for encryption. Hence it makes AES algorithm more robust against hacking.

➨It is most common security protocol used for wide variety of applications such as wireless communication, financial transactions, e-business, encrypted data storage etc.

➨It is one of the most widely used commercial and open source solutions across the world.

➨No one can hack your personal information.

➨For 128 bit, about 2128 attempts are needed to break. This makes it very difficult to hack it as a result it is very safe protocol.

**Disadvantages of AES algorithm**

➨It uses too simple algebraic structure.

➨Every block is always encrypted in the same way.

➨Hard to implement with software.

➨AES in counter mode is complex to implement in software taking both performance and security into considerations

**Real world application of AES algorithm**

* **Wireless security:** AES is used in securing wireless networks, such as Wi-Fi networks, to ensure data confidentiality and prevent unauthorized access.
* **Database Encryption:** AES can be applied to encrypt sensitive data stored in databases. This helps protect personal information, financial records, and other confidential data from unauthorized access in case of a data breach.
* **Secure communications:** AES is widely used in protocols like such as internet communications, email, instant messaging, and voice/video calls.It ensures that the data remains confidential.
* **Data storage:** AES is used to encrypt sensitive data stored on hard drives, USB drives, and other storage media, protecting it from unauthorized access in case of loss or theft.
* **Virtual Private Networks (VPNs):** AES is commonly used in VPN protocols to secure the communication between a user’s device and a remote server. It ensures that data sent and received through the VPN remains private and cannot be deciphered by eavesdroppers.
* **Secure Storage of Passwords**: AES encryption is commonly employed to store passwords securely. Instead of storing plaintext passwords, the encrypted version is stored. This adds an extra layer of security and protects user credentials in case of unauthorized access to the storage.
* **File and Disk Encryption:** AES is used to encrypt files and folders on computers, external storage devices, and cloud storage. It protects sensitive data stored on devices or during data transfer to prevent unauthorized access.

Next RSA algorithm

**RSA Algorithm**

**RSA algorithm** is an asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. **Public Key** and **Private Key.** As the name describes that the Public Key is given to everyone and the Private key is kept private.

**An example of asymmetric cryptography:**

1. A client (for example browser) sends its public key to the server and requests some data.
2. The server encrypts the data using the client’s public key and sends the encrypted data.
3. The client receives this data and decrypts it.

Since this is asymmetric, nobody else except the browser can decrypt the data even if a third party has the public key of the browser.

**The idea!** The idea of RSA is based on the fact that it is difficult to factorize a large integer. The public key consists of two numbers where one number is a multiplication of two large prime numbers. And private key is also derived from the same two prime numbers. So if somebody can factorize the large number, the private key is compromised. Therefore encryption strength totally lies on the key size and if we double or triple the key size, the strength of encryption increases exponentially. RSA keys can be typically 1024 or 2048 bits long, but experts believe that 1024-bit keys could be broken in the near future. But till now it seems to be an infeasible task.

**Let us learn the mechanism behind the RSA algorithm : >> Generating Public Key:**

Select two prime no's. Suppose **P = 53 and Q = 59.**

**Now First part of the Public key : n = P\*Q = 3127.**

We also need a small exponent say **e :**

**But e Must be**

**An integer.**

**Not be a factor of Φ(n).**

**1 < e < Φ(n)**

Let us now consider it to be equal to 3.

Our Public Key is made of n and e

**>> Generating Private Key:**

We need to calculate Φ(n) :

Such that **Φ(n) = (P-1)(Q-1)**

**so, Φ(n) = 3016**

Now calculate Private Key, **d :**

**d = (k\*Φ(n) + 1) / e for some integer k**

**For k = 2, value of d is 2011.**

Now we are ready with our – Public Key ( n = 3127 and e = 3) and Private Key(d = 2011) Now we will encrypt **“HI”**:

Convert letters to numbers : H = 8 and I = 9

Thus **Encrypted Data c = (89e)mod n**

**Thus our Encrypted Data comes out to be 1394**

Now we will decrypt **1394** :

**Decrypted Data = (cd)mod n**

**Thus our Encrypted Data comes out to be 89**

**8 = H and I = 9 i.e. "HI".**

**Advantages:**

* **Security:**RSA algorithm is considered to be very secure and is widely used for secure data transmission.
* **Public-key cryptography:**RSA algorithm is a public-key cryptography algorithm, which means that it uses two different keys for encryption and decryption. The public key is used to encrypt the data, while the private key is used to decrypt the data.
* **Key exchange:**RSA algorithm can be used for secure key exchange, which means that two parties can exchange a secret key without actually sending the key over the network.
* **Digital signatures:**RSA algorithm can be used for digital signatures, which means that a sender can sign a message using their private key, and the receiver can verify the signature using the sender’s public key.
* **Speed:** The RSA technique is suited for usage in real-time applications since it is quite quick and effective.
* **Widely used:** Online banking, e-commerce, and secure communications are just a few fields and applications where the RSA algorithm is extensively developed.

**Disadvantages:**

* **Slow processing speed:**RSA algorithm is slower than other encryption algorithms, especially when dealing with large amounts of data.
* **Large key size:**RSA algorithm requires large key sizes to be secure, which means that it requires more computational resources and storage space.
* **Vulnerability to side-channel attacks:** RSA algorithm is vulnerable to side-channel attacks, which means an attacker can use information leaked through side channels such as power consumption, electromagnetic radiation, and timing analysis to extract the private key.
* **Limited use in some applications:** RSA algorithm is not suitable for some applications, such as those that require constant encryption and decryption of large amounts of data, due to its slow processing speed.
* **Complexity:** The RSA algorithm is a sophisticated mathematical technique that some individuals may find challenging to comprehend and use.
* **Key Management:** The secure administration of the private key is necessary for the RSA algorithm, although in some cases this can be difficult.
* **Vulnerability to Quantum Computing:** Quantum computers have the ability to attack the RSA algorithm, potentially decrypting the data.

**Real Life Applications of RSA algorithm**

* Creating coded transmissions or messages.
* Used with other encryption methods to enhance security.
* Used to secure internet-enabled software to protect data.
* Securing the connection between VPN servers and clients.

**MD5 Algorithm**

MD5 is a cryptographic hash function algorithm that takes the message as input of any length and changes it into a fixed-length message of 16 bytes. MD5 algorithm stands for the message-digest algorithm. MD5 was developed as an improvement of MD4, with advanced security purposes. The output of MD5 (Digest size) is always 128 bits. MD5 was developed in 1991 by Ronald Rivest.

MD5 algorithm follows the following steps

**1. Append Padding Bits:**In the first step, we add padding bits in the original message in such a way that the total length of the message is 64 bits less than the exact multiple of 512. 

Suppose we are given a message of 1000 bits. Now we have to add padding bits to the original message. Here we will add 472 padding bits to the original message.  After adding the padding bits the size of the original message/output of the first step will be 1472 i.e. 64 bits less than an exact multiple of 512 (i.e. 512\*3 = 1536).

**Length(original message + padding bits) =  512 \* i – 64**where i = 1,2,3 . . .

**2. Append Length Bits:**In this step, we add the length bit in the output of the first step in such a way that the total number of the bits is the perfect multiple of 512. Simply, here we add the 64-bit as a length bit in the output of the first step.   
i.e. output of first step = 512 \* n – 64   
length bits = 64.

After adding both we will get**512 \* n**i.e. the exact multiple of 512.

**3. Initialize MD buffer:**Here, we use the 4 buffers i.e. J, K, L, and M. The size of each buffer is 32 bits.

- J = 0x67425301

- K = 0xEDFCBA45

- L = 0x98CBADFE

- M = 0x13DCE476

**4. Process Each 512-bit Block:**This is the most important step of the MD5 algorithm. Here, a total of 64 operations are performed in 4 rounds. In the 1st round, 16 operations will be performed, 2nd round 16 operations will be performed, 3rd round 16 operations will be performed, and in the 4th round, 16 operations will be performed. We apply a different function on each round i.e. for the 1st round we apply the F function, for the 2nd G function, 3rd for the H function, and 4th for the I function.   
We perform OR, AND, XOR, and NOT (basically these are logic gates) for calculating functions. We use 3 buffers for each function i.e. K, L, M.

- F(K,L,M) = (K AND L) OR (NOT K AND M)

- G(K,L,M) = (K AND L) OR (L AND NOT M)

- H(K,L,M) = K XOR L XOR M

- I(K,L,M) = L XOR (K OR NOT M)

After applying the function now we perform an operation on each block. For performing operations we need

* add modulo 232
* M[i] – 32 bit message.
* K[i] – 32-bit constant.
* <<<n – Left shift by n bits.

Now take input as initialize MD buffer i.e. J, K, L, M. Output of  K will be fed in L, L will be fed into M, and M will be fed into J. After doing this now we perform some operations to find the output for J.

* In the first step, Outputs of K, L, and M are taken and then the function F is applied to them. We will add modulo 232bits for the output of this with J.
* In the second step, we add the M[i] bit message with the output of the first step.
* Then add 32 bits constant i.e. K[i] to the output of the second step.
* At last, we do left shift operation by n (can be any value of n) and addition modulo by 232.

After all steps, the result of J will be fed into K. Now same steps will be used for all functions G, H, and I. After performing all 64 operations we will get our message digest.

**Advantages of MD5 Algorithm:**

* MD5 is faster and simple to understand.
* MD5 algorithm generates a strong password in 16 bytes format. All developers like web developers etc use the MD5 algorithm to secure the password of users.
* To integrate the MD5 algorithm, relatively low memory is necessary.
* It is very easy and faster to generate a digest message of the original message.

**Disadvantages of MD5 Algorithm:**

* MD5 generates the same hash function for different inputs.
* MD5 provides poor security over SHA1.
* MD5 has been considered an insecure algorithm. So now we are using SHA256 instead of MD5
* MD5 is neither a symmetric nor asymmetric algorithm.

**Application Of MD5 Algorithm:**

* We use message digest to verify the integrity of files/ authenticates files.
* MD5 was used for data security and encryption.
* It is used to Digest the message of any size and also used for Password verification.
* For Game Boards and Graphics.

Implementation of RSA in Java

Code:

import java.io.\*;

import java.math.\*;

import java.util.\*;

public class RSA {

    public static *double* gcd(*double* *a*, *double* *h*)

    {

*double* temp;

        while (true) {

            temp = a % h;

            if (temp == 0)

                return h;

            a = h;

            h = temp;

        }

    }

    public static *void* main(*String*[] *args*)

    {

*double* p = 3;

*double* q = 7;

*double* n = p \* q;

*double* e = 2;

*double* phi = (p - 1) \* (q - 1);

        while (e < phi) {

            if (gcd(e, phi) == 1)

                break;

            else

                e++;

        }

*int* k = 2;

*double* d = (1 + (k \* phi)) / e;

*double* msg = 12;

        System.out.println("Message data = " + msg);

*double* c = Math.pow(msg, e);

        c = c % n;

        System.out.println("Encrypted data = " + c);

*double* m = Math.pow(c, d);

        m = m % n;

        System.out.println("Original Message Sent = " + m);

    }

}

Output:

