VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SMART BRIDGE – Ethical Hacking

Assessment - 3

NAME : LOKESHWARAN M

REG NO : 20BCE2599

DATE : 22-5-2023

SCHOOL : SCOPE

Assignment-3: Cryptography Analysis and Implementation

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

Instructions:
Research: Begin by conducting research on different cryptographic algorithms such as symmetric key algorithms (e.g., AES, DES), asymmetric key algorithms (e.g., RSA, Elliptic Curve Cryptography), and hash functions (e.g., MD5, SHA-256). Understand their properties, strengths, weaknesses, and common use cases.

Analysis: Choose three cryptographic algorithms (one symmetric, one asymmetric, and one hash function) and write a detailed analysis of each. Include the following points in your analysis:

Briefly explain how the algorithm works.
Discuss the key strengths and advantages of the algorithm.
Identify any known vulnerabilities or weaknesses.
Provide real-world examples of where the algorithm is commonly used.

Implementation:
Select one of the cryptographic algorithms you analyzed and implement it in a practical scenario. You can choose any suitable programming language for the implementation.
Clearly define the scenario or problem you aim to solve using cryptography.
Provide step-by-step instructions on how you implemented the chosen algorithm. Include code snippets and explanations to demonstrate the implementation.
Test the implementation and discuss the results.

Security Analysis:
Perform a security analysis of your implementation, considering potential attack vectors and countermeasures.

Identify potential threats or vulnerabilities that could be exploited.
Propose countermeasures or best practices to enhance the security of your implementation.
Discuss any limitations or trade-offs you encountered during the implementation process.
Conclusion: Summarize your findings and provide insights into the importance of cryptography in cybersecurity and ethical hacking.

Submission Guidelines:
Prepare a well-structured report that includes the analysis, implementation steps, code snippets, and security analysis.
Use clear and concise language, providing explanations where necessary. Include any references or sources used for research and analysis.
Compile all the required files (report, code snippets, etc.) into a single zip file for submission.

Analysis:

Symmetric Algorithm :DES

DES (Data Encryption Standard) is a symmetric-key block cipher algorithm that was developed in the 1970s by IBM in collaboration with the National Bureau of Standards (now known as the National Institute of Standards and Technology, NIST). It became a widely adopted encryption standard for protecting sensitive data. DES operates on 64-bit blocks of plaintext using a 56-bit encryption key. The algorithm employs a series of substitutions, permutations, and bitwise operations to transform the input data into encrypted ciphertext. The encryption process involves multiple rounds of transformation, each using a different 48-bit subkey derived from the original key through a key schedule process.

How does DES work?

<u>Key Generation:</u> A 56-bit encryption key is generated, with 8 of those bits used for parity, resulting in a 64-bit key. This key is used for both encryption

and decryption.

Subkey Generation: The 64-bit key is transformed into sixteen 48-bit subkeys through a process called key schedule. These subkeys are used in different rounds of encryption and provide additional security.

Encryption Process: The input plaintext, divided into 64-bit blocks, undergoes a series of transformations known as Feistel network. Each block is divided into two halves, and a set of substitutions and permutations are applied to the data using the subkeys.

Decryption Process: The decryption process is similar to encryption, but the subkeys are used in reverse order. The ciphertext is divided into 64-bit blocks, and the reverse permutations and substitutions are applied using the corresponding subkeys.

Strengths of DES:

Speed and Efficiency: DES is designed to be relatively fast and efficient, making it suitable for various applications that require real-time or high-speed data encryption.

Widely Adopted: DES has been widely adopted and implemented in numerous software and hardware systems, ensuring compatibility and interoperability.

Advantages of the DES algorithm

- **It supports functionality to save a file in an encrypted format which can only be accessed by supporting the correct password.**
- **It can change the system to create the directories password protected.**
- **It can review a short history of DES and represent the basic structures.**

- **It can define the building block component of DES.**

- **It can define the round keys generation process and to interpret data encryption standard.**
- **It can provide that private information is not accessed by other users.**
- **Some users can use the similar system and still can work individually.**

Disadvantages of the DES algorithm

- **The 56 bit key size is the largest defect of DES and the chips to implement one million of DES encrypt or decrypt operations a second are applicable (in 1993).**
- **Hardware implementations of DES are very quick.**
- **DES was not designed for application and therefore it runs relatively slowly.**
- **In a new technology, it is improving a several possibility to divide the encrypted code, therefore AES is preferred than DES.**

Real-World Example of DES: Legacy Systems:

DES was widely used in the past and is still present in some legacy systems where backward compatibility is required.

Asymmetric Algorithm: RSA

RSA (Rivest-Shamir-Adleman) is a widely used asymmetric cryptographic algorithm that is named after its inventors: Ron Rivest, Adi Shamir, and Leonard Adleman. It is primarily used for secure communication and data encryption. RSA is widely used for secure communication, digital signatures, and key exchange in various applications, including secure email, SSL/TLS encryption for web browsing, and secure communication protocols like SSH.

How RSA Algorithm works:

Key Generation:

✉**Choose two distinct prime numbers, p and q.**

✉Compute the modulus, N, by multiplying p and q: N = p * q.

✉Calculate Euler's totient function, φ(N), which is the number of positive integers less than N that are coprime (do not share any common factors) with N. In RSA, φ(N) = (p
- 1) * (q - 1).

✉Select a public exponent, e, which is a small odd integer greater than 1 and less than φ(N). Typically, e = 65537 (2^16 + 1) is used as it has good cryptographic properties and is efficient for computation.

✉Compute the private exponent, d, as the modular multiplicative inverse of e modulo φ(N). In other words, d is the number such that (d * e) mod φ(N) = 1.

The public key is (N, e), and the private key is (N, d). N is the modulus and e is the public exponent.

<u>Encryption:</u>

To encrypt a message M, the sender uses the recipient's public key (N, e).

The sender converts the message into a numerical representation, usually using a predetermined scheme like ASCII or Unicode.

Each block of the numerical representation is encrypted separately.

For each block, the sender computes the ciphertext C using the encryption formula: C
= M^e mod N.

The sender sends the ciphertext C to the

recipient. <u>Decryption:</u>

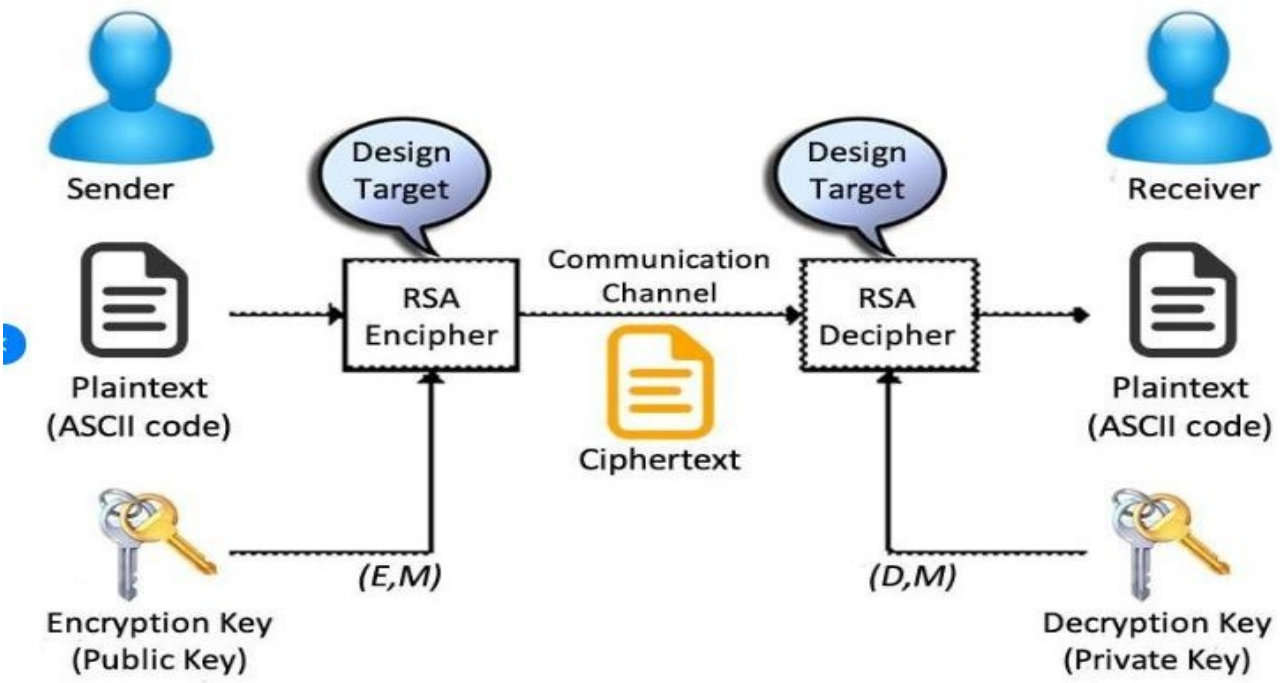
To decrypt the ciphertext C, the recipient uses their private key (N, d).

The recipient applies the decryption formula: M = C^d mod N.

The recipient obtains the original numerical representation of the message.

The recipient converts the numerical representation back into the original message format.

RSA ALGORITHM:



Strengths of RSA:

RSA offers strong security by leveraging the difficulty of factoring large numbers. RSA's use of different keys for encryption and decryption enables secure communication without a shared secret. it supports the generation and verification of digital signatures, ensuring data integrity and authenticity.RSA facilitates secure key exchange, allowing parties to establish a shared secret key over an insecure channel. RSA is a well-established and widely used algorithm, with broad support and integration in various cryptographic systems and protocols.

Benefits or advantages of RSA

- **Strong Security: RSA provides a high level of security, relying on the computational difficulty of factoring large numbers into their prime factors. As long as an appropriate key length is used, RSA encryption is considered secure against brute-force attacks.**

- **Public-Key Encryption: RSA uses a pair of keys: a public key for encryption and a private key for decryption. This enables secure communication between parties without the need to exchange a secret key beforehand. The public key can be freely distributed, while the private key remains confidential.**

- **Digital Signatures: RSA supports the generation and verification of digital signatures. Digital signatures ensure the integrity and authenticity of data, providing a mechanism for non-repudiation. This is crucial in applications such as verifying the authenticity of software, securing financial transactions, and ensuring message integrity.**

- **Key Exchange: RSA allows for secure key exchange, enabling parties to establish a shared secret key over an insecure channel. This key can then be used for symmetric encryption, offering efficiency and confidentiality.**

Overall, RSA's benefits include strong security, public-key encryption, digital signature support, key exchange capabilities, efficiency, and widespread adoption, making it a versatile and widely used cryptographic algorithm.

Drawbacks or disadvantages of RSA

- **Complexity – RSA algorithm is a complex mathematical method that can be difficult for some people to understand and implement.**
- **Key Size – RSA algorithm requires large prime numbers as part of the encryption process. The larger the prime numbers, the more secure the encryption, but it also increases the key size and processing time.**
- **Speed – RSA algorithm can be slower than other encryption methods, especially when encrypting large amounts of data.**
- **Vulnerability to Quantum Computing – RSA algorithm is vulnerable to attacks by quantum computers, which can potentially break the encryption.**
- **Key Management – RSA algorithm requires the secure management of the private key, which can be a challenge in certain scenarios.**

Weakness:

RSA is vulnerable to attacks from future quantum computers, which can efficiently factor large numbers and compromise its security. Longer key lengths are needed to maintain the same level of security as computing power advances, which can impact performance and increase storage and bandwidth requirements. RSA implementations can be vulnerable to side-channel attacks and improper padding schemes, potentially leaking information about the private key or the plaintext.

Examples : OpenPGP
RSA is widely used in secure email protocols . SSL/TLS encryption for web browsing, and virtual private networks (VPNs).

Hash function: MD5

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.

How does MD5 work?

MD5 runs entire files through a mathematical hashing algorithm to generate a signature that can be matched with an original file. That way, a received file can be authenticated as matching the original file that was sent, ensuring that the right files get where they need to go.

The MD5 hashing algorithm converts data into a string of 32 characters. For example, the word "frog" always generates this hash: 938c2cc0dcc05f2b68c4287040cfcf71. Similarly, a file of 1.2 GB also generates a hash with the same number of characters. When you send that file to someone, their computer authenticates its hash to ensure it matches the one you sent.

If you change just one bit in a file, no matter how large the file is, the hash output will be completely and irreversibly changed. Nothing less than an exact copy will pass the MD5 test.

Strengths of MD5:

MD5 (Message Digest Method 5) is a cryptographic hash algorithm used to generate a 128-bit digest from a string of any length. It represents the digests as 32 digit hexadecimal numbers. Ronald Rivest designed this algorithm in 1991 to provide the means for digital signature verification.

Advantages of the MD5 algorithm

- It's easier to compare and store smaller hashes using MD5 Algorithms than it is to store a large variable-length text.

- **By using MD5, passwords are stored in 128-bit format.**

- **You may check for file corruption by comparing the hash values before and after transmission. To prevent data corruption, file integrity tests are valid once the hashes match.**

- **A message digest can easily be created from an original message using MD5.**

Disadvantages of the MD5 algorithm

- **When compared to other algorithms like the SHA algorithm, MD5 is comparatively slow.**
- **It is possible to construct the same hash function for two distinct inputs using MD5.**
- **MD5 is less secure when compared to the SHA algorithm since MD5 is more vulnerable to collision attacks.**

Real-World Example of Hashing: Online Passwords

Every time you attempt to log in to your email account, your email provider hashes the password YOU enter and compares this hash to the hash it has saved. Only when the two hashes match are you authorized to access your email.

IMPLEMENTATIONS:

Aim and abstract:

To provide a secure method for encryption, decryption, digital signatures, and key exchange in information and communication systems. It aims to ensure the confidentiality, integrity, and authenticity of data, as well as facilitate secure communication between parties without the need for a pre-shared secret key. RSA utilizes the mathematical properties of prime factorization and modular arithmetic. It involves the generation of a public-private key pair, where the public key is used for encryption and verification, while the private key is used for decryption and signing. RSA's aim is to provide a robust and secure method for secure communication and data protection in various applications, such as secure messaging, secure online transactions, and digital certificates. Its effectiveness relies on the computational difficulty of factoring large numbers into their prime factors, ensuring the security of encrypted data and digital signatures.

Key words: Security, cryptosystem, Prime numbers , encryption, private keys,public

Existing Method

The existing method of RSA encryption and decryption involves the following steps:

1. Key Generation:

Generate two large prime numbers, p and q.

Calculate the modulus n by multiplying p and q: n = p * q.

Choose an encryption exponent e, which is relatively prime to (p-1)(q-1).

Calculate the decryption exponent d using the modular inverse of e modulo (p-1)(q-1).

2. Encryption:

Convert the plaintext message into numerical form using a predetermined

mapping or encoding scheme.

Apply the encryption algorithm using the recipient's public key (n, e).

The encryption algorithm raises the numerical representation of the plaintext to the power of e modulo n, resulting in the ciphertext.

3.Decryption:

Apply the decryption algorithm using the recipient's private key (n, d).

The decryption algorithm raises the ciphertext to the power of d modulo n,resulting in the numerical representation of the original plaintext.

Convert the numerical representation back into the plaintext message using the decoding scheme.

PROPOSED SYSTEM:

The proposed system of RSA refers to a set of enhancements or modifications to the existing RSA algorithm with the aim of improving its security, efficiency, and functionality. It involves introducing new features or techniques that can address the limitations or vulnerabilities of the original RSA algorithm.

Implementation Details and Analysis:

I'm Implementing using Java:

Generating Public Key

1. Select two prime no's. Suppose P = 53 and Q = 59. Now First part of the Public key : n = P*Q = 3127.

2. We also need a small exponent say e : But e Must be

-An integer.

-Not be a factor of n.

-1 < e < Φ(n) [Φ(n) is discussed below], Let us now consider it to be equal to 3.

The public key has been made of n and

e <u>Generating Private Key</u>

1. We need to calculate Φ(n) : Such that Φ(n) = (P-1)(Q-1)
so, Φ(n) = 3016

2. Now calculate Private Key, d :
d = (k*Φ(n) + 1) / e for some integer k
3. For k = 2, value of d is 2011.

The private key has been made of d

JAVA CODE :

Rsa,java

```
import java.math.*;
import java.util.*;


class Rsa {
   public static void main(String args[]) {
      int p, q, n, z, d = 0, e, i;


      // The number to be encrypted and decrypted int msg = 12;
      double c;
      BigInteger msgback;


      // 1st prime number p p = 3;


      // 2nd prime number q q = 11;
      n = p * q;
      z = (p - 1) * (q - 1);
      System.out.println("the value of z = " + z);
      for (e = 2; e < z; e++) {
         if (gcd(e, z) == 1) {
            break;
         }
      }
      System.out.println("the value of e = " + e);
      for (i = 0; i <= 9; i++) {
         int x = 1 + (i * z);


         if (x % e == 0) {
            d = x / e;
            break;
         }
      }
      System.out.println("the value of d = " + d);
      c = (Math.pow(msg, e)) % n;
      System.out.println("Encrypted message is : " + c);


      BigInteger N = BigInteger.valueOf(n);
```

```
    BigInteger C = BigDecimal.valueOf(c).toBigInteger();
    msgback = (C.pow(d)).mod(N);
    System.out.println("Decrypted message is : " + msgback);
  }


  static int gcd(int e, int z) {
    if (e == 0)
      return z;
    else
      return gcd(z % e, e);
  }


}
```

Output :

Output :

Security Analysis:

IBM RSA Security Analytics is an enterprise security information and event management (SIEM) product. The purpose of a SIEM is to harvest, analyze and report on security log data across an enterprise, including network-based security controls and host operating systems and applications.

Limitations:

1. **RSA only employs asymmetric encryption and complete encryption requires both symmetric and asymmetric encryption, it might occasionally fail.**
2. **Sometimes, it's necessary for a third party to confirm the dependability of public keys.**
3. **Since so many people are engaged, the data transfer rate is slow.**
4. **RSA cannot be used for public data encryption, such as electoral voting.**

5. Decryption requires intensive processing on the receiver’s end.

Conclusion:

In conclusion, RSA (Rivest-Shamir-Adleman) is a widely used asymmetric cryptographic algorithm that provides encryption, decryption, digital signatures, and key exchange capabilities. It offers several strengths and advantages, including strong security based on the difficulty of factoring large numbers, proven effectiveness in real-world applications, and wide industry support. RSA does have limitations and vulnerabilities that need to be carefully considered. These include the need for larger key sizes to maintain security against advances in computing power, potential vulnerabilities to quantum computing, performance considerations for computationally intensive operations, and the importance of proper key management and secure implementation

Overall, RSA provides a valuable foundation for secure communication and data protection, but it is important to stay informed about its strengths, weaknesses, and evolving cryptographic landscape to make informed decisions about its usage and consider alternative algorithms when appropriate.