Expt 1: Implementing Password Security

CODE - 1a

```
#include <iostream>
using namespace std;
int main(){
   string pass = "admin";
   cout<<"Enter Password:"<<endl;
   string passin;
   getline(cin, passin);
   if(passin == pass){
       cout<<"Correct Password"<<endl;
   }
     else{
       cout<<"Wrong password"<<endl;
   }
}
```

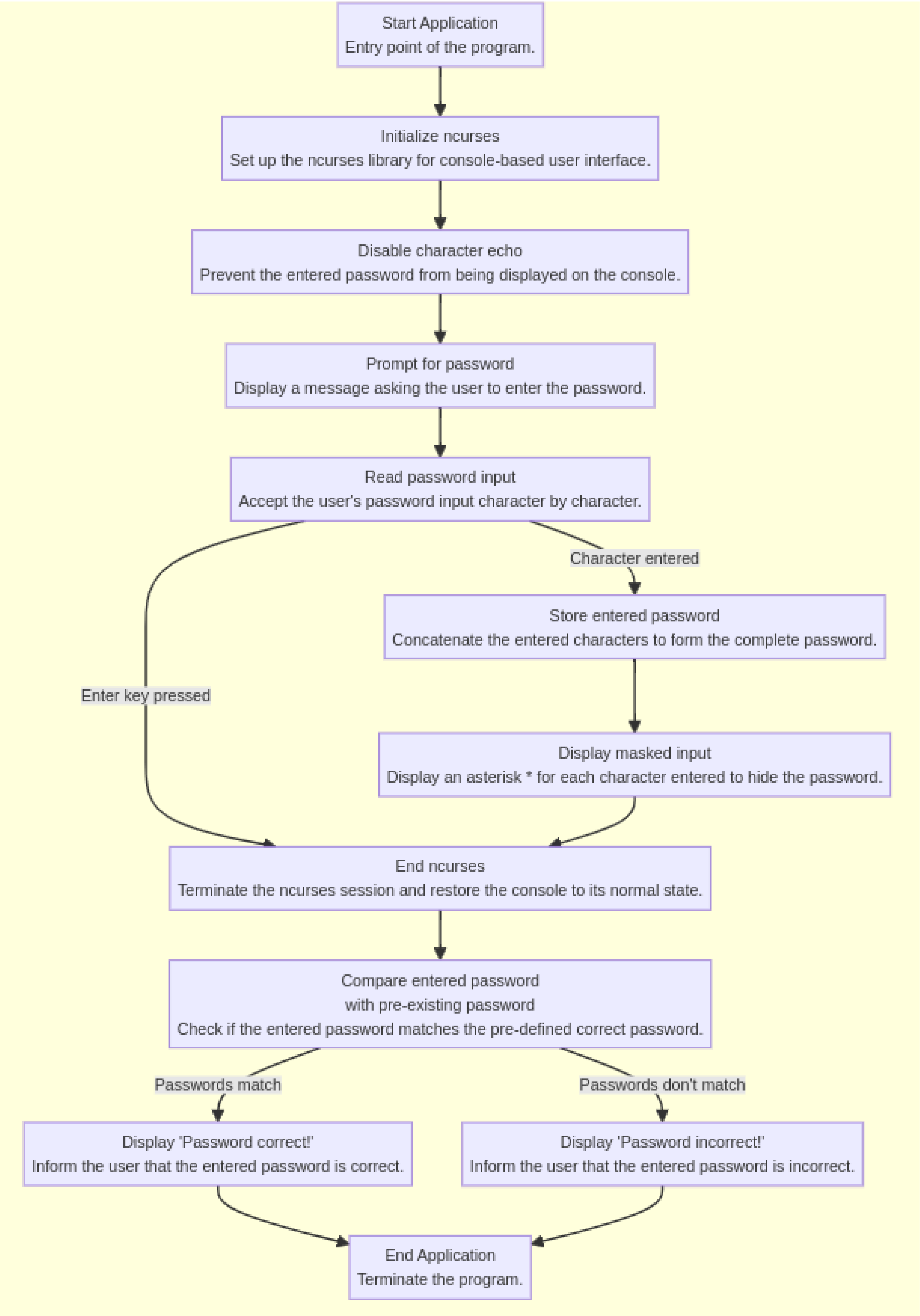
CODE - 1b

```
#include <iostream>
#include <ncurses.h>
using namespace std;
int main() {
   string preexisting_password = "password123";
   string entered_password;
   char ch;
   initscr(); // Initialize ncurses
   noecho();  // Don't echo characters to the console
   printw("Enter password: ");
   refresh();

   while (true) {
       ch = getch();
       if (ch == '\n') // Enter key pressed
           break;
       entered_password += ch;
```

```
        printw("*");
        refresh();
    }

    endwin(); // End ncurses
    cout << endl;
    if (entered_password == preexisting_password) {
        cout << "Password correct!" << endl;
    } else {
        cout << "Password incorrect!" << endl;
    }
    return 0;
}
```

```
sudo apt-get install libncurses-dev
 g++ -o a 1b.cpp -lncurses
```

Install MinGW


Start Application
Entry point of the program.
Initialize ncurses
Set up the ncurses library for console-based user interface.
Disable character echo
Prevent the entered password from being displayed on the console.
Prompt for password
Display a message asking the user to enter the password.
Read password input
Accept the user's password input character by character.
Character entered
Store entered password
Concatenate the entered characters to form the complete password.
Display masked input
Display an asterisk * for each character entered to hide the password.
Enter key pressed
End ncurses
Terminate the ncurses session and restore the console to its normal state.
Compare entered password
with pre-existing password
Check if the entered password matches the pre-defined correct password.
Passwords match
Passwords don't match
Display 'Password correct!'
Inform the user that the entered password is correct.
Display 'Password incorrect!'
Inform the user that the entered password is incorrect.
End Application
Terminate the program.

Explain RSA Algorithms with example.

RSA

The RSA (Rivest-Shamir-Adleman) algorithm is a widely used asymmetric encryption algorithm named after its inventors. It is based on the practical difficulty of factoring the product of two large prime numbers, the factoring problem. The security of RSA relies on the difficulty of factoring large composite numbers, known as the RSA problem.

Key Generation:

Choose two large prime numbers, p and q.
Compute n = p * q, which is the modulus for the public and private keys.
Compute the totient of n, φ(n) = (p-1) * (q-1).
Choose an integer e such that 1 < e < φ(n) and gcd(e, φ(n)) = 1; e is the public exponent.
Compute the private exponent d such that d * e ≡ 1 (mod φ(n)); d is the private exponent.

Encryption:

Given a message M, the ciphertext C is computed as C ≡ M^e (mod n).

Decryption:

Given a ciphertext C, the original message M is computed as M ≡ C^d (mod n).

Example:

Let's take a simple example with small numbers for illustration purposes:

Choose p = 5 and q = 7.
Compute n = p * q = 5 * 7 = 35.
Compute φ(n) = (p-1) * (q-1) = 4 * 6 = 24.
Choose e = 5 (any number coprime with 24 will work).
Compute d = 5^-1 mod 24 = 5 (since 5 * 5 ≡ 1 mod 24).

Public key (e, n) = (5, 35)
Private key (d, n) = (5, 35)

Encryption:

Let's encrypt a message M = 10.
Ciphertext C = 10^5 mod 35 = 100000 mod 35 = 10.

Decryption:

To decrypt the ciphertext C = 10.
Message M = 10^5 mod 35 = 100000 mod 35 = 10.

Explain any two of the following.

- **Hill Cipher:**

  Hill cipher is a polygraphic substitution cipher based on linear algebra.Each letter is represented by a number modulo 26. Often the simple scheme A = 0, B = 1, ..., Z = 25 is used, but this is not an essential feature of the cipher. To encrypt a message, each block of n letters (considered as an n-component vector) is multiplied by an invertible n × n matrix, against modulus 26. To decrypt the message, each block is multiplied by the inverse of the matrix used for encryption.

  Encryption

  We have to encrypt the message 'ACT' (n=3).The key is 'GYBNQKURP' which can be written as the nxn matrix:

$$\begin{bmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{bmatrix}$$

  The message 'ACT' is written as vector:

$$\begin{bmatrix} 0 \\ 2 \\ 19 \end{bmatrix}$$

  The enciphered vector is given as:

$$\begin{bmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{bmatrix} \begin{bmatrix} 0 \\ 2 \\ 19 \end{bmatrix} = \begin{bmatrix} 67 \\ 222 \\ 319 \end{bmatrix} \equiv \begin{bmatrix} 15 \\ 14 \\ 7 \end{bmatrix} \pmod{26}$$

  which corresponds to ciphertext of 'POH'

- **Vigenere Cipher.**

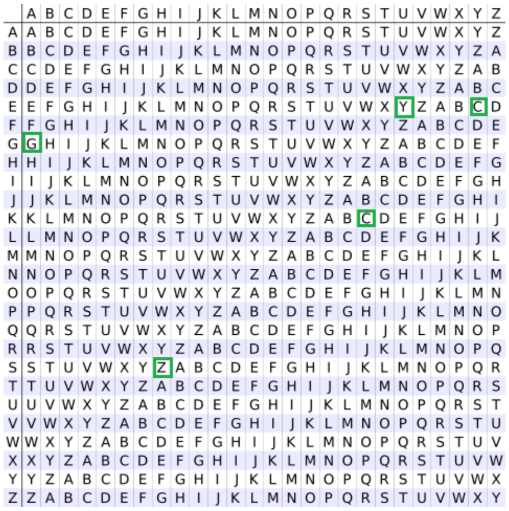
  Vigenere Cipher is a method of encrypting alphabetic text. It uses a simple form of polyalphabetic substitution. A polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets. The encryption of the original text is done using the Vigenère square or Vigenère table.

The table consists of the alphabets written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible Caesar Ciphers.
At different points in the encryption process, the cipher uses a different alphabet from one of the rows.
The alphabet used at each point depends on a repeating keyword.

Example:
Input : Plaintext : GEEKSFORGEEKS
Keyword : AYUSH
Output : Ciphertext : GCYCZFMLYLEIM

For generating key, the given keyword is repeated in a circular manner until it matches the length of the plain text.
The keyword "AYUSH" generates the key "AYUSHAYUSHAYU"
The plain text is then encrypted using the process explained below.
Encryption:

The first letter of the plaintext, G is paired with A, the first letter of the key. So use row G and column A of the Vigenère square, namely G. Similarly, for the second letter of the plaintext, the second letter of the key is used, the letter at row E, and column Y is C. The rest of the plaintext is enciphered in a similar fashion.
Table to encrypt – Geeks



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Decryption:

Decryption is performed by going to the row in the table corresponding to the key, finding the position of the ciphertext letter in this row, and then using the column's label as the plaintext. For example, in row A (from AYUSH), the ciphertext G appears in column G, which is the first plaintext letter. Next, we go to row Y (from AYUSH), locate the ciphertext C which is found in column E, thus E is the second plaintext letter.

A more easy implementation could be to visualize Vigenère algebraically by converting [A-Z] into numbers [0–25].

Encryption

The plaintext(P) and key(K) are added modulo 26.

Ei = (Pi + Ki) mod 26

Decryption

Di = (Ei - Ki) mod 26

Note: Di denotes the offset of the i-th character of the plaintext. Like offset of A is 0 and of B is 1 and so on.

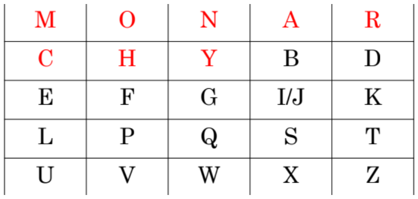
- Vernam Cipher:

- Playfair Cipher:

The Playfair cipher is a cryptographic technique used for encryption and decryption. It was invented by Charles Wheatstone in 1854, but it was named after Lord Playfair, who promoted its use. The cipher is a type of substitution cipher that employs a 5x5 grid of letters, called a Playfair square or Playfair matrix, to encrypt pairs of letters from the plaintext.

Encryption Process:

1. Key Preparation: The Playfair cipher requires a keyword to generate the Playfair matrix. The key is used to fill the matrix, excluding duplicate letters and the letters 'J' and 'I' (which are treated as the same letter).
2. Playfair Matrix: The key is used to fill a 5x5 grid (Playfair matrix) with unique letters, typically starting with the letters of the keyword (excluding duplicates) followed by the remaining letters of the alphabet (excluding 'J/I').



M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

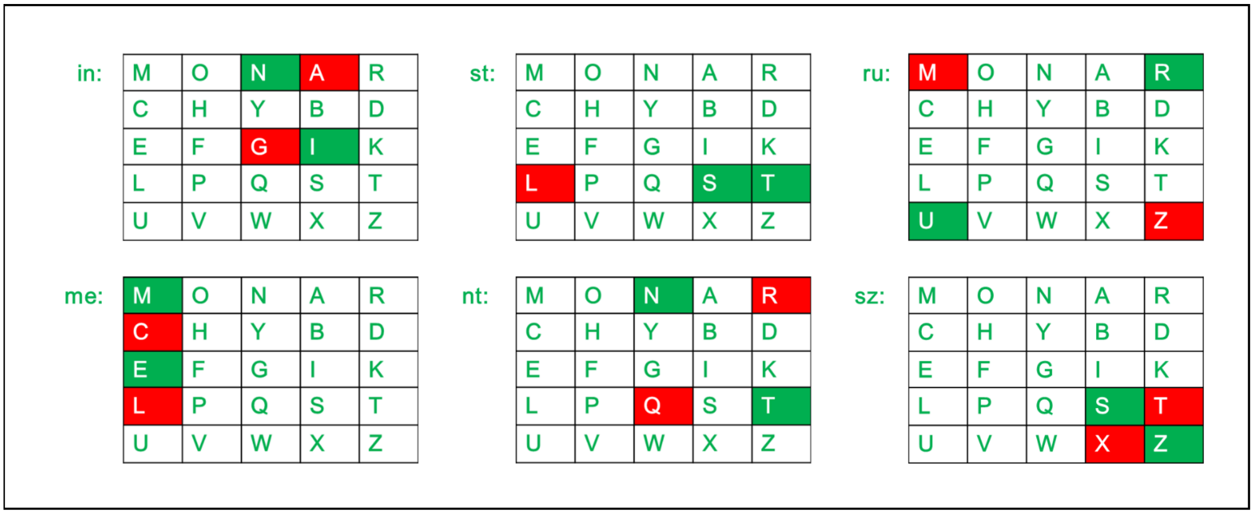
3. Pairing: The plaintext message is divided into pairs of letters. If there is an odd number of letters, a dummy letter (often 'X') is added at the end.
4. Encryption: Each pair of letters is encrypted using the Playfair matrix rules:
   - If the letters are in the same row, replace them with the letters to their immediate right (wrapping around to the left if needed).
   - If the letters are in the same column, replace them with the letters immediately below (wrapping around to the top if needed).
   - If the letters form a rectangle, replace them with the letters on the same row but at the opposite corners of the rectangle.
5. If both letters are the same, insert a dummy letter (often 'X') between them and proceed as usual.

Decryption Process:

1. Playfair Matrix: The same Playfair matrix used for encryption is used for decryption.
2. Decryption: Each pair of letters in the ciphertext is decrypted using the Playfair matrix rules in reverse:
   - If the letters are in the same row, replace them with the letters to their immediate left (wrapping around to the right if needed).

- If the letters are in the same column, replace them with the letters immediately above (wrapping around to the bottom if needed).
- If the letters form a rectangle, replace them with the letters on the same row but at the opposite corners of the rectangle.

Plain Text: "instrumentsz"
Encrypted Text: gatlmzclrqtx



in:

M	O	🟩 N	🟥 A	R
C	H	Y	B	D
E	F	🟥 G	🟩 I	K
L	P	Q	S	T
U	V	W	X	Z

st:

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
🟥 L	P	Q	🟩 S	🟩 T
U	V	W	X	Z

ru:

🟥 M	O	N	A	🟩 R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
🟩 U	V	W	X	🟥 Z

me:

🟩 M	O	N	A	R
🟥 C	H	Y	B	D
🟩 E	F	G	I	K
🟥 L	P	Q	S	T
U	V	W	X	Z

nt:

M	O	🟩 N	A	🟥 R
C	H	Y	B	D
E	F	G	I	K
L	P	🟥 Q	S	🟩 T
U	V	W	X	Z

sz:

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	🟩 S	🟥 T
U	V	W	🟥 X	🟩 Z

Difference between Mono & Polyalphabetic Cipher.

SR.NO	Monoalphabetic Cipher	Polyalphabetic Cipher
1	Monoalphabetic cipher is one where each symbol in plain text is mapped to a fixed symbol in cipher text.	Polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets.
2	The relationship between a character in the plain text and the characters in the cipher text is one-to-one.	The relationship between a character in the plain text and the characters in the cipher text is one-to-many.
3	Each alphabetic character of plain text is mapped onto a unique alphabetic character of a cipher text.	Each alphabetic character of plain text can be mapped onto 'm' alphabetic characters of a cipher text.
4	A stream cipher is a monoalphabetic cipher if the value of key does not depend on the position of the plain text character in the plain text stream.	A stream cipher is a polyalphabetic cipher if the value of key does depend on the position of the plain text character in the plain text stream.
5	It includes additive, multiplicative, affine and monoalphabetic substitution cipher.	It includes autokey, Playfair, Vigenere, Hill, one-time pad, rotor, and Enigma cipher.
6	It is a simple substitution cipher.	It is multiple substitutions cipher.

7	Monoalphabetic Cipher is described as a substitution cipher in which the same fixed mappings from plain text to cipher letters across the entire text are used.	Polyalphabetic Cipher is described as substitution cipher in which plain text letters in different positions are enciphered using different cryptoalphabets.
8	Monoalphabetic ciphers are not that strong as compared to polyalphabetic cipher.	Polyalphabetic ciphers are much stronger.

Compare in Tabular fashion, Confidentiality, Integrity, Authentication and Non repudiation for following security techniques

a. Classical Encryption/Decryption

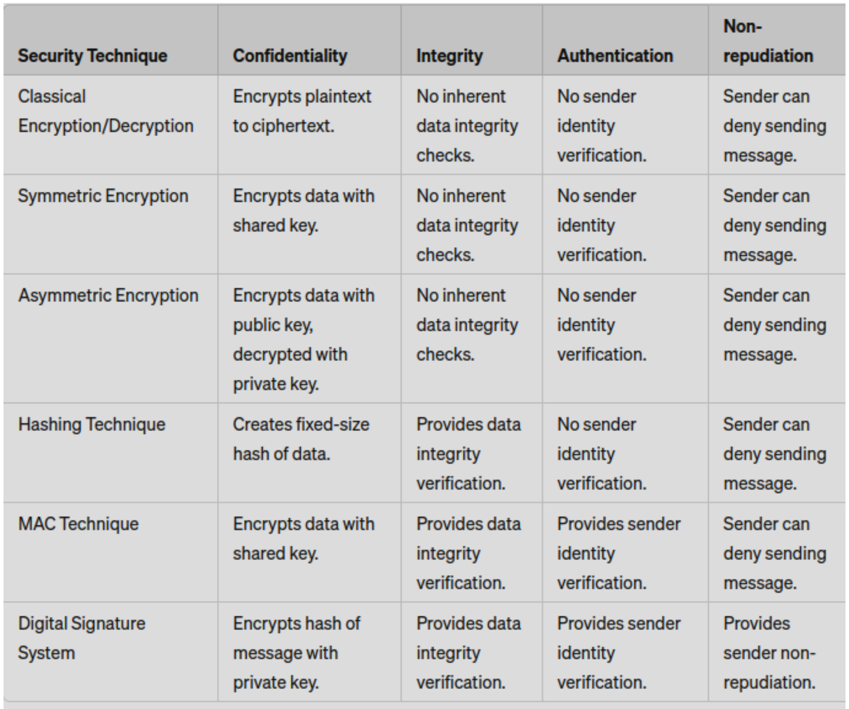
b. Symmetric Encryption

c. Asymmetric Encryption

d. Hashing Technique

e. MAC technique

f. Digital Signature System



Security Technique	Confidentiality	Integrity	Authentication	Non-repudiation
Classical Encryption/Decryption	Encrypts plaintext to ciphertext.	No inherent data integrity checks.	No sender identity verification.	Sender can deny sending message.
Symmetric Encryption	Encrypts data with shared key.	No inherent data integrity checks.	No sender identity verification.	Sender can deny sending message.
Asymmetric Encryption	Encrypts data with public key, decrypted with private key.	No inherent data integrity checks.	No sender identity verification.	Sender can deny sending message.
Hashing Technique	Creates fixed-size hash of data.	Provides data integrity verification.	No sender identity verification.	Sender can deny sending message.
MAC Technique	Encrypts data with shared key.	Provides data integrity verification.	Provides sender identity verification.	Sender can deny sending message.
Digital Signature System	Encrypts hash of message with private key.	Provides data integrity verification.	Provides sender identity verification.	Provides sender non-repudiation.

Explain DES in detail.

DES

The Data Encryption Standard (DES) is a symmetric-key block cipher algorithm that encrypts data in 64-bit blocks. It was adopted by the U.S. government in 1977 and became a widely used encryption standard. DES operates on 64-bit plaintext blocks and uses a 56-bit key, where 8 bits are used for parity and not for encryption. Here is a basic overview of the DES algorithm:

Key Generation:

The 56-bit key is used to generate 16 subkeys, one for each round of encryption and one for the final permutation.

The original 56-bit key is permuted to generate a 64-bit key, with one bit added to each 8-bit byte for parity.

Initial Permutation (IP):

The 64-bit plaintext block is permuted according to a fixed table.

16 Rounds of Encryption:

Expansion: The 32-bit half-block is expanded to 48 bits using a fixed table.

Key Mixing: The expanded half-block is XORed with the 48-bit subkey for the round.

Substitution: The result is divided into 8 6-bit blocks, each block is substituted using one of 8 S-boxes, producing 32 output bits.

Permutation: The 32 output bits are permuted according to a fixed table.

Swap: The two 32-bit halves of the block are swapped.

Final Permutation (FP):

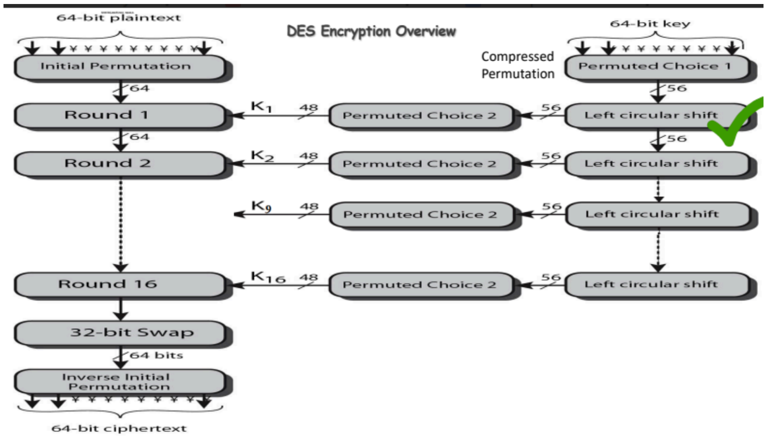
The final 64-bit ciphertext block is permuted according to a fixed table, which is the inverse of the initial permutation.

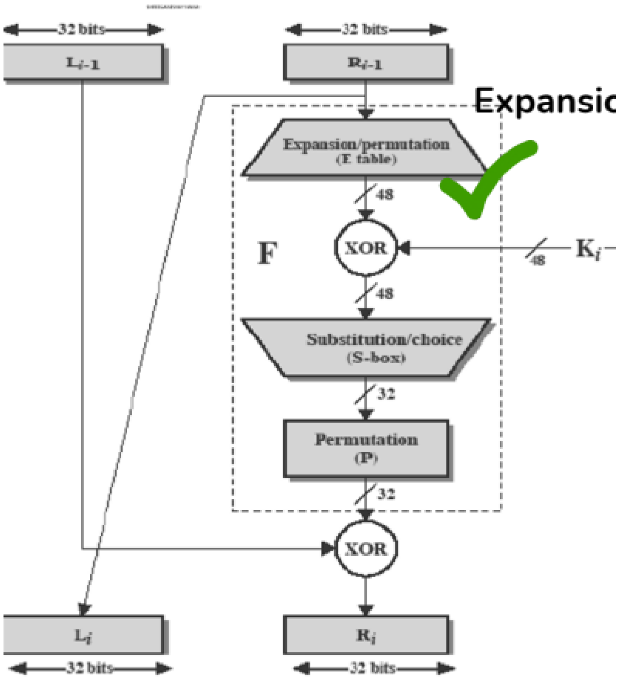
Decryption:

The decryption process is the same as encryption, except that the subkeys are used in reverse order.

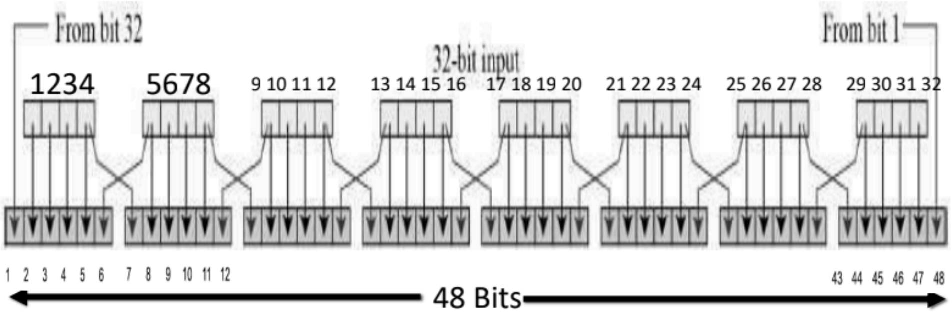
Key Points:

- DES has a 56-bit effective key size, as 8 of the 64 bits are used for parity and not for encryption.
- DES was the standard encryption algorithm for many years but is now considered insecure due to its small key size.
- Triple DES (3DES) is a variant of DES that applies DES three times with different keys, providing a higher level of security.

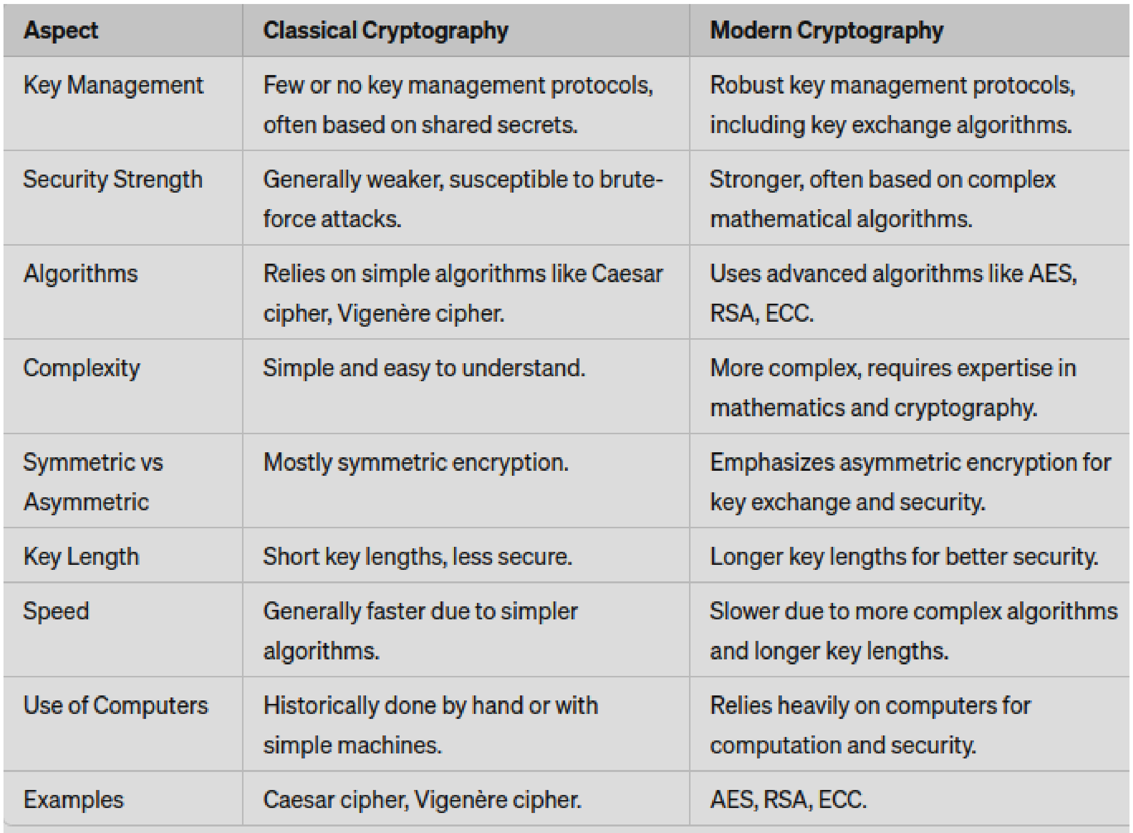




Expansion Permutation Box (32 to 48 Bits)



Compare Classical & Modern Cryptography



Aspect	Classical Cryptography	Modern Cryptography
Key Management	Few or no key management protocols, often based on shared secrets.	Robust key management protocols, including key exchange algorithms.
Security Strength	Generally weaker, susceptible to brute-force attacks.	Stronger, often based on complex mathematical algorithms.
Algorithms	Relies on simple algorithms like Caesar cipher, Vigenère cipher.	Uses advanced algorithms like AES, RSA, ECC.
Complexity	Simple and easy to understand.	More complex, requires expertise in mathematics and cryptography.
Symmetric vs Asymmetric	Mostly symmetric encryption.	Emphasizes asymmetric encryption for key exchange and security.
Key Length	Short key lengths, less secure.	Longer key lengths for better security.
Speed	Generally faster due to simpler algorithms.	Slower due to more complex algorithms and longer key lengths.
Use of Computers	Historically done by hand or with simple machines.	Relies heavily on computers for computation and security.
Examples	Caesar cipher, Vigenère cipher.	AES, RSA, ECC.

What is DoS attack? What is DDoS attack? How to Mitigate it?

A Denial-of-Service (DoS) attack is an attempt to make a computer resource unavailable to its intended users. This can be achieved by flooding the target with a large volume of traffic or by sending it malicious requests that consume its resources, such as bandwidth, memory, or processing power. The goal of a DoS attack is to disrupt the normal functioning of the targeted system or network.

A Distributed Denial-of-Service (DDoS) attack is a variant of the DoS attack where multiple compromised systems, often referred to as a botnet, are used to launch the attack. These systems, controlled by the attacker, simultaneously flood the target with traffic or requests, making it more difficult to mitigate the attack.

Mitigating DoS and DDoS attacks involves implementing various measures to protect the targeted system or network. Some common mitigation techniques include:

1. **Network Filtering:** Implementing firewalls, routers, and intrusion detection systems (IDS) to filter out malicious traffic and allow only legitimate traffic to reach the target.
2. **Traffic Analysis**: Monitoring network traffic patterns to detect and block abnormal traffic flows that may indicate a DoS or DDoS attack.
3. **Rate Limiting**: Limiting the rate of incoming traffic to prevent overwhelming the target system.
4. **Load Balancing**: Distributing incoming traffic across multiple servers to distribute the load and prevent any single server from being overwhelmed.
5. **Application Security**: Ensuring that applications and services are secure against common vulnerabilities that could be exploited in a DoS or DDoS attack.
6. **Cloud-Based Protection**: Using cloud-based DDoS protection services that can absorb and filter out malicious traffic before it reaches the target network.
7. **Incident Response Plan**: Having an incident response plan in place to quickly detect, respond to, and mitigate the effects of a DoS or DDoS attack.
8. **Regular Updates and Patches**: Keeping systems and software up to date with the latest security patches and updates to protect against known vulnerabilities.

Expt 2: Transposition Cipher

```
#include <bits/stdc++.h>
using namespace std;

//Encryption function
string Encryption(int no_rows, int len_key, int len_msg, string msg,
int col_val[])
{
   int x = 0;
   char enc_mat[no_rows + 1][len_key];
 //creating the matrix
   for (int i = 0; i < no_rows + 1; i++)
   {
       for (int j = 0; j < len_key; j++)
       {
         //initializes the positions with '_' after the end of message
           if (x >= len_msg)
           {
               enc_mat[i][j] = '_';
           }
           else
           {
               enc_mat[i][j] = msg[x];
           }
           x++;
       }
   }
    int t = 1;
   string cipher = "";
 //finding the cipher text according to the value of col_val matrix
   while (t <= len_key)
   {
       for (int i = 0; i < len_key; i++)
       {
           int k = col_val[i];
           if (k == t)
           {
               for (int j = 0; j < no_rows + 1; j++)
               {
                   cipher += enc_mat[j][i];
               }
               t++;
           }
       }
   }
   return cipher;
}

//decryption function
string Decryption(int no_rows, int len_key, string cipher,int
col_val[])
{
   char dec_mat[no_rows + 1][len_key];
   int x = 0,t = 1;
 //rearrange the matrix according to the col_val
   while (t <= len_key)
   {
       for (int i = 0; i < len_key; i++)
       {
           int k = col_val[i];
           if (k == t)
           {
               for (int j = 0; j < no_rows + 1; j++)
               {
                   dec_mat[j][i]=cipher[x];
                   x++;
               }
               t++;
           }
       }
   }

   string message = "";
   for (int i = 0; i < no_rows + 1; i++)
   {
       for (int j = 0; j < len_key; j++)
       {
         //replacing the '_' with space
           if (dec_mat[i][j] == '_')
           {
               dec_mat[i][j] = ' ';
           }
           message += dec_mat[i][j];
       }
   }
   return message;
}

int main()
{
 //message
   string msg =
"pleasetransferonemilliondollarstomyswissbankaccountsixtwotwofour";
 //key
   string key = "megabuck";

   int len_key = key.length();
   int len_msg = msg.length();

   int val = 1,count = 0,ind;

   int col_val[len_key];
 //intializing col_val matrix with 0
   memset(col_val, 0, sizeof(col_val));
 //numbering the key alphabets according to its ASCII value
   while (count < len_key)
   {
       int min = 999;
       for (int i = 0; i < len_key; i++)
       {
           if ((min > int(key[i])) && (col_val[i] == 0))
           {
               min = int(key[i]);
               ind = i;
           }
       }
       col_val[ind] = val;
       count++;
       val++;
   }

   int no_rows = len_msg / len_key;
//encrypted text
   string cipher_text = " ";
   cipher_text = Encryption(no_rows, len_key, len_msg, msg, col_val);
   cout << "Encrypted Message : " << cipher_text << endl;
//decrypted text
   string original_msg = " ";
   original_msg = Decryption(no_rows, len_key, cipher_text,col_val);
   cout << "Decrypted Message : " << original_msg << endl;
}
```

Explain C,I,A, Authentication & Non-Repudiation.

1. **Confidentiality** (C): Confidentiality ensures that information is only accessible to those who are authorized to view it. This is often achieved through encryption, which scrambles data in a way that only authorized parties can decrypt and read it.
2. **Integrity** (I): Integrity ensures that information remains unaltered and accurate during transmission or storage. This is typically achieved through mechanisms such as checksums or digital signatures, which can detect if data has been tampered with.
3. **Availability** (A): Availability ensures that information and resources are accessible and usable when needed. This involves implementing measures to prevent or mitigate disruptions, such as network failures or cyberattacks, that could result in information becoming unavailable.
4. **Authentication**: Authentication verifies the identity of a user or system. It ensures that the entity requesting access to information or resources is who or what it claims to be. Authentication mechanisms include passwords, biometric authentication, and cryptographic techniques.
5. **Non-Repudiation**: Non-repudiation ensures that a user cannot deny the authenticity or integrity of a message or action they have performed. This is often achieved through the use of digital signatures, which provide a way to verify that a message was indeed sent by a particular sender and has not been altered since.

Explain the Interruption, Interception, Modification and Fabrication attack. Correlate the said attacks with C,I,A, Authentication and Non-Repudiation.

1. **Interruption**: Interruption attacks aim to disrupt the availability of information or services. This can be achieved by launching denial-of-service (DoS) attacks, which overwhelm a system with traffic, making it inaccessible to legitimate users. Interruption attacks primarily affect the availability (A) aspect of CIA, as they prevent users from accessing the resources they need.

2. **Interception**: Interception attacks involve unauthorized parties gaining access to sensitive information while it is being transmitted. This compromises the confidentiality (C) of the information. For example, in a man-in-the-middle (MitM) attack, an attacker intercepts and possibly modifies the communication between two parties without their knowledge.
3. **Modification**: Modification attacks involve altering data in transit or at rest to achieve a malicious objective. This undermines the integrity (I) of the data. For example, an attacker could modify the contents of a message to deceive the recipient or change the value of a financial transaction.
4. **Fabrication**: Fabrication attacks involve creating and inserting counterfeit data into a system to deceive users or gain unauthorized access. This can compromise both the integrity (I) and authenticity (Authentication) of the system. For example, an attacker could fabricate authentication credentials to gain unauthorized access to a system.

Correlation with C, I, A, Authentication, and Non-Repudiation:

- **Confidentiality** (C): Interception attacks directly target confidentiality by exposing sensitive information. Protection mechanisms such as encryption can help mitigate interception attacks.
- **Integrity** (I): Modification attacks undermine the integrity of data by altering it. Mechanisms such as digital signatures and checksums can help verify the integrity of data.
- **Availability** (A): Interruption attacks aim to disrupt availability by denying access to resources. Measures such as redundancy and DoS protection can help maintain availability.
- **Authentication**: Fabrication attacks can be prevented through strong authentication mechanisms that verify the identity of users or systems. This helps ensure that only legitimate entities can access resources.
- **Non-Repudiation**: Non-repudiation helps prevent denial of actions or events. For example, digital signatures can provide evidence that a message was sent by a specific sender, preventing them from later denying it.

E**xplain RSA algorithm.**

Compare in Tabular fashion, Confidentiality, Integrity, Authentication and Non repudiation for following security techniques
a. Classical Encryption/Decryption

b. Symmetric Encryption
c. Asymmetric Encryption
d. Hashing Technique
e. MAC technique
f. Digital Signature System

Explain any two of following with example

i. Playfair Cipher
ii. Hill Cipher
iii. Vigenère cipher
iv. One-Time Pad cipher

Expt 3: Substitution Cipher

```
#include <iostream>
#include <string>
#include <algorithm>
using namespace std;

// Function to encrypt a message using a substitution cipher with a
fixed numeric key
string encrypt(string message, int key) {
   string encrypted_text = message;
   for (char& c : encrypted_text) {
       if (isalpha(c)) {
           c = toupper(c);
           c = 'A' + ((c - 'A' + key) % 26);
       }
   }
   return encrypted_text;
}
// Function to decrypt a message using a substitution cipher with a
fixed numeric key
string decrypt(string message, int key) {
   string decrypted_text = message;
   for (char& c : decrypted_text) {
       if (isalpha(c)) {
           c = toupper(c);
           c = 'A' + ((c - 'A' - key + 26) % 26);
       }
   }
   return decrypted_text;
}
int main() {
   string message =
"pleasetransferonemilliondollarstomyswissbankaccountsixtwotwofour";
   int key = 3; // Fixed numeric key

   string encrypted_message = encrypt(message, key);
   cout << "Encrypted: " << encrypted_message << endl;

   string decrypted_message = decrypt(encrypted_message, key);
   cout << "Decrypted: " << decrypted_message << endl;
   return 0;
```

}
Use key
a. 3, or
b. 5, or
c. 7

d. for message: please transfer one million dollars to my swiss bank account six two two four

Encrypted:
SOHDVHWUDQVIHURQHPLOOLRQGROODUVWRPBVZLVVEDQNDFFRXQWVLAWZRWZRIRXU
Decrypted:
PLEASETRANSFERONEMILLIONDOLLARSTOMYSWISSBANKACCOUNTSIXTWOTWOFOUR

Explain C, I, A, Authentication and Non-Repudiation.

Explain the Interruption, Interception, Modification and Fabrication attack. Correlate the said attacks with C,I,A, Authentication and Non-Repudiation.

Explain the RSA algorithm in detail with examples.

Explain any two of following with example
i. Playfair Cipher
ii. Hill Cipher
iii. Vigenère cipher
iv. One-Time Pad cipher
v. Monoalphabetic cipher

Compare in Tabular fashion, Confidentiality, Integrity, Authentication and Non repudiation for following security techniques
a. Classical Encryption/Decryption
b. Symmetric Encryption
c. Asymmetric Encryption
d. Hashing Technique
e. MAC technique
f. Digital Signature System

Expt 4: Mobile Security Demo

What are WEP and WAP security techniques? Explain the details.

WEP (Wired Equivalent Privacy) and WPA (Wi-Fi Protected Access) are both security protocols used to secure wireless networks. However, they differ significantly in terms of their security features and effectiveness.

1. **WEP (Wired Equivalent Privacy):**
    - **Overview**: WEP was the first security protocol introduced for wireless networks and aimed to provide security comparable to that of a wired network. However, it has several significant vulnerabilities and is considered highly insecure.
    - **Security Features**:
        - WEP uses a shared key authentication mechanism, where all devices on the network use the same key to encrypt and decrypt data.
        - It uses the RC4 encryption algorithm with a 40-bit or 104-bit key.
    - **Vulnerabilities**:
        - WEP is vulnerable to several attacks, including the IV (Initialization Vector) attack, which can be used to recover the WEP key.
        - The small key size makes it susceptible to brute-force attacks, where an attacker tries all possible key combinations.
    - **Recommendation**: WEP is considered insecure and should not be used to secure wireless networks. It is recommended to upgrade to a more secure protocol like WPA or WPA2.
2. **WPA (Wi-Fi Protected Access):**
    - Overview: WPA was introduced as a replacement for WEP and addressed many of its security weaknesses. It provides stronger encryption and enhanced security features.
    - **Security Features**:
        - WPA uses a stronger encryption algorithm called TKIP (Temporal Key Integrity Protocol) to encrypt data.
        - It also introduced the concept of a passphrase, which is used to generate the encryption keys dynamically, making them more difficult to crack.
        - WPA also includes a mechanism called MIC (Message Integrity Check) to detect and prevent tampering with data packets.
    - **Improvements over WEP**:

    - WPA addresses the vulnerabilities present in WEP, such as the weak encryption algorithm and key management issues.
    - It provides stronger security measures to protect wireless networks from unauthorized access and attacks.
  - **Variants**: WPA has several variants, including WPA-PSK (Pre-Shared Key) and WPA-Enterprise, which differ in their authentication mechanisms.
  - **Recommendation**: WPA is considered more secure than WEP but has also been largely superseded by WPA2, which provides even stronger security features.

In summary, WEP is an outdated and insecure security protocol that should be avoided, while WPA provides stronger security features and is a better choice for securing wireless networks. However, WPA2 or WPA3 are recommended for the highest level of security.

What are the different wireless components & protocols used for Wi-Fi, Bluetooth Communications?

Wireless communication for Wi-Fi and Bluetooth involves various components and protocols that enable devices to connect and communicate wirelessly. Here's an overview of the components and protocols used for each:

Wi-Fi Components and Protocols:

1. **Access Point** (AP): A hardware device that allows Wi-Fi devices to connect to a wired network using wireless connections.
2. **Wi-Fi Router**: A device that performs the functions of an access point, router, and often includes a built-in switch for wired Ethernet devices.
3. **Wireless Network Interface Card (NIC):** A hardware component that enables devices to connect to wireless networks.
4. **Wireless Access Point (WAP):** A device that allows wireless communication devices to connect to a wired network using Wi-Fi.
5. **Wi-Fi Protocol Standards:**
    - **802.11a/b/g/n/ac/ax:** These are the various Wi-Fi standards that define the specifications for wireless networks, including data rates, frequencies, and protocols.
6. **Wi-Fi Protected Access (WPA)**: A security protocol designed to secure wireless networks, replacing the insecure WEP protocol.
7. **Wi-Fi Direct:** A protocol that allows devices to connect to each other without the need for an access point, enabling peer-to-peer communication.

Bluetooth Components and Protocols:

1. **Bluetooth Radio:** The physical layer of Bluetooth communication, responsible for transmitting and receiving data.
2. **Bluetooth Stack:** The software component of Bluetooth that implements the Bluetooth protocol stack, including the Radio Layer, Baseband Layer, L2CAP Layer, and higher layers.
3. **Bluetooth Profiles:** Profiles define how different Bluetooth devices communicate with each other based on their functionality, such as Hands-Free Profile (HFP) for hands-free calling and Advanced Audio Distribution Profile (A2DP) for streaming audio.
4. **Bluetooth Low Energy (BLE):** A power-efficient version of Bluetooth designed for low-power devices, such as fitness trackers and smartwatches.
5. **Bluetooth Protocol:** The Bluetooth protocol defines how devices communicate, establish connections, and manage data transfer.

Both Wi-Fi and Bluetooth are widely used wireless technologies, each with its own set of components and protocols that enable wireless communication between devices.

Write details about ISM band frequencies, BT standards & Wi-Fi standards.

ISM Band Frequencies:

The Industrial, Scientific, and Medical (ISM) bands are portions of the radio spectrum reserved internationally for industrial, scientific, and medical purposes. These bands are used for various applications, including radio-frequency heating, microwave ovens, and medical diathermy machines. They are also used for communication purposes, such as Bluetooth and Wi-Fi. The most commonly used ISM bands for communication are:

1. **2.4 GHz Band:** This band is widely used for Wi-Fi and Bluetooth communication. It offers good range and penetration through walls but can suffer from interference due to the number of devices using this frequency range.
2. **5 GHz Band:** This band is also used for Wi-Fi communication, particularly for newer standards like 802.11ac and 802.11ax (Wi-Fi 6 and Wi-Fi 6E). It offers higher data rates and less interference compared to the 2.4 GHz band but has shorter range and poorer wall penetration.

Bluetooth Standards:

Bluetooth is a wireless technology standard used for exchanging data over short distances using short-wavelength radio waves in the ISM band. The different Bluetooth standards include:

1. **Bluetooth 1.x:** The initial versions of Bluetooth with limited data rates and security features.
2. **Bluetooth 2.0 +** EDR: Enhanced Data Rate (EDR) introduced higher data rates for faster file transfers.

3. **Bluetooth 3.0 +** HS: Introduced High-Speed (HS) for even faster data transfers using Wi-Fi technology for data exchange.
4. **Bluetooth 4.x:** Introduced Bluetooth Low Energy (BLE) for low-power applications like wearables and IoT devices.
5. **Bluetooth 5.x:** The latest standard offering increased range, speed, and data broadcasting capacity.

Wi-Fi Standards:

Wi-Fi is a wireless networking technology that allows devices to connect to a local area network (LAN) wirelessly. The different Wi-Fi standards include:

1. **802.11b:** Introduced in 1999, offering data rates up to 11 Mbps in the 2.4 GHz band.
2. **802.11a:** Introduced in 1999, offering data rates up to 54 Mbps in the 5 GHz band.
3. **802.11g:** Introduced in 2003, offering data rates up to 54 Mbps in the 2.4 GHz band.
4. **802.11n:** Introduced in 2009, offering data rates up to 600 Mbps in the 2.4 GHz and 5 GHz bands.
5. **802.11ac:** Introduced in 2013, offering data rates up to 6.77 Gbps in the 5 GHz band.
6. **802.11ax (Wi-Fi 6):** Introduced in 2019, offering improved efficiency and performance in dense environments.

These standards define the specifications for wireless communication, including data rates, frequencies, and protocols, ensuring compatibility and interoperability between devices from different manufacturers.

Draw & Explain DES Standard

DES

The Data Encryption Standard (DES) is a symmetric-key block cipher algorithm that encrypts data in 64-bit blocks. It was adopted by the U.S. government in 1977 and became a widely used encryption standard. DES operates on 64-bit plaintext blocks and uses a 56-bit key, where 8 bits are used for parity and not for encryption. Here is a basic overview of the DES algorithm:

Key Generation:

The 56-bit key is used to generate 16 subkeys, one for each round of encryption and one for the final permutation.

The original 56-bit key is permuted to generate a 64-bit key, with one bit added to each 8-bit byte for parity.

Initial Permutation (IP):

The 64-bit plaintext block is permuted according to a fixed table.

16 Rounds of Encryption:

Expansion: The 32-bit half-block is expanded to 48 bits using a fixed table.

Key Mixing: The expanded half-block is XORed with the 48-bit subkey for the round.

Substitution: The result is divided into 8 6-bit blocks, each block is substituted using one of 8 S-boxes, producing 32 output bits.

Permutation: The 32 output bits are permuted according to a fixed table.

Swap: The two 32-bit halves of the block are swapped.

Final Permutation (FP):

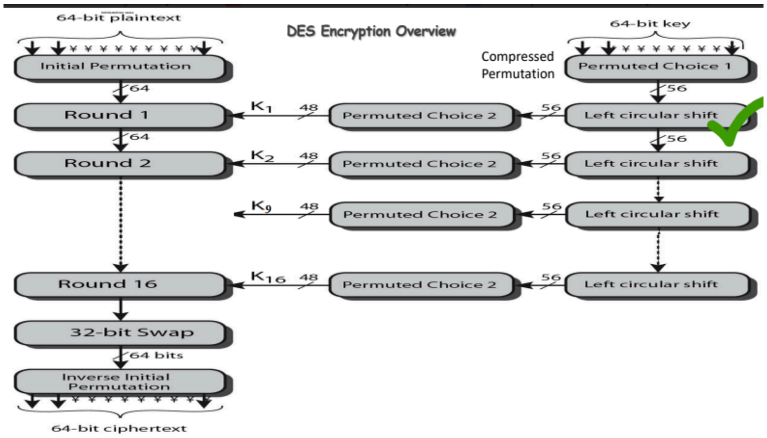
The final 64-bit ciphertext block is permuted according to a fixed table, which is the inverse of the initial permutation.

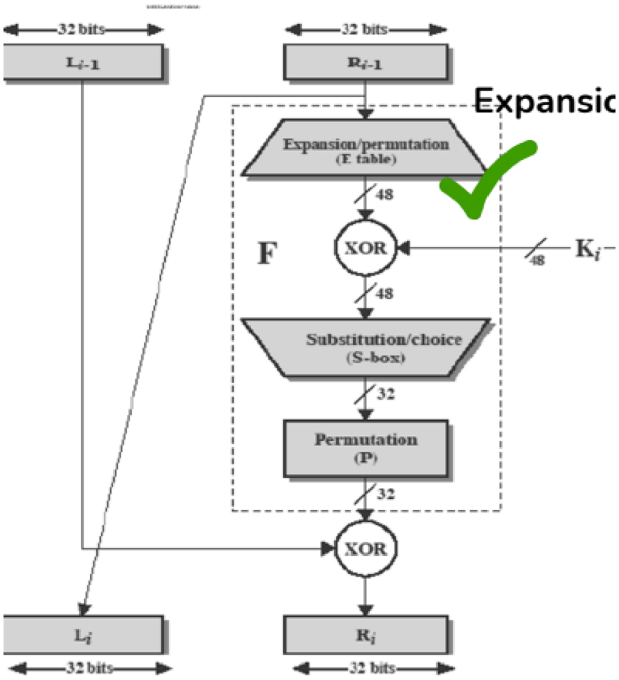
Decryption:

The decryption process is the same as encryption, except that the subkeys are used in reverse order.

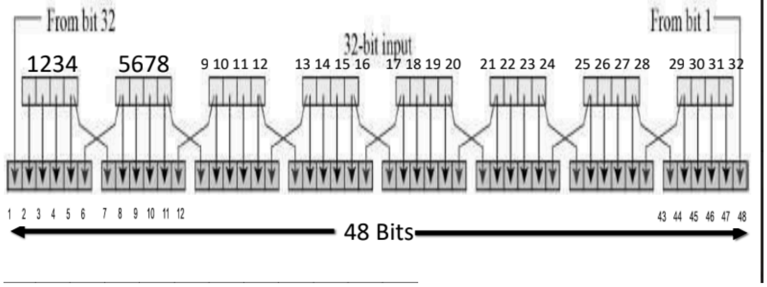
Key Points:

- DES has a 56-bit effective key size, as 8 of the 64 bits are used for parity and not for encryption.
- DES was the standard encryption algorithm for many years but is now considered insecure due to its small key size.
- Triple DES (3DES) is a variant of DES that applies DES three times with different keys, providing a higher level of security.





Expansion Permutation Box (32 to 48 Bits)



Explain Transport and tunnel mode in IPSec.

Transport Mode:

In IPSec, the transport mode is used to protect the payload of IP packets. It provides end-to-end security between two hosts, encrypting and authenticating the data portion of the IP packet while leaving the IP header unencrypted. The original IP packet's header is preserved, with only the payload (upper-layer protocol data) encrypted and authenticated.

Characteristics of Transport Mode:

- Only the data payload is encrypted and authenticated.
- The original IP header is preserved.
- Typically used for securing communication between two hosts.

Use Cases:

- Secure communication between two hosts in a network.
- Protecting the data payload while leaving the IP header visible.

Tunnel Mode:

In IPSec, the tunnel mode is used to protect entire IP packets, including both the IP header and the payload. It is often used to create virtual private networks (VPNs), where entire packets are encrypted and authenticated, then encapsulated within a new IP packet for transmission over the network. The original IP packet becomes the payload of the new, encrypted IP packet.

Characteristics of Tunnel Mode:

- Encrypts and authenticates the entire IP packet.
- Adds a new IP header for the encrypted packet.
- Used for creating secure tunnels between networks.

Use Cases:

- Establishing secure connections between two networks over the internet.
- Creating VPNs to secure traffic between remote sites and a central location.

Comparison:

- Transport mode is used for host-to-host communication, while tunnel mode is used for network-to-network or host-to-network communication.
- Transport mode encrypts only the data payload, while tunnel mode encrypts the entire packet.
- Tunnel mode adds a new IP header, while transport mode preserves the original IP header.
- Tunnel mode provides more security and privacy, especially for communication over untrusted networks, but may introduce additional overhead due to the encapsulation process

Expt 5: Steganography

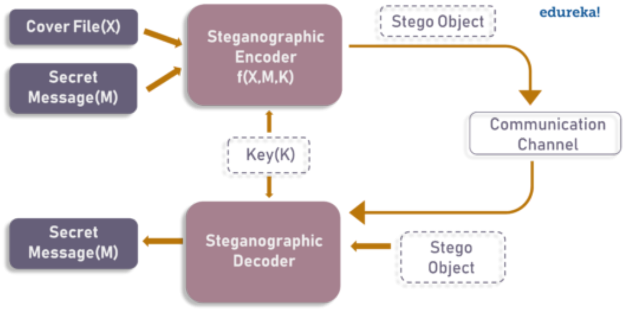
List features of Steganography.

Steganography is the practice of concealing a message, image, or file within another message, image, or file. It involves hiding the existence of the information to ensure privacy and confidentiality. Some key features of steganography include:

1. Concealment: Steganography hides the existence of secret information within seemingly innocuous carrier files or messages.
2. Security: It provides a layer of security by making the hidden information invisible to unintended recipients.
3. Invisibility: Steganographic techniques aim to make the embedded information undetectable to the human eye or automated systems.
4. Capacity: The capacity of steganographic techniques varies based on the carrier file and the chosen method, allowing for varying amounts of hidden data.
5. Resistance to Attacks: Steganography can be resistant to attacks that focus on the detection or alteration of encrypted data, as the existence of the hidden data is not apparent.
6. Multiple Carrier Types: Steganography can use various types of carrier files, such as images, audio files, video files, or text, making it versatile in different contexts.
7. Digital and Analog Formats: Steganography techniques can be applied to both digital and analog formats, providing flexibility in usage.
8. Authentication: Some steganographic methods include authentication mechanisms to ensure that the hidden information is not corrupted or tampered with.
9. Applications: Steganography finds applications in covert communication, digital watermarking, and copyright protection, among others.
10. Complementary to Encryption: Steganography can be used in conjunction with encryption for added security, as it hides the presence of encrypted data.

Draw and explain the block diagram of Steganography for Image as data.

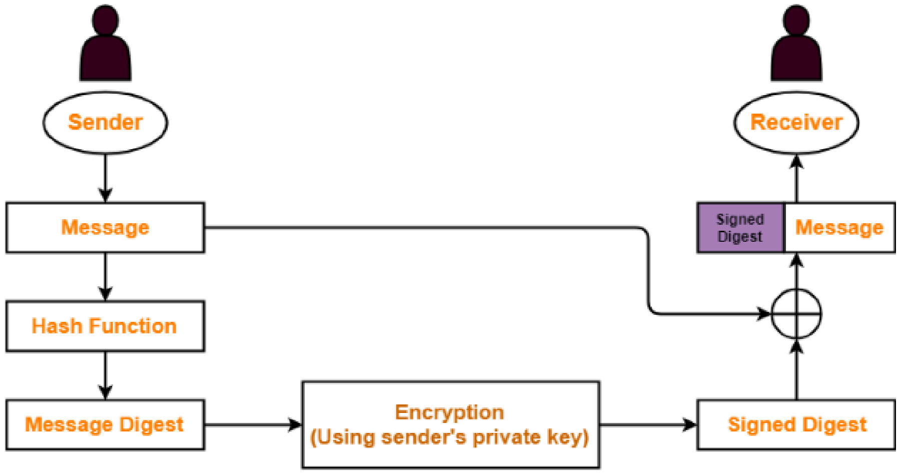
Steganography is the practice of concealing messages or information within other non-secret data. When it comes to images, steganography involves hiding information within the pixels of the image itself. Here's a block diagram illustrating the process:



1. **Input Image:** This is the original image that contains the hidden message.
2. **Message**: This is the secret message or data that needs to be hidden within the image.
3. **Embedding Algorithm:** This algorithm is responsible for embedding the message into the image. It takes the input image and the message as input and produces a steganographic image as output. The algorithm ensures that the embedded message is not easily detectable.
4. **Steganographic Image:** This is the output of the embedding algorithm. It looks like a normal image to the naked eye, but it contains the hidden message within its pixels.
5. **Steganalysis**: Steganalysis is the process of detecting the presence of hidden messages within images. It involves analyzing the steganographic image to see if it deviates from a normal image in a way that suggests the presence of hidden data.
6. **Output Image**: If the steganographic image passes through the steganalysis process undetected, it can be considered the output image. This image can be shared or transmitted like any other normal image, and the hidden message can be extracted using a decoding algorithm.

Draw and explain how the DES algorithm works in detail.
Explain the RSA algorithm in detail with examples.
Compare symmetric and asymmetric key cryptography.(Min 8 Points).

Draw and explain the following block diagrams
a. Digital Signature system.

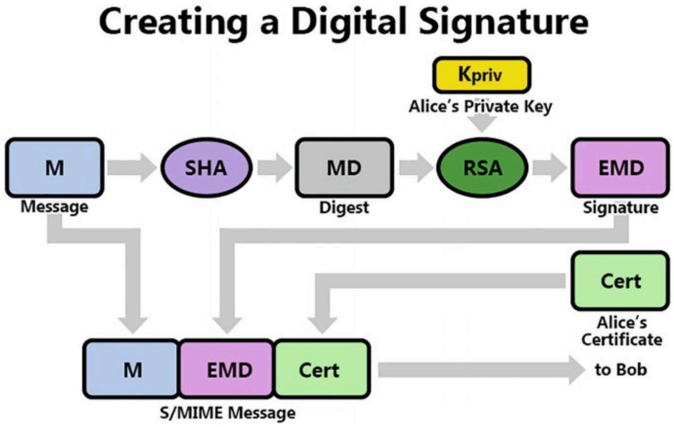


A digital signature system is used to verify the authenticity and integrity of digital messages or documents. Here's a block diagram illustrating the components of a digital signature system:

1. Message: This is the digital message or document that needs to be signed.
2. Hash Function: The message is passed through a hash function to generate a fixed-size hash value. The hash function ensures that even a slight change in the message will result in a significantly different hash value.
3. Private Key: The hash value is encrypted using the sender's private key. This step ensures that only the sender, who possesses the private key, can create the digital signature.
4. Digital Signature: The encrypted hash value, known as the digital signature, is appended to the message. This signature serves as a unique identifier for the message and is used to verify its authenticity and integrity.
5. Public Key: The sender's public key is used by the recipient to decrypt the digital signature. The public key is widely distributed and is used by anyone to verify the sender's signature.
6. Decryption: The recipient decrypts the digital signature using the sender's public key to obtain the original hash value.
7. Hash Function (again): The recipient passes the original message through the same hash function used by the sender to generate a hash value.
8. Comparison: The recipient compares the hash value obtained from the decrypted signature with the newly generated hash value. If the two values match, the message is considered authentic and has not been tampered with.

In summary, a digital signature system uses a combination of hashing, encryption, and public-key cryptography to ensure the authenticity and integrity of digital messages. The sender signs the message with their private key, and the recipient verifies the signature using the sender's public key, thereby establishing trust in the communication process.

b. End to End Email Communication system with Hashing, Digital signature and Digital Envelope processing blocks.



1. Sender's Side:
    - Message: The sender creates an email message.
    - Hashing: The message is hashed using a cryptographic hash function to create a fixed-size hash value.
    - Digital Signature: The hash value is encrypted using the sender's private key to create a digital signature.
    - Digital Envelope: The original message, along with the digital signature, is encrypted using a symmetric encryption algorithm with a randomly generated session key.
    - Email Transmission: The encrypted message is sent over the network to the recipient.
2. Receiver's Side:
    - Email Reception: The encrypted message is received by the recipient.
    - Digital Envelope Decryption: The recipient uses their private key to decrypt the session key used for encrypting the message.
    - Digital Signature Verification: The recipient decrypts the digital signature using the sender's public key to obtain the hash value.
    - Hashing (again): The recipient hashes the received message to generate a hash value.
    - Signature Verification: The recipient compares the hash value obtained from the decrypted signature with the newly generated hash value. If they match, the message is considered authentic and has not been tampered with.

- Decryption: Finally, the recipient decrypts the original message using the session key.

This process ensures the authenticity, integrity, and confidentiality of email messages exchanged betwee the sender and the recipient.

Expt 6: Configuring Firewall Security

What is a firewall and how does it function, also list different types of firewall at different layers, also list features

A firewall is a network security device or software that monitors and controls incoming and outgoing network traffic based on predetermined security rules. Its main function is to establish a barrier between a trusted internal network and untrusted external networks, such as the internet, to prevent unauthorized access and attacks.
Functionality:

1. **Packet Filtering**: Examines each packet of data passing through the network and blocks or allows it based on predefined rules.

2. **Stateful Inspection:** Keeps track of the state of active connections and allows only legitimate traffic based on the state table.

3. **Proxy Service:** Acts as an intermediary between clients and servers, forwarding requests on behalf of the clients and returning responses from the servers.

4. **Network Address Translation (NAT):** Translates private IP addresses to public IP addresses and vice versa, allowing multiple devices in a network to share a single public IP address.

5. **Virtual Private Network (VPN):** Provides secure access to a private network over a public network, such as the internet, by encrypting data traffic.

Types of Firewalls:

1. **Packet Filtering Firewall (Network Layer)**: Examines packets and filters them based on IP addresses, ports, and protocols. Stateless and stateful packet filtering firewalls are examples.

2. **Application-Level Gateway (Application Layer):** Acts as a proxy server for specific applications, such as FTP or HTTP, providing more detailed inspection and control.

3. **Circuit-Level Gateway (Transport Layer):** Works at the session layer of the OSI model and monitors TCP handshakes, allowing or denying connections based on predefined rules.

4. **Stateful Multilayer Inspection Firewall (Network to Application Layer):** Combines the functionality of packet filtering, stateful inspection, and application layer inspection for comprehensive security.

5. **Next-Generation Firewall (NGFW):** Integrates advanced features, such as intrusion detection and prevention, deep packet inspection, and application awareness, for enhanced security.

6. **Hardware vs. Software Firewalls:** Hardware firewalls are physical devices installed between the network and the internet, while software firewalls are software programs installed on individual devices or servers.

Features:

1. **Access Control:** Controls access to the network based on predefined rules.

2. **Logging and Monitoring:** Keeps detailed logs of network traffic and security events for analysis and auditing.

3. **Intrusion Detection and Prevention:** Detects and prevents malicious activities, such as hacking attempts and malware infections.
4. **Virtual Private Network (VPN):** Provides secure remote access to the network for remote users or branch offices.

5. **User Authentication:** Requires users to authenticate before accessing the network, enhancing security.

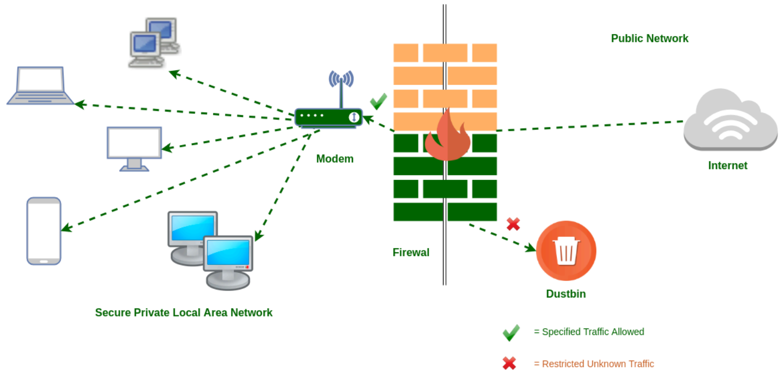
6. **Content Filtering**: Filters web content to block access to malicious or inappropriate websites.

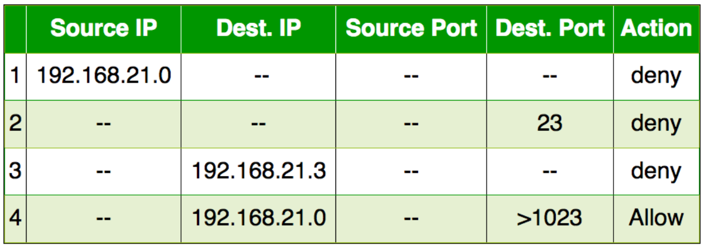
7. **High Availability:** Ensures continuous network availability even in the event of hardware or software failures.

Draw and explain the following. A)Packet filtering firewall B)Application Layer Firewall C) Circuit Level Gateway firewall

A) Packet Filtering Firewall:

A packet filtering firewall operates at the network layer (Layer 3) of the OSI model and examines packets of data as they pass through the firewall. It filters packets based on criteria such as source and destination IP addresses, source and destination port numbers, and the protocol used (e.g., TCP, UDP, ICMP). Here's a diagram illustrating how a packet filtering firewall works:


Public Network
Modem
Internet
Firewall
Dustbin
Secure Private Local Area Network
✓ = Specified Traffic Allowed
✗ = Restricted Unknown Traffic



	Source IP	Dest. IP	Source Port	Dest. Port	Action
1	192.168.21.0	--	--	--	deny
2	--	--	--	23	deny
3	--	192.168.21.3	--	--	deny
4	--	192.168.21.0	--	>1023	Allow

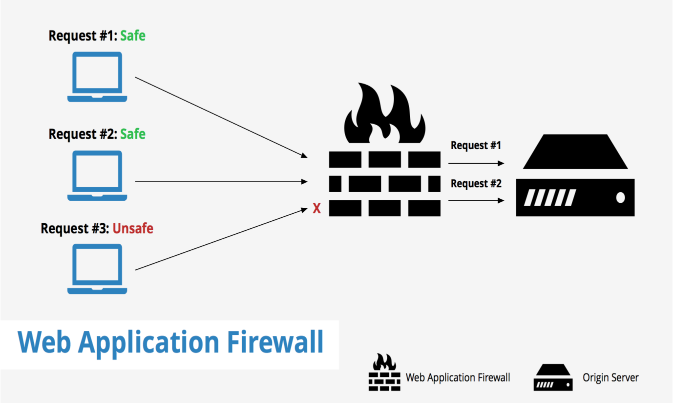
Sample Packet Filter Firewall Rule

Explanation:

1. Incoming Packets: Data packets arriving at the firewall from the network.
2. Packet Filtering Firewall: The firewall examines each packet based on predefined rules (e.g., allow or deny based on IP address, port number, protocol).
3. Accepted Packets: Packets that meet the criteria are allowed to pass through the firewall to the destination.
4. Rejected Packets: Packets that do not meet the criteria are dropped or rejected by the firewall.

B) Application Layer Firewall:

An application layer firewall, also known as a proxy firewall, operates at the application layer (Layer 7) of the OSI model. It acts as an intermediary between client applications and servers, providing additional security by filtering and inspecting traffic at the application level. Here's a diagram illustrating how an application layer firewall works:



Explanation:

1. **Client Application:** The application running on the client device that initiates a connection.
2. **Application Layer Firewall:** The firewall intercepts requests from the client application and forwards them to the server application. It inspects and filters the traffic based on application-specific rules.
3. **Server Application:** The application running on the server that responds to requests from the client application.

C) Circuit Level Gateway Firewall:

A circuit-level gateway firewall operates at the session layer (Layer 5) of the OSI model. It does not inspect the contents of packets but instead verifies the validity of the session before allowing traffic to pass through.
alt: This works as the Sessions layer of the OSI Model's . This allows for the simultaneous setup of two Transmission Control Protocol (TCP) connections. It can effortlessly allow data packets to flow without using quite a lot of computing power. These firewalls are ineffective because they do not inspect data packets; if malware is found in a data packet, they will permit it to pass provided that TCP connections are established properly Here's a diagram illustrating how a circuit-level gateway firewall works:

Explanation:

1. **Incoming Connection Request:** A request from a client to establish a connection with a server.
2. **Circuit Level Gateway Firewall:** The firewall verifies the validity of the connection request (e.g., based on TCP handshake) without inspecting the contents of the packets.
3. **Validated Connection**: If the connection is valid, the firewall allows traffic to pass through to the destination.
4. **Rejected Connection:** If the connection is not valid, the firewall rejects the connection request.

Compare Firewall & Antivirus.
Firewall vs. Antivirus:

1. **Purpose:**
   - **Firewall**: Monitors and controls incoming and outgoing network traffic based on predetermined security rules to protect against unauthorized access.
   - **Antivirus**: Detects, prevents, and removes malicious software (malware) such as viruses, worms, and trojans from a computer or network.
2. **Scope**:
   - **Firewall**: Protects the network by filtering traffic at the network level, typically at the boundary between a private network and the internet.
   - **Antivirus**: Protects individual devices (e.g., computers, smartphones) by scanning files and programs for malware.
3. **Functionality**:
   - **Firewall**: Controls traffic based on predefined rules, such as allowing or blocking specific IP addresses, ports, or protocols.
   - **Antivirus**: Scans files and programs for known malware signatures or suspicious behavior, and quarantines or removes infected files.
4. **Prevention vs**. **Detection**:
   - **Firewall**: Primarily focuses on preventing unauthorized access and attacks from reaching the network.
   - **Antivirus**: Primarily focuses on detecting and removing malware that has already infected a device.
5. **Deployment**:
   - **Firewall**: Can be deployed as hardware devices or software programs on individual devices or network infrastructure.
   - **Antivirus**: Typically installed as software programs on individual devices, although some network-level antivirus solutions exist.

6. **Effectiveness**:
    - **Firewall**: Effective in preventing unauthorized access and controlling network traffic, but may not detect or remove malware.
    - **Antivirus**: Effective in detecting and removing malware, but may not prevent all types of attacks or unauthorized access.
7. **Complementary Security Measures**:
    - **Firewall**: Often used in conjunction with antivirus software and other security measures to provide comprehensive network security.
    - **Antivirus**: Often used in conjunction with firewalls, anti-malware software, and other security measures to provide comprehensive device security.

Compare IDS and IPS in detail.

Intrusion Detection System (IDS) vs. Intrusion Prevention System (IPS):

1. **Purpose:**
    - IDS: Monitors network traffic or system activity for suspicious patterns or anomalies that may indicate unauthorized access or attacks.
    - IPS: Similar to IDS but with the ability to take automated action to block or prevent detected threats in real-time.
2. **Detection vs. Prevention:**
    - IDS: Focuses on detecting and alerting security personnel about potential threats or attacks.
    - IPS: Goes a step further by actively preventing detected threats from reaching their target, either by blocking or dropping malicious packets or by reconfiguring firewalls or routers.
3. **Response Mechanism:**
    - IDS: Passive monitoring system that does not actively interfere with network traffic.
    - IPS: Active system that can modify or block traffic based on predefined rules or policies.
4. **Deployment Location:**
    - IDS: Can be deployed at various points within a network, such as at the perimeter, on specific servers, or within the network infrastructure.
    - IPS: Typically deployed at the network perimeter or within critical network segments to provide immediate threat prevention.

5. **Processing and Analysis:**
    - IDS: Analyzes traffic or logs to detect patterns or anomalies, often using signature-based or anomaly-based detection methods.
    - IPS: Similar to IDS but with the ability to take immediate action based on detection results, often using signature-based, anomaly-based, or behavior-based detection methods.
6. **Alerting:**
    - IDS: Generates alerts for security personnel to investigate and respond to potential threats.
    - IPS: Generates alerts and can also take automated actions to block or prevent threats without human intervention.
7. **Scalability:**
    - IDS: Can be scaled to monitor large networks but may require additional resources as the network size increases.
    - IPS: Similar scalability to IDS but with potentially higher resource requirements due to the need for real-time response capabilities.
8. **False Positives/Negatives:**
    - IDS: May generate false positives (incorrectly identifying normal traffic as malicious) or false negatives (failing to detect actual attacks).
    - IPS: Similar potential for false positives and false negatives but with the added risk of blocking legitimate traffic if not configured properly.

Draw and explain how DES algorithm works in detail.

Explain the RSA algorithm in detail with example.

Draw and explain the following block diagrams
a.Digital Signature system.
b. End to End Email Communication system with Hashing, Digital signature and Digital Envelope processing blocks.

Expt 7: Browser Security

What are different types of cookies?

1. **Session Cookies:** These are temporary cookies that are erased from your device when you close your web browser. They are used to track your navigation on a website during a single session.

2. **Persistent Cookies:** These cookies remain on your device even after you close your browser or restart your computer. They are used to remember your preferences or actions across multiple sites and sessions.

3. **Secure Cookies:** These cookies are transmitted over HTTPS and are only sent to the server if the browser encrypts the data during the transmission. They are more secure than regular HTTP cookies.

4. **HTTP-Only Cookies:** These cookies are only accessible to the server and are not accessible to JavaScript, which helps prevent cross-site scripting (XSS) attacks.

5. **Third-party Cookies:** These cookies are set by domains other than the one you are visiting. They are commonly used for tracking and advertising purposes.

6. **Zombie Cookies:** Also known as persistent cookies, these cookies are automatically recreated after being deleted, making them difficult to remove permanently.

7. **Super Cookies:** These cookies use various storage mechanisms, such as Flash or HTML5, to store user data, making them more persistent and difficult to delete.

8. **Same-Site Cookies**: These cookies are designed to mitigate the risk of cross-origin information leakage. They allow servers to mitigate the risk of CSRF and information leakage attacks by asserting that a particular cookie should only be sent with requests initiated from the same registrable domain.

What are the advantages and drawbacks of cookies?
Advantages of Cookies:

1. Customized User Experience: Cookies can store user preferences and settings, allowing websites to personalize the user experience based on past interactions.
2. Session Management: Cookies are commonly used for session management, enabling websites to remember users and their actions as they navigate the site.
3. Tracking and Analytics: Cookies are essential for tracking user behavior and collecting analytics data, which helps website owners understand how users interact with their site.
4. E-commerce Functionality: Cookies are often used in e-commerce websites to remember items in a shopping cart or user login details.
5. Targeted Advertising: Cookies enable advertisers to deliver more targeted ads to users based on their browsing behavior and interests.
6. Authentication: Cookies can be used for authentication, allowing users to stay logged in to a website without having to re-enter their credentials.

Drawbacks of Cookies:

1. Privacy Concerns: Cookies can track user behavior across different websites, raising privacy concerns about the collection and use of personal data.
2. Security Risks: Cookies can be exploited by malicious actors for cross-site scripting (XSS) attacks or to steal sensitive information.
3. Limited Storage: Cookies have size limitations, which can restrict the amount of data that can be stored, impacting their usefulness for certain applications.
4. Browser Compatibility: Not all browsers handle cookies in the same way, which can lead to inconsistencies in how websites function across different browsers.
5. User Control: While users can delete cookies or disable them, this can affect their browsing experience and may require them to re-enter preferences or login details.
6. Misuse for Tracking: Cookies can be used for tracking users without their consent, leading to concerns about online privacy and user tracking practices.

Explain “Session Hijacking”

Session hijacking refers to the malicious act of taking control of a user’s web session. A session, in the context of web browsing, is a series of interactions between two communication endpoints, sharing a unique session token to ensure continuity and security.

It’s a form of attack where a bad actor steals or manipulates the session token to gain unauthorized access to information or services. The hijacking process typically begins when an attacker intercepts this token, which can be likened to a secret handshake between the user and the website. Once in possession of this token, the attacker gains the ability to masquerade as the legitimate user, potentially causing havoc. The methods of interception can vary, ranging from network eavesdropping to sophisticated phishing attacks.

1. **Cookie Generation:** When a user logs into a website, the server generates a session cookie and sends it to the user's browser. This cookie contains a unique identifier that associates the user with their session on the server.

2. **Cookie Transmission:** The browser sends the session cookie along with every subsequent request to the website, allowing the server to identify the user and maintain their session.

3. **Cookie Interception:** An attacker intercepts the session cookie, usually through means like packet sniffing on an unsecured network, cross-site scripting (XSS) attacks, or by gaining access to the victim's device.

   Among the arsenal of techniques at a hijacker's disposal, certain methods stand out due to their prevalence and effectiveness.

   - **Session sniffing:** This technique involves monitoring network traffic to capture valid session tokens.
   - **Cross-site scripting (XSS):** Attackers inject malicious scripts into web pages, which then allow them to steal session cookies from unsuspecting users.
   - **Session fixation:** Here, an attacker forces a user to use a specific session ID, which the attacker has already obtained, to compromise the session.

4. **Cookie Usage**: The attacker uses the stolen session cookie to impersonate the user. By inserting the stolen cookie into their own browser or sending it with their requests, the attacker gains access to the victim's session, effectively hijacking it.

5. **Unauthorized Actions**: With the hijacked session, the attacker can perform various unauthorized actions on behalf of the user, such as accessing sensitive information, changing account settings, or making purchases.

Fortunately, there are tools and techniques designed to detect session hijacking. Intrusion detection systems (IDS), for example, can monitor network traffic for signs of session token misuse. Additionally, anomaly-based detection mechanisms can alert administrators to irregular session activities that may indicate hijacking attempts.

Tools & Techniques for detection

Implementing robust detection systems is only part of the solution. Continuous monitoring and regular security assessments are essential for identifying and addressing vulnerabilities before they're exploited. Security teams must stay vigilant and look for the following:

- Unexpected changes in session durations or locations
- Multiple concurrent sessions from different IP addresses
- Unusual patterns of session activity that could indicate scripted attacks

By combining advanced detection tools with proactive monitoring, organizations can reduce the risk of session hijacking.

To mitigate session hijacking attacks, websites can implement the following measures:

- Use HTTPS: Encrypting the communication between the user's browser and the server helps prevent attackers from intercepting the session cookie.
- Secure Cookies: Set the Secure and HttpOnly flags on cookies to prevent them from being accessed by JavaScript and transmitted over unencrypted connections.
- Session Expiry: Set a short session timeout and require users to reauthenticate after a certain period of inactivity.
- Token-based Authentication: Use tokens instead of session cookies for authentication, which can be validated on each request and are less susceptible to hijacking.
- Monitoring: Monitor for unusual activity, such as multiple logins from different locations or devices, which could indicate a session hijacking attempt.

Draw and explain the following block diagrams
a. Digital Signature system.
b. End to End Email Communication system with Hashing, Digital signature and Digital Envelope processing blocks.

Explain RSA algorithm in detail with example.

Explain TLS and S/MIME used in Email Security
In detail

TLS (Transport Layer Security) and S/MIME (Secure/Multipurpose Internet Mail Extensions) are cryptographic protocols used to enhance the security of email communications.

TLS (Transport Layer Security):

1. **Purpose**: TLS is a protocol that ensures privacy and data integrity between two communicating applications. It is commonly used to secure connections between email servers (SMTP) and clients (POP3, IMAP).
2. **Encryption**: TLS encrypts the data exchanged between the email client and server, preventing unauthorized access to the contents of the emails.
3. **Authentication**: TLS also provides authentication mechanisms, allowing servers and clients to verify each other's identities, ensuring that the communication is secure and not intercepted by attackers.
4. **Implementation**: To use TLS, both the email server and client must support TLS. The email server needs to have a TLS certificate, and the client needs to be configured to use TLS for the connection.
5. **Benefits**: TLS protects email communications from eavesdropping and tampering, ensuring that sensitive information remains confidential and secure during transmission.

S/MIME (Secure/Multipurpose Internet Mail Extensions):

1. **Purpose**: S/MIME is a standard for secure email messaging that provides encryption, authentication, message integrity, and non-repudiation features.
2. **Encryption**: S/MIME uses public-key cryptography to encrypt email messages, ensuring that only the intended recipient can decrypt and read the message.
3. **Authentication**: S/MIME provides a mechanism for digitally signing email messages, allowing recipients to verify the sender's identity and ensure that the message has not been altered in transit.
4. **Implementation**: To use S/MIME, both the sender and recipient must have a digital certificate issued by a trusted Certificate Authority (CA). The email client needs to be configured to use S/MIME for signing and encrypting emails.
5. **Benefits**: S/MIME ensures the confidentiality, integrity, and authenticity of email messages, making it a secure method for exchanging sensitive information over email.

In summary, TLS and S/MIME are essential security protocols used to protect email communications from interception, tampering, and unauthorized access. They provide encryption, authentication, and integrity features to ensure the security of email messages exchanged between users.

Compare PGP Vs S/MIME.

PGP (Pretty Good Privacy) and S/MIME (Secure/Multipurpose Internet Mail Extensions) are both cryptographic protocols used to secure email communications, but they have some differences in terms of implementation, security features, and popularity:

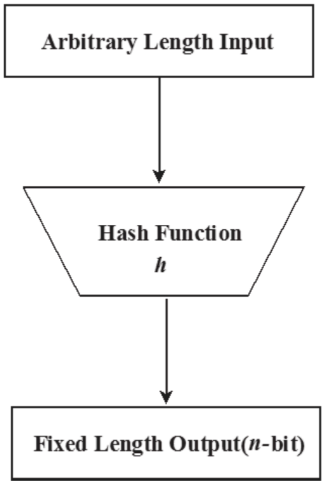
1. **Popularity and Adoption:**
    - **PGP**: PGP was developed in the early 1990s and gained popularity among privacy advocates and individuals concerned about email security. It is widely used in both personal and professional settings.
    - **S/MIME**: S/MIME is a standard developed by the IETF (Internet Engineering Task Force) and is widely supported by email clients and servers. It is commonly used in corporate environments and by organizations that require secure email communication.
2. **Key Management:**
    - **PGP**: PGP uses a web of trust model for key management, where users can sign each other's keys to establish trust. This decentralized approach allows users to verify the authenticity of keys.
    - **S/MIME:** S/MIME relies on a hierarchical trust model, where a trusted Certificate Authority (CA) issues digital certificates to users. This centralized approach simplifies key management but requires users to trust the CA.
3. **Encryption and Authentication:**
    - **PGP**: PGP provides both encryption and digital signatures for email messages. It uses the RSA algorithm for encryption and digital signatures.
    - **S/MIME:** S/MIME also provides encryption and digital signatures, but it supports multiple encryption algorithms, including RSA, DSA, and ECC (Elliptic Curve Cryptography).
4. **Compatibility:**
    - PGP: PGP is not natively supported by all email clients, so users may need to install third-party software or plugins to use PGP encryption.
    - S/MIME: S/MIME is supported by many email clients and servers, making it easier to use without additional software or plugins.
5. **Interoperability**:
    - **PGP**: PGP is a standalone protocol that can be used with any email service or client that supports PGP encryption.
    - **S/MIME:** S/MIME is an extension of the MIME standard and is designed to work seamlessly with existing email infrastructure.

In conclusion, both PGP and S/MIME are effective ways to secure email communications, but they differ in terms of key management, encryption algorithms, and compatibility. PGP is more popular among privacy-conscious individuals, while S/MIME is widely used in corporate environments due to its ease of implementation and compatibility with existing email systems.

Expt 8: Hash Function

What is a Hash Function?

A hash function is a mathematical function that converts an input (or 'message') into a fixed-size string of bytes, typically a sequence of numbers and letters. The output, known as the hash value or hash code, is unique to the input data and serves as a digital fingerprint of the input. Hash functions are widely used in computer science and cryptography for various purposes, including data integrity verification, digital signatures, and password storage.



How a Hash Function Works:

1. **Input Data**: The hash function takes input data of any length, such as a file, message, or password.

2. **Hashing Process**: The hash function processes the input data using a specific algorithm to generate a fixed-size hash value. The algorithm applies a series of mathematical operations to the input data, resulting in a unique hash value.

3. **Fixed Size Output**: The hash value is typically a fixed size, regardless of the size of the input data. For example, the SHA-256 hash function produces a 256-bit hash value.

4. **Deterministic**: The hash function is deterministic, meaning that the same input data will always produce the same hash value. Even a small change in the input data will result in a significantly different hash value.

5. **Uniformity**: A good hash function produces hash values that are uniformly distributed, meaning that each possible hash value is equally likely to occur.

6. **Irreversibility**: Hash functions are designed to be one-way functions, meaning that it is computationally infeasible to reverse the process and recover the original input data from the hash value. This property is crucial for password hashing and digital signatures.
7. **Collision Resistance:** A good hash function should also be collision-resistant, meaning that it is computationally infeasible to find two different inputs that produce the same hash value. This property ensures the uniqueness of the hash value for a given input.

List different applications of SHA2.

SHA-2 (Secure Hash Algorithm 2) is a family of cryptographic hash functions designed by the National Security Agency (NSA) in the United States. It is part of the larger SHA family of hash functions and is widely used in various security applications, including digital signatures, certificate authorities, and message integrity checks.
The SHA-2 family consists of six hash functions, each with a different bit length for the hash output: SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, and SHA-512/256. The numbers in the names indicate the length of the hash output in bits. For example, SHA-256 produces a 256-bit hash value.

SHA-2 (Secure Hash Algorithm 2) is widely used in various applications that require secure hashing and data integrity. Some common applications of SHA-2 include:

1. **Digital Signatures:** SHA-2 is used in conjunction with asymmetric encryption algorithms (such as RSA) to create digital signatures, providing authentication and integrity for digital documents and transactions.

2. **Certificate Authorities (CAs)**: SHA-2 is used in digital certificates to ensure the integrity and authenticity of the certificate data, helping to establish trust on the Internet.

3. **Message Integrity:** SHA-2 is used to verify the integrity of messages transmitted over insecure channels, ensuring that the message has not been altered in transit.

4. **Data Integrity Checks:** SHA-2 is used to verify the integrity of data stored or transmitted over networks, ensuring that the data has not been tampered with.

5. **Password Storage**: SHA-2 is used to hash passwords for storage in databases, ensuring that passwords are not stored in plain text and providing a secure way to verify user passwords during authentication.

6. **Blockchain Technology:** SHA-2 is used in blockchain technology to create a secure, tamper-proof record of transactions, ensuring the integrity and immutability of the blockchain.

7. **Secure Software Updates:** SHA-2 is used to verify the integrity of software updates, ensuring that updates have not been tampered with or altered before installation.

8. **Secure Hashing of Files:** SHA-2 is used to generate hash values for files, allowing users to verify the integrity of files downloaded from the Internet or stored on local systems.

Explain following applications of Hash functions in detail
a. Protection to password storage

Hash functions are commonly used to protect passwords stored in databases. When a user creates or updates their password, the password is passed through a hash function, and the resulting hash value is stored in the database instead of the actual password. This process provides several benefits:

1. **Security**: Storing passwords in plaintext is risky because if the database is compromised, attackers can easily access all the passwords. By using hash functions, passwords are converted into irreversible hash values, making it difficult for attackers to retrieve the original passwords.
2. **Data Confidentiality**: Hashing passwords ensures that even system administrators cannot access the plaintext passwords. This protects user privacy and confidentiality.
3. **Collision Resistance**: A good hash function has a low probability of producing the same hash value for two different inputs (collision). This ensures that even if two users have the same password, their hash values will be different.
4. **Salting**: To further enhance security, a random value called a "salt" can be added to the password before hashing. The salt is stored alongside the hash value and is unique for each user. Salting prevents attackers from using precomputed hash tables (rainbow tables) to crack passwords.
5. **Performance**: Hashing is computationally efficient, making it suitable for use in high-traffic systems.

b. Data Integrity Check:

Hash functions are used to verify the integrity of data during transmission or storage. When data is transmitted over a network or stored on a disk, a hash value of the data is calculated and sent/stored along with the data. When the data is received or accessed, the hash value is recalculated, and if it matches the original hash value, it indicates that the data has not been tampered with. This process provides the following benefits:

1. **Data Integrity**: Hashing ensures that data has not been altered or corrupted during transmission or storage. Any change in the data will result in a different hash value.
2. **Checksums**: Hash functions are used to generate checksums for data blocks, which are then used to detect errors in data transmission or storage. If the checksum calculated at the receiving end does not match the original checksum, it indicates that the data has been corrupted.
3. **File Integrity Checking**: Hash functions are used to verify the integrity of files downloaded from the Internet. The hash value provided by the source can be used to verify that the downloaded file is identical to the original file.
4. **Digital Signatures**: Hash functions are used in digital signatures to ensure the authenticity and integrity of digital documents. The hash value of a document is encrypted with the sender's private key, and the recipient can verify the signature using the sender's public key.

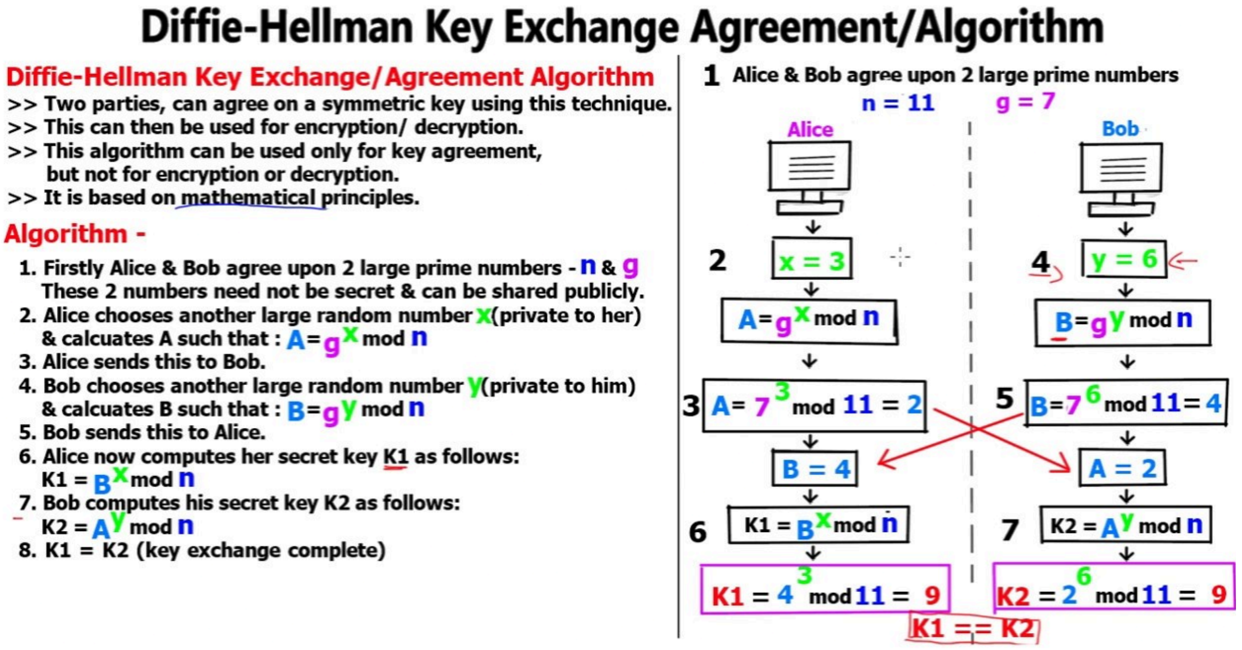
Explain Transport and tunnel mode in IPSec.

Explain TLS and S/MIME used in Email Security. Compare PGP Vs S/MIME.

Explain RSA algorithm in detail with example.

Expt 9: Diffie Hellman Key Exchange

Explain Diffie Hellman Key exchange Agreement.



Diffie-Hellman Key Exchange Agreement/Algorithm

Diffie-Hellman Key Exchange/Agreement Algorithm

>> Two parties, can agree on a symmetric key using this technique.
>> This can then be used for encryption/ decryption.
>> This algorithm can be used only for key agreement, but not for encryption or decryption.
>> It is based on mathematical principles.

Algorithm -

1. Firstly Alice & Bob agree upon 2 large prime numbers - n & g These 2 numbers need not be secret & can be shared publicly.
2. Alice chooses another large random number x(private to her) & calcuates A such that : $A = g^x \bmod n$
3. Alice sends this to Bob.
4. Bob chooses another large random number y(private to him) & calcuates B such that : $B = g^y \bmod n$
5. Bob sends this to Alice.
6. Alice now computes her secret key K1 as follows: $K1 = B^x \bmod n$
7. Bob computes his secret key K2 as follows: $K2 = A^y \bmod n$
8. K1 = K2 (key exchange complete)

1 Alice & Bob agree upon 2 large prime numbers
$n = 11$ $g = 7$
Alice
Bob
2 $x = 3$
$A = g^x \bmod n$
3 $A = 7^3 \bmod 11 = 2$
$B = 4$
6 $K1 = B^x \bmod n$
$K1 = 4^3 \bmod 11 = 9$
4 $y = 6$
$B = g^y \bmod n$
5 $B = 7^6 \bmod 11 = 4$
$A = 2$
7 $K2 = A^y \bmod n$
$K2 = 2^6 \bmod 11 = 9$
K1 == K2

Draw and explain how the DES algorithm works in detail.

Explain the RSA algorithm in detail with examples.

What is Hash function? Explain How it works briefly? List the different applications of SHA2.

Compare symmetric and asymmetric key cryptography.(Min 8 Points).

Explain any two of following with example
i. Playfair Cipher
ii. Hill Cipher
iii. Vigenère cipher
iv. One-Time Pad cipher
v. Monoalphabetic cipher

Write LONG note on the following
a. Transport and tunnel mode in IPSec.
b. S/MIME for Email security.
c. TLS explanation with Suitable example

a. Transport and Tunnel Mode in IPSec:

IPSec (Internet Protocol Security) is a protocol suite used to secure Internet Protocol (IP) communications by authenticating and encrypting each IP packet in a data stream. IPSec can operate in two modes: Transport mode and Tunnel mode.

1. **Transport Mode:**
   - **Purpose**: Transport mode is used to encrypt the payload (data) of the IP packet while leaving the IP header unencrypted. It is typically used for end-to-end communications between two hosts.
   - **Authentication**: In transport mode, the IP payload is encrypted and authenticated, ensuring that the data is not tampered with during transmission.
   - **Header Changes:** The IP header is not modified in transport mode, except for the addition of IPSec headers for authentication and encryption.
   - **Suitability**: Transport mode is suitable for securing communications between individual hosts or between a host and a gateway.
2. **Tunnel Mode**:
   - **Purpose**: Tunnel mode is used to encrypt the entire IP packet, including the IP header and payload. It is often used to create a secure tunnel between two networks, such as between two routers or a router and a gateway.
   - **Authentication**: In tunnel mode, the entire IP packet is encrypted and authenticated, providing a higher level of security than transport mode.
   - **Header Changes**: In tunnel mode, a new IP header is added to the encrypted packet, with the original packet encapsulated within. This allows the packet to be transmitted securely over an insecure network.
   - **Suitability**: Tunnel mode is suitable for securing communications between networks or for establishing virtual private networks (VPNs).

b. S/MIME for Email Security:

S/MIME (Secure/Multipurpose Internet Mail Extensions) is a standard for securing email messages using public key cryptography. It provides authentication, message integrity, and confidentiality for email communication.

1. **Authentication**: S/MIME allows the sender of an email to digitally sign the message using their private key. The recipient can then verify the signature using the sender's public key, ensuring that the message has not been tampered with and was indeed sent by the claimed sender.
2. **Message Integrity**: S/MIME also provides message integrity by generating a message digest (hash) of the email content and encrypting it with the sender's private key. The recipient can verify the integrity of the message by decrypting the digest with the sender's public key and comparing it to a newly calculated digest of the received email.
3. **Confidentiality**: S/MIME supports message encryption, where the email content is encrypted using the recipient's public key. Only the recipient, who possesses the corresponding private key, can decrypt and read the message.

c. TLS Explanation with Suitable Example:

TLS (Transport Layer Security) is a protocol used to secure communications over a computer network. It ensures privacy and data integrity between communicating applications. An example of TLS usage is securing HTTP traffic to create HTTPS, which is widely used for secure web browsing.

1. **Handshake Protocol**: The TLS handshake protocol allows the client and server to authenticate each other and to negotiate encryption algorithms and cryptographic keys before any data is transmitted. This ensures that both parties agree on the level of security for the connection.
2. **Data Encryption**: Once the TLS handshake is complete, the client and server can begin securely exchanging data. TLS encrypts the data using symmetric encryption, which is faster than asymmetric encryption used during the handshake.
3. **Data Integrity**: TLS uses cryptographic hash functions to ensure that data has not been tampered with during transmission. This provides assurance that the data received is the same as the data sent.
4. : When you visit a website using HTTPS (HTTP over TLS), your web browser and the web server perform a TLS handshake to establish a secure connection. The web server sends its digital certificate to the browser to prove its identity. The browser verifies the certificate and generates a symmetric encryption key for secure data exchange. All data transmitted between the browser and the server is encrypted and protected from eavesdroppers.

Expt 10: Vernam Cipher

Explain Vernam Cipher its uses features and write a short note

The Vernam Cipher, also known as the one-time pad, is a symmetric encryption algorithm that uses a random or pseudo random key of the same length as the plaintext. It is named after its inventor, Gilbert Vernam. The key is used only once and must be kept completely secret from everyone except the sender and receiver.
Features and Uses:

1. **Perfect Secrecy:** When used correctly, the Vernam Cipher provides perfect secrecy, meaning that the ciphertext reveals no information about the plaintext without the key.
2. **Unbreakable**: If the key is truly random, is as long as the plaintext, and is never reused, the Vernam Cipher is unbreakable, even with unlimited computational power.
3. **Key Distribution**: One of the main challenges of the Vernam Cipher is key distribution. The key must be securely distributed to both the sender and receiver without interception.
4. **Inefficiency**: The Vernam Cipher is not practical for large-scale communication due to the need for a key as long as the plaintext, and the need to securely distribute this key.

Short Note:
The Vernam Cipher, also known as the one-time pad, is a symmetric encryption algorithm that provides perfect secrecy when used correctly. It uses a random or pseudorandom key of the same length as the plaintext, which is only used once and must be kept completely secret. Despite its unbreakable nature, the Vernam Cipher is not widely used due to the challenges of key distribution and the inefficiency of requiring a key as long as the plaintext.

Plaintext: HELLO
Key: RANDOM

1. **Convert the plaintext and key into binary:**
   - HELLO -> 01001000 01000101 01001100 01001100 01001111
   - RANDOM -> 01010010 01000001 01001110 01000100 01001111 01001101
2. **XOR each bit of the plaintext with the corresponding bit of the key:**
   - Ciphertext: 00011010 00000100 00000110 00001000 00000000
3. **Convert the ciphertext back into text:**
   - Ciphertext: 00011010 00000100 00000110 00001000 00000000 -> 01000110 01010110 01001100 01001111 01001101
   - Encrypted Message: FVLOM
   -

Draw and explain how DES algorithm works in detail.
Explain RSA algorithm in detail with example.
Compare symmetric and asymmetric key cryptography.(Min 8 Points).
Explain any two of following with example
i. Playfair Cipher
ii. Hill Cipher
iii. Vigenère cipher
iv. One-Time Pad cipher
v. Monoalphabetic cipher

Write short note on the following
a. Transport and tunnel mode in IPSec.
b. S/MIME for Email security.
c. TLS explanation with Suitable example

Explain AH and ESP working in IPSec.
AH, the authentication header, provides integrity only; that is, with AH there is no encryption. The AH integrity protection applies to everything beyond the IP header and some fields of the header. As previously mentioned, not all fields of the IP header can be integrity protected (TTL, for example). AH classifies IP header fields as mutable or immutable, and it applies its integrity protection to all of the immutable fields. ESP, the encapsulating security payload, provides integrity and confidentiality. Both the confidentiality and integrity protection are applied to everything beyond the IP header, that is, the data (from the perspective of IP).

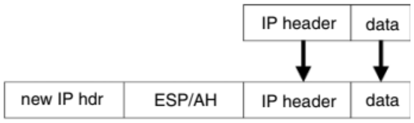


Figure 10.15. IPSec tunnel mode.

There is a clever trick whereby ESP can be used for integrity only. In ESP, Alice and Bob negotiate the cipher that they will use. One of the ciphers that must be supported is the “NULL” cipher, which is described in RFC 2410 [92]:

• NULL encryption “is a block cipher the origins of which appear to be lost in antiquity”
• “Despite rumors,” there is no evidence that NSA “suppressed publication of this

algorithm”

- Evidence suggests it was developed in Roman times as exportable version of Caesar’s cipher
- NULL encryption can make use of keys of varying length
- No IV is required
- NULL encryption is defined by Null(P , K) = P for any plaintext P and any key K

This RFC proves that security people are strange.

If the NULL cipher is selected in ESP, then no encryption is applied but the data is still integrity protected. Since this looks like the same service provided by AH, why does AH exist?

There are three reasons given for the existence of AH. As previously noted, the IP header can’t be encrypted since routers must see the header in order to route packets. But AH does provide integrity protection to the immutable fields in the IP header, whereas ESP provides no protection to the header. A second reason for the existence of AH is that ESP encrypts everything beyond the IP header, provided a non-NULL cipher is selected. If ESP is used and the packet is encrypted, a firewall can’t look inside the packet to examine the TCP header. Surprisingly, ESP with NULL encryption doesn’t solve this problem. When the firewall sees the ESP header, it will know that ESP is being used. However, the header does not say that the NULL cipher is used (that was negotiated between Alice and Bob), so the firewall can’t know whether it can read the TCP header or not.

Both of these reasons for the existence of AH are fairly weak. The designers of AH/ESP could have made minor modifications to the protocol so that ESP alone could overcome these drawbacks. But a third reason has been given for the existence of AH. At one meeting where the IPSec standard was being developed, “someone from Microsoft gave an impassioned speech about how AH was useless ...” and “... everyone in the room looked around and said, Hmm. He’s right, and we hate AH also, but if it annoys Microsoft let’s leave it in since we hate Microsoft more than we hate AH” [122]. So now you know the real reason why AH exists.