MAHENDRACOLLEGEOFENGINEERING

Approved by AICTE and Affiliated by Anna University, Chennai

NAACA ccredited – RecognizedU/S2(F)&amp;12(B)of UGC Act 1956

Salem-ChennaiHighwayNH79,Minnampalli, Salem-636106

DEPARTMENT OF INFORMATION TECHNOLOGY

CCS354- NETWORK SECURITY

RECORD NOTE BOOK

CLASS:III YEAR/VI SEMESTER

MAHENDRA COLLEGE OF ENGINEERING

Approved by AICTE and Affiliated by Anna University, Chennai NAAC
Accredited – Recognized U/S 2 (F) & 12 (B) of UGC Act 1956 **Salem-Chennai Highway NH 79, Minnampalli, Salem-636106**

DEPARTMENT OF INFORMATION TECHNOLOGY

REGISTER NUMBER :

STUDENT NAME :

SEMESTER :

YEAR :

SUBJECT CODE :

SUBJECT NAME :

MAHENDRA COLLEGE OF ENGINEERING

NAACA credited

ApprovedbyAICTE and Affiliated byAnnaUniversity,Chennai

Salem-Chennai Highway NH 79, Minnampalli, Salem-636106

Department of

____

____

LABORATORY RECORD

Certified to be t he bona fide of work done by

Name: ________________________________Register No: ________________________

Class: __________________________________Branch: _________________________

Laboratory Name: ____

HEAD OF THE DEPARTMENT STAFF-IN-CHARGE
DATE:

Submitted for the University Practical Examination on

____

INTERNALEXAMINER EXTERNALEXAMINER

INSTITUTION VISION AND MISSION

VISION

Mahendra College of Engineering is committed to be a leader in Higher Education achieving excellence through world class learning environment for Science and Technology with a blend of advanced research to create ethical and competent professionals.

MISSION

- To provide a conductive atmosphere to impart innovative knowledge and commendable skills through quality education by continuous improvement and customization of teaching.
- To nurture research attitude and bring about tangible developments with dynamic Industry - Institute Interaction.
- To create society oriented citizens with professional ethics.

MAHENDRA COLLEGE OF ENGINEERING

SALEM-CAMPUS, ATTUR MAIN ROAD, MINNAMPALLI, SALEM -636 106.

DEPARTMENT OF INFORMATION TECHNOLOGY

DEPARTMENT VISION AND MISSION

VISION

To become a department, producing graduates with good technical skills in emerging areas of Information Technology, through value based education and research.

MISSION

- To provide exposure to students to the emerging technologies in Hardware and Software.
- To inculcate students with sound application knowledge.
- To establish strong Industry- Institute Interaction.

PROGRAMME SPECIFIC OUTCOMES (PSOs)

To ensure graduates

- Have proficiency in programming skills to design, develop and apply appropriate techniques, to solve complex engineering problems.
- Have knowledge to build, automate and manage business solutions using cutting edge technologies.
- Have excitement towards research in applied computer technologies.

PROGRAM OUTCOMES (POs)

1. Engineering knowledge:

Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis:

Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions:

Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems:

Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern tool usages:

Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The engineer and society:

Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7.Environment and sustainability:

Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics:

Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work:

Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. Communications:

Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance:

Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and imultidisciplinary environments.

12.Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological chang

CCS354 –NETWORK SECURITY LABORATORY

COURSE OBJECTIVES:

- To learn the fundamentals of cryptography.
- To learn the key management techniques and authentication approaches.
- To explore the network and transport layer security techniques.
- To understand the application layer security standards.
- To learn the real time security practices.

LIST OF EXPERIMENTS:

1. Implement symmetric key algorithms.
2. Implement asymmetric key algorithms and key exchange algorithms.
3. Implement digital signature schemes.
4. Installation of Wire shark, tcpdump and observe data transferred in Client-server communication using UDP/TCP and identify the UDP/TCP datagram.
5. Check message integrity and confidentiality using SSL.
6. Experiment Eavesdropping, Dictionary attacks, MITM attacks.
7. Experiment with Sniff Traffic using ARP Poisoning.
8. Demonstrate intrusion detection system using any tool.
9. Explore network monitoring tools.
10. Study to configure Firewall, VPN.

TOTAL: 30 PERIODS

SOFTWARE AND HARDWARE REQUIREMENTS:

Software Requirements	Java or equivalent compiler GnuPG, N-Stalker or Equivalent Compiler
Hardware Requirements	Desktop Computer

COURSE OUTCOMES:

At the end of the course, the student should be able to:

CO1: Classify the encryption techniques.

CO2: Illustrate the key management technique and authentication.

CO3: Evaluate the security techniques applied to network and transport layer.

CO4: Discuss the application layer security standards.

CO5: Apply security practices for real time applications.

CO's-PO & PSO's MAPPING:

CO's	PO's											PSO's		
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**	**3**
1	3	3	2	2	2	-	-	-	2	1	2	1	2	3	1
2	1	1	3	2	2	-	-	-	2	2	1	1	3	1	2
3	1	2	1	1	2	-	-	-	3	3	1	3	2	1	3
4	2	2	3	2	3	-	-	-	3	3	2	1	2	1	3
5	2	1	3	2	2	-	-	-	2	1	1	3	2	1	1
AVg.	1.8	1.8	2.4	1.8	2.2	-	-	-	2.4	2	1.4	1.8	2.2	1.4	2

INDEX

S.NO	DATE	LIST OF EXPERIMENTS	PAGE NO	MARK	SIGN
1(a)					
1(b)					
1(c)					
2(a)					
2(b)					
3					
4					
5					
6					
7					
8					
9					
10					
CONTEND BEYOND SYLLABUS					
11					
12					

Ex.No:1 (A) Date:	**Implement Symmetric Key Algorithm Using AES**

AIM:

To implement the symmetric key algorithm using AES.

ALGORITHM:

1. Start the program.
2. Import the required Java libraries for encryption and decryption: javax.crypto.Cipher, javax.crypto.spec.SecretKeySpec, java.nio.charset.StandardCharsets, and java.util.Base64
3. Create an instance of the AES Example class.
4. Prompt the user to enter a key (16, 24, or 32 bytes) and store it in the key variable.
5. Prompt the user to enter plaintext and store it in the plaintext variable.
6. Call the encrypt method with the provided key and plaintext, print the cipher text.
   - 6.1 Accept a key and plaintext as input parameters.
   - 6.2 Create a SecretKeySpec using the key and UTF-8 encoding.
   - 6.3 Initialize a cipher instance for AES/ECB/PKCS5Padding.
   - 6.4 Initialize the cipher in encryption mode with the secret key.
   - 6.5 Encrypt the plaintext using doFinal and convert the result to Base64.
   - 6.6 Return the Base64-encoded cipher text.
7. Call the decrypt method with the provided key and cipher text, print the decrypted plaintext.
   - 7.1 Accept a key and cipher text as input parameters.
   - 7.2 Create a SecretKeySpec using the key and UTF-8 encoding.
   - 7.3 Initialize a cipher instance for AES/ECB/PKCS5Padding.
   - 7.4 Initialize the cipher in decryption mode with the secret key.
   - 7.5 Decode the Base64-encoded cipher text using Base64.getDecoder().
   - 7.6 Decrypt the cipher text using doFinal and convert the result to a UTF-8 string.
   - 7.7 Accept a key and ciphertext as input parameters.
   - 7.8 Return the decrypted plaintext.
8. Close the scanner.
9. Wrap the code in a try-catch block to handle exceptions.
10. If an exception occurs, print the stack trace for debugging purposes.
11. Stop the program.

PROGRAM:

```
import  javax.crypto.Cipher;
import.javax.crypto.spec.SecretKeySpec;
import java.nio.charset.StandardCharsets;
import java.util.Base64;
import java.util.Scanner;


public class AESUserInput {

   public static String encrypt(String key, String plaintext) throws Exception {
      SecretKeySpec secretKey=new SecretKeySpec(key.getBytes(StandardCharsets.UTF_8),"AES");
      Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
      cipher.init(Cipher.ENCRYPT_MODE, secretKey);
      byte[] encryptedBytes=cipher.doFinal(plaintext.getBytes(StandardCharsets.UTF_8));
      return Base64.getEncoder().encodeToString(encryptedBytes);
   }
   public static String decrypt(String key,String ciphertext)throws Exception {
      SecretKeySpec secretKey=new SecretKeySpec(key.getBytes(StandardCharsets.UTF_8),"AES");
      Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
      cipher.init(Cipher.DECRYPT_MODE, secretKey);
  byte[] decryptedBytes=cipher.doFinal(Base64.getDecoder().decode(ciphertext));
      return new String(decryptedBytes, StandardCharsets.UTF_8);
   }

   Public static void main(String [] args){
      try {
         Scanner scanner=new Scanner(System.in);

         //Input keyfrom user
         System.out.print("Enter the key(16,24,or32bytes):"); String
         key = scanner.nextLine();
```

```
            // Input plaintext from user
            System.out.print("Entertheplaintext:");
            String plaintext = scanner.nextLine();

            // Encryption
            String ciphertext = encrypt(key, plaintext);
            System.out.println("Ciphertext:"+ciphertext);

            // Decryption
            String decryptedText = decrypt(key, ciphertext);
            System.out.println("DecryptedText:"+decryptedText);

            scanner.close();
        }catch(Exception e){
            e.printStackTrace();
        }
    }
}
```

SAMPLE OUTPUT:

```
Enterthekey(16,24,or32bytes):1234567890123456
Enter the plaintext: Hello, AES!
Ciphertext:F1L+RZpVTAv7vxhYxypGsw==
Decrypted Text: Hello, AES!
```

RESULT:

Implementation of symmetric key using AES was successfully executed and output is verified

Ex.No:1 (B) Date:	Implement symmetric key algorithm using RC6

AIM:

To implement the symmetric key algorithm using RC6.

ALGORITHM:

1. Start the program.
2. Import the required Java libraries.
3. Print a message asking the user to enter a 32-bit.input.
4. Convert the input key bytes into an array of 32-bit words, denoted as L.
5. Initialize Constants (NUM_ROUNDS) to 20, word size (WORD_SIZE) to 32, key size (KEY_SIZE) to 16 bytes (128 bits).
6. Initialize Variables: int c to the number of words in the key (key.length / 4), int[] S with size 2 * NUM_ROUNDS + 4.
7. Set S[0] = 0xB7E15163.
8. Initialize variables A, B, i, and j to 0.
9. Key Expansion Loop:
   9.1 Iterate from i = 1 to S.length – 1.
   9.2 Set S[i] = S[i - 1] + 0x9E3779B9
   9.3 Initialize a variable v = 3 * Math.max(c, S.length).
10. RC6 Key Schedule:
    10.1 Iterate s from 0 to v – 1,Update A and S[i] using rotations and additions.
    10.2 Update B and L[j] using rotations and additions.
    10.3 Update indices i and j (with modulo c for j).
11. Store Key Schedule: Set the class variable S to the final key schedule.
12. In Main Method (`main`):- Initialize a key (`"0123456789ABCDEF"`) and plaintext (`"Hello, RC6!"`).
13. Create an instance of the `RC6` class with the key.
14. Encrypt the plaintext
    14.1 Accept a byte array `plaintext` as input.Divide the `plaintext` into blocks of size `blockSize` (16 bytes).
    14.2 For each block, convert it into an array of integers (`block`), each representing a 32-bit word.
    14.3 Initialize variables `A`, `B`, `C`, and `D` with values from the block.
    14.4 Perform the encryption rounds for each block using the RC6 algorithm:
    14.5 Update `A`, `B`, `C`, and `D` using bitwise rotations, XOR operations, and additions.
    14.6 Wrap the resulting `A`, `B`, `C`, and `D` values into a byte array `ciphertext`.
    14.7 Return the `ciphertext` array.
15. Decrypt the ciphertext and print the decrypted message.
    15.1 Accept a byte array `ciphertext` as input.
    15.2 Divide the `ciphertext` into blocks of size `blockSize` (16 bytes).
    15.3 For each block, convert it into an array of integers (`block`), each representing a 32-bit word.

15.4 Initialize variables `A`, `B`, `C`, and `D` with values from the block.

15.5 Perform the decryption rounds for each block using the RC6 algorithm in reverse:
15.6 Update `A`, `B`, `C`, and `D` using bitwise rotations, XOR operations, and subtractions.
15.7 Subtract the initial key-related values (`S[0]`, `S[1]`, `S[2]`, `S[3]`) from `A`, `B`, `C`, and `D`.
15.8 Wrap the resulting `A`, `B`, `C`, and `D` values into a byte array `plaintext`.
15.9 Return the `plaintext` array.

16. Print the original plaintext, encrypted ciphertext (in hexadecimal), and the decrypted plaintext.

PROGRAM:

```
import java.nio.ByteBuffer;
import java.nio.charset.StandardCharsets;
public class RC6 {
    private static final int WORD_SIZE = 32;
    private static final int NUM_ROUNDS=20;
    private static final int KEY_SIZE=16;//16bytes=128bits
    private int[] S;
    public RC6(byte[]key){
        keyExpansion(key);
    }
    Private void key Expansion(byte[]key){
        int[] L = new int[key.length / 4];
        for(int i = 0; i< key.length; i +=4) {
            L[i/4]=ByteBuffer.wrap(key,i, 4).getInt();
        }
        intc=key.length/4;

int[]S=new int[2*NUM_ROUNDS+4]; S[0] = 0xB7E15163;
        for (inti=1;i<S.length;i++){
        S[i] =S[i-1]+0x9E3779B9;
        }
        int A,B,i, j;
        A =B =i = j =0;
    int v=3*Math.max(c,S.length);

    for (int s = 0; s < v; s++) {
        A=S[i] =Integer.rotateLeft((S[i] +A+B) <<3,3);
        B= L[j] = Integer.rotateLeft((L[j]+A+B)<<(A+B),A+B);
        i= (i +1)% S.length;
        j= (j +1)%c;
    }
    this.S =S;
}
public byte[] encrypt(byte[] plaintext) {
    int blockSize=16; //16bytesperblock
    int numBlocks = plaintext.length / blockSize;
    byte[] ciphertext=new byte[plaintext.length];
    for (int i = 0; i < numBlocks; i++) {
        int[] block = new int[blockSize / 4];
        for(intj =0; j<block.length;j++) {
            block[j]=ByteBuffer.wrap(plaintext,i*blockSize+j*4, 4).getInt();
        }
        int     A=block[0]+S[0];
        int B=block[1]+S[1];

         int C=block[2]+S[2];

        int D =block[3]+S[3];
        for (int round=1;round<=NUM_ROUNDS;round++){
            A = Integer.rotateLeft(A ^ B, B) + S[2 * round];
            C=Integer.rotateLeft(C^D,D)+S[2*round+1];
            B=Integer.rotateLeft(B^A,A)+S[2*round+2];
            D=Integer.rotateLeft(D ^C,C) +S[2* round+3];


}
        ByteBuffer.wrap(ciphertext, i * blockSize, 4).putInt(A);
        ByteBuffer.wrap(ciphertext, i * blockSize + 4, 4).putInt(B);
        ByteBuffer.wrap(ciphertext, i * blockSize + 8, 4).putInt(C);
        ByteBuffer.wrap(ciphertext,i*blockSize+12,4).putInt(D);
    }
    return ciphertext;
}
```

```
public byte[] decrypt(byte[] ciphertext) {
intblockSize=16; //16 bytesper block
   intnumBlocks=ciphertext.length/blockSize;
   byte[]plaintext=newbyte[ciphertext.length];
   for (int i = 0; i < numBlocks; i++) {
      int[] block = new int[blockSize / 4];
      for(intj =0; j<block.length;j++) {
         block[j] =ByteBuffer.wrap(ciphertext,i*blockSize+j*4, 4).getInt();
      }
      intA=block[0];
      intB=block[1];
      intC=block[2];
      int D = block[3];
      for(intround=NUM_ROUNDS;round>=1;round--){  D  =
         Integer.rotateRight(D - S[2 * round + 3], C) ^ C;
         B= Integer.rotateRight(B -S[2* round+2], A)^ A;
         C=Integer.rotateRight(C-S[2*round+1],D)^D;   A   =
         Integer.rotateRight(A - S[2 * round], B) ^ B;
      }
      A-=S[0];
      B-=S[1];
      C-=S[2];
      D-=S[3];
      }

      ByteBuffer.wrap(plaintext, i * blockSize, 4).putInt(A);
      ByteBuffer.wrap(plaintext,i*blockSize+4,4).putInt(B);
      ByteBuffer.wrap(plaintext,i*blockSize+8,4).putInt(C);
       ByteBuffer.wrap(plaintext,i*blockSize+12,4).putInt(D);
   }
   return plaintext;
}
```

```
    public static void main(String[]args){
        // Exampleusage
        byte[]key="0123456789ABCDEF".getBytes(StandardCharsets.UTF_8);//16-bytekey byte[]
        plaintext = "Hello, RC6!".getBytes(StandardCharsets.UTF_8);
        RC6rc6=new RC6(key);
        byte[]ciphertext=rc6.encrypt(plaintext.clone());//Clonetoavoidmodifyingtheoriginalarray byte[]
        decryptedText = rc6.decrypt(ciphertext);
        System.out.println("Original:"+newString(plaintext,StandardCharsets.UTF_8));
        System.out.print("Encrypted: ");
        printByteArrayAsHex(ciphertext);
        System.out.println("\nDecrypted:"+newString(decryptedText,StandardCharsets.UTF_8));
    }
    Private static void printByteArrayAsHex(byte[]array){
        for (byte b : array) {
            System.out.print(String.format("%02X",b));
        }
        System.out.println();
    }
}
```

SAMPLE OUTPUT:

```
Original: Hello, RC6!
Encrypted: 2A42B88F5D7998C36FD09F7C6D54C5D1
Decrypted: Hello, RC6!
```

RESULT:

Implementation of symmetric key using RC6 was successfully executed and output is verified.

Ex.No:1 (C) **Date:**	**Implement Symmetric Key Algorithm Using BLOWFISH**

AIM:

To implement the symmetric key algorithm using blowfish.

ALGORITHM:

1. Start the program.
2. Import the required Java libraries.
3. Create a `Scanner` to get input from the user.
4. Prompt the user to enter a key (up to 56 bytes) and read it from the console.
5. Prompt the user to enter plaintext and read it from the console.
6. Call the `encrypt` method with the key and plaintext.
   6.1 Convert the input key into a `SecretKey` using `SecretKeySpec` and UTF-8 encoding.
   6.2 Create a `Cipher` instance for Blowfish encryption.
   6.3 Initialize the cipher in encryption mode using the secret key.
   6.4 Convert the plaintext into bytes using UTF-8 encoding.
   6.5 Use the initialized cipher to perform encryption on the plaintext bytes.
   6.6 Return the resulting cipher text as a byte array
   6.7 Print the Base64-encoded cipher text.
7. Call the `decrypt` method with the key and the ciphertext obtained from the encryption step.
   7.1 Convert the input key into a `SecretKey` using `SecretKeySpec` and UTF-8 encoding.
   7.2 Create a `Cipher` instance for Blowfish decryption.
   7.3 Initialize the cipher in decryption mode using the secret key.
   7.4 Use the initialized cipher to perform decryption on the input ciphertext bytes.
   7.5 Convert the decrypted bytes into a string using UTF-8 encoding.
   7.6 Return the resulting decrypted text.
   7.7 Print the decrypted text.
8. If an exception occurs, print the stack trace for debugging purposes.
9. Stop the program.

PROGRAM:

```
import      javax.crypto.Cipher;
import javax.crypto.SecretKey;
import  javax.crypto.spec.SecretKeySpec;
importjava.nio.charset.StandardCharsets;
import java.util.Base64;
import java.util.Scanner;

public class BlowfishImplementationWithInput{

    public static byte[] encrypt(String key, String plaintext) throws Exception {
        SecretKey    secretKey    =    new    SecretKeySpec(key.getBytes(StandardCharsets.UTF_8),
"Blowfish");

        Cipher cipher = Cipher.getInstance("Blowfish");
        cipher.init(Cipher.ENCRYPT_MODE,secretKey);
        return cipher.doFinal(plaintext.getBytes(StandardCharsets.UTF_8));
    }
    public static String decrypt(String key,byte[] ciphertext)throws Exception
    {
 SecretKey secretKey=new SecretKeySpec(key.getBytes(StandardCharsets.UTF_8),"Blowfish");
        Cipher cipher = Cipher.getInstance("Blowfish");
        cipher.init(Cipher.DECRYPT_MODE, secretKey);
        byte[] decryptedBytes=cipher.doFinal(ciphertext);
        return new String(decryptedBytes,StandardCharsets.UTF_8);
    }
    public static void main(String[] args){
        try {
            Scanner scanner=new Scanner(System.in);

            //Input key from user
            System.out.print("Enter the key (upto 56 bytes):");
            String key = scanner.nextLine();

            // Input plaintext from user
            System.out.print("Enter the plain text:");
            String plaintext = scanner.nextLine();

            // Encryption
            byte[] ciphertext = encrypt(key, plaintext);
            System.out.println("Ciphertext(Base64):"+
Base64.getEncoder().encode To String(ciphertext));

            // Decryption
            String decryptedText = decrypt(key, ciphertext);
            System.out.println("Decrypted Text:"+decryptedText);

            scanner.close();
        } catch (Exceptione){
            e.printStackTrace();
        }
    }
```

SAMPLE OUTPUT:

```
Enterthekey(upto56bytes):mySecretKey
Enter the plaintext: Hello, Blowfish!
Ciphertext(Base64):DrnuwAAwC8C0Ec0zxLb9Yw== Decrypted
Text: Hello, Blowfish!
```

VIVA QUESTIONS:

1. What is Advanced Encryption Standard?
2. What is the parameter of AES algorithm?
3. What is Key Length of AES?
4. What is overview of RC6 Algorithm?
5. What is Key Generation in RC6?
6. What is the Key Length and Security?
7. What is the Feistel Network Structure?
8. What are the Performance Evaluation in blowfish algorithm?

RESULT:

Implementation of symmetric key using blowfish was successfully executed and output is verified.

Ex.No:2 (A) Date:	**Implement Asymmetric Key Algorithm**

AIM:

To implement the asymmetric key algorithm.

ALGORITHM:

1. Start the program.
2. Choose two large prime numbers `p` and `q`.
3. Compute `n` as the product of `p` and `q`: `n = p * q`, Which is part of the public key.
4. Compute `phi` (Euler's totient function) as `(p - 1) * (q - 1)`.
5. Choose a public exponent `e` such that `1 < e < phi` and `gcd(e, phi) = 1`.
6. In the provided code, `e = 2` is chosen initially, and then it is incremented until `gcd(e, phi) = 1`.
7. Compute the private exponent `d` using the equation `d * e ≡ 1 (mod phi)`.
8. In the provided code, `d = (1 + (k * phi)) / e`, where `k` is an integer (`k = 2` in the code).
9. Choose a message `msg` to be encrypted. (In the provided code, `msg = 12` is chosen for demonstration.)
10. Compute `c` (Cipher text):Compute `c ≡ (msg^e) mod n`.
11. Print the encrypted message `c`.
12. Decrypt the cipher text
13. Compute `m` (Original Message):Compute `m ≡ (c^d) mod n`.
14. Print the original message `m`.
15. Print the original message (`msg`), encrypted message (`c`), and decrypted message (`m`).

PROGRAM:

```
import java.io.*;
importjava.math.*;
import java.util.*;
public class GFG
{
Public static doublegcd(doublea, double h)
{
doubletemp;
while (true)
{
temp = a % h;
if (temp == 0)
return h;
a=h;
h =temp;
}
}
Public static void main(String[]args)
{
double p = 3;
double q = 7;
doublen=p*q;
double e = 2;
doublephi=(p-1)*(q-1); while (e
< phi) {
if(gcd(e,phi)==1) break;
elsee++;
}
int k =2;
doubled=(1+(k*phi))/e; double
msg = 12;
System.out.println("Messagedata="+msg);doublec=Math.pow(msg,e); c = c
% n;
System.out.println("Encrypteddata="+c);doublem=Math.pow(c,d); m = m
% n;
System.out.println("OriginalMessageSent= "+ m);
}
```

SAMPLE OUTPUT:

```
Message data=12.000000

Encrypted data=3.000000
Original Message Sent= 12.000000
```

RESULT:

Implementation of Asymmetric key was successfully executed and output is verified.

Ex.No:2 (B) Date:	**Implement Asymmetric Key Exchange Algorithm**

AIM:

To implement the asymmetric key exchange algorithm.

ALGORITHM:

1. Start the program.
2. Initialize large prime number P and primitive root G.
3. Choose private keys for Alice(a) and Bob(b).
4. Calculate public keys for Alice(x) and Bob(y) using the chosen primitive root and private keys.
   4.1. Alice computes her public value `x` using the formula: `x = G^a mod P`.
   4.2. Bob computes his public value `y` using the formula: `y = G^b mod P`.
5. Exchange Public Values Alice and Bob exchange their public values (`x` and `y`).
6. Calculate the shared secrets for Alice(ka) and Bob (kb)using their private keys and the other party's public key.
   6.1. Alice computes the shared secret key (`ka`) using Bob's public value:
   `ka = y^a mod P`.
   6.2. Bob computes the shared secret key (`kb`) using Alice's public value:
   `kb = x^b mod P`.
7. Print the computed shared secrets for Alice and Bob.

PROGRAM:

```
Class GFG{
Private static long power(longa, longb, longp)
{
if(b==1)
return a;
else
return(((long)Math.pow(a,b))%p);
}
Public static void main(String[]args)
{
longP,G,x,a,y,b,ka,kb;

P = 23;
System.out.println("ThevalueofP:"+P);
 G = 9;
System.out.println("ThevalueofG:"+G);
a = 4;


                System.out.println("The private key
                a for Alice:"+a); x = power(G, a, P);
                b =3;
                System.out.println("The private key
                b for Bob:"+b); y = power(G, b, P);
                ka=power(y,a,P);//Secret
                key for Alice kb =
                power(x, b, P); // Secret
                key for Bob
                System.out.println("Secret key for the Alice is:"+ka);
                System.out.println("Secret key for the Bob is:"+ kb);
                }
}
```

SAMPLE OUTPUT:

```
The value of P: 23

The value of G: 9
The private key a for Alice:4
The private key b for Bob : 3
Secret key for the Alice is : 9
Secret Key for the Bob is : 9
```

VIVA QUESTIONS:

1. What is Asymmetric Key Algorithm?
2. What is Key Pair Generation?
3. How to consider Key Length in Asymmetric Key algorithm?
4. How Key Exchange Mechanisms works?
5. What are Common Asymmetric Algorithms?

RESULT:

Implementation of Asymmetric key exchange was successfully executed and output is verified.

Ex.No:3 **Date:**	**Implement Digital Signature Scheme**

AIM:

To implement the signature scheme-Digital Signature Standard.

ALGORITHM:

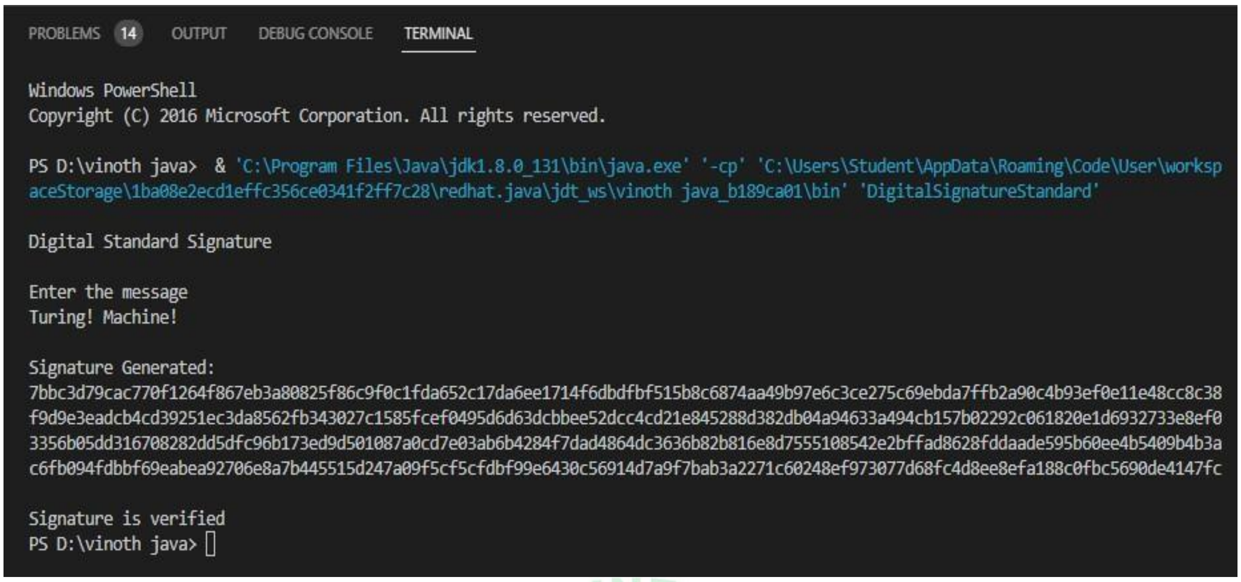
1. Declare the class and required variables.
2. Create the object for the class in the main program.
3. Access the member functions using the objects.
4. Implement the SIGNATURESCHEME-Digital Signature Standard.
5. Generate an RSA key pair (public and private keys) using a key size of 2048 bits.
6. Prompt the user to enter a message.
7. Compute the digital signature of the message using the private key.
   7.1. Create a `Signature` instance with the algorithm "SHA256withRSA".
   7.2. Initialize the signature with the private key.
   7.3. Update the signature with the message bytes.
   7.4. Generate the digital signature.
8. Print the generated digital signature in hexadecimal format.
9. Verify the generated signature using the public key.
   9.1. Create a `Signature` instance with the algorithm "SHA256withRSA".
   9.2. Initialize the signature with the public key.
   9.3. Update the signature with the message bytes.
   9.4. Verify the signature against the generated signature.
10. Print whether the signature is verified or not.

PROGRAM:

```
import java.security.KeyPair;

import java.security.KeyPairGenerator;

import java.security.PrivateKey;

import java.security.PublicKey;

import java.security.SecureRandom;

import java.security.Signature;

import java.util.Scanner;

public class DigitalSignatureStandard{

   public static byte[] generateSignature(byte[]messageBytes,PrivateKeyKey)throws Exception
{

      Signaturesignature=Signature.getInstance("SHA256withRSA");

      signature.initSign(Key);


      signature.update(messageBytes);

      return signature.sign();

   }

publicstaticKeyPairgenerateRSAKeyPair()throwsException{ SecureRandom

      secureRandom = new SecureRandom();

      KeyPairGeneratorkeyPairGenerator=KeyPairGenerator.getInstance ("RSA");

      keyPairGenerator.initialize(2048, secureRandom);

returnkeyPairGenerator.generateKeyPair();

   }

   public static Boolean isVerifiedSignature(byte[]messageBytes,byte[]signatureGenerated,
PublicKey publicKey) {

      try{

         Signaturesignature=Signature.getInstance("SHA256withRSA"); signature.initVerify(publicKey);

         signature.update(messageBytes);

         returnsignature.verify(signatureGenerated);}catch (Exception e)

         { e.printStackTrace();
      }
```

```
     returnfalse;
}
public static void main(String[] args) {
    Scannersc=newScanner(System.in);
    System.out.println("\nDigitalStandardSignature\n");
    System.out.println("Enter the message");
    Stringmessage=sc.next();
    try{
        KeyPairkeyPair=generateRSAKeyPair();
        byte[]signature=generateSignature(message.getBytes(),keyPair.getPrivate());
        //bytes to hex

        System.out.println();
        System.out.println("Signature Generated:");
        for (byte b : signature) {
            Stringhex=String.format("%02x",b);
            System.out.print(hex);
        }
        System.out.println("\n");
        if(isVerifiedSignature(message.getBytes(),signature,keyPair.getPublic())){
            System.out.println("Signature is verified");
        }else{
            System.out.println("Signatureisnotverified");
        }
    }catch(Exceptione){
        e.printStackTrace();
    }
}
```

SAMPLE OUTPUT:



```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS D:\vinoth java>  & 'C:\Program Files\Java\jdk1.8.0_131\bin\java.exe' '-cp' 'C:\Users\Student\AppData\Roaming\Code\User\worksp
aceStorage\1ba08e2ecd1effc356ce0341f2ff7c28\redhat.java\jdt_ws\vinoth java_b189ca01\bin' 'DigitalSignatureStandard'

Digital Standard Signature

Enter the message
Turing! Machine!

Signature Generated:
7bbc3d79cac770f1264f867eb3a80825f86c9f0c1fda652c17da6ee1714f6dbdfbf515b8c6874aa49b97e6c3ce275c69ebda7ffb2a90c4b93ef0e11e48cc8c38
f9d9e3eadcb4cd39251ec3da8562fb343027c1585fcef0495d6d63dcbbee52dcc4cd21e845288d382db04a94633a494cb157b02292c061820e1d6932733e8ef0
3356b05dd316708282dd5dfc96b173ed9d501087a0cd7e03ab6b4284f7dad4864dc3636b82b816e8d7555108542e2bffad8628fddaade595b60ee4b5409b4b3a
c6fb094fdbbf69eabea92706e8a7b445515d247a09f5cf5cfdbf99e6430c56914d7a9f7bab3a2271c60248ef973077d68fc4d8ee8efa188c0fbc5690de4147fc

Signature is verified
PS D:\vinoth java> 
```

VIVA QUESTIONS:

1. What is digital signature?
2. List two general schemes of Digital Signatures
3. List the digital signature algorithm function
4. An example for digital signature
5. Advantage of Digital signature
6. Disadvantage of Digital signature
7. Real time example for digital signature

RESULT:

Thus the Digital Signature Standard Signature Scheme has been implemented and the output has been verified successfully.

Ex.No:4 **Date:**	**Installation of Wire shark,tcp-dump and observe data transferred In client-server communication using UDP/TCP and identify the UDP/TCP datagram.**

AIM:

Installation of Wire shark, TCP-DUMP using UDP/TCP.

ALGORITHM/STEPSTOINSTALLWIRESHARK:

Step1: Visit the official Wire shark website using any web browser.

Step2: Click on Download, anew webpage will open with different installers of Wireshark.

Step3: Downloading of the executable file will start shortly. It is a small 73.69MB file that will take some time.

Step4: Now check for the executable file in downloads in your system and run it.

Step5: It will prompt confirmation to make changes to your system. Click on Yes. Step 6: Setup screen will appear, click on Next.

Step7:The next screen will be of License Agreement, click on Noted.

Step 8: This screen is for choosing components, all components are already marked so don't change anything just clicks on the Next button.

Step 9: This screen is of choosing shortcuts like start menu or desktop icon along with file extensions which can be intercepted by Wireshark, tick all boxes and click on Next button.

Step 10: The next screen will be of installing location so choose the drive which will have sufficient memory space for installation. It needed only a memory space of 223.4 MB.

Step 11: Next screen has an option to install Npcap which is used with Wireshark to capture packets pcap means packet capture so the install option is already checked don't change anything and click the next button.

Step 12: Next screen is about USB network capturing so it is one's choice to use it or not, click on Install.

Step13: After this installation process will start.

Step 14: This installation will prompt for Npcap installation as already checked so the license agreement of Npcap will appear to click on the I Agree button.

Step15: Next screen is about different installing options of Npcap, don't do anything click on Install.

Step16: After this installation process will start which will take only a minute.

Step17: After this installation process will complete click on the Next button.

Step 18: Click on Finish after the installation process is complete.

Step19: After this installation process of Wireshark will complete click on the Next button.

Step 20: Click on Finish after the installation process of Wireshark is complete.

PROCESS:

Introduction The first part of the lab introduces packet sniffer,Wireshark.Wireshark is a free open- source network protocol analyzer. It is used for network troubleshooting and communication protocol analysis. Wireshark captures network packets in real time and display them in human-readable format. It provides many advanced features including live capture and offline analysis, three-pane packet browser, coloring rules for analysis. This document uses Wireshark for the experiments, and it covers Wireshark installation, packet capturing, and protocol analysis.

In the CSC 4190 Introduction to Computer Networking (one of the perquisite courses), TCP/IP network stack is introduced and studied. This background section briefly explains the concept of TCP/IP network stack to help you better understand the experiments. TCP/IP is the most commonly used network model for Internet services. Because its most important protocols, the Transmission Control Protocol (TCP) and the Internet Protocol (IP) were the first networking protocols defined in this standard, it is named as TCP/IP. However, it contains multiple layers including application layer, transport layer, network layer, and data link layer.

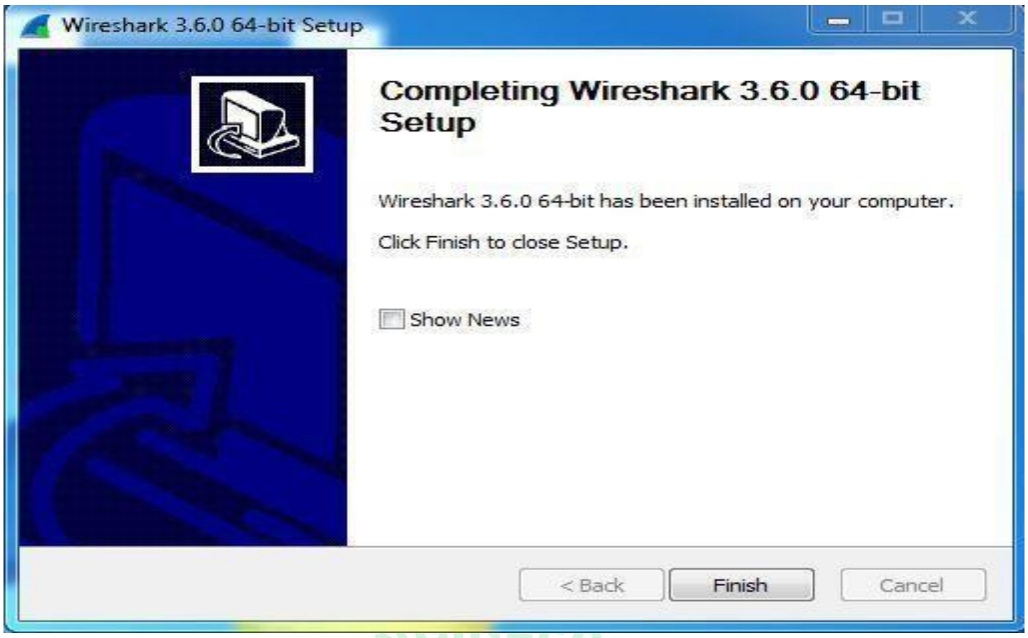
Application Layer: The application layer includes the protocols used by most applications for providing user services. Examples of application layer protocols are Hypertext Transfer Protocol (HTTP), Secure Shell (SSH), File Transfer Protocol (FTP), and Simple Mail Transfer Protocol (SMTP).

Transport Layer: The transport layer establishes process-to-process connectivity, and it provides end-to-end services that are independent of underlying user data. To implement the process-to-process communication, the protocol introduces a concept of port. The examples of transport layer protocols are Transport Control Protocol (TCP) and User Datagram Protocol (UDP). The TCP provides flow- control, connection establishment, and reliable transmission of data, while the UDP is a connectionless transmission model.

Internet Layer: The Internet layer is responsible for sending packets to across networks. It has two functions: 1) Host identification by using IP addressing system (IPv4 and IPv6); and 2) packets routing from source to destination. The examples of Internet layer protocols are Internet Protocol (IP), Internet Control Message Protocol (ICMP), and Address Resolution Protocol (ARP).

Link Layer: The link layer defines the networking methods within the scope of the local network link. It is used to move the packets between two hosts on the same link. An common example of link layer protocols is Ethernet.

SAMPLE OUTPUT:

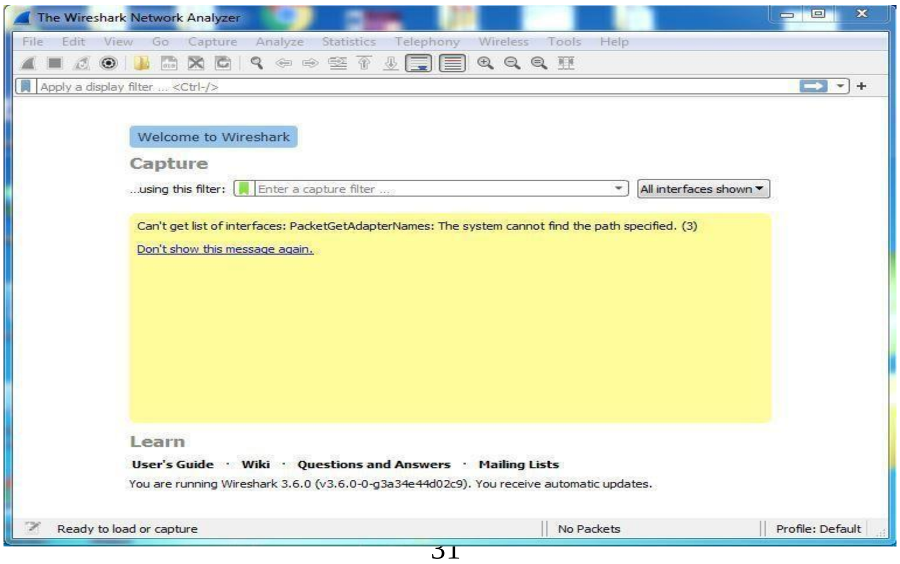


Completing Wireshark 3.6.0 64-bit Setup

Wireshark 3.6.0 64-bit has been installed on your computer.

Click Finish to close Setup.

Show News


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
All interfaces shown
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists

Can't get list of interfaces: PacketGetAdapterNames: The system cannot find the path specified. (3)

Don't show this message again.

You are running Wireshark 3.6.0 (v3.6.0-0-g3a34e44d02c9). You receive automatic updates.

Ready to load or capture | No Packets | Profile: Default

VIVA QUESTIONS:

1. What is Wire shark, and what is its primary purpose in network analysis?
2. Explain the purpose of Tcp dump and how it differs fromWireshark interms of functionality.
3. What are the common protocols associated with client-server communication over a network?
4. Describe the fundamental differences between TCP and UDP in terms of connection-oriented versus connectionless communication.
5. What is a UDP datagram, and how does it differ from a TCP segment?
6. How can filtering be utilized in Wireshark to focus specifically on UDP or TCP traffic?

RESULT:

Thus, the Wireshark tool is installed successfully

Ex.No:5 Date:	**Check message integrity and confidentiality using SSL**

AIM:

To check message integrity and confidentiality using SSL.

SERVER ALGORITHM:

1. Start the program.
2. Set system properties for the server's key store, key store password, trust store, and trust store password.
3. Set system properties for the server's key store, key store password, trust store, and trust store password.
4. Create an SSLServer Socket Factory and use it to create an SSLServer Socket on port 9999.
5. Wait for a client to connect by accepting an SSL socket connection.
6. Receive Data from Client
   6.1. Create a BufferedReader to read data from the client.
   6.2. Receive a message from the client.
7. Process the received message (in this case, prefix it with "Processed: ").
8. Send Processed Data to Client:
   8.1. Create an OutputStream to send the processed message back to the client.
   8.2. Send the processed message to the client.
9. Close the BufferedReader, OutputStream, SSL socket, and SSL server socket.

PROGRAM:

SERVER

```
import java.io.BufferedReader;
importjava.io.InputStreamReader;
import java.io.OutputStream;
import javax.net.ssl.SSLServerSocket;
import javax.net.ssl.SSLServerSocketFactory;
import javax.net.ssl.SSLSocket;
public class SecureServer{
public static void main(String[]args){
try {
System.setProperty("javax.net.ssl.keyStore", "server key store.jks");
System.setProperty("javax.net.ssl.keyStorePassword","key store password")
```

```
 System.setProperty("javax.net.ssl.trustStore", "servertruststore.jks");
System.setProperty("javax.net.ssl.trustStorePassword", "truststorepassword");
SSLServerSocketFactorysslServerSocketFactory=(SSLServerSocketFactory)
SSLServerSocketFactory.getDefault();
SSLServerSocket serverSocket = (SSLServerSocket)
sslServerSocketFactory.createServerSocket(9999);
System.out.println("Serverwaitingforclientconnection...");
SSLSocket sslSocket = (SSLSocket) serverSocket.accept();
System.out.println("Client connected.");
BufferedReader reader = new BufferedReader(new
InputStreamReader(sslSocket.getInputStream()));
OutputStream outputStream = sslSocket.getOutputStream();
String clientMessage = reader.readLine();
System.out.println("Receivedfromclient:"+clientMessage);
String processedMessage = "Processed: " + clientMessage;
outputStream.write(processedMessage.getBytes());
System.out.println("Sent to client: " + processedMessage);
reader.close();
outputStream.close();
sslSocket.close();
serverSocket.close();
}catch(Exceptione){
e.printStackTrace();
}
}
}
```

CLIENT ALGORITHM:

1. Set system properties for the server's key store, key store password, trust store, and trust store password.
2. Create an SSLSocketFactory and use it to create an SSLSocket connecting to the server on local host at port 9999.
3. Send Data to Server:
   3.1. Create a BufferedReader to read data from the server.
   3.2. Create an OutputStream to send a message to the server (in this case, "Hello from the client!").
4. Receive Processed Data from Server:
   4.1. Send the message to the server.
   4.2. Receive the processed message from the server.
5. Print the received processed message from the server.
6. Close the BufferedReader, OutputStream, and SSLSocket.
7. Stop the program.

PROGRAM:

CLIENT

```
import java.io.BufferedReader;

importjava.io.InputStreamReader;

import java.io.OutputStream;

import javax.net.ssl.SSLSocket;

importjavax.net.ssl.SSLSocketFactory;

public class SecureClient {

public static void main(String[]args){

try {

System.setProperty("javax.net.ssl.keyStore", "clientkeystore.jks");

System.setProperty("javax.net.ssl.keyStorePassword", "key store password");
System.setProperty("javax.net.ssl.trustStore", "client trust store.jks");

System.setProperty("javax.net.ssl.trustStorePassword", "trust store password");

SSLSocketFactory sslSocketFactory = (SSLSocketFactory) SSLSocketFactory.getDefault();



SSLSocket sslSocket = (SSLSocket) sslSocketFactory.createSocket("localhost", 9999);

System.out.println("Connected to server.");

BufferedReader reader=newBufferedReader(new
(sslSocket.getInputStream()));

OutputStream outputStream = sslSocket.getOutputStream();

String message = "Hello from the client!";

System.out.println("Sendingmessagetoserver:"+message);

outputStream.write(message.getBytes());

StringserverResponse=reader.readLine();

System.out.println("Received response from server:"+serverResponse);
reader.close();

outputStream.close();

sslSocket.close();

}catch (Exceptione){

e.printStackTrace();

}

}

}
```

SERVER SAMPLE OUTPUT:

```
    Server waiting for client connection...
Client connected.
            Received from client: Hello from the client!
            Sent to client: Processed: Hello from the client!
```

CLIENT SAMPLE OUTPUT:

```
            Connected to server.
            Sending message to server: Hello from the client!
            Received response from server: Processed: Hello from the client!
```

VIVA QUESTIONS:

1. What is the primary purpose of SSL (Secure Sockets Layer) in web communication?
2. Explain the concepts of message integrity and confidentiality in the context of secure communication.
3. How does SSL ensure message integrity during data transmission?
4. Describe the encryption process used by SSL to ensure the confidentiality of transmitted data.
5. What role do SSL certificates play in establishing the authenticity of a website or server?
6. Describe the steps involved in setting up SSL for a web server.
7. Discuss the role of symmetric and asymmetric encryption in SSL.

RESULT:

Thus the program to check message integrity and confidentiality using SSL was successfully executed and output is verified.

Ex.No:6 Date:	**Experiment On Eavesdropping, Dictionary Attacks, MitmAttacks**

AIM:

To study on Eavesdropping, Dictionary attacks and MITM attacks.

EAVESDROPPING:

Definition:

An Eavesdropping attack occurs when a hacker intercepts, deletes, or modifies data that is transmitted between two devices. Eavesdropping, also known as sniffing or snooping, relies on unsecured network communications to access data in transit between devices. The data is transmitted across an open network, which gives an attacker the opportunity to exploit vulnerability and intercept it via various methods. Eavesdropping attacks can often be difficult to spot.

Attacks:

Eavesdropping attacks are a big threat to the integrity and confidentiality of the data. It allows an attacker to gather sensitive information, such as login credentials, financial data, or personal conversations, without the victim's knowledge. Furthermore, attackers can use the extracted information for various malicious purposes, such as identity theft, extortion, or espionage.

Let's look at the general steps in order to launch an Eavesdropping attack:

The first step is identifying a target for the attack, such as a specific individual or organization. As soon as the attacker identifies the target, it starts gathering information about it. Some useful information the attacker wants to extract includes the communication systems and vulnerabilities that can be exploited.

The next step is to choose an appropriate method for the successful execution of the attack. There're several different methods that an attacker can use. Some examples are intercepting communication over unsecured networks, using malware to gain access to a device, or using hardware devices.

The next step is to execute the chosen method in the target system and intercept the target's communication. Finally, the attacker analyzes the intercepted communication and extracts valuable information.

Prevention Techniques:

We can use several techniques to prevent eavesdropping attacks. Some popular techniques include encryption, virtual private networks, secure communication protocols, firewalls, and network segmentation. Encrypting communication makes it difficult for attackers to intercept and read messages. In order toencrypt communication, we can use different types of encryption algorithms, such as symmetric key algorithms and public key algorithms. Advanced Encryption Standard (AES) is an example of a symmetric key algorithm. Additionally, Rivest–Shamir–Adleman (RSA) is a widely used public key algorithm.

Virtual private networks (VPNs) create a secure, encrypted connection between a device and a remote server. Theycan help to prevent eavesdroppingattacks byencryptingcommunication and makingit difficult for attackers to intercept.

DICTIONARYATTACKS:

Definition:

A dictionary attack is a method of breaking into a password-protected computer, network or other IT resourcebysystematicallyenteringeverywordin adictionaryasapassword.Adictionaryattack canalsobe used in an attempt to find the key necessary to decrypt an encrypted message or document.

Dictionary attacks work because many computer users and businesses insist on using ordinary words as passwords. These attacks are usually unsuccessful against systems using multiple-word passwords and are also often unsuccessful against passwords made up of uppercase and lowercase letters and numbers in random combinations.

How do dictionary attacks work?

A dictionary attack uses a preselected library of words and phrases to guess possible passwords.It operates under the assumption that users tend to pull from a basic list of passwords, such as "password,""123abc" and "123456."

These lists include predictable patterns that can varybyregion. For example, hackers looking to launch a dictionary attack on a New York-based group of targets might look to test phrases like "knicksfan2020" or "newyorkknicks1234." Attackers incorporate words related to sports teams, monuments, cities, addresses and other regionally specific items when building their attack library dictionaries.

How effective is a dictionary attack?

How successful a dictionary attack is depending on how strong the passwords are for the individuals a hacker is targeting? Because weak passwords are still common, attackers continue to have success with these attacks. Individual users, however, aren't the only ones who are subject to weak password security.

Take steps to prevent a dictionary attack

Dictionary hacking is a very common type of cybercrime that hackers use to gain access to an individual's personal accounts, including bank accounts, social media profiles, and emails. With this access, hackers can perpetrate all sorts of actions, from financial fraud and malicious social media posts to further cybercrimes like phishing. However, dictionary attack prevention can be as simple as implementing certain safeguards to minimize the risk of falling victim to these attacks. Using smart password management habits, employing different types of authentications, and using readily available password managers, for example, can all help keep passwords and accounts secure.

MITMATTACKS:

What is a Man-in-the-Middle(MITM)Attack?

Man-in-the-middle attacks (MITM) are a common type of cybersecurity attack that allows attackers to eavesdrop on the communication between two targets. The attack takes place in between two legitimately communicating hosts, allowing the attacker to "listen" to a conversation they should normally not be able to listen to, hence the name "man-in-the-middle.

Types of Man-in-the-Middle Attacks

Rogue Access Point

Devices equipped with wireless cards will often try to auto-connect to the access point that is emittingthestrongestsignal.Attackerscansetuptheirownwirelessaccess pointandtricknearbydevicesto join its domain.

ARP Spoofing

ARP is the Address Resolution Protocol. It is used to resolve IP addresses to physical MAC (media access control) addresses in a local area network. When a host needs to talk to a host with a given IP address, it references the ARP cache to resolve the IP address to a MAC address.

MDNS Spoofing

Multicast DNS is similar to DNS, but it's done on a local area network (LAN) using broadcast like ARP. This makes it a perfect target for spoofing attacks. The local name resolution system is supposed to make the configuration of network devices extremely simple.

DNS Spoofing

Similar to the way ARP resolves IP addresses to MAC addresses on a LAN, DNS resolves domain names to IP addresses. When using a DNS spoofing attack, the attacker attempts to introducecorrupt DNS cache information to a host in an attempt to access another host using their domain name, such as www.onlinebanking.com.

Man-in-the-Middle Attack Techniques

Sniffing

Attackers use packet capture tools to inspect packets at lowlevel.Using specific wireless devices that are allowed to be put into monitoring or promiscuous mode can allow an attacker to see packets that are not intended for it to see, such as packets addressed to other hosts.

Packet Injection

An attacker can also leverage their device's monitoring mode to inject malicious packets intodata communication streams. The packets can blend in with valid data communication streams, appearing to be part of the communication, but malicious in nature. Packet injection usually involves first sniffing to determine how and when to craft and send packets.

Session Hijacking

Most web applications use a login mechanism that generates a temporarysession token to use for future requests to avoid requiring the user to type a password at every page. An attacker can sniff sensitive traffic to identify the session token for a user and use it to make requests as the user.

The attacker does not need to spoof once he has a session token.

SSL Stripping

Since using HTTPS is a common safeguard against ARP or DNS spoofing, attackers use SSL stripping to intercept packets and alter their HTTPS-based address requests to go to their HTTP equivalent endpoint, forcing the host to make requests to the server unencrypted. Sensitive information can be leaked in plain text.

How to Detect a Man-in-the-Middle Attack

Detecting a Man-in-the-middle attack can be difficult without taking the proper steps. If you aren't actively searching to determine if your communications have been intercepted, a Man-in-the-middle attack can potentially go unnoticed until it's too late. Checking for proper page authentication and implementing some sort of tamper detection are typically the key methods to detect a possible attack, but these procedures might require extra forensic analysis after-the-fact.

VIVA QUESTION:

1. Define eavesdropping and explain how it can compromise the security of communication.
2. What is a dictionary attack, and how does it differ from brute-force attacks?
3. Explain the importance of strong and unique passwords in preventing dictionary attacks.
4. Define a Man-in-the-Middle (MitM)attack and provide examples of how it can be exploited.
5. How can encryption help protect against eavesdropping?
6. Provide examples of encryption methods used to secure communication.
7. Describe the process of a MitM attack on a network.

RESULT:

Thus the Eavesdropping, dictionary attacks and MITM attacks are observed.

Ex.No:7 Date:	Experiment With Sniff Traffic Using Arp Poisoning

AIM:

To study on sniff traffic using ARP poisoning.

Sniffing Attack:

Sniffing Attack in context of network security corresponds to theft or interception of data by capturing the network traffic using a packet sniffer (an application aimed at capturing network packets). When data is transmitted across networks, if the data packets are not encrypted, the data within the network packet can be read using a sniffer. Using a sniffer application, an attacker can analyse the network and gain information to eventually cause the network to crash or to become corrupted,or read the communications happening across the network.

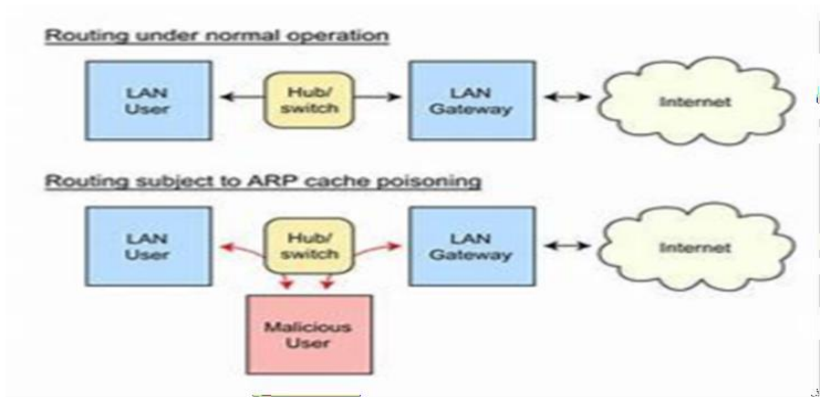
Sniffing attacks can be compared to tapping of phone wires and get to know about the conversation, and for this reason, it is also referred as wiretapping applied to computer networks. Using sniffing tools, attackers can sniff sensitive information from a network, including email (SMTP, POP, IMAP), web (HTTP), FTP (Telnet authentication, FTP Passwords, SMB, NFS) and many more types of network traffic. The packet sniffer usually sniffs the network data without making any modifications in the network's packets. Packet sniffers can just watch, display, and log the traffic, and this information can be accessed by the attacker.

What is ARP Poisoning?

ARP Poisoning consists of abusing the weaknesses in ARP to corrupt the MAC-to-IP mappings of other devices on the network. Security was not a paramount concern when ARP was introduced in 1982, so the designers of the protocol never included authentication mechanisms to validate ARP messages. Any device on the network can answer an ARP request, whether the original message was intended for it or not. For example, if Computer A “asks” for the MAC address of Computer B, an attacker at Computer C can respond and Computer A would accept this response as authentic. This oversight has made a variety of attacks possible. By leveraging easily available tools, a threat actor can “poison” the ARP cache of other hosts on a local network, filling the ARP cache with inaccurate entries.

Sniff traffic using ARP poisoning:

Sniffing traffic using ARP poisoning is a type of cyber attack those abuses weaknesses in the Address Resolution Protocol (ARP) to intercept, modify, or block network traffic between two devices. ARP is a protocol that maps an IP address to a MAC address with in a local network. However, ARP lacks authentication mechanisms, and this is what the attack exploits. The attacker sends fake ARP responses to a specific host on the network, thus linking the attacker’s MAC address to the IP address of another host, such as the network’s gateway. As a result, the target host sends all its network traffic to the attacker instead of the intended host



Toper form this attack, , and use a tool that can send out forged ARP responses, such as Arp spoof or Driftnet. The attacker configures the tool with their MAC address and the IP addresses of the two devices they want to intercept traffic between. The forged responses tell both devices that the correct MAC address for each of them is the attacker's MAC address. As a result, both devices start sending all their network traffic to the attacker's machine, thinking it's the other device they want to communicate with.

The attacker can then do various things with the incorrectly directed traffic. If the attacker chooses to inspect the traffic, they can steal sensitive information, such as passwords, account details, or credit card numbers. If they decide to modify the traffic, they can inject malicious scripts, such as malware, ransomware, or phishing links. Finally, if they choose to block the traffic, they can perform a Denial of Service (DoS) attack, where they completely stop the communication between the two devices.

ARP poisoning is a serious threat to network security, as it can compromise the confidentiality, integrity, and availability of network data. To prevent ARP poisoning attacks, some possible countermeasures are:

Using network monitoring tools, such as Wireshark or Nmap, to detect any anomalies or suspicious activities on the network, such as duplicate MAC addresses, ARP requests, or ARP responses.

VIVA QUESTIONS:

1. What is the Purpose of the Experiment?
2. What is Ethical Considerations
3. How is Safety Measures
4. What is Documentation and Analysis
5. What is Equipment and Software?
6. What is Security Risks and Mitigations
7. What is Data Handling?

RESULT:

Thus, the sniff traffic using ARP poisoning observed.

Ex.No:8 **Date:**	**Demonstrate Intrusion Detection System Using Any Tool**

AIM:

To demonstrate intrusion detection system using any tool.

ALGORITHM:

1. Start the program.
2. Initialize Network Traffic and Intrusion Pattern

    2.1. `networkTraffic`: The string representing network traffic.

    2.2. `intrusionPattern`: The pattern to search for in the network traffic.
3. Compile Pattern:

    3.1. Use `Pattern.compile()` to compile the intrusion pattern into a `Pattern` object.

    3.2. In the given program, the pattern is compiled with the string "suspicious pattern".
4. Match Pattern in Network Traffic.

    4.1. Use the compiled pattern and create a `Matcher` object by invoking `matcher()` on the network traffic string.

    4.2. Use `find()` method on the matcher to search for the pattern in the network traffic.
5. Intrusion Detection:

    5.1. If the pattern is found in the network traffic (i.e., `matcher.find()` returns true), print "Intrusion detected: {intrusionPattern}".

    5.2. - If the pattern is not found, print "No intrusion detected."
6. Stop the program.

PROGRAM:

```
import java.util.regex.*;
public class SimpleIDS{
public static void main(String[]args){
String networkTraffic="Somenetworktrafficwithasuspiciouspattern";StringintrusionPattern= "suspicious
pattern";
Pattern pattern = Pattern.compile(intrusionPattern); Matcher matcher = pattern.matcher(networkTraffic);
if (matcher.find()) {


System.out.println("Intrusiondetected:"+ intrusionPattern);
}else {
System.out.println("Nointrusiondetected.");
}
}
}
```

SAMPLE OUTPUT:

```
Intrusion detected: suspicious pattern
```

VIVA QUESTION:

1. What is Intrusion detection system?
2. Different tools for Intrusion detection system
3. Advantage of Intrusion detection system
4. How to download snort?
5. List the step for install process

RESULT:

Program to demonstrate intrusion detection system was successfully executed.

Ex.No:9 Date:	**Explore Network Monitoring Tool**

AIM:

To explore network monitoring tool.

ALGORITHM:

1. Start the program.
2. Get List of Network Interfaces: Use `JpcapCaptor.getDeviceList()` to obtain an array of available network interfaces.
3. Select a Network Interface:
    3.1. If no network interfaces are found, print an error message and exit.
    3.2. Choose a network interface from the list (for this example, the first one is selected).
4. Open a `JpcapCaptor` on the selected network interface with the following parameters:
    4.1. `selectedDevice`: The chosen network interface.
    4.2. `2000`: Maximum packet size.
    4.3. 3.3 true`: Promiscuous mode (capture all packets on the network)
    4.4. 3.4. `20`: Capture timeout in milliseconds.
5. Capture and Display Packets:
6. Enter an infinite loop to continuously capture and display network packets.
    6.1. Use `captor.getPacket()` to retrieve the next captured packet.
    6.2. If a packet is captured (`packet! = null`), print the packet details.
7. Stop the Program.

PROGRAM:

```
        import jpcap.*;

        import jpcap.packet.*;
        public class NetworkMonitor{
        public static void main(String[] args) throws Exception { NetworkInterface[] devices =
        JpcapCaptor.getDeviceList(); if (devices.length == 0) {
        System.out.println("Nonetworkinterfacefound.Makesureyouhavetherequiredpermissions.");
        return;
        }
        Int selectedDeviceIndex=0;
        NetworkInterface selectedDevice = devices[selectedDeviceIndex]; JpcapCaptor captor =
        JpcapCaptor.openDevice(selectedDevice, 2000, true, 20); System.out.println("Monitoring " +
        selectedDevice.name + "...");
        while (true){
        Packet packet=captor.getPacket();if (packet!=null){System.out.println(packet);
        }
        }
    }
```

SAMPLE OUTPUT:

```
Ethernetpacket(source MAC:00:11:22:33:44:55,destination MAC:AA:BB:CC:DD:EE:FF) IPv4packet
(source IP: 192.168.1.2, destination IP: 8.8.8.8, protocol: TCP)
TCP packet (source port: 12345, destination port: 80, flags: SYN) Payload data: Hello, this is a sample
packet!
```

VIVA QUESTION:

1. How Effectiveness of the Network Monitoring Tool
2. What is Training and Familiarization?
3. What is Data Analysis and Interpretation?
4. What is Scalability and Performance?
5. What is Recommendations for Improvement?

RESULT:

Program for network monitoring tool are explored successfully.

Ex.No:10 **Date:**	**Study To Configure Firewall VPN**

AIM:

To study configure firewall VPN.

Create and Configure the Network

Initialize the network:

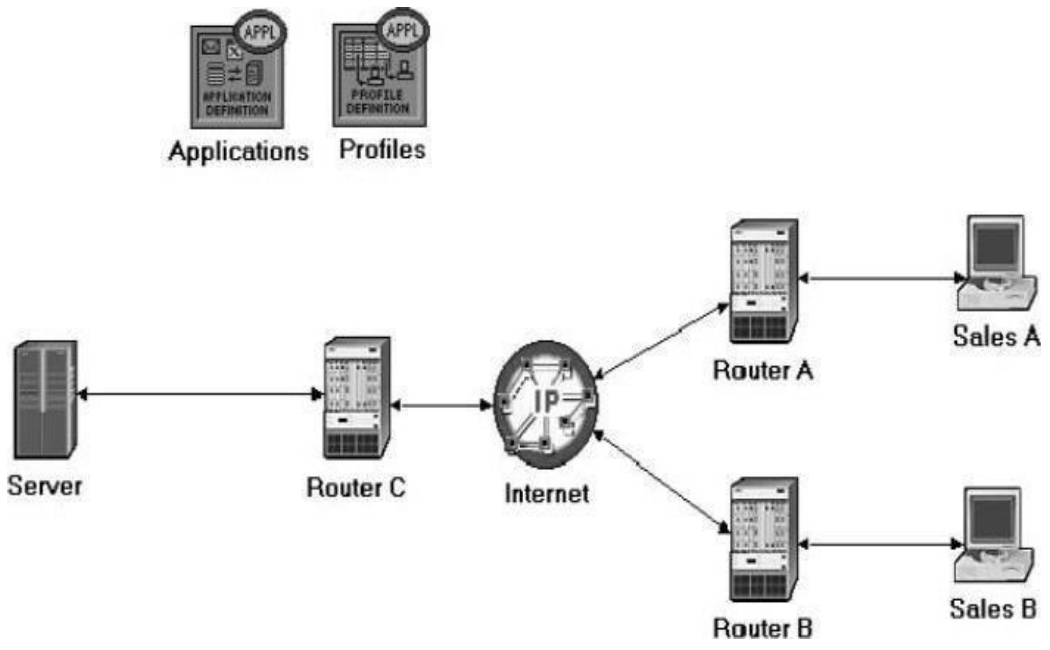
1. Open the Object Palette dialog box by clicking. Make sure that the internet toolbox item is selected from the pull-down menu on the object palette.
2. Add the following objects from the palette to the project workspace (see the following figure for placement): Application Config, Profile Config, an ip32_cloud, one ppp_server, three ethernet4_slip8_gtwy routers, and two pppwkstn hosts.
3. Rename the objects you added and connect them using PPP_DS1links,as shown.

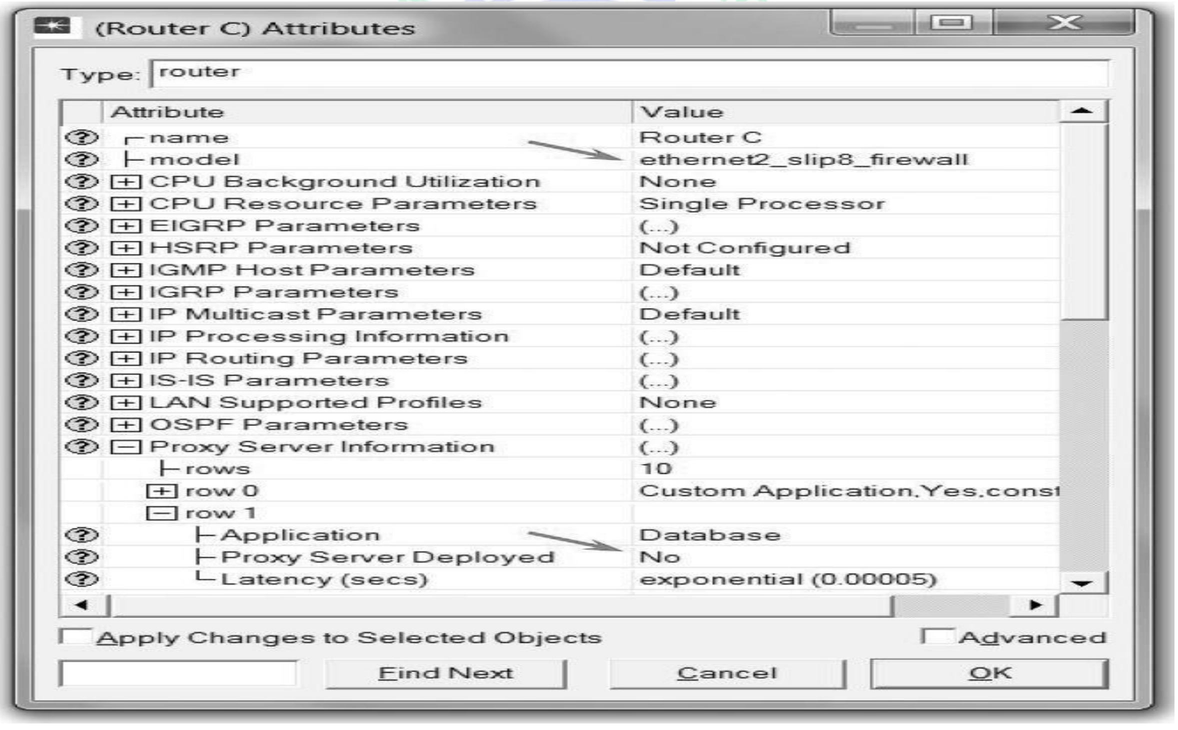
The Firewall Scenario:

In the network we just created, the Sales Person profile allows both sales sites to access applications such as database access, email, and Web browsing from the server (check the Profile Configuration of the Profiles node). Assume that we need to protect the database in the server from external access, including the salespeople. One way to do that is to replace

Router C with a fire wall as follows:

1. SelectDuplicateScenariofromtheScenariosmenuandnameitFirewall.ClickOK.
2. In the new scenario, right-click on Router C· Edit Attributes.
3. Assignethernet2_slip8_firewalltothemodelattribute.
4. Expand the hierarchy of the Proxy Server Information attribute · Expand the row 1, which is for the database application hierarchy · Assign No to the Proxy Server Deployed Attribute as shown. Click OK, and Save your project.





Type: router

Attribute	Value
name	Router C
model	ethernet2_slip8_firewall
⊞ CPU Background Utilization	None
⊞ CPU Resource Parameters	Single Processor
⊞ EIGRP Parameters	(...)
⊞ HSRP Parameters	Not Configured
⊞ IGMP Host Parameters	Default
⊞ IGRP Parameters	(...)
⊞ IP Multicast Parameters	Default
⊞ IP Processing Information	(...)
⊞ IP Routing Parameters	(...)
⊞ IS-IS Parameters	(...)
⊞ LAN Supported Profiles	None
⊞ OSPF Parameters	(...)
⊟ Proxy Server Information	(...)
rows	10
⊞ row 0	Custom Application,Yes,const
⊟ row 1	
Application	Database
Proxy Server Deployed	No
Latency (secs)	exponential (0.00005)

The Firewall VPN Scenario:

In the Firewall scenario, we protected the databases in the server from “any” external access using a firewall router. Assume that we want to allow the people in the Sales A site to have access to the databases in the server. Because the firewall filters all database-related traffic regard less of the source of the traffic, we need to consider the VPN solution. A virtual tunnel can be used by Sales A to senddatabase requests to the server. The firewall will not filter the traffic created by Sales A because the IP packets in the tunnel will be encapsulated inside an IP datagram.

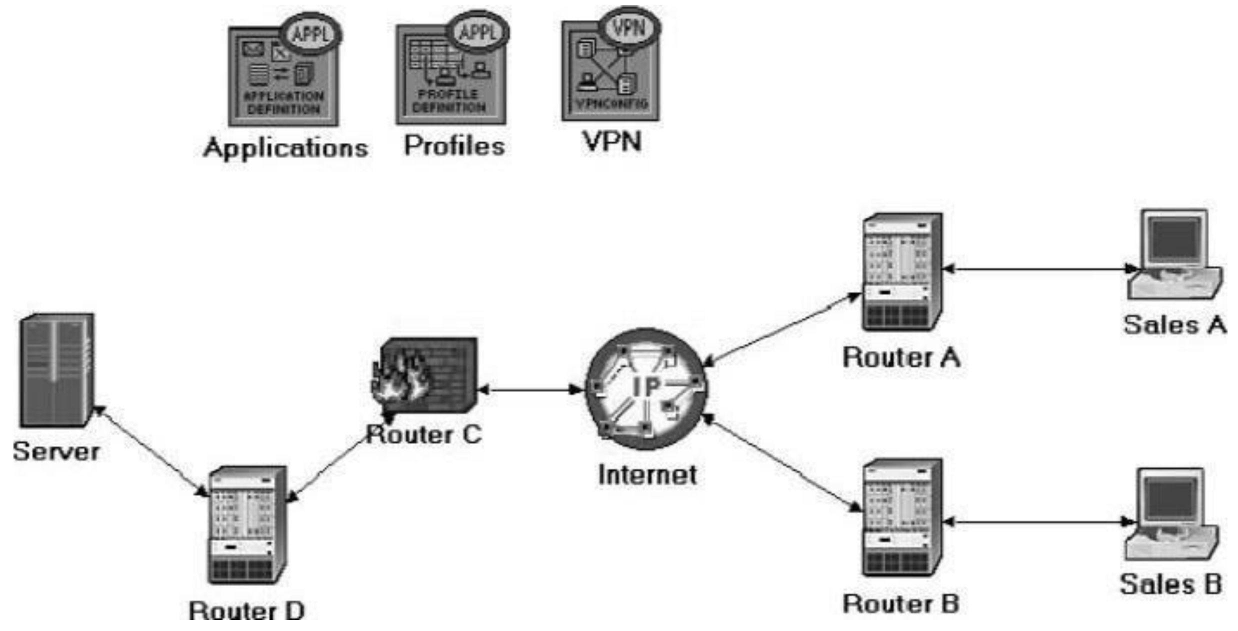
1. While you are in the Firewall scenario, select Duplicate Scenario from the Scenarios menu and give it the name Firewall_VPN · Click OK.
2. Remove the link between Router C and the Server.
3. OpentheObjectPalettedialogboxbyclicking.Makesurethattheinternettoolboxisselected from the pull-down menu on the object palette.

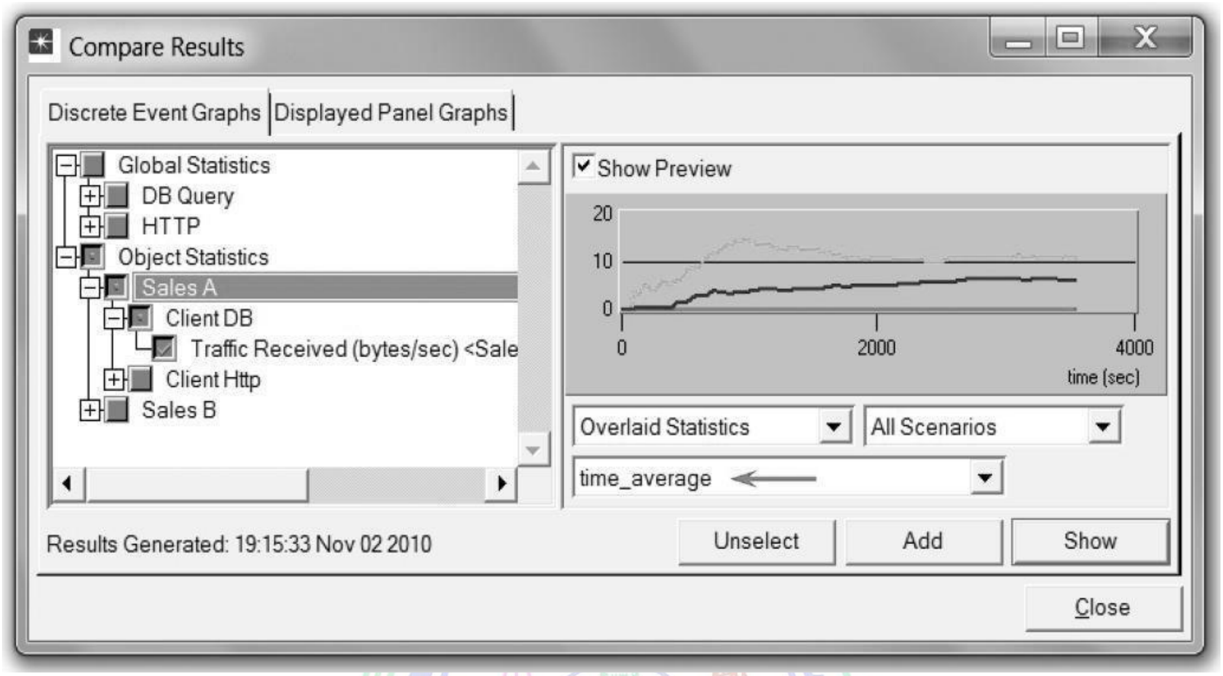
a. Add to the project work space one ethernet4_slip8_gtwy and one IPVPN Config(seethe following figure for placement).

b. From the Object palette, use two PPP_DS1links to connect the new router to the Router C(the firewall) and to the Server, as shown in the following figure.

c. Close the Object Palette dialogbox.

4. Rename the IP VPN Config object to VPN.
5. Rename the new router to Router Das shown in the following fig.





Results Generated: 19:15:33 Nov 02 2010

VIVA QUESTION:

1. What is VPN Protocol and Configuration?
2. What is Firewall Plat form and Version?
3. What is Network Topology?
4. What is Security Policies?
5. What is Permissions and Authorization?
6. What is Monitoring and Logging?

RESULT:

Thus, the configure to firewall VPN observed.

CONTENT BEYOND SYLLABUS

Ex.No:1 Date:	**CAESAR CIPHER**

AIM:

To implement a Caesar cipher substitution technique in Java.

ALGORITHM:

1. Assign the 26 letters in alphabet to the variable named ALPHABET.
2. Convert the plaintext letters into lowercase.
3. To encrypt a plaintext letter, the first set of plaintext letters and slides it to LEFT by the number of positions of the secret shift.
4. The plaintext letter is then encrypted to the ciphertext letter on the sliding ruler underneath.
5. On receiving the ciphertext, the receiver who also knows the secret shift, positions his sliding ruler underneath the ciphertext alphabet and slides it to RIGHT by the agreed shift number, 3 in this case.
6. Then replaces the ciphertext letter by the plaintext letter on the sliding ruler underneath.

PROGRAM:

```
import java.util.Scanner;
public class ceasercipher
{
        public static final String ALPHABET="abcdefghijklmnopqrstuvwxyz";
        public static String encrypt(String plainText,int shiftKey)
        {
                plainText=plainText.toLowerCase();
                String cipherText="";
                for (int i=0; i<plainText.length();i++)
               {
                        int charPosition=ALPHABET.indexOf(plainText.charAt(i));
                        int keyVal=(shiftKey+charPosition)%26;
                        char replaceVal=ALPHABET.charAt(keyVal);
                        cipherText+=replaceVal;
                }
                return cipherText;
        }
        public static String decrypt(String cipherText,int shiftKey)
        {
                cipherText = cipherText.toLowerCase();
                String plainText = "";
                for(int i=0;i<cipherText.length();i++)
               {
                        int charPosition= ALPHABET. indexOf(cipherText. charAt(i));
                        int keyVal=(charPosition-shiftKey)%26;
                        if (keyVal< 0)
                        {




                             keyVal=ALPHABET.length()+keyVal;
                        }
                        char replaceVal=ALPHABET.charAt(keyVal

                        plainText+=replaceVal;


                   }
                   return plainText;
              }
              public static void main(String[] args)
              {
                   Scanner sc=new Scanner(System.in);
                   System.out.println("Enter the Plain text for Encryption: ");
                   String message=new String();
                   message=sc.next();
                   System.out.println("Encrypted message:Cipher Text="+encrypt(message,3));
                   System.out.println("Decrypted message:Plain Text="+decrypt (encrypt(
                   message,3),3));
                   sc.close();
              }
         }
```

SAMPLE OUTPUT:

```
F:\bin>javac ceasercipher.java
F:\bin>java ceasercipher
Enter the Plain text for Encryption:
covid
Encrypted message:Cipher Text=frylg
Decrypted message:Plain Text=covid
```

RESULT:

Thus the Caesar cipher substitution technique was implemented and executed successfully.

Ex.No:2 Date:	**HILL CIPHER**

AIM:

To implement a Hill cipher substitution technique in Java.

ALGORITHM:

1. Obtain a plaintext message to encode in standard English with no spaces.
2. Split the plaintext into group of length three. To fill this, add X at the end.
3. Convert each group of letters with length three into plaintext vectors.
4. Replace each letter by the number corresponding to its position in the alphabet i.e. A=1, B=2, C=3…Z=0.
5. Create the keyword in a 3*3 matrix.
6. Multiply the two matrices to obtain the cipher text of length three.
7. For decryption, convert each entry in the ciphertext vector into its plaintext vector by multiplying the cipher text vector and inverse of a matrix.
8. Thus plain text is obtained fromcorresponding plaintext vector by corresponding position in the alphabet.

PROGRAM:

```
import java.util.Scanner;
import javax.swing.JOptionPane;
public class hillcipher
{
  //the 3x3 key matrix for 3 characters at once
  public static int[][] keymat = new int[][]
  {
          { 1, 2, 1 },
          { 2, 3, 2 },
          { 2, 2, 1 },
  };
  public static int[][] invkeymat = new int[][]
  {
          { -1, 0, 1 },
          { 2, -1, 0 },
          { -2, 2, -1},
  };
  public static String key = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

  public static void main(String[] args)
  {
    String text,outtext ="",outtext1="";
    int ch, n;
      Scanner sc=new Scanner(System.in);
    System.out.println("Enter the Plain text for Encryption: ");
    //String text=new String();
      text=sc.next();

      text = text.toUpperCase();
    text = text.replaceAll("\\s",""); //removing space
```

```
        n = text.length() % 3;
        if(n!=0)
        {
           for(int i = 1; i<= (3-n);i++)
           {
              text+= 'X';
           }
        }
        System.out.println("Padded Text:" + text);
        char[] ptextchars = text.toCharArray();
                for(int i=0;i< text.length(); i+=3)
                      {
                   outtext += encrypt(ptextchars[i],ptextchars[i+1],ptextchars[i+2]);
                }
                               System.out.println("Encypted Message: " + outtext);

                 char[] ptextchars1 = outtext.toCharArray();
           for(int i=0;i< outtext.length(); i+=3)
                      {
                   outtext1 += decrypt(ptextchars1[i],ptextchars1[i+1],ptextchars1[i+2]);
                }
           System.out.println("Decrypted Message: " + outtext1);
  }

  private static String encrypt(char a, char b, char c)
  {
           String ret = "";
           int x,y, z;
           int posa = (int)a - 65;
           int posb = (int)b - 65;
           int posc = (int)c - 65;
           x = posa * keymat[0][0] + posb * keymat[1][0] + posc * keymat[2][0];
           y = posa * keymat[0][1] + posb * keymat[1][1] + posc * keymat[2][1];
           z = posa * keymat[0][2] + posb * keymat[1][2] + posc * keymat[2][2];
           a = key.charAt(x%26);
           b = key.charAt(y%26);
           c = key.charAt(z%26);
           ret = "" + a + b + c;
           return ret;
  }

  private static String decrypt(char a, char b, char c)
  {
           String ret = "";
           int x,y,z;
           int posa = (int)a - 65;
           int posb = (int)b - 65;
           int posc = (int)c - 65;
            x = posa * invkeymat[0][0]+ posb * invkeymat[1][0] + posc *  invkeymat[2][0];
             y = posa * invkeymat[0][1]+ posb * invkeymat[1][1] + posc * invkeymat[2][1];
             z  =  posa  *  invkeymat[0][2]+  posb  *  invkeymat[1][2]  +  posc  *
            invkeymat[2][2]; a = key.charAt((x%26<0)?(26+x%26):(x%26));
            b                                         =
            key.charAt((y%26<0)?(26+y%26):(y%26
            ));                      c                =
            key.charAt((z%26<0)?(26+z%26):(z%26)
            );ret = "" + a + b + c;
            return ret;
        }
        }
```

SAMPLE OUTPUT:

```
F:\bin>javac hillcipher.java

F:\bin>java hillcipher
Enter the Plain text for Encryption:
mothertheresa
Padded Text:MO THERTHERESAXX
Encypted Message:
AAHXIGPPLJEROLR Decrypted
Message: MOTHERTHERESAXX

F:\bin>java hillcipher
Enter the Plain text for Encryption:
hilcipher
Padded Text:HILCIPHER
Encypted Message:
TIIWGHXIGDecrypted
Message: HILCIPHER
```

RESULT:

Thus the Hill cipher substitution technique was implemented and executedsuccessfully.

VIVA QUESTIONS ANSWERS

EXERCISE-1

1. What is Advanced Encryption Standard?

The Advanced Encryption Standard (AES) is an algorithm that uses the same key to encrypt and decrypt protected data. Instead of a single round of encryption, data is put through several rounds of substitution, transposition, and mixing to make it harder to compromise.

2. What is the parameter of AES algorithm?

In the AES algorithm, we need three parameters: input data, secret key, and IV. IV is not used in ECB mode.

3. What is Key Length of AES?

The key length of AES (Advanced Encryption Standard) can be 128, 192, or 256 bits. These different key lengths correspond to the AES-128, AES-192, and AES-256 encryption algorithms, respectively.

4. What is overview of RC6 Algorithm?

RC6 (Rivest Cipher 6) is a symmetric key block cipher designed by Ronald Rivest in 1998. It operates on 32-bit words and has a variable block size and key size. RC6 is a Feistel network-based algorithm that uses modular addition, bitwise rotation, and bitwise XOR operations.

5. What is Key Generation in RC6?

In RC6, key generation involves expanding a given user key into a larger set of round keys. This process typically involves mixing the user key with constants, performing modular arithmetic operations, and iterating through a key schedule algorithm to generate the required round keys for encryption and decryption.

6. What is the Key Length and Security?

The key length in RC6 can vary, but it typically ranges from 128 to 256 bits. The security of RC6 depends on factors such as the key length chosen, the number of rounds used, and the quality of the key schedule algorithm. Generally, RC6 with longer key lengths provides higher security against brute-force attacks.

7. What is the Feistel Network Structure?

A Feistel network is a structure used in many block cipher algorithms, including RC6. It consists of multiple rounds, each of which applies a function to a portion of the input data and then mixes the result with another portion of the input. This process is repeated for several rounds, typically with alternating encryption and decryption operations, to achieve confusion and diffusion in the encryption process.

8. What is the Performance Evaluation in blowfish algorithm?

Performance evaluation in the Blowfish algorithm involves assessing its efficiency in terms of encryption and decryption speed, memory usage, and suitability for various platforms and applications. Factors such as algorithm complexity, round count, key schedule efficiency, and data-dependent operations influence the overall performance of Blowfish. Performance benchmarks and comparisons with other encryption algorithms help in evaluating its effectiveness in different scenarios.

EXERCISE-2

1. What is Asymmetric Key Algorithm?

   An asymmetric key algorithm, also known as public-key cryptography, is a cryptographic algorithm that uses two separate keys for encryption and decryption. These keys are mathematically related but are not identical. One key, known as the public key, is widely distributed and used for encryption, while the other key, known as the private key, is kept secret and used for decryption. Asymmetric key algorithms provide a secure means of communication and digital signatures without requiring the sharing of secret keys.

2. What is Key Pair Generation?

   Key pair generation is the process of creating a matched set of cryptographic keys consisting of a public key and a private key. These keys are generated together in such a way that the private key can decrypt data encrypted with the corresponding public key, and vice versa. Key pair generation typically involves selecting appropriate key lengths and using algorithms such as RSA, Elliptic Curve Cryptography (ECC), or Diffie-Hellman to generate the keys.

3. How to consider Key Length in Asymmetric Key algorithm?

   The key length in an asymmetric key algorithm is an important factor in determining the security level of the encryption scheme. Generally, longer key lengths provide higher security against brute-force attacks but may require more computational resources for encryption and decryption. Key lengths are often chosen based on the desired level of security and the capabilities of the cryptographic algorithm being used. Common key lengths for asymmetric key algorithms range from 1024 bits to 4096 bits, with longer key lengths typically offering greater resistance to cryptographic attacks.

4. How Key Exchange Mechanisms works?

   Key exchange mechanisms allow two parties to securely exchange cryptographic keys over an insecure communication channel. In asymmetric key cryptography, this is typically achieved using a protocol such as Diffie-Hellman key exchange. In this protocol, both parties contribute to the generation of a shared secret without directly exchanging their private keys. The shared secret can then be used to establish a symmetric encryption key for secure communication between the parties.

5. What are Common Asymmetric Algorithms?

Common asymmetric algorithms include:

- RSA (Rivest-Shamir-Adleman)
- Elliptic Curve Cryptography (ECC)
- Diffie-Hellman Key Exchange
- Digital Signature Algorithm (DSA)
- ElGamal Encryption
- Schnorr Signature Algorithm

EXERCISE-3

1. What is digital signature?

A digital signature is a cryptographic technique used to verify the authenticity and integrity of a digital message, document, or transaction. It provides assurance that the message has been created by a particular sender and has not been altered in transit.

2. List two general schemes of Digital Signatures?

Two general schemes of digital signatures are:

a. Public Key Cryptography: In this scheme, a sender uses their private key to digitally sign a message, and the recipient verifies the signature using the sender's public key.

b. Hash-Based Signatures: In this scheme, a hash function is applied to the message to produce a fixed-size digest, which is then signed using the sender's private key. The recipient verifies the signature by hashing the received message and comparing it to the decrypted signature.

3. List the digital signature algorithm function?

Digital signature algorithms typically involve the following functions:

a. Key Generation: Creation of a public-private key pair.

b. Signing: Using the private key to generate a digital signature for a given message.

c. Verification: Using the corresponding public key to verify the authenticity of a received digital signature.

4. An example for digital signature?

Example: Alice wants to digitally sign a contract before sending it to Bob. She generates a digital signature by applying a cryptographic algorithm to the contract using her private key. Bob receives the contract along with the digital signature. He verifies the signature using Alice's public key to ensure that the contract is authentic and unaltered.

5. Advantage of Digital signature?

- Provides authentication: Ensures the identity of the sender.
- Ensures data integrity: Guarantees that the message has not been tampered with.
- Non-repudiation: Prevents the sender from denying their involvement in the message.
- Efficient and convenient for electronic transactions.

6. Disadvantage of Digital signature?

- Requires a secure key management system to protect private keys.
- Complexity in implementation and integration into existing systems.
- Legal and regulatory challenges related to acceptance and enforceability.

7. Real time example for digital signature?

- Signing documents electronically using software such as Adobe Sign or DocuSign.
- Securely authenticating online transactions, such as banking transactions or e-commerce purchases, using digital signatures.
- Digitally signing emails to ensure the sender's identity and the integrity of the message content.

EXERCISE-4

1. What is Wire shark, and what is its primary purpose in network analysis?

   Wireshark is a widely-used network protocol analyzer that allows users to capture and interactively browse the traffic running on a computer network. Its primary purpose in network analysis is to capture and analyze data packets flowing over a network in real-time. Wireshark provides detailed information about network protocols, packet headers, and payload contents, aiding in troubleshooting network issues, monitoring network performance, and identifying security vulnerabilities.

2. Explain the purpose of Tcp dump and how it differs fromWireshark interms of functionality.

   cpdump is a command-line packet analyzer available on Unix-like operating systems. Its primary purpose is similar to Wireshark in that it allows users to capture and analyze network traffic. However, Tcpdump operates solely in the command line interface, making it more lightweight and suitable for use in scripting and automated tasks. Unlike Wireshark, Tcpdump does not provide a graphical user interface and offers limited interactivity for packet analysis.

3. What are the common protocols associated with client-server communication over a network?

Common protocols associated with client-server communication over a network include:

- HTTP (Hypertext Transfer Protocol)
- HTTPS (HTTP Secure)
- FTP (File Transfer Protocol)
- SMTP (Simple Mail Transfer Protocol)
- POP3 (Post Office Protocol version 3)
- IMAP (Internet Message Access Protocol)
- DNS (Domain Name System)

4. Describe the fundamental differences between TCP and UDP in terms of connection-oriented versus connectionless communication.

TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) are two primary transport layer protocols that differ in their approach to communication:

- TCP is connection-oriented, providing reliable, ordered, and error-checked delivery of data packets. It establishes a connection between the sender and receiver before data exchange, manages flow control, and ensures data integrity.

- UDP is connectionless and does not guarantee reliable delivery or order of data packets. It is faster and more lightweight than TCP but lacks features such as acknowledgment, retransmission, and congestion control.

5. What is a UDP datagram, and how does it differ from a TCP segment?

A UDP datagram is a unit of data transmitted over a network using the User Datagram Protocol (UDP). It consists of a header and payload, where the header contains information such as source and destination ports, length, and checksum. UDP datagrams are connectionless and provide unreliable delivery of data, meaning there is no guarantee of packet delivery or order preservation. In contrast, a TCP segment is the unit of data transmitted over a network using the Transmission Control Protocol (TCP). TCP segments are connection-oriented and provide reliable, ordered delivery of data with features such as acknowledgment, retransmission, and flow control.

6. How can filtering be utilized in Wireshark to focus specifically on UDP or TCP traffic?

Filtering in Wireshark can be utilized to focus specifically on UDP or TCP traffic by using display filters. For example, to display only UDP traffic, you can apply a display filter such as “udp” in the filter bar. Similarly, to focus on TCP traffic, you can apply a filter like “tcp”. This allows users to isolate and analyze specific types of network traffic based on protocol

EXERCISE-5

1. What is the primary purpose of SSL (Secure Sockets Layer) in web communication?

   The primary purpose of SSL (Secure Sockets Layer) in web communication is to provide a secure and encrypted connection between a web server and a web browser. SSL ensures that data transmitted between the client and server remains confidential, authenticates the server's identity, and prevents tampering or eavesdropping by malicious entities.

2. Explain the concepts of message integrity and confidentiality in the context of secure communication.

   Message integrity refers to the assurance that data transmitted between parties has not been altered or tampered with during transmission. Confidentiality, on the other hand, ensures that the transmitted data remains private and inaccessible to unauthorized parties.

3. How does SSL ensure message integrity during data transmission?

   SSL ensures message integrity during data transmission through the use of cryptographic hash functions, such as SHA (Secure Hash Algorithm). Each transmitted message is accompanied by a hash value generated by applying the hash function to the message. The recipient recalculates the hash value upon receiving the message and compares it to the transmitted hash value. If the two hash values match, it indicates that the message has not been altered in transit.

4. Describe the encryption process used by SSL to ensure the confidentiality of transmitted data.

   SSL ensures the confidentiality of transmitted data through encryption. When a secure connection is established between a client and server, SSL encrypts the data using symmetric encryption algorithms, such as AES (Advanced Encryption Standard) or 3DES (Triple Data Encryption Standard). This encryption process ensures that even if intercepted, the transmitted data appears as meaningless ciphertext to unauthorized entities.

5. What role do SSL certificates play in establishing the authenticity of a website or server?

   SSL certificates play a crucial role in establishing the authenticity of a website or server. These certificates are issued by trusted Certificate Authorities (CAs) and contain information such as the website's domain name, public key, expiration date, and the CA's digital signature. When a client connects to a website secured with SSL, it receives the SSL certificate from the server. The client then verifies the certificate's authenticity using its trusted root certificate store. If the certificate is valid and issued by a trusted CA, the client establishes a secure connection with the server.

6. Describe the steps involved in setting up SSL for a web server.

   Setting up SSL for a web server involves the following steps:

- Purchase or obtain an SSL certificate from a trusted Certificate Authority.

- Install the SSL certificate on the web server.

- Configure the web server to enable SSL/TLS encryption.

- Optionally, configure the server to redirect HTTP traffic to HTTPS for secure communication.

- Test the SSL configuration to ensure proper functioning.

7. Discuss the role of symmetric and asymmetric encryption in SSL.

   - Asymmetric encryption is used during the initial phase of the SSL handshake to establish a secure connection and exchange session keys. This involves the server's public key being used for encryption and the server's private key for decryption.

   - Symmetric encryption is then used for the actual data transmission, where a shared session key is generated and used for encrypting and decrypting data between the client and server. This approach ensures efficient and secure communication once the initial handshake is complete.

EXERCISE-6

1. Define eavesdropping and explain how it can compromise the security of communication.

Eavesdropping refers to the act of secretly listening to or intercepting communication between two parties without their knowledge or consent. It can compromise the security of communication by allowing unauthorized individuals to access sensitive information such as passwords, personal conversations, financial transactions, or proprietary data. Eavesdropping attacks can occur over various communication channels, including phone calls, emails, instant messaging, or wireless networks, and can lead to identity theft, data breaches, or espionage.

2. What is a dictionary attack, and how does it differ from brute-force attacks?

A dictionary attack is a type of cyberattack that involves systematically trying to guess passwords or encryption keys by testing a large set of known words or phrases against a target system. Unlike brute-force attacks, which try every possible combination of characters, numbers, and symbols, dictionary attacks rely on a predefined list of commonly used passwords or words from dictionaries. This approach makes dictionary attacks faster and more efficient but less exhaustive compared to brute-force attacks.

3. Explain the importance of strong and unique passwords in preventing dictionary attacks.

   Strong and unique passwords are crucial in preventing dictionary attacks because they are less susceptible to being guessed or cracked using commonly used words or phrases. Strong passwords typically consist of a combination of uppercase and lowercase letters, numbers, and special characters, making them more resistant to automated guessing attempts. Additionally, using unique passwords for each account or system reduces the likelihood of multiple accounts being compromised if one password is discovered through a dictionary attack.

4. Define a Man-in-the-Middle (MitM)attack and provide examples of how it can be exploited.

A Man-in-the-Middle (MitM) attack is a cybersecurity attack where an attacker intercepts communication between two parties without their knowledge and may alter or manipulate the communication. Examples of MitM attacks include:

- Intercepting unencrypted Wi-Fi traffic to capture sensitive information such as login credentials or financial data.
- Impersonating a legitimate website or server to intercept and modify data exchanged between the user and the server.
- Hijacking an SSL/TLS session to decrypt and eavesdrop on encrypted communication between a client and server.

5. How can encryption help protect against eavesdropping?

Encryption helps protect against eavesdropping by encoding the transmitted data in such a way that only authorized parties with the appropriate decryption key can decipher it. Even if intercepted, encrypted data appears as random ciphertext to unauthorized individuals, making it unreadable and maintaining the confidentiality and integrity of the communication.

6. Provide examples of encryption methods used to secure communication.

- Symmetric encryption: Uses a single shared key for both encryption and decryption, such as AES (Advanced Encryption Standard).
- Asymmetric encryption: Utilizes a pair of public and private keys for encryption and decryption, such as RSA (Rivest-Shamir-Adleman).
- SSL/TLS encryption: Provides secure communication over the internet by encrypting data transmitted between a client and server using cryptographic protocols like HTTPS.

7. Describe the process of a MitM attack on a network.

The process of a Man-in-the-Middle (MitM) attack on a network typically involves:

- Intercepting communication between two parties without their knowledge, often by exploiting vulnerabilities in network protocols or devices.
- Monitoring and possibly altering the communication flow to eavesdrop on sensitive information or inject malicious content.
- Impersonating one or both parties to establish a false sense of trust and manipulate the communication for malicious purposes.
- Redirecting traffic to malicious servers or websites to steal credentials, financial information, or other sensitive data.

EXERCISE-7

1. What is the Purpose of the Experiment?

The purpose of the experiment is to investigate a specific hypothesis or research question, gather data through observation or experimentation, and analyze the results to draw conclusions and contribute to scientific knowledge or understanding.

2. What is Ethical Considerations

Ethical considerations involve ensuring that the experiment is conducted in an ethical manner, taking into account the rights and well-being of participants, respecting confidentiality and privacy, obtaining informed consent when necessary, and adhering to relevant ethical guidelines and regulations.

3. How is Safety Measures

   Safety measures refer to precautions taken to ensure the safety of participants, researchers, and others involved in the experiment. This may include providing proper training and supervision, using appropriate protective gear and equipment, following safety protocols, and minimizing risks of injury or harm.

4. What is Documentation and Analysis

   Documentation and analysis involve keeping detailed records of the experiment methodology, procedures, observations, and results. This documentation ensures transparency, reproducibility, and accountability in scientific research. Analysis involves interpreting the collected data, identifying patterns or trends, and drawing conclusions based on statistical or qualitative analysis methods.

5. What is Equipment and Software?

Equipment and software refer to the tools, instruments, and technology used to conduct the experiment and analyze the data. This may include laboratory equipment, sensors, measurement devices, computers, software programs for data analysis, and specialized tools or materials specific to the experiment's objectives.

6. What is Security Risks and Mitigations

Security risks and mitigations involve identifying potential threats to the experiment's integrity, data confidentiality, or participant safety, and implementing measures to mitigate those risks. This may include securing physical facilities, protecting sensitive data from unauthorized access, using encryption or authentication mechanisms, and implementing cybersecurity measures to prevent data breaches or tampering.

7. What is Data Handling?

Data handling involves the proper collection, storage, transmission, and analysis of data generated during the experiment. This includes ensuring data accuracy, reliability, and integrity, maintaining data confidentiality and privacy, following data management protocols and best practices, and complying with relevant legal and regulatory requirements for data handling and storage.

EXERCISE-8

1. What is Intrusion detection system?

   An Intrusion Detection System (IDS) is a security tool or software application designed to monitor network or system activities for malicious activities or policy violations. It analyzes incoming network traffic, system logs, or event data to identify suspicious behavior or patterns indicative of unauthorized access, misuse, or attacks on the network or systems.

2. Different tools for Intrusion detection system

   Different tools for Intrusion Detection System include:

   a. Snort

   b. Suricata

   c. Bro (Zeek)

   d. OSSEC

   e. Snort-based commercial products (e.g., Cisco Firepower)

   f. Intrusion Detection and Prevention Systems (IDPS) such as Snort inline mode or Suricata with IPS capabilities

3. Advantage of Intrusion detection system

   a. Early detection of security incidents and potential threats.

   b. Enhanced network and system security by monitoring and analyzing network traffic.

   c. Improved incident response capabilities by providing alerts and notifications for suspicious activities.

   d. Compliance with security standards and regulations by maintaining a proactive security posture.

   e. Reduction in security breaches and data loss by identifying and mitigating security vulnerabilities.

4. How to download snort?

   To download Snort, you can visit the official Snort website or use package managers available on your operating system. Snort is open-source software, and its source code can be obtained from the Snort website or downloaded from GitHub.

5. List the step for install process

. Steps for installing Snort may vary depending on the operating system and distribution. However, a general process for installing Snort on a Linux system may include the following steps:

a. Download the Snort source code or package from the official website or repository.

b. Extract the downloaded files to a directory on your system.

c. Navigate to the extracted directory and run the configuration script (./configure).

d. Optionally, customize the configuration options based on your requirements.

e. Compile the Snort source code by running the make command.

f. Install Snort by running the make install command with appropriate permissions (e.g., sudo make install).

g. Optionally, configure Snort rules and configuration files as needed for your environment.

h. Start the Snort service or daemon using the appropriate command or init script.

i. Verify that Snort is running correctly and monitoring network traffic by checking log files or using command-line tools.

EXERCISE-9

1. How Effectiveness of the Network Monitoring Tool

The effectiveness of a network monitoring tool refers to its ability to accurately and efficiently detect, analyze, and respond to network issues, anomalies, or security threats. Key factors that contribute to the effectiveness of a network monitoring tool include its capability to capture and analyze network traffic, generate meaningful alerts or notifications, provide real-time visibility into network performance, and support proactive monitoring and troubleshooting activities. Additionally, the tool's ease of use, scalability, integration with other systems, and customization options also influence its effectiveness in meeting the organization's network monitoring requirements.

2. What is Training and Familiarization?

Training and familiarization involve providing users with the knowledge, skills, and understanding necessary to effectively utilize a network monitoring tool. This includes training sessions, documentation, tutorials, and hands-on experience to familiarize users with the tool's features, functionalities, and best practices for monitoring and managing network infrastructure. Training and familiarization help ensure that users can maximize the benefits of the network monitoring tool, accurately interpret monitoring data, and respond appropriately to network events or incidents.

3. What is Data Analysis and Interpretation?

Data analysis and interpretation involve analyzing the data collected by the network monitoring tool to gain insights into network performance, security, and operational trends. This includes identifying patterns, anomalies, or deviations from normal behavior, correlating data from multiple sources, and interpreting the findings to make informed decisions or take appropriate actions. Data analysis and interpretation help organizations understand their network environment, diagnose issues, optimize performance, and enhance security posture.

4. What is Scalability and Performance?

Scalability refers to the ability of a network monitoring tool to handle increasing volumes of network traffic, devices, and data without compromising performance or functionality. A scalable network monitoring tool can accommodate growth in network infrastructure and data volume while maintaining reliability, responsiveness, and efficiency. Performance, on the other hand, relates to the speed, responsiveness, and resource utilization of the network monitoring tool under normal operating conditions. A high-performance network monitoring tool can efficiently process and analyze network data in real-time, providing timely insights and alerts to users.

5. What is Recommendations for Improvement?

Recommendations for improvement involve identifying areas where the network monitoring tool can be enhanced or optimized to better meet the organization's needs and objectives. This may include suggestions for improving monitoring coverage, enhancing alerting mechanisms, optimizing data collection and storage, integrating with other systems or tools, enhancing user interface and usability, or addressing scalability and performance limitations. Recommendations for improvement aim to continuously enhance the effectiveness and efficiency of the network monitoring tool in supporting the organization's network infrastructure and operations.

EXERCISE-10

1. What is VPN Protocol and Configuration?

- A VPN (Virtual Private Network) protocol is a set of rules and procedures used to establish a secure and encrypted connection over a public network, such as the internet. VPN protocols determine how data is transmitted, encrypted, and authenticated between the VPN client and server. Common VPN protocols include OpenVPN, IPSec (Internet Protocol Security), L2TP/IPSec (Layer 2 Tunneling Protocol with IPSec), PPTP (Point-to-Point Tunneling Protocol), and SSTP (Secure Socket Tunneling Protocol).

- VPN configuration involves setting up and configuring VPN servers, clients, and connections. This includes configuring authentication methods, encryption settings, IP addressing, routing, and other parameters to establish secure and reliable VPN connections between users and networks.

2. What is Firewall Plat form and Version?

- A firewall platform refers to the hardware or software-based system used to enforce network security policies by controlling incoming and outgoing network traffic based on predefined rules or criteria. Common firewall platforms include hardware firewalls, such as those provided by Cisco, Palo Alto Networks, Juniper Networks, and software firewalls like iptables (Linux), Windows Firewall (Microsoft Windows), and pfSense (open-source firewall distribution).

- The firewall version refers to the specific release or version of the firewall software or firmware installed on the firewall platform. Keeping firewall software up-to-date with the latest version is essential for ensuring optimal security and protection against emerging threats and vulnerabilities.

3. What is Network Topology?

   - Network topology refers to the physical or logical layout or structure of interconnected devices, nodes, and links within a computer network. It describes how devices are connected and the paths through which data flows within the network. Common network topologies include bus, ring, star, mesh, and hybrid topologies. Network topology plays a crucial role in determining network performance, scalability, fault tolerance, and security.

4. What is Security Policies?

   - Security policies are a set of rules, guidelines, and procedures established by an organization to define and enforce security requirements, controls, and practices. Security policies address various aspects of information security, including access control, data protection, authentication, authorization, encryption, incident response, and compliance. Security policies help ensure the confidentiality, integrity, and availability of information assets and resources within the organization.

6. What is Permissions and Authorization?

   - Permissions refer to the access rights or privileges granted to users, groups, or entities to perform specific actions or operations on resources within a computer network or system. Permissions dictate what actions users are allowed or denied to perform, such as reading, writing, executing, modifying, or deleting files, folders, or network services.

   - Authorization, on the other hand, is the process of determining whether a user or entity has the necessary permissions to access a particular resource or perform a specific action. Authorization mechanisms verify user identities, authenticate credentials, and enforce security policies to grant or deny access based on predefined rules or criteria.

7. What is Monitoring and Logging?

   - Monitoring involves continuously observing and analyzing network traffic, system activities, and security events to detect anomalies, threats, or unauthorized activities. Monitoring tools and techniques collect and analyze data from various sources, such as network devices, servers, applications, and security systems, to provide real-time visibility into the network environment and identify potential security incidents.

   - Logging refers to the process of recording and storing detailed information, events, and activities generated by network devices, systems, applications, and users. Logs contain valuable data for troubleshooting, auditing, compliance, and forensic analysis. Logging systems collect and store log entries in centralized repositories or log files, which can be analyzed, searched, and reviewed for security and operational purposes.