Department of Computer Science

Gujarat University

Certificate

Roll No: 13 Seat No: _____

This is to certify that Mr./Ms. _____Antra koul_____ student of MCA Semester – V has duly completed his/her term work for the semester ending in December 2020, in the subject of _____Network Security_____ towards partial fulfillment of his/her Degree of Masters in Computer Applications.

11-12-2020
Date of Submission

Internal Faculty

Head of Department

Department Of Computer Science
Rollwala Computer Centre
Gujarat University

MCA –5

Subject: - Network Security

Name: - Antra koul

Roll No.: - 13 **Exam Seat No.: -**

Sr. No.	Contents	Pg. No	Date	Signature
1	Assignment 1	1-3	12-Aug-2020	
2	Assignment 2	3-6	15-Sept-2020	
3	Assignment 3	7-8	10-Nov-2020	
4	Assignment 4	9-11	24-Nov-2020	

Assignment 1

1. List all symmetric key algorithms.

DES :-
Data Encryption Standard (DES) It is a block cipher published by NIST. It is an implementation of Feistal Cipher. It uses 16 round feestal structure. It has an effective key length of 56 bits

Triple DES:-
Triple DES is an encryption technique which uses 3 instances of DES on same plain text. It uses three different types of key choosing techniques in first all used keys are different and in second 2 keys are same and one is different. and in third all keys are same.

AES!-
Advanced Encryption Standard (AES) is atleast 6 times faster than Triple DES. It is iterative rather than Feistal Cipher. It computes all permutation on bytes rather than bits. It treats the 128 bits of a plaintext block as 16 bytes.

2. List all asymmetric key algorithms.

RSA:- It is an asymmetric key algorithm. and is considered as most secure in encryption. It has following features!-

1. It is a popular exponentiation in a finite field over integers including prime numbers.
2. The integers used by this method are sufficiently large making it difficult to solve

- There are two sets of keys used : private & public

Diffie Hellman Key Exchange:-
It is a method of securely exchanging cryptographic keys over a public channel. It is a method of digital encryption, that uses number raised to the power to produces decryption keys on the basis of components that are never directly. trans

3. List the algorithm for message digest.

1. MD5:-
It is a one way cryptographic algorithms. Its family comprises of hash functions. MD2, MD4, MD5. and MD6. It have been widely used in the software worldwide. For example Servers often provide a pre computed MD5 checksum of all files

2- Secure Hash Function (SHA)
The original version of SHA is SHA 0, a 160 bit Hash function was published by the NIST. SHA1 is the most widely used of the existing algorithms. It is employed in several widely used applications and protocols. including Secure Socket Layer (SSL) security.

Assignment 2:-

Discuss briefly

a) PII (Personally Identifiable Information)

- PII identifiable is an data that could potentially identify a specific individual.
- Any information that can be used to distinguish one person from another and can be used for deanonymizing previously anonymous data can be considered as PII.

b) Us Privacy Act of 1974

- It establishes a code of affair information practices that govern the collection, maintenance and use and dissemination of information about individuals that is maintained in systems of records by federal agencies.

c) FOIA:-

It generally provides any person with the statutory right, enforceable in court to obtain access to Government. Information in executive branch agency records.

This right to access is limited when such information is protected from disclosure by one of FOIA's nine statutory exemptions.

d) FERPA:-

The Family Educational Rights and Privacy Act (FERPA) is a federal law that affords parents the right to have access to their children's education records, the right to seek to have

the records amended and the right to have some control over the disclosure of personally identifiable information from the education record.

e) CFAA :-
It stands for Computer Fraud and Abuse Act. It imposes criminal and civil liability for unauthorized access or damage to a protected computer. This law reaches every computer connected to the internet and non networked computers used by the US govt & financial institutions.

f) COPAA
Childrens Online Privacy Protection Rule imposes certain requirements on operators of websites or online services directed to children under age of 13 years and on operators of other web-sites or online services that have actual knowledge that they are collecting personal information online from child under 13 yrs.

g) VPPA :-
Video Privacy Protection Act was created to prevent what it refers to as "wrongful" disclosure of video tape rental or sale records to cover items such as media games and the future DVD. It makes any video tape service provider that discloses rental information outside the ordinary course of ~~bu~~ business liable for up to $2500 in actual damages.

h) HIPAA:-
The Health Insurance Portability and Accountability Act required the Secretary of the US department of Health and Services to develop regulations protecting the privacy and Security of certain health information.

i) GLBA:-
The Gramm Leach Bliley Act is also known as Financial Modernization Act of 1999. It requires law that requires financial institutions to explain how they share and protect their customer private information.

j) PCI DSS:-
The Payment Card Industry Data Security Standard (PCI DSS) is a set of security standards designed to ensure all Companies that accept, process, store or transit credit card information maintain a secure environment

k) FCRA:-
The Foreign Contribution Regulation Act lasts certain obligation on banks in regard to acceptance of foreign inward remittances for onward credit to the accounts of association in India.
These are applicable to all the scheduled commercial Banks.

b) FACTA:- The Fair and Accurate Credit Transactions Act (FACT Act) was enacted in 2003 and amends the Fair Credit Reporting Act (FCRA) a federal law that regulates, in part who is permitted to access your consumer report information. and how it can be used. The Act entitles consumers to obtain one free copy of their consumer files from certain customer reporting agencies during each 12 month period.

Assignment :- 3

State the full form for:-

1. RADIUS :-
   Remote Authentication Dial-in User Services

2. TACAS:-
   Terminal Access Controllee Access Control System.

3. L2TP & PPTP.
   Layer 2 Tunneling Protocol
   Point-to-Point Tunneling Protocol.

4. PPP
   Point to Point Protocol.

5. EAP
   Extensible Authentication Protocol.

6. CHAP
   Challenge - Handshake Authentication Protocol.

7. NTLM:
   New Technology LAN Manager.

8. PAP
   Password Authentication Protocol.

9. SSH
   Secure Shell.

10. LDAP

Lightweight Directory Access Protocol.

Assignment 4.

1. List the name of software for.

1. Firewall.
ZoneAlarm, PeerBlock, Private Firewall, Norton, AVS Firewall, Tinywall, System Mechanic Ultimate Defense, OpenDNS Home, Glasswire

2. Intrusion Detection and Prevention.
McAfee Network Security Platform, Trend Micro Tipping Point, Hillstone NIPS, MazeTrace Enterprise Immune System.

3. Antivirus:-
PCProtect, SCANGUARD, NORTON, MacKEEPER

4. Packet Sniffing
Wireshark, Kismet, Fiddler, Tcpdump, Ethereage

2. State any 10 security software used by ethical hackers.

1. Wireshark
2. Metasploit
3 Nessus
4 Aircrack
5 Snort
6. Cain and Abel
7. Back Track
8 10. Netcat
9. tcp Dump
10. John the Ripper.

3. What is the role of CERT?

Reactive:-

1. Provides a single point of contact for reporting local problems.
2. Assist the organisational constituency and general computing in preventing and handling computer security incidents.
3. Share information and lessons with CERT/CC other CERT's.
4. Incident Response
5. Provide 24x7 security service.
6. Offer recovery procedures.
7. Artifact Analysis.
8. Incident tracing.

Proactive:-

1. Issues security guidelines
2. Vulnerability analysis and response
3. Risk Analysis
4. Collaboration with vendors
5. National Repo. of and a referral policy for cyber instructions.
6. Profiling hackers
7. Conduct training
8. Interact with vendors and others at large to
9. investigate and provide solutions for incidents

4. List OSA OWASP Top ten.

1. Injection
2. Broken Authentication
3. Sensitive Data Exposure
4. XML External Entities (XXE)
6. Broken Access Control
6. Security Misconfiguration
7. Cross Site Scripting
8. Insecure Deserialization
9. Using components with known Vulnerabilities
10. Insufficient Logging and Monitoring.

D E P A R T M E N T O F C O M P U T E R S C I E N C E
R O L L W A L A C O M P U T E R C E N T R E
G U J A R A T U N I V E R S I T Y
M.C.A. –5

R O L L NO : 13
N A M E : Antra koul
S U B J E C T :Network Security

NO.	TITLE	PAGE NO.	DATE	SIGN
1.	Caesar Cipher	**1**	**13-Aug-2020**	
2	DES	**3**	**15-Aug-2020**	
3	MD5	**7**	**23-Aug-2020**	
4	One time pad	**8**	**30-Aug-2020**	
5	RSA	**12**	**2-Sept-2020**	
6	SHA1	**15**	**10-Sept-2020**	
7	Substitution cipher	**18**	**14-Sept-2020**	
8	Substitution Client server	**21**	**17-Sept-2020**	
9	Transposition cipher	**31**	**20-Sept-2020**	
10	AES	**35**	**25-Sept-2020**	
11.	Implement authentication Service. The sender sends password in encrypted format to the receiver, the receiver checks the password (after decrypting) in its database/array and replies back as success or failure.(Keys are already shared)	**38**	**15-Oct-2020**	
12	Implement authentication Service. The sender sends password in encrypted format to the receiver, the receiver checks the password (after decrypting and applying hash) in its database/array and replies back as success or failure. (Note: Here the password is stored as hash in database).	**44**	**20-Oct-2020**	
13	Implement a firewall that behaves like forwarder. It does not forward the packet that contains the word "terrorist".	**52**	**12-Nov-2020**	

D E P A R T M E N T O F C O M P U T E R S C I E N C E
R O L L W A L A C O M P U T E R C E N T R E
G U J A R A T U N I V E R S I T Y
M.C.A. –5

R O L L NO : 13
N A M E : Antra koul
S U B J E C T :Network Security

14	Implement NAT functionality. The NAT works like forwarder, that will forward to appropriate receiver.	**61**	**15-Nov-2020**	
15	**Key Distribution** Implement a program to demonstrate the functioning of a KDC. There are three entities: sender, receiver and KDC. Assume that Sender and Receiver have already established their own individual permanent secret keys with KDC. The sender requests the KDC to issue a session key to communicate with receiver. The KDC is supposed to give session key information to sender in a secure way. The same session key is also to be communicated to the receiver securely. Use a suitable protocol to achieve the above functionality.	**72**	**27-Nov-2020**	

Assignment-1

1.”Caesar Cipher”

```
import java.util.Scanner;
class CaesarCypher{
	public static void main(String args[]){
		Scanner in = new Scanner(System.in);

		String plain_text;

		System.out.print("\nEnter text to be encrypted:");
		plain_text=in.nextLine();

		String encrypted_Text=encryptText(plain_text);
		System.out.print("\nEncrypted Text:"+encrypted_Text);

		System.out.print("\nDecrypted Text:"+decryptText(encrypted_Text));
	}
	public static String encryptText(String plain_text)
	{
		String encrypted_Text="";
		for(int i=0;i<plain_text.length();i++){
			if(((int)plain_text.charAt(i)) == 32)
```

```
                encrypted_Text=encrypted_Text+" ";
            else{
                int n=(((int)plain_text.charAt(i))-65-3);
                if(n<0)
                    n=26-Math.abs(n);
                encrypted_Text=encrypted_Text+((char)(n+65));
            }
        }
        return encrypted_Text;
}
public static String decryptText(String plain_text)
{
        String encrypted_Text="";
        for(int i=0;i<plain_text.length();i++){
            if(((int)plain_text.charAt(i)) == 32)
                encrypted_Text=encrypted_Text+" ";
            else{
                int n=(((int)plain_text.charAt(i))-65+3);
                if(n>25)
                    n=Math.abs(n)-26;
                encrypted_Text=encrypted_Text+((char)(n+65));
            }
```

```
            }
            return encrypted_Text;
        }
}
/*Output:
Enter text to be encrypted:THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

Encrypted Text:QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD
Decrypted Text:THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG*/
```

2.”DES”

```
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;

import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;
```

```
import java.util.Base64;


public class DESString {


   private static Cipher ecipher;
   private static Cipher dcipher;
   private static SecretKey key;
   public static void main(String[] args) {
                try {
                        // generate secret key using DES algorithm
        key = KeyGenerator.getInstance("DES").generateKey();
                        ecipher = Cipher.getInstance("DES");
        dcipher = Cipher.getInstance("DES");
                        // initialize the ciphers with the given key
                        ecipher.init(Cipher.ENCRYPT_MODE, key);
                        dcipher.init(Cipher.DECRYPT_MODE, key);
                        String original="the quick brown fox jumps over a lazy dog!!";
                        String encrypted = encrypt(original);
                        System.out.print("Original text:"+original);
                        System.out.println("\nEncrypted: " + encrypted);
                        String decrypted = decrypt(encrypted);
                        System.out.println("\nDecrypted: " + decrypted);
```

```
                }
        catch (Exception e) {
            System.out.println("Exception occured!" + e.getMessage());
            return;
        }
 }
        public static String encrypt(String str) {
        try {
                        // encode the string into a sequence of bytes using the named
charset
                        byte[] string_bytes= str.getBytes("UTF8");
                        // storing the result into a new byte array.
                        byte[] enc = ecipher.doFinal(string_bytes);
                        // encode to base64
                        enc = Base64.getEncoder().encode(enc);
                        return new String(enc);
        }
        catch (Exception e) {
                e.printStackTrace();
        }
        return null;
}
```

```
 public static String decrypt(String str) {
        try {
        // decode with base64 to get bytes
        byte[] dec = Base64.getDecoder().decode(str.getBytes());
        byte[] utf8 = dcipher.doFinal(dec);
        return new String(utf8, "UTF8");
        }
        catch (Exception e) {
                e.printStackTrace();
        }
                return null;
        }
}
/*Output:
D:\Semester-5-\Network security\Assignment 1>javac DESString.java

D:\Semester-5-\Network security\Assignment 1>java DESString
Original text:the quick brown fox jumps over a lazy dog!!
Encrypted:
OjizH7l/ytY90hUHfnmBG3GNDavJKuke2AZBmq/6MRY1PozpF6MxpZij9fpEDJFc

Decrypted: the quick brown fox jumps over a lazy dog!!*/
```

3.”MD5”

```
import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
public class MD5 {
	public static void main(String args[]) throws NoSuchAlgorithmException  {
		String message = "Ankita";
		System.out.println("HashCode:" + getMessagedigest(message));
		}

	public static String getMessagedigest(String message)
	{
	try {
	 MessageDigest md = MessageDigest.getInstance("MD5");
	 byte[] messageDigest = md.digest(message.getBytes());
	 BigInteger no = new BigInteger(1, messageDigest);
	 String hashtext = no.toString(16);
			while (hashtext.length() < 32) {
				hashtext = "0" + hashtext;
			}
			return hashtext;
```

```
            }
        catch (NoSuchAlgorithmException e) {
                    throw new RuntimeException(e);
            }
        }
}
/*Output
D:\Semester-5-\Network security\Assignment 1>javac MD5.java

D:\Semester-5-\Network security\Assignment 1>java MD5
HashCode:d265e24340d83487e7740d67927e4003*/
```

4.”One Time Pad”

```
import java.util.Scanner;
import java.util.Random;

class OneTimePad{
        public static void main(String args[]){
            Scanner in = new Scanner(System.in);

            String plain_text;
```

```
        StringBuffer key=new StringBuffer();
        StringBuffer encrypted_text=new StringBuffer();
        StringBuffer decrypted_text=new StringBuffer();
        System.out.print("\nEnter plain text:");
        plain_text=in.nextLine();

        key=generateKey(plain_text);
        System.out.print("\nKey:"+key);

        encrypted_text=encryptText(plain_text,key);

        System.out.print("\nEncrypted text is :"+encrypted_text);

        decrypted_text=decryptText(encrypted_text,key);

        System.out.print("\nDecrypted text is :"+decrypted_text);

}
public static StringBuffer generateKey(String plain_text)
{
        Random random = new Random();
```

```
            StringBuffer key=new StringBuffer("");
            for(int i=0;i<plain_text.length();i++)
            {
                char a=(char)(random.nextInt(128));
                key.append((char)(a));
            }
            return key;
        }
        public static StringBuffer encryptText(String plain_text,StringBuffer key)
        {
            StringBuffer encrypted_text=new StringBuffer("");
            for(int i=0;i<plain_text.length();i++)
            {

        encrypted_text.append((char)(((int)(plain_text.charAt(i)))^((int)(key.charAt(
i)))));
            }
            return encrypted_text;
        }
        public static StringBuffer decryptText(StringBuffer
encrypted_text,StringBuffer key)
        {
            StringBuffer decrypted_text=new StringBuffer("");
```

```
            for(int i=0;i<encrypted_text.length();i++)
            {

        decrypted_text.append((char)(((int)(encrypted_text.charAt(i)))^((int)(key.charAt(i)))));
            }
            return decrypted_text;
        }
}
/*Output:
C:\Users\user\Desktop\Sem 5\Semester-5-\Network security>javac OneTimePad.java

C:\Users\user\Desktop\Sem 5\Semester-5-\Network security>java OneTimePad

Enter plain text:Antra koul

Key:><□"~
```

5.”RSA “

```
import java.security.KeyPair;
import java.security.KeyPairGenerator;
```

```
import java.security.PrivateKey;
import java.security.PublicKey;
import java.util.Base64;

import javax.crypto.Cipher;

public class RSAEncryption
{
   static String plainText = "My name is Antra!";

   public static void main(String[] args) throws Exception
   {
      // Get an instance of the RSA key generator
      KeyPairGenerator keyPairGenerator = KeyPairGenerator.getInstance("RSA");
      keyPairGenerator.initialize(4096);

      // Generate the KeyPair
      KeyPair keyPair = keyPairGenerator.generateKeyPair();

      // Get the public and private key
      PublicKey publicKey = keyPair.getPublic();
      PrivateKey privateKey = keyPair.getPrivate();
```

```
        System.out.println("Original Text  : "+plainText);

        // Encryption
        byte[] cipherTextArray = encrypt(plainText, publicKey);
        String encryptedText =
Base64.getEncoder().encodeToString(cipherTextArray);
        System.out.println("Encrypted Text : "+encryptedText);

        // Decryption
        String decryptedText = decrypt(cipherTextArray, privateKey);
        System.out.println("DeCrypted Text : "+decryptedText);
    }

    public static byte[] encrypt (String plainText,PublicKey publicKey ) throws
Exception
    {
        //Get Cipher Instance RSA With ECB Mode and OAEPWITHSHA-
512ANDMGF1PADDING Padding
        Cipher cipher = Cipher.getInstance("RSA/ECB/OAEPWITHSHA-
512ANDMGF1PADDING");

        //Initialize Cipher for ENCRYPT_MODE
```

```
        cipher.init(Cipher.ENCRYPT_MODE, publicKey);

        //Perform Encryption
        byte[] cipherText = cipher.doFinal(plainText.getBytes()) ;

        return cipherText;
    }

    public static String decrypt (byte[] cipherTextArray, PrivateKey privateKey)
throws Exception
    {
        //Cipher Instance RSA With ECB Mode and OAEPWITHSHA-
512ANDMGF1PADDING
        Cipher cipher = Cipher.getInstance("RSA/ECB/OAEPWITHSHA-
512ANDMGF1PADDING");

        //Initialize Cipher for DECRYPT_MODE
        cipher.init(Cipher.DECRYPT_MODE, privateKey);

        //Perform Decryption
        byte[] decryptedTextArray = cipher.doFinal(cipherTextArray);

        return new String(decryptedTextArray);
```

```
    }
}
/*Output:
D:\Semester-5-\Network security\Assignment 1>javac RSAEncryption.java


D:\Semester-5-\Network security\Assignment 1>java RSAEncryption
Original Text  : My name is Antra!
Encrypted Text :
jst++XKd8nZDkB/bbhLNfDN7IRHr63SBEvHp7hIxvUw0ACwq6Ckcv4zyh0x7hAmA72
/SpdnkTCq8v+aV8CVV+OhhoUatPU6JxMg1yIHuoLNvzfLD5fGGKIrs8zJyWgXg24nzs
cMG3e1KNdo8HmDreF+iTvrLZh33JMgHhQ68KHns2/aHQkBVQYipEf2eqotJhsGZW
7uBEHO2LcHyVCtPpwg8RSYzWo8eRtZ1L4lXlgn8F53lBuS8WNgjCUBf5iLGqx0njsYbc
cQjUMdh8JbEPMJVECOh2zlyyX8unugjQ0gLg0GSAD6ENKkkoRN5RwIojMF9kIIXovZ
F0OJopZW3WYZsnkx/NFrEGGTOGdpVW3kHIyGNtoYmZOJiO4d/CUgYiaK2OFdDYkv
lVfhr7bwwuZooymcHSDKe89xiu0cKoZSHZA5IP0BFiaGcksa70rIH/Pzr/dRzvuM9+ZP
X1aMaJKE5UtSIz44m2wxDoquvvlrEvLdWnJN2tz5I0c08eGlH2JDQV6iNng+YOCOe9
AEkzb1xHCUvyNXGV6FlqeWjVd0fdagerXC1cMqdI5juCSG/gzs7IJjau0PlgsnCduLyy2
u4lBQcmMct52ZwMCYJ5zmkSH+b2fuktbLaM4c2e9epD3TnvZaCxXprHQb378261B
+w9w7UJVYP8tkV/j296jA=

DeCrypted Text : My name is Antra!*/
```

6.”SHA1”

```
import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
public class SHA1 {
```

```
    public static void main(String args[]) throws NoSuchAlgorithmException  {
            String message = "Ankita";
            System.out.println("HashCode:" + getMessagedigest(message));
            }

    public static String getMessagedigest(String message)
    {
    try {
     MessageDigest md = MessageDigest.getInstance("SHA1");
     byte[] messageDigest = md.digest(message.getBytes());
     BigInteger no = new BigInteger(1, messageDigest);
     String hashtext = no.toString(16);
                while (hashtext.length() < 32) {
                        hashtext = "0" + hashtext;
                }
                return hashtext;
            }
      catch (NoSuchAlgorithmException e) {
                throw new RuntimeException(e);
            }
    }
}
```

/*Output:

D:\Semester-5-\Network security\Assignment 1>javac SHA1.java

D:\Semester-5-\Network security\Assignment 1>java SHA1

HashCode:7aa4ca80d0edd35cb05f88b5a7bd4cf228af3f21*/

#Xx

C^eL-ted text is :NPH

Decrypted text is :Antra koul

*/

7.”<u>Substitution cipher”</u>

```
import java.util.Scanner;
import java.io.*;
class Substitution{
	public static void main(String args[]){
		Scanner in = new Scanner(System.in);

		String plain_text;
		try{
		BufferedReader br = new BufferedReader(new FileReader(new File("SubstitutionKey.txt")));
		int key=Integer.parseInt(br.readLine());

	System.out.print("\nEnter text to be encrypted:");
		plain_text=in.nextLine();
```

```
            String encrypted_Text=encryptText(plain_text,key);


            System.out.print("\nEncrypted Text:"+encrypted_Text);


            System.out.print("\nDecrypted
Text:"+decryptText(encrypted_Text,key));
            }
            catch(Exception e){
                  e.printStackTrace();
            }
      }
      public static String encryptText(String plain_text,int key)
      {
            String encrypted_Text="";
            for(int i=0;i<plain_text.length();i++){
                  if(((int)plain_text.charAt(i)) == 32)
                        encrypted_Text=encrypted_Text+" ";
                  else{
                        int n=(((int)plain_text.charAt(i))-65-key);
                        if(n<0)
                              n=26-Math.abs(n);
                        encrypted_Text=encrypted_Text+((char)(n+65));
```

```
                }
            }
            return encrypted_Text;
        }
        public static String decryptText(String plain_text,int key)
        {
            String encrypted_Text="";
            for(int i=0;i<plain_text.length();i++){
                if(((int)plain_text.charAt(i)) == 32)
                    encrypted_Text=encrypted_Text+" ";
                else{
                    int n=(((int)plain_text.charAt(i))-65+key);
                    if(n>25)
                        n=Math.abs(n)-26;
                    encrypted_Text=encrypted_Text+((char)(n+65));
                }
            }
            return encrypted_Text;
        }
}
/*Output:
C:\Users\user\Desktop\Sem 5\Network security>javac Substitution.java
```

C:\Users\user\Desktop\Sem 5\Network security>java Substitution

Enter text to be encrypted:THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

Encrypted Text:QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD

Decrypted Text:THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

*/

Substitution key

3

8.”Sunstitution Client Server:”

```
import java.net.*;
import java.io.*;
import java.util.*;
class SubstitutionUDP
{
	public static void main(String args[])throws Exception
	{
		DatagramSocket datagram_socket = new DatagramSocket();

		DatagramPacket datagram_packet = null;
```

```
InetAddress ip = InetAddress.getLocalHost();

Scanner in = new Scanner(System.in);

byte[] send_packet = new byte[65535];

byte[] receive_packet = new byte[65535];

BufferedReader br = new BufferedReader(new FileReader(new
File("SubstitutionKey.txt")));
int key=Integer.parseInt(br.readLine());

while(true)
{
    String message_string ;

    System.out.print("\nClient:");
    message_string= in.nextLine();

    send_packet = message_string.getBytes();
```

```
                    datagram_packet = new
DatagramPacket(send_packet,send_packet.length,ip,1234);

                    datagram_socket.send(datagram_packet);

                    datagram_packet = new
DatagramPacket(receive_packet,receive_packet.length);

                    datagram_socket.receive(datagram_packet);

                    message_string =  convert_to_string(receive_packet);

                    System.out.print("Server : "+message_string);

                    System.out.print("\nDecrypted
Text:"+decryptText(message_string,key));

                    if(message_string.equals("EXIT") ||
message_string.equals("BYE"))
                    {
                        System.out.print("Client exiting.....");
                        break;
                    }
```

```
                send_packet = new byte[65535];


                receive_packet = new byte[65535];


        }
}
public static String convert_to_string(byte[] buffer)
{
        if ( buffer == null )
        {
                System.out.print("\n No message sent !");
                return null;
        }
        String message = "";
        int i = 0;
        while ( buffer[i] != 0 )
        {
                message = message + (char)buffer[i];
                i++;
        }
        return message;
}
```

```
	public static String decryptText(String plain_text,int key)
	{
		String encrypted_Text="";
		for(int i=0;i<plain_text.length();i++){
			if(((int)plain_text.charAt(i)) == 32)
				encrypted_Text=encrypted_Text+" ";
			else{
				int n=(((int)plain_text.charAt(i))-65+key);
				if(n>25)
					n=Math.abs(n)-26;
				encrypted_Text=encrypted_Text+((char)(n+65));
			}
		}
		return encrypted_Text;
	}
}

import java.io.*;
import java.net.*;
import java.util.*;
class SubstitutionServer
{
```

```
public static void main(String args[])throws Exception
{
        DatagramSocket datagram_socket = new DatagramSocket(1234);

        InetAddress ip = InetAddress.getLocalHost();

        byte[] recieve_packet = new byte[65535];

        byte[] send_packet = new byte[65535];

        DatagramPacket datagram_packet = null;

        BufferedReader br = new BufferedReader(new FileReader(new
File("SubstitutionKey.txt")));
        int key=Integer.parseInt(br.readLine());

        while(true)
        {
                datagram_packet = new DatagramPacket(recieve_packet ,
recieve_packet.length);

                datagram_socket.receive(datagram_packet);
```

```
                String message = convert_to_String(recieve_packet);

                System.out.print("\nClient's Plain text:"+message);

                message=encryptText(message,key);

                System.out.print("\nServer Encrypted text:"+message);

                send_packet = message.getBytes();

                ip = datagram_packet.getAddress();

                int port = datagram_packet.getPort();

            datagram_packet = new DatagramPacket(send_packet,
send_packet.length , ip , port);

                datagram_socket.send(datagram_packet);

                if(message.equals("EXIT") || message.equals("BYE"))

                {
```

```
                System.out.print("\n Client exiting!");
                datagram_socket.close();
                break;
            }

            recieve_packet = new byte[65535];

            send_packet = new byte[65535];
        }
}
public static String convert_to_String(byte[] buf)
{
        if(buf == null)
        {
            System.out.print("\n No message recieved !");
            return null;
        }
        String message = "";
        int i = 0;
        while ( buf[i] != 0 )
        {
            message = message + (char)buf[i];
```

```
                i++;
            }
        return message;
    }
    public static String encryptText(String plain_text,int key)
    {
        String encrypted_Text="";
        for(int i=0;i<plain_text.length();i++){
            if(((int)plain_text.charAt(i)) == 32)
                encrypted_Text=encrypted_Text+" ";
            else{
                int n=(((int)plain_text.charAt(i))-65-key);
                if(n<0)
                    n=26-Math.abs(n);
                encrypted_Text=encrypted_Text+((char)(n+65));
            }
        }
        return encrypted_Text;
    }
}
/*Output:
C:\Users\user\Desktop\Sem 5\Network security>javac SubstitutionServer.java
```

C:\Users\user\Desktop\Sem 5\Network security>java SubstitutionServer

Client's Plain text:THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

Server Encrypted text:QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD

C:\Users\user\Desktop\Sem 5\Network security>javac SubstitutionUDP.java

C:\Users\user\Desktop\Sem 5\Network security>java SubstitutionUDP

Client:THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

Server : QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD

Decrypted Text:THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

Client:

***/**

9.”Transposition Cipher”

```
import java.util.Scanner;
import java.util.TreeMap;
import java.util.Iterator;
import java.util.Map;
import java.util.ArrayList;
```

```
import java.io.File;
import java.io.FileReader;
import java.io.BufferedReader;

class TranspositionCypher{
	public static void main(String args[]){
		Scanner in = new Scanner(System.in);

		System.out.print("Enter the text to be encrypted:");
		String plain_text=in.nextLine();
		try{
			BufferedReader br = new BufferedReader(new FileReader(new
File("TranspositionKey.txt")));
			String key = br.readLine();

			TreeMap<String,String> encrypt_text = new
TreeMap<String,String>();
			encrypt_text=insert_key(key,encrypt_text,plain_text);
			System.out.print("\nEncrypted text:");
			int len=printEncryptedText(encrypt_text);
			System.out.print("\nDecrypted text:");
```

```
                printDecryptedText(encrypt_text,key,len);
            }
            catch(Exception e){
                System.out.print("Exception occured!"+e.getMessage());
            }
        }
        public static void printDecryptedText(TreeMap<String,String>
encrypt_text,String key,int length)
        {
            String str[] = key.split("");
            int counter=0,i=0;
            ArrayList<String> text = new ArrayList<String>();
            for(String s:str){
                text.add(encrypt_text.get(s));
            }
            while(counter<length)
            {
                for(int j=0;j<text.size();j++)
                {
                    String s=text.get(j);
                    if(i<s.length()){
                        System.out.print(s.charAt(i));
```

```
                                counter++;
                            }
                        }
                        i++;
                    }
                }
        public static int printEncryptedText(TreeMap<String,String> encrypt_text)
        {
                int count=0;
                Iterator itr =encrypt_text.entrySet().iterator();
                while (itr.hasNext()) {
         Map.Entry mapElement = (Map.Entry)itr.next();
                        String str=mapElement.getValue().toString();
                        System.out.print(str);
                        count=count+str.length();
      }
                return count;
        }
        public static TreeMap<String,String> insert_key(String key,
TreeMap<String,String>encrypt_text,String plain_text)
        {
                String[] str = key.split("");
```

```
        String text="";
        int i,j,key_length,text_length;
        j=0;
        key_length=key.length();
        text_length=plain_text.length();
        for(String s:str){
            i=j;
            while(i<text_length){
                text=text+(plain_text.charAt(i));
                if(i+key_length < text_length){
                    i+=key_length;
                    }
                else
                    i=text_length;
            }
            encrypt_text.put(s,text);
            j+=1;
            text="";
        }
        return encrypt_text;
    }
}
```

/*Output:

C:\Users\user\Desktop\Sem 5\Network security>javac TranspositionCypher.java

C:\Users\user\Desktop\Sem 5\Network security>java TranspositionCypher

Enter the text to be encrypted:please transfer one million dollars to my swiss bank account six two two

Encrypted text:as wktosfmdti rll sciwlanor autenenssnnwt llm cxoproiaybo eeioosast

Decrypted text:please transfer one million dollars to my swiss bank account six two two*/

Transposition key:

MEGABUCK

10**.”AES”**

```
import javax.crypto.*;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;
class AES{
    static String Iv="AAAAAAAAAAAAAAAA";
    static String plain_text="test text 123\0\0\0";
    static String key= "0123456789abcdef";
```

```
public static void main(String args[]){
        try{
                System.out.print("\nPlain text:"+plain_text);

                byte[] encrypted_text=encrypt(plain_text,key);

                System.out.print("\nEncrypted text:");
                for(int i=0;i<encrypted_text.length;i++){
                        System.out.print(new Integer(encrypted_text[i]+""));

                }

                String decrypted_text=decrypt(encrypted_text,key);

                System.out.print("\nDecrypted text:"+decrypted_text);
        }
        catch(Throwable e){
                e.printStackTrace();
        }
}
public static byte[] encrypt(String plain_text,String key)throws Throwable{
        Cipher cipher=Cipher.getInstance("AES/CBC/NoPadding", "SunJCE");
```

```
            SecretKeySpec secret_key_spec = new
SecretKeySpec(key.getBytes("UTF-8"),"AES");

            cipher.init(Cipher.ENCRYPT_MODE,secret_key_spec,new
IvParameterSpec(Iv.getBytes("UTF-8")));

            return cipher.doFinal(plain_text.getBytes());

        }

        public static String decrypt(byte[] cipher_text,String key)throws Throwable{

            Cipher cipher=Cipher.getInstance("AES/CBC/NoPadding", "SunJCE");

            SecretKeySpec secret_key_spec = new
SecretKeySpec(key.getBytes("UTF-8"),"AES");

            cipher.init(Cipher.DECRYPT_MODE,secret_key_spec,new
IvParameterSpec(Iv.getBytes("UTF-8")));

            return new String(cipher.doFinal(cipher_text),"UTF-8");

        }

}

/*Output:

C:\Users\user\Desktop\Sem 5\Semester-5-\Network security\AES>javac AES.java

Note: AES.java uses or overrides a deprecated API.

Note: Recompile with -Xlint:deprecation for details.


C:\Users\user\Desktop\Sem 5\Semester-5-\Network security\AES>java AES


Plain text:test text 123
```

Encrypted text:16-12441-83-16-12361-64-15-74872863306478

Decrypted text:test text 123*/

“Assignment2”

1. Implement authentication Service. The sender sends password in encrypted format to the receiver, the receiver checks

the password (after decrypting) in its database/array and replies back as success or failure.(Keys are already shared)

“Client”

```
import java.net.DatagramSocket;

import java.net.DatagramPacket;

import java.net.InetAddress;

import java.util.Scanner;

import java.io.IOException;


class Question1Sender

{

   public static void main(String args[]) throws IOException

   {

                DatagramSocket client = new DatagramSocket();

                InetAddress ip = InetAddress.getLocalHost();
```

```
                byte[] send = new byte[65536];
                byte[] recieve = new byte[65536];
                Scanner in = new Scanner(System.in);
                System.out.print("\nEnter username:");
                String username=in.nextLine();
                System.out.print("\nEnter password:");
                String password = in.nextLine();
                password = encryptPassword(password);
                String data=username+","+password;
                send=data.getBytes();
                DatagramPacket packet = new
DatagramPacket(send,send.length,ip,1234);
                client.send(packet);
                packet = new DatagramPacket(recieve,recieve.length);
                client.receive(packet);
                System.out.print("Server:"+convertToString(recieve));
        }
        public static String convertToString(byte[] a)
  {
     if (a == null)
        return null;
     String s = "";
```

```
        int i = 0;
        while (a[i] != 0)
        {
           s=s+(char)a[i];
           i++;
        }
        return s;
    }
        public static String encryptPassword(String password){
                char[] tokens = password.toCharArray();
                String encrypted_text="";
                for(char c:tokens){
                        encrypted_text=encrypted_text+((char)((int)c+3))+"";
                }
         return encrypted_text;
        }
}
```

“Server”

```
import java.net.DatagramSocket;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.util.Scanner;
```

```
import java.io.IOException;
import java.util.HashMap;


class Question1Reciever
{
        static HashMap<String,String> user_data = new HashMap<String,String>();
   public static void main(String args[]) throws IOException
   {
                user_data.put("ANTRA","ANTRA");
            user_data.put("ANKITA","ANKITA");
                DatagramSocket server = new DatagramSocket(1234);
                byte[] send = new byte[65536];
                byte[] recieve = new byte[65536];
                DatagramPacket packet = new
DatagramPacket(recieve,recieve.length);
                server.receive(packet);
                String[] message=(convertToString(recieve)).split(",");
                String username=message[0];
                String password=decryptPassword(message[1]);
                InetAddress ip = packet.getAddress();
                int port=packet.getPort();
                System.out.print("Username"+username+"Password"+password);
```

```
                String status=check(password,username);
                packet=new
DatagramPacket(status.getBytes(),status.getBytes().length,ip,port);
                server.send(packet);
        }
        //authoriziing function
        public static String check(String password,String username){
                if(password.equals((String)user_data.get(username))){
                        return "success";
                }
                return "failure";
        }
        //converting bytes to string
        public static String convertToString(byte[] a)
  {
    if (a == null)
      return null;
    String s = "";
    int i = 0;
    while (a[i] != 0)
    {
      s=s+(char)a[i];
```

```
            i++;
        }
        return s;
    }
        //decrypting password
        public static String decryptPassword(String password){
                char[] tokens = password.toCharArray();
                String decrypted_text="";
                for(char c:tokens){
                        decrypted_text=decrypted_text+((char)((int)c-3))+"";
                }
         return decrypted_text;
        }
}
/*Output:
PS D:\Semester-5-\Network security\Assignment 2> javac Question1Sender.java
PS D:\Semester-5-\Network security\Assignment 2> java Question1Sender

Enter username:ANKITA

Enter password:ANKITA
Server:success
```

PS D:\Semester-5-\Network security\Assignment 2> javac Question1Sender.java

PS D:\Semester-5-\Network security\Assignment 2> java Question1Sender

Enter username:ABC

Enter password:ABC

Server:failure

PS D:\Semester-5-\Network security\Assignment 2> javac Question2Reciever.java

PS D:\Semester-5-\Network security\Assignment 2> java Question2Reciever

UsernameANKITAPasswordANKITA

PS D:\Semester-5-\Network security\Assignment 2> javac Question2Reciever.java

PS D:\Semester-5-\Network security\Assignment 2> java Question2Reciever

UsernameABCPasswordABC*/

2. **Implement authentication Service. The sender sends password in encrypted format to the receiver,**

the receiver checks the password (after decrypting and applying hash) in its database/array and replies

back as success or failure. (Note: Here the password is stored as hash in database).*/

“Client”

```
import java.net.DatagramSocket;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.util.Scanner;
import java.io.IOException;

public class Question2Sender
{
   public static void main(String args[]) throws IOException
   {
                DatagramSocket client = new DatagramSocket();
                InetAddress ip = InetAddress.getLocalHost();
                byte[] message = new byte[65536];
                Scanner in = new Scanner(System.in);
                System.out.print("\nEnter username:");
                String username=in.nextLine();
                System.out.print("\nEnter password:");
                String password = in.nextLine();
                password = encryptPassword(password);
                String data=username+","+password;
                message=data.getBytes();
```

```
                DatagramPacket packet = new
DatagramPacket(message,message.length,ip,1234);
                client.send(packet);
                message = new byte[65536];
                packet = new DatagramPacket(message,message.length);
                client.receive(packet);
                System.out.print("Server:"+convertToString(message));
        }
        public static String convertToString(byte[] a)
   {
      if (a == null)
         return null;
      String s = "";
      int i = 0;
      while (a[i] != 0)
      {
         s=s+(char)a[i];
         i++;
      }
      return s;
   }
        public static String encryptPassword(String password){
```

```
                char[] tokens = password.toCharArray();
                String encrypted_text="";
                for(char c:tokens){
                        encrypted_text=encrypted_text+((char)((int)c+3))+"";
                }
         return encrypted_text;
        }
}
```

“Server”

```
import java.net.DatagramSocket;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.util.Scanner;
import java.io.IOException;
import java.util.HashMap;
import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

class Question2Receiver
{
```

```
        static HashMap<String,String> user_data = new HashMap<String,String>();
    public static void main(String args[]) throws IOException
    {
                user_data.put("Antra","182c4b89a27c52474a0a532d5519f5e1");
            user_data.put("Ankita","d265e24340d83487e7740d67927e4003");

                DatagramSocket server = new DatagramSocket(1234);
                byte[] message = new byte[65536];
                DatagramPacket packet = new
DatagramPacket(message,message.length);
                server.receive(packet);
                InetAddress ip=packet.getAddress();
                int port=packet.getPort();
                String[] str=(convertToString(message)).split(",");
                String username=str[0];
                String password=str[1];
                password=decryptPassword(password);
                System.out.print("Username:"+username+"Password:"+password);
                message=null;
                message=(check(password,username)).getBytes();
                packet=new DatagramPacket(message,message.length,ip,port);
                server.send(packet);
```

```
    }

    //authoriziing function
    public static String check(String password,String username){

    if((getMessagedigest(password)).equals((String)user_data.get(username))){
                    return "success";
            }
            return "failure";
    }
    //converting bytes to string
    public static String convertToString(byte[] a)
{
   if (a == null)
      return null;
   String s = "";
   int i = 0;
   while (a[i] != 0)
   {
      s=s+(char)a[i];
      i++;
   }
```

```
    return s;
}
    //decrypting password
    public static String decryptPassword(String password){
            char[] tokens = password.toCharArray();
            String decrypted_text="";
            for(char c:tokens){
                    decrypted_text=decrypted_text+((char)((int)c-3))+"";
            }
     return decrypted_text;
    }

    public static String getMessagedigest(String message)
    {
    try {
     MessageDigest md = MessageDigest.getInstance("MD5");
     byte[] messageDigest = md.digest(message.getBytes());
     BigInteger no = new BigInteger(1, messageDigest);
     String hashtext = no.toString(16);
                    while (hashtext.length() < 32) {
                            hashtext = "0" + hashtext;
                    }
```

```
                return hashtext;
            }
        catch (NoSuchAlgorithmException e) {
                throw new RuntimeException(e);
            }
        }
}
/*Output:
PS D:\Semester-5-\Network security\Assignment 2> javac Question2Sender.java
PS D:\Semester-5-\Network security\Assignment 2> java Question2Sender

Enter username:Antra

Enter password:ANtra
Server:failure
PS D:\Semester-5-\Network security\Assignment 2> javac Question2Sender.java
PS D:\Semester-5-\Network security\Assignment 2> java Question2Sender

Enter username:Antra

Enter password:Antra
Server:success
```

PS D:\Semester-5-\Network security\Assignment 2> javac Question2Receiver.java

PS D:\Semester-5-\Network security\Assignment 2> java Question2Receiver

Username:AntraPassword:ANtra

PS D:\Semester-5-\Network security\Assignment 2> javac Question2Receiver.java

PS D:\Semester-5-\Network security\Assignment 2> java Question2Receiver

Username:AntraPassword:Antra*/

3. Implement a firewall that behaves like forwarder. It does not forward the packet that contains the word "terrorist".

“Client”

```
import java.net.InetAddress;
import java.util.Scanner;
import java.io.IOException;
import java.net.DatagramSocket;
import java.net.DatagramPacket;
class Question3Client
{
   public static void main(String args[]) throws IOException
   {
                DatagramSocket client = new DatagramSocket();
                DatagramPacket packet = null;
```

```
            InetAddress ip = InetAddress.getLocalHost();
            byte[] packetBytes = new byte[65536];
            Scanner in = new Scanner(System.in);
            String message="";
            while(!message.equalsIgnoreCase("bye")){
                System.out.print("\nClient:");
                message = in.nextLine();
                if(message.equalsIgnoreCase("bye"))
                {
                    System.out.print("\nConversation ended!");
                    break;
                }
                packetBytes = message.getBytes();
                packet = new
DatagramPacket(packetBytes,packetBytes.length,ip,1234);
                client.send(packet);
                packetBytes = new byte[65536];
                packet = new
DatagramPacket(packetBytes,packetBytes.length);
                client.receive(packet);
                System.out.print("Server:"+convertToString(packetBytes));
            }
```

```
        }
        public static String convertToString(byte[] a)
    {
        if (a == null)
            return null;
        String s = "";
        int i = 0;
        while (a[i] != 0)
        {
            s=s+(char)a[i];
            i++;
        }
        return s;
    }
}
```

“Firewall”

```
import java.net.DatagramSocket;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.util.Scanner;
import java.io.IOException;
import java.util.HashMap;
```

```
class Question3Firewall
{
   public static void main(String args[]) throws IOException
   {
                DatagramSocket server = new DatagramSocket(1234);
                byte[] packetBytes = new byte[65536];
                String message="",status;
                InetAddress ip;
                int port;
                DatagramPacket packet=null;
                while(!message.equalsIgnoreCase("bye")){
                        packet=new
DatagramPacket(packetBytes,packetBytes.length);
                        server.receive(packet);
                        System.out.print("\nClient:"+message);
                        message=convertToString(packetBytes);
                        if(message.equalsIgnoreCase("bye")){
                                System.out.print("\n Connection terminated!");
                                break;
                        }
                        status=check(message);
```

```
                    if(status.equals("not sent!")){
                            ip = packet.getAddress();
                            port=packet.getPort();
                            packetBytes=status.getBytes();
                            packet=new
DatagramPacket(packetBytes,packetBytes.length,ip,port);
                            server.send(packet);
                    }
                    else
                    {

                            ip = packet.getAddress();
                            port=packet.getPort();
                            packetBytes=status.getBytes();
                            packet=new
DatagramPacket(packetBytes,packetBytes.length,ip,port);
                            server.send(packet);
                            packetBytes = new byte[65536];
                            ip=InetAddress.getLocalHost();
                            packetBytes=message.getBytes();
                            packet=new
DatagramPacket(packetBytes,packetBytes.length,ip,1111);
                            server.send(packet);
```

```
                }
                  packetBytes = new byte[65536];
                }
        }
        //authoriziing function
        public static String check(String message){
                String[] tokens = message.split(" ");
                for(String s:tokens){
                        if(s.equalsIgnoreCase("terrorist"))
                        {
                                System.out.println("Can't be forwarded!");
                                return "not sent!";
                        }
                }
                return "sent!";
        }
        //converting bytes to string
        public static String convertToString(byte[] a)
{
    if (a == null)
        return null;
    String s = "";
```

```
        int i = 0;
        while (a[i] != 0)
        {
           s=s+(char)a[i];
           i++;
        }
        return s;
    }
}
```

“Server”

```
import java.net.DatagramSocket;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.util.Scanner;
import java.io.IOException;
import java.util.HashMap;

class Question3Server
{
    public static void main(String args[]) throws IOException
    {
                DatagramSocket server = new DatagramSocket(1111);
```

```
                byte[] packetBytes = new byte[65536];
                String message="";
                DatagramPacket packet=null;
                while(!message.equalsIgnoreCase("bye")){
                        packet=new
DatagramPacket(packetBytes,packetBytes.length);
                        server.receive(packet);
                        message=convertToString(packetBytes);
                        System.out.print("\nClient:"+message);
                        if(message.equalsIgnoreCase("bye")){
                                System.out.print("\n Connection terminated!");
                                break;
                        }
                  packetBytes = new byte[65536];
                }
        }
        //converting bytes to string
        public static String convertToString(byte[] a)
   {
      if (a == null)
         return null;
      String s = "";
```

```
        int i = 0;
        while (a[i] != 0)
        {
           s=s+(char)a[i];
           i++;
        }
        return s;
    }
}
/*Output:
PS D:\Semester-5-\Network security\Assignment 2> javac Question3Client.java
PS D:\Semester-5-\Network security\Assignment 2> java Question3Client

Client:Hello
Server:sent!
Client:How are you?
Server:sent!
Client:terrorist
Server:not sent!
Client:Hey again?
Server:sent!
Client:
```

PS D:\Semester-5-\Network security\Assignment 2> javac Question3Firewall.java

PS D:\Semester-5-\Network security\Assignment 2> java Question3Firewall

Client:

Client:Hello

Client:How are you?Can't be forwarded!

Client:terrorist

PS D:\Semester-5-\Network security\Assignment 2> javac Question3Server.java

PS D:\Semester-5-\Network security\Assignment 2> java Question3Server

Client:Hello

Client:How are you?

Client:Hey again?*/

4. **Implement NAT functionality. The NAT works like forwarder, that will forward to appropriate receiver.**

“Sender”

```
import java.net.InetAddress;

import java.util.Scanner;
```

```
import java.io.IOException;
import java.net.DatagramSocket;
import java.net.DatagramPacket;
class Question4Sender
{
   public static void main(String args[]) throws IOException
   {
                DatagramSocket client = new DatagramSocket();
                DatagramPacket packet = null;
                InetAddress ip = InetAddress.getLocalHost();
                byte[] packetBytes = new byte[65536];
                Scanner in = new Scanner(System.in);
                String message="";
                while(!message.equalsIgnoreCase("bye")){
                        System.out.print("\nClient:");
                        message = in.nextLine();
                        if(message.equalsIgnoreCase("bye"))
                        {
                                System.out.print("\nConversation ended!");
                                break;
                        }
                        packetBytes = message.getBytes();
```

```
                        packet = new
DatagramPacket(packetBytes,packetBytes.length,ip,1234);
                        client.send(packet);
                }
        }
        public static String convertToString(byte[] a)
   {
      if (a == null)
         return null;
      String s = "";
      int i = 0;
      while (a[i] != 0)
      {
         s=s+(char)a[i];
         i++;
      }
      return s;
   }
}
```

“NAT”

```
import java.net.DatagramSocket;
```

```
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.io.IOException;
class Question4NAT
{
   public static void main(String args[]) throws IOException
   {
                DatagramSocket server = new DatagramSocket(1234);
                byte[] packetBytes = new byte[65536];
                String message="",status;
                InetAddress ip;
                int port;
                DatagramPacket packet=null;
                while(!message.equalsIgnoreCase("bye")){
                        packet=new
DatagramPacket(packetBytes,packetBytes.length);
                        server.receive(packet);
                        message=convertToString(packetBytes);
                        System.out.print("\nClient:"+message);
                        if(message.equalsIgnoreCase("bye")){
                                System.out.print("\n Connection terminated!");
                                break;
```

```
                    }
                    status=check(message);
                    ip=InetAddress.getLocalHost();
                    packetBytes=message.getBytes();
                    if(status.equals("even")){
                            packet=new
DatagramPacket(packetBytes,packetBytes.length,ip,1111);
                            server.send(packet);
                    }
                    else if(status.equals("odd"))
                    {
                            packet=new
DatagramPacket(packetBytes,packetBytes.length,ip,2222);
                            server.send(packet);
                    }
                    else{
                            System.out.print("\n Invalid message!");
                    }
                  packetBytes = new byte[65536];
                }
        }
        //authoriziing function
```

```
    public static String check(String message){
            if((Integer.parseInt(message)) % 2 == 0)
                    return "even";
            else if((Integer.parseInt(message)) % 2 != 0)
                    return "odd";
            else
                    return "failure!";
    }
    //converting bytes to string
    public static String convertToString(byte[] a)
{
   if (a == null)
      return null;
   String s = "";
   int i = 0;
   while (a[i] != 0)
   {
      s=s+(char)a[i];
      i++;
   }
   return s;
}
```

}

“Receiver1”

```
import java.net.DatagramSocket;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.io.IOException;
class Question4Server1
{
   public static void main(String args[]) throws IOException
   {
                DatagramSocket server = new DatagramSocket(1111);
                byte[] packetBytes = new byte[65536];
                String message="";
                DatagramPacket packet=null;
                while(!message.equalsIgnoreCase("bye")){
                        packet=new
DatagramPacket(packetBytes,packetBytes.length);
                        server.receive(packet);
                        message=convertToString(packetBytes);
                        System.out.print("\nClient:"+message);
                        if(message.equalsIgnoreCase("bye")){
```

```
                    System.out.print("\n Connection terminated!");
                    break;
                }
              packetBytes = new byte[65536];
            }
        }
        //converting bytes to string
        public static String convertToString(byte[] a)
    {
        if (a == null)
            return null;
        String s = "";
        int i = 0;
        while (a[i] != 0)
        {
            s=s+(char)a[i];
            i++;
        }
        return s;
    }
}
```

“Receiver2”

```
import java.net.DatagramSocket;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.io.IOException;
class Question4Server2
{
   public static void main(String args[]) throws IOException
   {
                DatagramSocket server = new DatagramSocket(2222);
                byte[] packetBytes = new byte[65536];
                String message="";
                DatagramPacket packet=null;
                while(!message.equalsIgnoreCase("bye")){
                        packet=new
DatagramPacket(packetBytes,packetBytes.length);
                        server.receive(packet);
                        message=convertToString(packetBytes);
                        System.out.print("\nClient:"+message);
                        if(message.equalsIgnoreCase("bye")){
                                System.out.print("\n Connection terminated!");
                                break;
                        }
```

```
                    packetBytes = new byte[65536];
                }
            }
            //converting bytes to string
            public static String convertToString(byte[] a)
    {
        if (a == null)
            return null;
        String s = "";
        int i = 0;
        while (a[i] != 0)
        {
            s=s+(char)a[i];
            i++;
        }
        return s;
    }
}
/*Output:
PS D:\Semester-5-\Network security\Assignment 2> javac Question4Sender.java
PS D:\Semester-5-\Network security\Assignment 2> java Question4Sender
```

Client:2

Client:23

Client:32

Client:11

Client:

PS D:\Semester-5-\Network security\Assignment 2> javac Question4NAT.java

PS D:\Semester-5-\Network security\Assignment 2> java Question4NAT

Client:2

Client:23

Client:32

Client:11

PS D:\Semester-5-\Network security\Assignment 2> javac Question4Server1.java

PS D:\Semester-5-\Network security\Assignment 2> java Question4Server1

Client:2

Client:32

PS D:\Semester-5-\Network security\Assignment 2> javac Question4Server2.java

PS D:\Semester-5-\Network security\Assignment 2> java Question4Server2

Client:23

Client:11*/

5. Key Distribution

Implement a program to demonstrate the functioning of a KDC. There are three entities: sender, receiver and KDC. Assume that Sender and Receiver have already established their own individual permanent secret keys with KDC. The sender requests the KDC to issue a session key to communicate with receiver. The KDC is supposed to give session key information to sender in a secure way. The same session key is also to be communicated to the receiver securely. Use a suitable protocol to achieve the above functionality.

"KDCSender"

```
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Scanner;
import java.util.Base64;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
 import java.io.*;
import javax.crypto.Cipher;
```

```
import javax.crypto.KeyGenerator;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;
 import javax.crypto.spec.SecretKeySpec;
class KDCSender{
    public static void main(String args[])throws Exception{
            DatagramSocket sender = new DatagramSocket();
            DatagramPacket packet = null;
            InetAddress ip = InetAddress.getLocalHost();
            String requestMessage = "Request for access to server!";
            byte[] message= new byte[65536];

            message=requestMessage.getBytes();
            packet = new
DatagramPacket(message,message.length,ip,1234);
            sender.send(packet);

            message= new byte[65536];
            packet = new DatagramPacket(message,message.length);
            sender.receive(packet);

            String str = convertToString(message);
            System.out.print("\nServer:"+str);

            BufferedReader br = new BufferedReader(new
FileReader(new File("SenderKey.txt")));
            String senderKey=br.readLine();

            String[] st=(decryptData(str,senderKey)).split(",");
            requestMessage=st[0];

            message= new byte[65536];
            message=requestMessage.getBytes();
```

```
            packet = new
DatagramPacket(message,message.length,ip,2222);
            sender.send(packet);



    }
    public static String decryptData(String message,String
senderKey)throws Exception{
            SecretKey key;
            byte[] keyByte = senderKey.getBytes();
            key = new SecretKeySpec(keyByte,0,keyByte.length,"DES");
            Cipher ecipher = Cipher.getInstance("DES");
            ecipher.init(Cipher.DECRYPT_MODE, key);
            byte[]
dec=Base64.getDecoder().decode(message.getBytes());
            byte[] utf8 = ecipher.doFinal(dec);
            return new String(utf8, "UTF8");
    }
    public static String convertToString(byte[] a)
   {
      if (a == null)
         return null;
      String s = "";
      int i = 0;
      while (a[i] != 0)
      {
         s=s+(char)a[i];
         i++;
      }
      return s;
   }
}
```

“KDC”

```
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Scanner;
import java.util.Base64;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
 import java.io.*;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;
 import javax.crypto.spec.SecretKeySpec;
class KDC{
    public static void main(String args[])throws Exception{
            DatagramSocket kdc = new DatagramSocket(1234);
            DatagramPacket packet = null;
            InetAddress ip ;
            byte[] messageBytes=new byte[65536];
            String sender="Antra";
            String receiver="Ankita";
            String sessionKey="Pizza";
            String message=sender+","+receiver+","+sessionKey;
            BufferedReader br = new BufferedReader(new
FileReader(new File("ReceiverKey.txt")));
            String receiverKey=br.readLine();

            message=encryptData(message,receiverKey);

            br = new BufferedReader(new FileReader(new
File("SenderKey.txt")));
            String senderKey = br.readLine();
```

```
            message=encryptData(message+","+sessionKey,senderKey);
            System.out.print("\nMessage:"+message);
            packet = new
DatagramPacket(messageBytes,messageBytes.length);
            kdc.receive(packet);
            String sender_request=convertToString(messageBytes);

            if(sender_request.equals("Request for access to server!")){
                  messageBytes=new byte[65536];
                  messageBytes=message.getBytes();
                  packet = new
DatagramPacket(messageBytes,messageBytes.length,packet.getAddr
ess(),packet.getPort());
                  kdc.send(packet);
            }
    }
    public static String encryptData(String message,String
receiverKey)throws Exception{
            SecretKey key;
            byte[] keyByte = receiverKey.getBytes();
            key = new SecretKeySpec(keyByte,0,keyByte.length,"DES");
            Cipher ecipher = Cipher.getInstance("DES");
            ecipher.init(Cipher.ENCRYPT_MODE, key);
            byte[] string_bytes=message.getBytes("UTF-8");
            byte[] encoded = ecipher.doFinal(string_bytes);
            encoded = Base64.getEncoder().encode(encoded);
            return new String(encoded);
    }
    public static String convertToString(byte[] a)
   {
      if (a == null)
         return null;
```

```
        String s = "";
        int i = 0;
        while (a[i] != 0)
        {
           s=s+(char)a[i];
           i++;
        }
        return s;
    }
}
```

“KDCReceiver”

```
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Scanner;
import java.util.Base64;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
 import java.io.*;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;
 import javax.crypto.spec.SecretKeySpec;
class KDCReceiver{
    public static void main(String args[])throws Exception{
            DatagramSocket sender = new DatagramSocket(2222);
            DatagramPacket packet = null;
            byte[] message= new byte[65536];
            packet = new DatagramPacket(message,message.length);
            sender.receive(packet);
            String str = convertToString(message);
```

```
            BufferedReader br = new BufferedReader(new
FileReader(new File("ReceiverKey.txt")));
            String senderKey=br.readLine();
            str=decryptData(str,senderKey);

            System.out.print("\nDecrypted :"+str);
    }
    public static String decryptData(String message,String
receiverKey)throws Exception{
            SecretKey key;
            byte[] keyByte = receiverKey.getBytes();
            key = new SecretKeySpec(keyByte,0,keyByte.length,"DES");
            Cipher ecipher = Cipher.getInstance("DES");
            ecipher.init(Cipher.DECRYPT_MODE, key);
            byte[]
dec=Base64.getDecoder().decode(message.getBytes());
            byte[] utf8 = ecipher.doFinal(dec);
            return new String(utf8, "UTF8");
    }
    public static String convertToString(byte[] a)
   {
      if (a == null)
         return null;
      String s = "";
      int i = 0;
      while (a[i] != 0)
      {
         s=s+(char)a[i];
         i++;
      }
      return s;
   }
```

```
}
/*Output:
D:\Semester-5-\Network security\Assignment 2>javac
KDCSender.java

D:\Semester-5-\Network security\Assignment 2>java KDCSender

Server:mOdm96LvgHSxIevA/AfCz8CT64IxlWviLf/EgRWP/VECqDpw5f
UZfA==


D:\Semester-5-\Network security\Assignment 2>java KDC

Message:mOdm96LvgHSxIevA/AfCz8CT64IxlWviLf/EgRWP/VECqDpw
5fUZfA==

D:\Semester-5-\Network security\Assignment 2>javac
KDCReceiver.java

D:\Semester-5-\Network security\Assignment 2>java KDCReceiver

Decrypted :Antra,Ankita,Pizza*/
```