

STELLA MARY'S COLLEGE OF ENGINEERING

(Approved by AICTE, New Delhi, Affiliated to Anna University, Chennai & Accredited by NAAC,
Accredited by NBA (CSE & MECH))

Aruthenganvilai, Kallukatti Junction, Azhikal Post
Kanyakumari District – 629 202, Tamil Nadu



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Name of the Laboratory : **NETWORK SECURITY**

Laboratory Code : **CCS354**

Name of the Student :

University Register No. :

Branch : **CSE**

Semester : **VI**

Academic Year : **2023-2024 (EVEN)**

INSTITUTE VISION

To emerge as a premiere institution, acknowledged as a center for excellence imparting technical education, creating technocrats who can address the needs of the society through exploration and experimentation and uplift mankind.

INSTITUTE MISSION

To provide an education that transforms students, through rigorous course-work and by providing an understanding of the needs of the society and the industry.

DEPARTMENT VISION

To produce Computer Science professionals who can accomplish path-breaking solutions for a better society, through quality technical education, on gaining the required inter- personal, entrepreneurial and computing skills.

DEPARTMENT MISSION

- To impart a holistic and experiential learning experience by making use of innovative teaching methodologies.
- To provide optimal technology solutions through collaborative and life-long learning for industry and societal needs with universal ethical values.
- To nurture leadership skills and facilitate various co-curricular and extra-curricular activities to implant the spirit of entrepreneurship.
- To provide industry-institute-interaction opportunities in order to motivate inter-disciplinary research capabilities with an inquiring mind.

Program Educational Objectives (PEOs)

PEO1:

Graduates will be competent in creating innovative technologies through inter- disciplinary research and comprehensive skills sets that are suitable for the global computing industry.

PEO2:

Graduates will be capable of managing leading positions with a broad understanding of the application of ethics in evolving computer-based solutions for the societal needs.

PEO3:

Graduates will imbibe entrepreneurial qualities and develop their career by upgrading their, communication, analytical and professional skills constantly.

Program Specific Outcomes (PSOs)

PSO1:

Use data management techniques and algorithmic thinking for Software Design and Development practices.

PSO2:

Develop reliable IT solutions based on the expertise in Distributed Applications Development, Web Designing and Networking for various societal needs and entrepreneurial practices ethically.

PSO3:

Manage multidisciplinary environments effectively through their interpersonal and analytical skills and be responsible members and leaders of the society.

STELLA MARY'S COLLEGE OF ENGINEERING

(Approved by AICTE, New Delhi, Affiliated to Anna University, Chennai & Accredited by NAAC,
Accredited by NBA (CSE & MECH))
Aruthenganvilai, kallukatti Junction, Azhikal Post
Kanyakumari District – 629 202, Tamil Nadu



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this is a bonafide record of the work done
by Mr./Ms. _____ of _____ semester
B.E. Department of Computer Science and Engineering, Register No.
_____ for the _____
laboratory during the academic year 20__ -20__ (Odd/Even).

Faculty in-charge

(Name, date, designation & Department)

Head of the Department

(Name, date & Department)

Submitted for the practical examination held on _____

Internal Examiner

External Examiner

OBJECTIVE

- 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.

OUTCOMES

At the **completion of the laboratory course** the student will 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.

CONTENTS

Name of the laboratory:

Laboratory code:

[illegible]

CONTENTS

Name of the laboratory:

Laboratory code:

[illegible]

Exp.no: 1

IMPLEMENT SYMMETRIC KEY ALGORITHM

Date:

AIM

To write a java program for implement symmetric key algorithm using AES.

ALGORITHM

Step1 :Start the program

Step2:Import the necessary packages

Step3:Generate the key

Step4:Get the original message from the user and create the cipher instance and initialize it to ENCRYPT_MODE

Step5:Encrypt the message

Step6:Convert the encrypted message to Base64 encoded string

Step7:Reinitialize the cipher to DECRYPT_MODE

Step8:Decrypt the message

Step9:Stop the program.

PROGRAM:

```
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import java.nio.charset.StandardCharsets;
import java.util.Base64;

public class Main {
    public static void main(String[] args) throws Exception {
        // Generate key
        SecretKey secretKey = KeyGenerator.getInstance("AES").generateKey();
        // Original message
        String originalMessage = "Hello, world!";
        // Create Cipher instance and initialize it to ENCRYPT_MODE
        Cipher cipher = Cipher.getInstance("AES");
        cipher.init(Cipher.ENCRYPT_MODE, secretKey);
        // Encrypt the message
```

```

byte[] encryptedMessage =
cipher.doFinal(originalMessage.getBytes(StandardCharsets.UTF_8));

// Convert the encrypted message to Base64 encoded string
String encodedMessage = Base64.getEncoder().encodeToString(encryptedMessage);

System.out.println("Original Message: " + originalMessage);

System.out.println("Encrypted Message: " + encodedMessage);

// Reinitialize the cipher to DECRYPT_MODE
cipher.init(Cipher.DECRYPT_MODE, secretKey);

// Decrypt the message
byte[] decryptedMessage = cipher.doFinal(Base64.getDecoder().decode(encodedMessage));

System.out.println("Decrypted Message: " + new String(decryptedMessage,
StandardCharsets.UTF_8));
}
}

```

OUTPUT

Original Message: Hello, world!

Encrypted Message: iWohhm/c89uBVaJ3j4YFkA==

Decrypted Message: Hello, world!

RESULT

Thus the program to implement symmetric key algorithm using AES was executed successfully.

Ex.no:2 IMPLEMENT ASYMMETRIC KEY ALGORITHM USING RSA ALGORITHM

AIM:

To develop an application to implement RSA public key cryptosystem using HTML and JavaScript.

ALGORITHM:

1. Choose two prime number p and q
2. Compute the value of n and p
3. Find the value of e (public key)
4. Compute the value of d (private key) using $\text{gcd}()$
5. Do the encryption and decryption

a. Encryption is given as,

$$c = te \bmod n$$

b. Decryption is given as,

$$t = cd \bmod n$$

PROGRAM:

rsa.html

```
<html>
<head>
<title>RSA Encryption</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
<center>
<h1>RSA Algorithm</h1>
<h2>Implemented Using HTML & Javascript</h2>
<hr>
<table>
<tr>
<td>Enter First Prime Number:</td>
<td><input type="number" value="53" id="p"></td>
</tr>
<tr>
<td>Enter Second Prime Number:</td>
<td><input type="number" value="59" id="q"></p> </td>
</tr>
```

```

<tr>
<td>Enter the Message(cipher text):<br>[A=1, B=2,...]</td>
<td><input type="number" value="89" id="msg"></p>
</td>
</tr>
<tr>
<td>Public Key:</td>
<td>
<p id="publickey"></p>
</td>
</tr>
<tr>
<td>Exponent:</td>
<td>
<p id="exponent"></p>
</td>
</tr>
<tr>
<td>Private Key:</td>
<td>
<p id="privatekey"></p>
</td>
</tr>
<tr>
<td>Cipher Text:</td>
<td>
<p id="ciphertext"></p>
</td>
</tr>
<tr>
<td><button onclick="RSA();">Apply RSA</button></td>
</tr>
</table>
</center>

```

```

</body>
<script type="text/javascript">
function RSA() {
var gcd, p, q, no, n, t, e, i, x;
gcd = function (a, b) { return (!b) ? a : gcd(b, a % b); };
p = document.getElementById('p').value;
q = document.getElementById('q').value;
no = document.getElementById('msg').value;
n = p * q;
t = (p - 1) * (q - 1);
for (e = 2; e < t; e++) {
if (gcd(e, t) == 1) {
break;
}
}
for (i = 0; i < 10; i++) {
x = 1 + i * t
if (x % e == 0) {
d = x / e;
break;
}
}
ctt = Math.pow(no, e).toFixed(0);
ct = ctt % n;
dtt = Math.pow(ct, d).toFixed(0);
dt = dtt % n;
document.getElementById('publickey').innerHTML = n;
document.getElementById('exponent').innerHTML = e;
document.getElementById('privatekey').innerHTML = d;
document.getElementById('ciphertext').innerHTML = ct;
}
</script>
</html>

```

Output

RSA Algorithm

Implemented Using HTML & Javascript

Enter First Prime Number:	<input type="text" value="53"/>
Enter Second Prime Number:	<input type="text" value="59"/>
Enter the Message(cipher text): [A=1, B=2,...]	<input type="text" value="89"/>
Public Key:	3127
Exponent:	3
Private Key:	2011
Cipher Text:	1394
<input type="button" value="Apply RSA"/>	

Result:

Thus the RSA algorithm has been implemented using HTML & CSS and the output has been verified successfully.

Ex.no:3

IMPLEMENT KEY EXCHANGE ALGORITHM USING DIFFIE-HELLMAN ALGORITHM

AIM:

To implement key exchange algorithm using Diffie-Hellman Algorithm.

ALGORITHM:

Diffie-Hellman algorithm is one of the most important algorithms used for establishing a shared secret. At the time of exchanging data over a public network, we can use the shared secret for secret communication.

1. We will take four variables, i.e., **P (prime)**, **G (the primitive root of P)**, and **a and b (private values)**.
2. The variables **P** and **G** both are publicly available. The sender selects a private value, either **a** or **b**, for generating a key to exchange publicly. The receiver receives the key, and that generates a secret key, after which the sender and receiver both have the same secret key to encrypt.

Steps	User 1	User 2
1.	P, G => available public keys.	P, G => available public keys.
2.	a is selected as a private key.	b is selected as a private key.
3.	Eq. to generate key: $x = G^a \text{ mod } P$	Eq. to generate key: $y = G^b \text{ mod } P$
4.	After exchanging keys, user1 receives key y.	After exchanging keys, user2 receives key x.
5.	User 1 generates a secret key by using the received key y: $k_a = y^a \text{ mod } P$	User 2 generates a secret key by using the received key x: $k_b = x^b \text{ mod } P$
	Algebraically, 5 th step can be shown as follows: $k_a = k_b$	

Program

```
import java.util.*;

// create class DiffieHellmanAlgorithmExample to calculate the key for two persons
class DiffieHellmanAlgorithmExample {

    // main() method start
```

```

public static void main(String[] args)
{
    long P, G, x, a, y, b, ka, kb;
    // create Scanner class object to take input from user
    Scanner sc = new Scanner(System.in);
    System.out.println("Both the users should be agreed upon the public keys G and P");
    // take inputs for public keys from the user
    System.out.println("Enter value for public key G:");
    G = sc.nextLong();
    System.out.println("Enter value for public key P:");
    P = sc.nextLong();
    // get input from user for private keys a and b selected by User1 and User2
    System.out.println("Enter value for private key a selected by user1:");
    a = sc.nextLong();
    System.out.println("Enter value for private key b selected by user2:");
    b = sc.nextLong();
    // call calculatePower() method to generate x and y keys
    x = calculatePower(G, a, P);
    y = calculatePower(G, b, P);
    // call calculatePower() method to generate ka and kb secret keys after the exchange of x
    and y keys
    // calculate secret key for User1
    ka = calculatePower(y, a, P);
    // calculate secret key for User2
    kb = calculatePower(x, b, P);
    // print secret keys of user1 and user2
    System.out.println("Secret key for User1 is:" + ka);
    System.out.println("Secret key for User2 is:" + kb);
}

// create calculatePower() method to find the value of  $x^y \text{ mod } P$ 
private static long calculatePower(long x, long y, long P)

```

```

{
    long result = 0;
    if (y == 1){
        return x;
    }
    else{
        result = ((long)Math.pow(x, y)) % P;
        return result;
    }
}
}

```

Output

Both the users should be agreed upon the public keys G and P

Enter value for public key G:

8

Enter value for public key P:

33

Enter value for private key a selected by user1:3

Enter value for private key b selected by user2:

2

Secret key for User1 is:25

Secret key for User2 is:25

Result:

Thus the program to implement key Exchange using Diffie-Hellman algorithm was executed successfully.

Ex.no:4

IMPLEMENT DIGITAL SIGNATURE SCHEME USING SHA 256

AIM:

To implement Digital Signature Scheme using SHA 256 Algorithm.

ALGORITHM:

Step 1: Start the program

Step 2: Append Padding bits.

Step3: Append Length - 64 bits are appended to the end.

Step 4: Prepare Processing Functions.

Step 5: Prepare Processing Constants.

Step 6: Initialize Buffers.

Step 7: Processing Message in 512-bit blocks (L blocks in total message).

Step 8: Produce 256 Hash Code.

Step 9: Stop the program.

Program

```
import java.math.BigInteger;
import java.nio.charset.StandardCharsets;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

// Java program to calculate SHA hash value
class GFG2 {
    public static byte[] getSHA(String input) throws NoSuchAlgorithmException
    {
        // Static getInstance method is called with hashing SHA
        MessageDigest md = MessageDigest.getInstance("SHA-256");
        // digest() method called
        // to calculate message digest of an input
        // and return array of byte
        return md.digest(input.getBytes(StandardCharsets.UTF_8));
    }
    public static String toHexString(byte[] hash)
```

```

{
    // Convert byte array into signum representation
    BigInteger number = new BigInteger(1, hash);
    // Convert message digest into hex value
    StringBuilder hexString = new StringBuilder(number.toString(16));
    // Pad with leading zeros
    while (hexString.length() < 64)
    {
        hexString.insert(0, '0');
    }
    return hexString.toString();
}

// Driver code
public static void main(String args[])
{
    try
    {
        System.out.println("HashCode Generated by SHA-256 for:");
        String s2 = "hello world";
        System.out.println("\n" + s2 + " : " + toHexString(getSHA(s2)));
        String s3 = "K1t4fo0V";
        System.out.println("\n" + s3 + " : " + toHexString(getSHA(s3)));
    }
    // For specifying wrong message digest algorithms
    catch (NoSuchAlgorithmException e) {
        System.out.println("Exception thrown for incorrect algorithm: " + e);
    }
}
}

```

Output

HashCode Generated by SHA-256 for:

hello world : b94d27b9934d3e08a52e52d7da7dabfac484efe37a5380ee9088f7ace2efcde9

K1t4fo0V : 0a979e43f4874eb24b740c0157994e34636eed0425688161cc58e8b26b1dcf4e

Result:

Thus the program to implement Digital Signature Scheme using SHA 256 algorithm was executed successfully.

Ex.no:5

Installation of Wire shark, tcpdump and observe data transferred in client-server communication using UDP/TCP and identify the UDP/TCP datagram.

AIM:

To install Wire Shark, tcpdump and observe data transferred in client-server communication using UDP/TCP and identify the UDP/TCP datagram.

What is Wireshark?

- Wireshark is a network packet analyzer. A network packet analyzer presents captured packet data in as much detail as possible. You could think of a network packet analyzer as a measuring device for examining what's happening inside a network cable, just like an electrician uses a voltmeter for examining what's happening inside an electric cable (but at a higher level, of course).
- Wireshark, a network analysis tool formerly known as Ethereal, captures packets in real time and display them in human-readable format. Wireshark includes filters, color-coding, and other features that let you dig deep into network traffic and inspect individual packets.

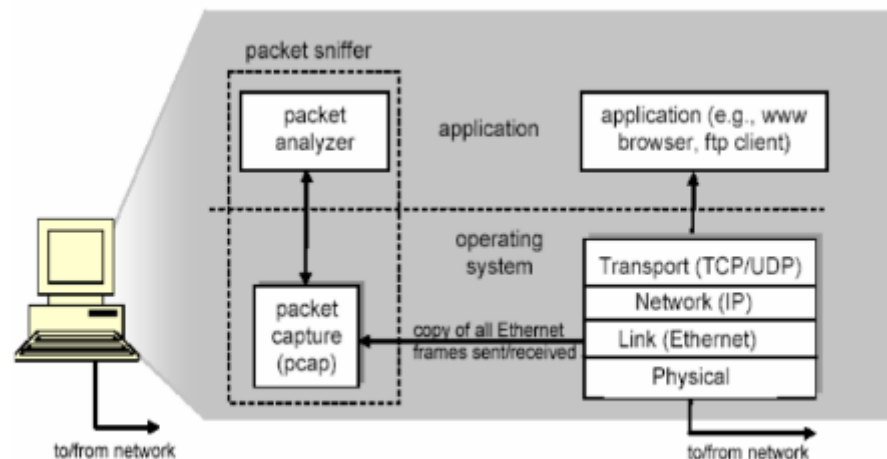


Figure 1: Packet sniffer structure

PACKET SNIFFER

The basic tool for observing the messages exchanged as packets on the network using a variety of protocols is called a packet sniffer. As the name suggests, a packet sniffer captures (“sniffs”) messages being sent/received from/by your computer; it will also typically store and/or display the contents of the various protocol fields in these captured messages. A packet sniffer itself is passive. It observes messages being sent and received by applications

and protocols running on your computer, but never sends packets itself. Similarly, received packets are never explicitly addressed to the packet sniffer. Instead, a packet sniffer receives a copy of packets that are sent / received from/by application and protocols executing on your machine.

Some intended purposes Here are some reasons people use Wireshark:

- Network administrators use it to troubleshoot network problems.
- Network security engineers use it to examine security problems.
- QA engineers use it to verify network applications.
- Developers use it to debug protocol implementations.
- People use it to learn network protocol internals.

Features

The following are some of the many features Wireshark provides:

- Available for UNIX and Windows.
- Capture live packet data from a network interface.
- Open files containing packet data captured with tcpdump/WinDump, Wireshark, and many other packetcapture programs.
- Import packets from text files containing hex dumps of packet data.
- Display packets with very detailed protocol information.
- Save packet data captured.
- Export some or all packets in a number of capture file formats.

Getting Wireshark

Wireshark can be installed in any operating system, just go to

[https:// www.wireshark.org/download.html](https://www.wireshark.org/download.html)

Running Wireshark

When you run the Wireshark program, the Wireshark graphical user interface will be displayed. Initially, no data will be displayed in the various windows. For Mac OSX users, you need to have XQuartz or X11 installed for Wireshark to work!!! Also, the first time you open Wireshark, it will take several seconds to start so be patient.

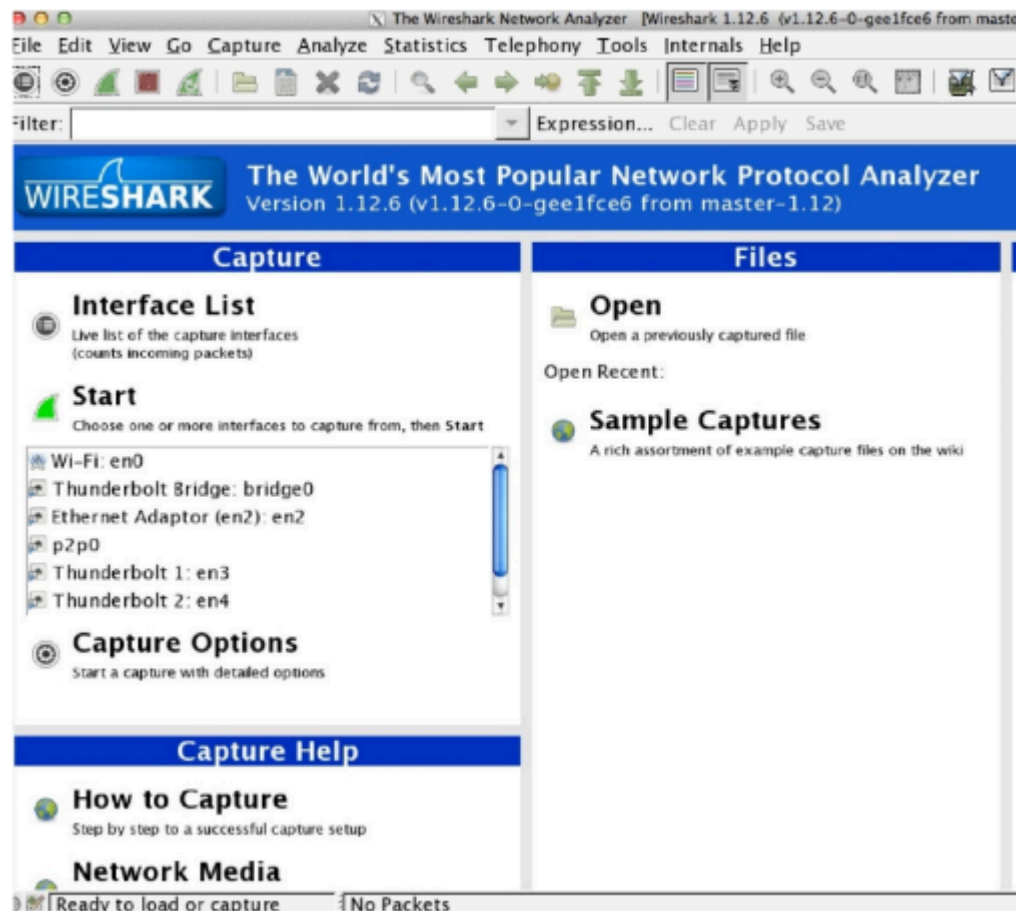


Figure 3: Initial screen for Wireshark.

Need to select one of the Wireshark interfaces, if you are using your laptop connected over the WiFi, then you need to select the WiFi interface. If you are at a server, you need to select the Ethernet interface being used. In general, you can select any interface but that does not mean that traffic will flow through that interface. The network interfaces (i.e., the physical connections) that your computer has to the network are shown. The attached snapshot was taken from my computer. You may not see the exact same entries when you perform a capture in the 237 Lab. You will notice that eth0 and eth1 will be displayed. Click “Start” for interface eth0. Packet capture will now begin- all packets being sent / received from/by your computer are now being captured by Wireshark! After you select the interface, you should click on “START”.

What is Wireshark?

Wireshark is a network packet analyzer. A network packet analyzer presents captured packet data in as much detail as possible. You could think of a network packet analyzer as a measuring device for examining what's happening inside a network cable, just like an electrician uses a voltmeter for examining

what's happening inside an electric cable (but at a higher level, of course)

Output

Capturing from Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
264	22.821735	2401:4900:2357:3220::	2603:1020:1001:6::39	TLSv1.2	112	Application Data
265	22.837508	62.128.100.47	192.168.43.66	TCP	54	443 → 54020 [ACK] Seq=1 Ack=2 Win=64028 Len=0
266	22.883131	192.168.43.66	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
267	23.051286	2603:1020:1001:6::39	2401:4900:2357:3220::	TLSv1.2	482	Application Data
268	23.051523	2401:4900:2357:3220::	2603:1020:1001:6::39	TCP	74	54029 → 443 [ACK] Seq=862 Ack=6614 Win=261376 Len=0
269	23.064778	2401:4900:2357:3220::	2603:1020:1001:6::39	TLSv1.2	157	Application Data
270	23.110804	2603:1020:1001:6::39	2401:4900:2357:3220::	TCP	74	443 → 54029 [ACK] Seq=6614 Ack=862 Win=4194048 Len=0
271	23.325827	2603:1020:1001:6::39	2401:4900:2357:3220::	TLSv1.2	213	Application Data
272	23.326066	2401:4900:2357:3220::	2603:1020:1001:6::39	TCP	74	54029 → 443 [ACK] Seq=945 Ack=6753 Win=261376 Len=0
273	23.341642	192.168.43.66	204.79.197.222	TLSv1.2	886	Application Data
274	23.417052	204.79.197.222	192.168.43.66	TCP	54	443 → 54021 [ACK] Seq=1 Ack=833 Win=16383 Len=0
275	23.419126	204.79.197.222	192.168.43.66	TLSv1.2	207	Application Data
276	23.419344	192.168.43.66	204.79.197.222	TCP	54	54021 → 443 [ACK] Seq=833 Ack=154 Win=1022 Len=0
277	23.421622	204.79.197.222	192.168.43.66	TLSv1.2	92	Application Data
278	23.421828	192.168.43.66	204.79.197.222	TCP	54	54021 → 443 [ACK] Seq=833 Ack=192 Win=1022 Len=0
279	23.791567	2401:4900:2357:3220::	2600:140f:2400::173::	TLSv1.2	334	Application Data

> Frame 1: 75 bytes on wire (600 bits), 75 bytes captured (600 bits) on interface \Device\NPF_{...}

> Ethernet II, Src: HonHaiPrecis_15:b1:d9 (9c:2a:70:15:b1:d9), Dst: XiaomiCommun_9c:0c:3d (00:ec:00:00:00:00)

> Internet Protocol Version 6, Src: 2401:4900:2357:3220:5057:33f2:a62e:9fcd, Dst: 2404:6800:4003:c1c1::

> Transmission Control Protocol, Src Port: 53929, Dst Port: 5228, Seq: 1, Ack: 1, Len: 1

> Data (1 byte)

0000 00 ec 0a 9c 0c 3d 9c 2a 70 15 b1 d9 86 dd 60 01* p.....
0010 02 36 00 15 06 ff 24 01 49 00 23 57 32 20 50 57\$ I #W2 PW
0020 33 f2 a6 2e 9f cd 24 04 68 00 40 03 0c 1c 00 00 3.....\$ h@.....
0030 00 00 00 00 00 bc d2 a9 14 6c 8a a2 36 f7 9f c51..6...
0040 cc 32 50 10 01 fe 33 90 00 00 002P...3... ..

*Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp

No.	Time	Source	Destination	Protocol	Length	Info
362	81.288576	2401:4900:2357:3220::	2603:1046:1400:1::1	TLSv1.2	109	Application Data
363	81.470710	2603:1046:1400:1::1	2401:4900:2357:3220::	TCP	74	443 → 54002 [ACK] Seq=1 Ack=106 Win=16386 Len=0
364	82.018424	2401:4900:2357:3220::	2603:1046:1400:1::1	TLSv1.2	109	Application Data
365	82.208248	2603:1046:1400:1::1	2401:4900:2357:3220::	TCP	74	443 → 54003 [ACK] Seq=1 Ack=106 Win=16385 Len=0
366	82.852549	192.168.43.66	62.128.100.47	TCP	55	[TCP Keep-Alive] 54020 → 443 [ACK] Seq=1 Ack=1 Win=63798 Len=1
368	83.165929	62.128.100.47	192.168.43.66	TCP	54	[TCP Keep-Alive ACK] 443 → 54020 [ACK] Seq=1 Ack=2 Win=64028 Len=0
369	84.471059	192.168.43.66	62.128.100.47	TCP	66	54031 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
370	84.823278	62.128.100.47	192.168.43.66	TCP	58	443 → 54031 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1300
371	84.823564	192.168.43.66	62.128.100.47	TCP	54	54031 → 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
372	84.823964	192.168.43.66	62.128.100.47	TCP	54	54031 → 443 [FIN, ACK] Seq=1 Ack=1 Win=64240 Len=0
373	85.090947	62.128.100.47	192.168.43.66	TCP	54	443 → 54031 [FIN, ACK] Seq=1 Ack=2 Win=64999 Len=0
374	85.090246	192.168.43.66	62.128.100.47	TCP	54	54031 → 443 [ACK] Seq=2 Ack=2 Win=64240 Len=0
377	87.060964	192.168.43.66	38.113.165.18	TCP	54	54030 → 443 [FIN, ACK] Seq=316 Ack=136 Win=131072 Len=0
378	87.569241	38.113.165.18	192.168.43.66	TCP	54	443 → 54030 [RST, ACK] Seq=136 Ack=317 Win=64128 Len=0
383	90.275490	2401:4900:2357:3220::	2404:6800:4003:c1c1::	TCP	75	[TCP Keep-Alive] 53929 → 5228 [ACK] Seq=1 Ack=1 Win=510 Len=1
384	90.506987	2404:6800:4003:c1c1::	2401:4900:2357:3220::	TCP	86	[TCP Keep-Alive ACK] 5228 → 53929 [ACK] Seq=1 Ack=2 Win=265 Len=0 SLE=1 SRE=2

> Frame 1: 75 bytes on wire (600 bits), 75 bytes captured (600 bits) on interface \Device\NPF_{...}

> Ethernet II, Src: HonHaiPrecis_15:b1:d9 (9c:2a:70:15:b1:d9), Dst: XiaomiCommun_9c:0c:3d (00:ec:00:00:00:00)

> Internet Protocol Version 6, Src: 2401:4900:2357:3220:5057:33f2:a62e:9fcd, Dst: 2404:6800:4003:c1c1::

> Transmission Control Protocol, Src Port: 53929, Dst Port: 5228, Seq: 1, Ack: 1, Len: 1

> Data (1 byte)

0000 00 ec 0a 9c 0c 3d 9c 2a 70 15 b1 d9 86 dd 60 01* p.....
0010 02 36 00 15 06 ff 24 01 49 00 23 57 32 20 50 57\$ I #W2 PW
0020 33 f2 a6 2e 9f cd 24 04 68 00 40 03 0c 1c 00 00 3.....\$ h@.....
0030 00 00 00 00 00 bc d2 a9 14 6c 8a a2 36 f7 9f c51..6...
0040 cc 32 50 10 01 fe 33 90 00 00 002P...3... ..

Result:

Thus the experiment to install Wire Shark, tcpdump and observe data transferred in client server communication using UDP/TCP and identify the UDP/TCP datagram was analyzed successfully.

Ex.no:6

Explore Network Monitoring Tools

AIM:

To explore network monitoring tools.

Tools Used

- SolarWinds Network Performance Monitor
- Auvik
- Datadog Network Monitoring
- Paessler PRTG Network Monitor
- ManageEngine OpManager
- Domotz
- Checkmk
- Progress Whatsup Gold

THEORY

- Network monitoring tools are software that can use to evaluate network connections. These software programs can help you monitor a network connection and identify network issues, which may include failing network components, slow connection speed, network outage or unidentifiable connections.
- The Network management and monitoring tools can also help you resolve these issues or establish solutions that prevent specific issues from occurring in the future.

1. SolarWinds Network Performance Monitor

- It is a multi-vendor monitoring tool. It allows users to monitor multiple vendors' networks at the same time. It also provides network insights for thorough visibility into the health of the networks. Some prominent features include network availability monitoring, intelligent network mapping, critical path visualisation, performance analysis and advanced alerting.

2. Auvik

- It is a network monitoring and management tool. It offers a quick implementation process that helps users to set up the tool easily. It also has a clean user interface that makes it easy to navigate and use. The tool provides in-depth network visibility that enables faster troubleshooting for network issues.
- Users can automate network visibility using Auvik. It provides real-time updates on network issues and configuration changes.

3. Datadog

- Network Monitoring offers services for on-premises devices and cloud networks. A highlighting feature of this tool is the visualisations. It offers various graphical representations of all the network connections on a system. It also allows users to track key metrics like network latency, connection churn and transmission control protocol (TCP) retransmits.

4. Paessler's

- The network connection monitoring tool provides a clean user interface and network visibility on multiple devices. Users can track the health of different connection types like local area networks (LAN), wide area network (WAN), servers, websites, applications and services.

5. ManageEngine OpManager

- It is a good network monitoring and managing tool for users that prefer in-depth view of network health and issues. This tool provides over 2000 network performance monitors that allow users to track and monitor their connections and perform detailed analyses on issues.

6. Domotz

- It is an expansive tool that provides a list of features for monitoring network connections. It allows users to customise their network monitoring preferences. Users can write scripts to retrieve the data they wish to evaluate. It also allows connection to open ports on remote devices while ensuring network security.

7. Checkmk

- It is a tool that allows users to automate it completely. You can customise its operations and enable it to perform tasks automatically. It also identifies network and security components without the user.

8. Progress Whatsup Gold

- It is a basic network monitoring software. It provides a minimal user interface with essential features like device monitoring, application monitoring, analysing network traffic and managing configurations. The tool allows users to monitor cloud devices.

RESULT

Thus to explore network monitoring tools was done successfully.

What is Wireshark?

Wireshark is a network packet analyzer. A network packet analyzer presents captured packet data in as much detail as possible. You could think of a network packet analyzer as a measuring device for examining what's happening inside a network cable, just like an electrician uses a voltmeter for examining

what's happening inside an electric cable (but at a higher level, of course)

Ex.no:7

MD5 Algorithm

AIM:

To Calculate the message digest of a text using the MD5 algorithm.

ALGORITHM:

1. Append Padding Bits
2. Append Length - 64 bits are appended to the end
3. Prepare Processing Functions
4. Prepare Processing Constants
5. Initialize Buffers
6. Processing Message in 512-bit blocks (L blocks in total message)

Program

```
import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

// Java program to calculate MD5 hash value
public class MD5 {
    public static String getMd5(String input)
    {
        try {
            // Static getInstance method is called with hashing MD5
            MessageDigest md = MessageDigest.getInstance("MD5");
            // digest() method is called to calculate message digest
            // of an input digest() return array of byte
            byte[] messageDigest = md.digest(input.getBytes());
```



```

        // Convert byte array into signum representation
        BigInteger no = new BigInteger(1, messageDigest);
        // Convert message digest into hex value
        String hashtext = no.toString(16);
        while (hashtext.length() < 32) {
            hashtext = "0" + hashtext;
        }
        return hashtext;
    }
    // For specifying wrong message digest algorithms
    catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(e);
    }
}
// Driver code
public static void main(String args[]) throws NoSuchAlgorithmException
{
    String s = "Stella Mary's";
    System.out.println("Your HashCode Generated by MD5 is: " + getMd5(s));
}
}

```

Output

Your HashCode Generated by MD5 is: 080d94bef2707a33e5bc20f336e4bc46

RESULT

Thus the Message Digest 5 (MD5) has been implemented and the output has been verified successfully.

What is Wireshark?

Wireshark is a network packet analyzer. A network

packet analyzer presents captured packet data in as much detail as possible.

You could think of a network packet analyzer as a measuring device for

examining what's happening inside a network cable, just like an electrician uses a voltmeter for

examining what's happening inside an electric cable (but at a higher level, of course)

Ex.no:8

Confidentiality and Authentication

AIM:

To write a java program for implement Message Authentication Code to examine confidentiality and authentication.

ALGORITHM:

1. Import Key generator library files.
2. Generate secret key.
3. Prepare the message to be sent to receiver.
4. Using SHA512 generate hash code.
5. Combine message and hash code and forward to receiver.
6. validate hash code.

Program

```
import javax.crypto.KeyGenerator;
import javax.crypto.Mac;
import javax.crypto.SecretKey;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;

public class MACExample {
    public static void main(String[] args) {
        try {
            // Step 1: Generate a symmetric key (K)
            KeyGenerator keygen = KeyGenerator.getInstance("HmacSHA512");
            SecretKey secretKey = keygen.generateKey();

            // Step 2: Compute MAC for a sample message (M)
            String message = "Hello, world!";
            Mac mac = Mac.getInstance("HmacSHA512");
            mac.init(secretKey);
            byte[] macBytes = mac.doFinal(message.getBytes());
            // Display the MAC (authentication tag)
            System.out.println("Message: " + message);
            System.out.println("MAC: " + bytesToHex(macBytes));
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        }
    }
}
```

```

    }
    catch (InvalidKeyException e) {
        e.printStackTrace();
    }
}
// Helper method to convert bytes to hexadecimal
private static String bytesToHex(byte[] bytes) {
    StringBuilder result = new StringBuilder();
    for (byte b : bytes) {
        result.append(String.format("%02x", b));
    }
    return result.toString();
}
}

```

Output

Message: Hello, world!

MAC:

4b2b5a76f88d272185b00549bbbea7218359c6f4bb4a6295d67d20808e0cb9219b6bdfb9e494e08b0e98e37c2830e8d21f513a1c2a2ae912794c57a6504f89c8

RESULT

Thus the java program for ensure confidentiality and authentication has been implemented and the output has been verified successfully.

What is Wireshark?

Wireshark is a network packet analyzer. A network packet analyzer presents captured packet data in as

much detail as possible.

You could think of a network packet analyzer as a measuring device for

examining what's happening inside a network cable, just like an electrician uses a voltmeter for examining

what's happening inside an electric cable (but at a higher level, of course)

Ex.no:9

Dictionary Attack

AIM:

Write a Java program to implement and show how to protect your system from the dictionary attack.

Dictionary Attack

- A dictionary attack is a type of password cracking technique used to gain unauthorized access to a system or account. In this method, an attacker utilizes a list of common words or phrases, often sourced from a dictionary or a wordlist, and systematically tries each entry as a password until access is granted.
- The process involves automated software that sequentially tests each word or phrase from the dictionary against the authentication mechanism (such as a login page) until it finds a match. This approach exploits the tendency of many users to choose weak passwords based on easily guessable terms, like common words, phrases, or easily predictable variations thereof.
- To enhance the effectiveness of a dictionary attack, attackers often use variations in spelling, common substitutions (such as replacing 'o' with '0' or 'e' with '3'), appending numbers or special characters, and other common password patterns.
- To protect against dictionary attacks, users are encouraged to choose strong, unique passwords that combine uppercase and lowercase letters, numbers, and special characters, making them more resistant to brute-force and dictionary-based cracking attempts.

ALGORITHM:

1. Import necessary library files.
2. Create class dictionary_atack_demo.
3. Prepare the more common password.
4. User scanner to take a input from keyboard.
5. Enter the password and check the strength.
6. Give the proper response based on the password.

Program

```
import java.util.Scanner;

public class DictionaryAttackDemo {

    public static void main(String[] args) {

        // Simulating a dictionary of common passwords
```

```

String[] dictionary = {
    "password", "123456", "qwerty", "admin", "letmein"
    // Add more common passwords here
};
Scanner scanner = new Scanner(System.in);
System.out.print("Enter your password: ");
String userPassword = scanner.nextLine();
// Check if the entered password is in the dictionary
boolean isPasswordInDictionary = false;
for (String commonPassword : dictionary) {
    if (userPassword.equals(commonPassword)) {
        isPasswordInDictionary = true;
        break;
    }
}
if (isPasswordInDictionary) {
    System.out.println("Password is weak! Please choose a stronger one.");
} else {
    System.out.println("Password is secure. Good job!");
}
scanner.close();
}
}

```

Output

Enter your password: 123456

Password is weak! Please choose a stronger one.

Enter your password: dfsdfds

Password is secure. Good job!

RESULT

Thus the java program for protect the password from dictionary attack has been implemented and the output has been verified successfully.

What is Wireshark?

Wireshark is a network packet analyzer. A network packet analyzer presents captured packet data in as much detail as possible.

You could think of a network packet analyzer as a measuring device for

examining what's happening inside a network cable, just like an electrician uses a voltmeter for examining

what's happening inside an electric cable (but at a higher level, of cours

Ex.no:10

Create Firewall Setup

AIM:

Write a Java program to create firewall setup.

Firewall

- A firewall can be defined as a special type of network security device or a software program that monitors and filters incoming and outgoing network traffic based on a defined set of security rules. It acts as a barrier between internal private networks and external sources (such as the public Internet).
- The primary purpose of a firewall is to allow non-threatening traffic and prevent malicious or unwanted data traffic for protecting the computer from viruses and attacks. A firewall is a cybersecurity tool that filters network traffic and helps users block malicious software from accessing the [Internet](#) in infected computers.

ALGORITHM:

1. Import necessary library files.
2. Open Windows Defender Firewall.
3. Click Advanced Settings->click Inbound Rules.
4. Apply firewall rules.

Program

```
public class FirewallConfigurator {  
  
    public static void addFirewallRule(String ruleName) {  
        try {  
            String command = "netsh advfirewall firewall add rule name=\"" + ruleName + "\"  
protocol=TCP dir=in localport=80 security=authdynenc action=allow";  
            Process process = Runtime.getRuntime().exec(command);  
        }  
    }  
}
```

```

process.waitFor();
if (process.exitValue() == 0) {
    System.out.println("Firewall rule added successfully.");
} else {
    System.out.println("Error adding firewall rule.");
}
} catch (Exception e) {
    e.printStackTrace();
}
}
public static void main(String[] args) {
    // Example usage:
    addFirewallRule("My Port 82");
}
}

```

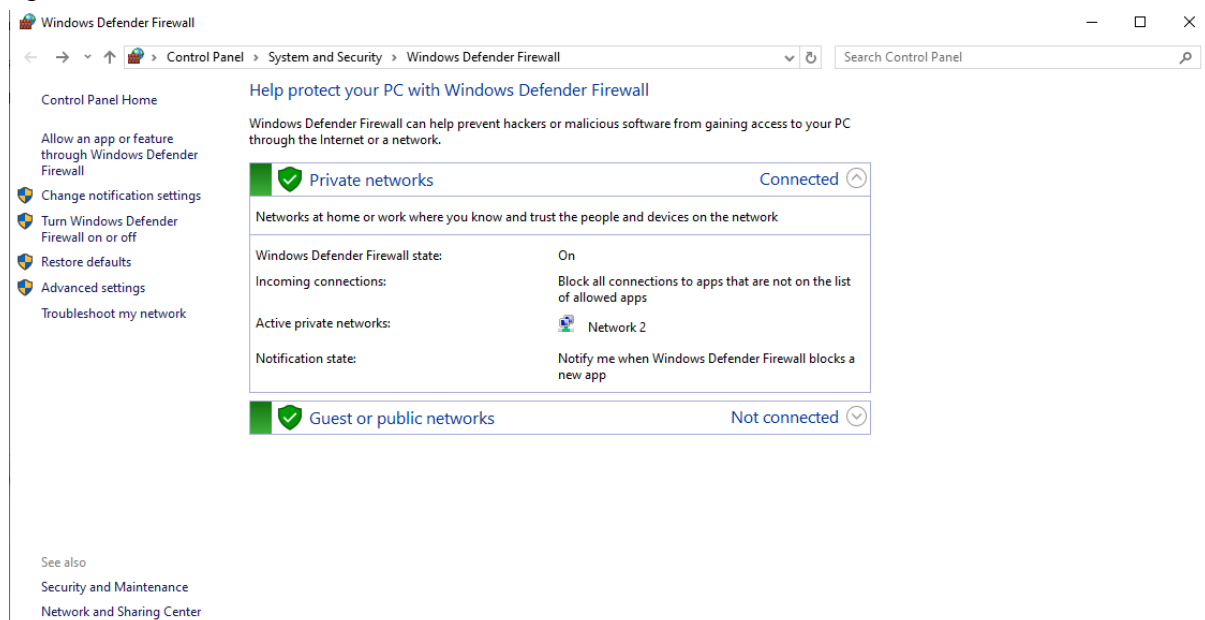
Output

Run this program in Administrator Command Prompt

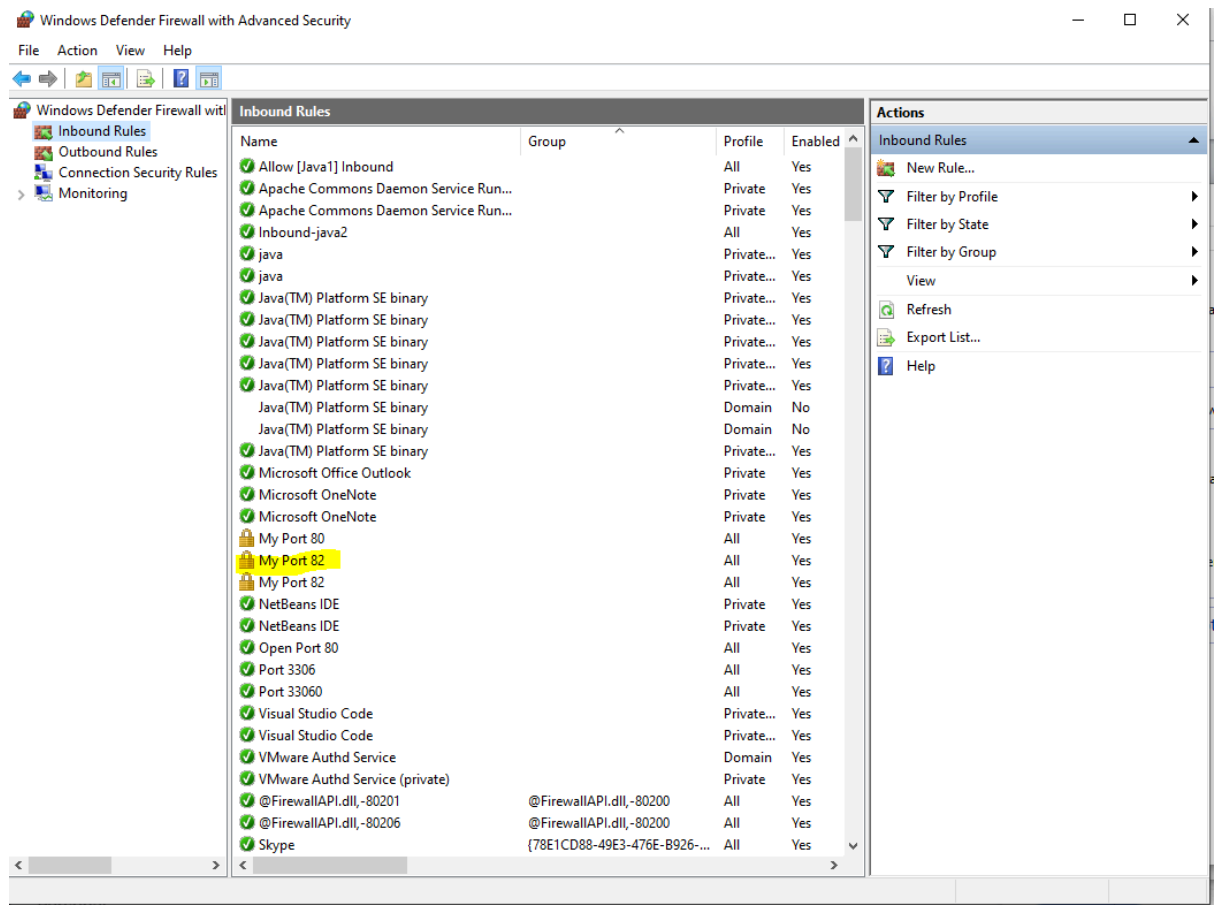
C:\rr>java FirewallConfigurator

Firewall rule added successfully.

1) Open Windows Defender Firewall



2) Click Advanced Settings->click Inbound Rules



Firewall Created Successfully

RESULT

Thus the java program for create firewall setup and the output has been verified successfully.

What is Wireshark?

Wireshark is a network packet analyzer. A network

packet analyzer presents captured packet data in as much detail as possible.

You could think of a network packet analyzer as a measuring device for

examining what's happening inside a network cable, just like an electrician uses a voltmeter for

examining what's happening inside an electric cable (but at a higher level, of course)

Ex.no:11**Setup and Configure VPN Connection using JAVA Commands****AIM:**

Write a Java program to Setup and Configure VPN Connection.

VPN

- VPN stands for Virtual Private Network. It's a technology that allows you to create a secure connection over the internet, typically between your device and a private network, such as a corporate network or a remote server.
- VPN configurations typically involve interacting with the underlying operating system's networking stack, which usually requires administrative privileges and access to system-level APIs, neither of which are readily available from within a Java program.
- However, you can interact with VPN configurations indirectly using Java by executing system commands. Below, I'll outline a basic approach using Java's ProcessBuilder class to execute shell commands for setting up a VPN connection.

ALGORITHM:

5. Import necessary library files.
6. Replace `sudo openvpn --config /path/to/config.ovpn` with the actual command you use to start your VPN connection.
7. ProcessBuilder is used to execute the command.
8. `redirectErrorStream(true)` merges the standard output and standard error streams, so you can handle both through the `InputStreamReader`.

Program

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
public class VPNConfiguration {
    public static void main(String[] args) {
        try {
            // Replace these commands with the actual commands for setting up the VPN
            connection
            String[] commands = {"sudo", "openvpn", "--config", "/path/to/config.ovpn"};
            ProcessBuilder builder = new ProcessBuilder(commands);
```

```

        builder.redirectErrorStream(true);
        Process process = builder.start();
        // Read output from the process
        BufferedReader reader = new BufferedReader(new
InputStreamReader(process.getInputStream()));
        String line;
        while ((line = reader.readLine()) != null) {
            System.out.println(line);
        }
        // Wait for the process to finish
        int exitCode = process.waitFor();
        System.out.println("Process exited with code " + exitCode);
    } catch (IOException | InterruptedException e) {
        e.printStackTrace();
    }
}
}
}

```

Output

Run this program in Administrator Command Prompt

waitFor() is called to wait for the process to finish and get the exit code.

Process exited with code 690

RESULT

Thus the java program for setup and configure VPN connection and the output has been verified successfully.

What is Wireshark?

Wireshark is a network packet analyzer. A network packet

analyzer presents captured packet data in as much detail as possible. You could think of a network packet analyzer as a measuring device for examining what's happening inside a network cable, just like an electrician uses a voltmeter for examining what's happening inside an electric cable (but at a higher level, of course)