P.T.LEE CHENGALVARAYA NAICKER COLLEGE OF ENGINEERING & TECHNOLOGY

(Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai)

Vallal P.T.Lee Chengalvaraya Naicker Nagar, Oovery, Kanchipuram – 631 502

Department of Computer Science and Engineering

Regulation - 2021

CCS354 – NETWORK SECURITY

RECORD

NAME :

REG.NO :

YEAR/SEMESTER :

P.T.LEE CHENGALVARAYA NAICKER COLLEGE OF ENGINEERING & TECHNOLOGY

(Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai)

Vallal P.T.Lee Chengalvaraya Naicker Nagar, Oovery, Kanchipuram – 631 502

BONAFIDE CERTIFICATE

This is to certify that record work done by_____with Reg. No: _____of_____Year_____Semester Bachelor of Computer Science and Engineering, in the **CCS354 – NETWORK SECURITY LABORATORY** during the academic year 2023-24.

Lab-in-charge Head of the Department

Submitted for the Practical Examination held on _____

Internal Examiner External Examiner

INDEX

S.No	DATE	EXPERIMENTS	PAGE NO	MARK	SIGN
1		IMPLEMENT SYMMETRIC KEY ALGORITHMS			
2		ASYMMETRIC KEY ALGORITHMS AND KEY EXCHANGE ALGORITHMS			
3		IMPLEMENT DIGITAL SIGNATURE SCHEMES			
4		INSTALLATION OF WIRE SHARK, TCPDUMP			
5		CHECK MESSAGE INTEGRITY AND CONFIDENTIALITY USING SSL			
6(a)		EAVESDROPPING			
6(b)		DICTIONARY ATTACKS			
6(c)		MAN-IN-THE-MIDDLE			
7		SNIFF TRAFFIC USING ARP POISONING			
8		INTRUSION DETECTION			
9		NETWOTK MONITORING TOOLS			
10(a)		FIREWALL			
10(b)		VPN			

EX.NO: 1	IMPLEMENT SYMMETRIC KEY ALGORITHMS
DATE:	

Aim:

To implement symmetric key algorithms for secure data encryption and decryption.

ALGORITHM:

Encryption Steps:

1. Initialize the round keys using the main symmetric key.
2. Break the plaintext into blocks, padding the last block if necessary.
3. For each block: a. Perform the initial round key addition. b. Perform multiple rounds (10, 12, or 14 rounds based on key length):

- Byte substitution using a substitution box (S-box).
- Row shifting within the block.
- Column mixing within the block.
- Round key addition using the current round key. c. Perform the final round without the column mixing step.

4. Combine the encrypted blocks to create the ciphertext.

Decryption Steps:

1. Initialize the round keys using the main symmetric key.
2. Break the ciphertext into blocks.
3. For each block: a. Perform the initial round key addition. b. Perform the reverse of the encryption rounds in the reverse order:

- Inverse byte substitution using an inverse S-box.
- Inverse row shifting within the block.
- Inverse round key addition using the current round key.
- Inverse column mixing within the block (if not in the final round). c. Perform the final round without the inverse column mixing step.

4. Combine the decrypted blocks to recover the original plaintext.
5. Remove any padding added during encryption to obtain the actual original plaintext.

Program:

```
importjavax.crypto.Cipher;
importjavax.crypto.KeyGenerator;
importjavax.crypto.SecretKey;
import java.util.Base64;

public class SymmetricEncryptionExample {

public static void main(String[] args) throws Exception {
        String plaintext = "Hello, symmetric encryption!";
System.out.println("Original Text: " + plaintext);

        // Generate a secret key
KeyGeneratorkeyGen = KeyGenerator.getInstance("AES");
keyGen.init(128); // You can choose 128, 192, or 256 bits
SecretKeysecretKey = keyGen.generateKey();

        // Encryption
        Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
cipher.init(Cipher.ENCRYPT_MODE, secretKey);
byte[] encryptedBytes = cipher.doFinal(plaintext.getBytes("UTF-8"));

System.out.println("Encrypted Text: " +
Base64.getEncoder().encodeToString(encryptedBytes));

        // Decryption
cipher.init(Cipher.DECRYPT_MODE, secretKey);
byte[] decryptedBytes = cipher.doFinal(encryptedBytes);
        String decryptedText = new String(decryptedBytes, "UTF-8");

System.out.println("Decrypted Text: " + decryptedText);
    }
}
```

Output:

Original Text: Hello, symmetric encryption!

Encrypted Text: XnO8JLmjRhKwx/mLPJ8zXyz6kfONJ6LYH8C75KxVvwk=

Decrypted Text: Hello, symmetric encryption!

Result:

Thus the java program to implement symmetric key algorithms is verified successfully and output is verified.

EX.NO.:2	ASYMMETRIC KEY ALGORITHMS AND KEY EXCHANGE ALGORITHMS
DATE:	

Aim:

To implement asymmetric key algorithms and key exchange algorithms using Java Program.

Algorithm:

Asymmetric Key Algorithms (RSA):

Key Generation:

1. Generate a pair of mathematically related keys: public key (for encryption) and private key (for decryption).
2. Select two large prime numbers, `p` and `q`.
3. Calculate the modulus `n = p * q`.
4. Calculate the totient `phi(n) = (p - 1) * (q - 1)`.
5. Choose an integer `e` (the public exponent) such that `1 < e <phi(n)` and `gcd(e, phi(n)) = 1`.
6. Calculate the modular multiplicative inverse `d` of `e` modulo `phi(n)` (`d * e ≡ 1 (mod phi(n))`).

Encryption:

1. Obtain the recipient's public key `(e, n)`.
2. Break the plaintext into smaller blocks.
3. For each block, calculate the ciphertext `c = m^e mod n`, where `m` is the plaintext block.

Decryption:

1. Use the recipient's private key `d` to calculate the original message `m = c^d mod n`, where `c` is the ciphertext block.

Key Exchange Algorithms (Diffie-Hellman):

Key Generation:

1. Each party selects a large prime number `p` and a primitive root modulo `p`, denoted as `g`.

Key Exchange:

1. Both parties select their secret private keys, `a` and `b`, respectively.

2. They each calculate their public keys: `A = g^a mod p` and `B = g^b mod p`.
3. They exchange these public keys.
4. Using the received public keys, each party calculates the shared secret: `s = B^a mod p` and `s = A^b mod p`. Both parties now have the same shared secret `s`.

Program:

```
importjava.security.KeyPair;
importjava.security.KeyPairGenerator;
importjava.security.KeyAgreement;
importjavax.crypto.Cipher;

public class AsymmetricAndKeyExchangeExample {

public static void main(String[] args) throws Exception {
        // Asymmetric Key Encryption using RSA
        String plaintext = "Hello, asymmetric encryption!";
System.out.println("Original Text: " + plaintext);

        // Generate key pair
KeyPairGeneratorkeyPairGenerator = KeyPairGenerator.getInstance("RSA");
keyPairGenerator.initialize(2048); // Key length
KeyPairkeyPair = keyPairGenerator.generateKeyPair();

        // Encryption
        Cipher cipher = Cipher.getInstance("RSA");
cipher.init(Cipher.ENCRYPT_MODE, keyPair.getPublic());
byte[] encryptedBytes = cipher.doFinal(plaintext.getBytes());

System.out.println("Encrypted Text: " + new String(encryptedBytes));

        // Decryption
cipher.init(Cipher.DECRYPT_MODE, keyPair.getPrivate());
byte[] decryptedBytes = cipher.doFinal(encryptedBytes);
```

```
        String decryptedText = new String(decryptedBytes);

System.out.println("Decrypted Text: " + decryptedText);

        // Key Exchange using Diffie-Hellman
KeyPairGeneratordhKeyPairGenerator = KeyPairGenerator.getInstance("DiffieHellman");
dhKeyPairGenerator.initialize(2048); // Key length
KeyPairdhKeyPair = dhKeyPairGenerator.generateKeyPair();

KeyAgreementkeyAgreement = KeyAgreement.getInstance("DiffieHellman");
keyAgreement.init(dhKeyPair.getPrivate());

        // Simulate another party by generating its key pair
KeyPairdhOtherKeyPair = dhKeyPairGenerator.generateKeyPair();
keyAgreement.doPhase(dhOtherKeyPair.getPublic(), true);

byte[] sharedSecret = keyAgreement.generateSecret();
System.out.println("Shared Secret: " + new String(sharedSecret));
    }
}
```

Output:

Original Text: Hello, asymmetric encryption!

Encrypted Text: ...

Decrypted Text: Hello, asymmetric encryption!

Shared Secret: ...

Result:

Thus the java program to asymmetric key algorithms and key exchange algorithms is executed successfully and output is verified.

EX.NO.:3	IMPLEMENT DIGITAL SIGNATURE SCHEMES
DATE:	

Aim:

To implement digital signature scheme using java program

Algorithm:

Step 1: Key Pair Generation

Generate an RSA key pair with a specified key length.

Share the public key and private key with appropriate security measures.

Step 2: Creating a Digital Signature

Choose a message to be digitally signed.

Use the private key to generate a digital signature for the message.

Attach the digital signature to the message.

Step 3: Signature Verification

Receive a signed message and the associated digital signature.

Extract the public key from the sender.

Verify the authenticity of the message using the digital signature and the public key.

Step 4: Observation and Analysis

Observe the successful verification of authentic messages.

Attempt to verify messages with altered content or incorrect signatures to understand the verification process.

Step 5: Discussion and Reflection

Discuss the importance of digital signatures in ensuring data integrity and authentication.

Reflect on how the private key's security impacts the overall security of digital signatures.

Program:

```
importjava.security.*;
import java.util.Base64;
public class DigitalSignatureExample {
public static void main(String[] args) throws Exception {
        String message = "Hello, this is a message for digital signature!";
System.out.println("Original Message: " + message);
        // Generate key pair
KeyPairGeneratorkeyPairGenerator = KeyPairGenerator.getInstance("RSA");
keyPairGenerator.initialize(2048); // Key length
KeyPairkeyPair = keyPairGenerator.generateKeyPair();
        // Create a digital signature
        Signature signature = Signature.getInstance("SHA256withRSA");
signature.initSign(keyPair.getPrivate());
signature.update(message.getBytes());
byte[] digitalSignature = signature.sign();
System.out.println("Digital Signature: " +
Base64.getEncoder().encodeToString(digitalSignature));
        // Verify the digital signature
        Signature verifier = Signature.getInstance("SHA256withRSA");
verifier.initVerify(keyPair.getPublic());
verifier.update(message.getBytes());
booleanisVerified = verifier.verify(digitalSignature);
if (isVerified) {
```

```
System.out.println("Digital Signature Verified: The message is authentic.");
        } else {
System.out.println("Digital Signature Verification Failed: The message may be tampered with.");
        }
    }
}
```

Output:

Original Message: Hello, this is a message for digital signature!

Digital Signature: ...

Digital Signature Verified: The message is authentic.

Result:

Thus the java program to digital signature schemes is executed successfully and output is verified.

EX.NO.:4	INSTALLATION OF WIRE SHARK, TCPDUMP
DATE:	

Aim:

The aim of this lab activity is to install Wireshark and tcpdump, observe data transferred in client-server communication using UDP/TCP protocols, and identify UDP/TCP datagrams for better understanding of network traffic analysis and protocol behavior.

Algorithm:

Step 1: Install Wireshark and tcpdump

Download and install Wireshark from the official website.

Install tcpdump using the package manager (e.g., apt-get, yum) on Linux systems.

Step 2: Set Up Client-Server Communication

Set up a basic client-server communication scenario using UDP or TCP (e.g., sending messages from a client to a server).

Step 3: Capture Network Traffic

Open Wireshark and choose the appropriate network interface.

Start capturing traffic to monitor the communication.

Step 4: Observe UDP Datagram

Filter captured packets to display only UDP packets.

Analyze the captured UDP datagrams, noting source and destination ports, payload data, and other relevant information.

Step 5: Observe TCP Segment

Filter captured packets to display only TCP packets.

Analyze the captured TCP segments, noting source and destination ports, sequence and acknowledgment numbers, flags (such as SYN, ACK), and payload data.

Step 6: Discussion and Analysis

Compare the characteristics of UDP datagrams and TCP segments observed in the captured network traffic.

Discuss the advantages and disadvantages of using UDP and TCP for different types of applications.

Step 7: Reflection

Reflect on the importance of network analysis tools in understanding network communication and troubleshooting.

Program:

Step 1: Install Wireshark and tcpdump:

Installing Wireshark:

1. Visit the official Wireshark website: https:/www.wireshark.org/

2. Download and install Wireshark for your operating system (Windows, macOS, or Linux).

Installing tcpdump:

- On Linux:

  - Open a terminal.

  - Use the package manager of your distribution to install tcpdump.

  - For example, on Debian/Ubuntu, you can use: `sudo apt-get install tcpdump`

  - On CentOS/RHEL, you can use: `sudo yum install tcpdump`

Step 2: Observe Data Transferred in Client-Server Communication:

Setting Up a Simple Communication Scenario:

1. Choose a client and server machine, preferably on the same network.

2. Decide whether you'll be using UDP or TCP for communication.

3. Write a simple client-server program or use command-line tools (e.g., `nc` for TCP, `ncat` for UDP) to simulate communication.

Capturing Network Traffic:

- Wireshark:

  1. Open Wireshark.

  2. Select the network interface through which the client and server communicate.

  3. Click the "Start" button to capture traffic.

- **tcpdump:**

  1. Open a terminal.

  2. Use the following command to capture traffic on a specific interface:

sudotcpdump -i <interface> -w capture.pcap

3. Replace `<interface>` with the appropriate network interface.

Step 3: Identify UDP/TCP Datagram:

1. After capturing traffic, you can stop the capture in Wireshark or tcpdump.

2. In Wireshark:

- You will see a list of captured packets.

- Use display filters to focus on UDP or TCP traffic. For example:

- `udp` to show only UDP packets.

- `tcp` to show only TCP packets.

- Click on a packet to view its details, including source and destination ports, payload data, and more.

3. Intcpdump:

- Open the captured pcap file with Wireshark for graphical analysis.

- Use the same display filters (`udp`, `tcp`) to focus on UDP or TCP packets.

Identifying UDP Datagram:

- In Wireshark or tcpdump, look for packets with a UDP protocol. You'll see source and destination port numbers and the payload data.

Identifying TCP Segment:

- In Wireshark or tcpdump, look for packets with a TCP protocol. You'll see source and destination port numbers, sequence numbers, acknowledgment numbers, flags (such as SYN, ACK), and the payload data.

Result:

Thus the command for various activities is executed successfully and output is verified

EX.NO.:5	CHECK MESSAGE INTEGRITY AND CONFIDENTIALITY USING SSL
DATE:	

Aim:

Write a Java program to check message integrity and confidentiality using ssl

Algorithm:

Step 1: Key Pair Generation

Generate an RSA key pair with a specified key length.

Share the public key and private key with appropriate security measures.

Step 2: Creating aintegrity and confidentiality

Choose a message to be data confidential.

Use the private key to generate a digital signature for the message.

Attach the SSL to the message.

Step 3: Verification for SSL

Receive a signed message and the associated SSL.

Extract the public key from the sender.

Verify the integrity of the message using the SSL and the public key.

Step 4: Observation and Analysis

Observe the successful verification of authentic messages.

Attempt to verify messages with altered content or incorrect SSl to understand the integrity and confidentiality process.

Step 5: Discussion and Reflection

Discuss the importance of SSL in ensuring data integrity and confidentiality.

Reflect on how the private key's security impacts the overall security of SSL.

Program:

Serverside:

```
importjavax.net.ssl.*;

import java.io.*;

public class SSLServer {

public static void main(String[] args) throws Exception {

SSLServerSocketFactoryserverSocketFactory = (SSLServerSocketFactory)
SSLServerSocketFactory.getDefault();

SSLServerSocketserverSocket = (SSLServerSocket)
serverSocketFactory.createServerSocket(9999);

System.out.println("Server started. Waiting for a connection...");

SSLSocketclientSocket = (SSLSocket) serverSocket.accept();

System.out.println("Connection established.");

BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));

      String message = in.readLine();

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

in.close();

clientSocket.close();

serverSocket.close();

   }

}
```

Client Side:

```
importjavax.net.ssl.*;
import java.io.*;
public class SSLClient {
public static void main(String[] args) throws Exception {
SSLSocketFactorysocketFactory = (SSLSocketFactory) SSLSocketFactory.getDefault();
SSLSocket socket = (SSLSocket) socketFactory.createSocket("localhost", 9999);
PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
out.println("Hello, server! This is a secure message.");
System.out.println("Message sent.");
out.close();
socket.close();
    }
}
```

Output:

Server:

Server is running and listening on port 4433

Received from client: Hello, Server!

Client:

Server Response: Server: Message received - Hello, Server!

Result:

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

EX.NO.:6(a)	EAVESDROPPING
DATE:	

AIM :

To Implement Eavesdropping Using Java Program.

ALGORITHM:

AIM:

To Implement dictionary attacks Using Java Program.

ALGORITHM:

Create a Java Project: Start by creating a Java project in your favorite integrated development environment (IDE) or a simple text editor.

STEP 1:

Import Required Libraries: You'll need libraries for hash functions and file handling. Import the necessary libraries.

STEP 2: Read the List of Hashed Passwords: Create a method to read the list of hashed passwords from a file. Assume the hashed passwords are stored in a file, one hash per line

STEP 3: Load the Dictionary: Create a method to load a dictionary file containing words or phrases you want to try as passwords

STEP 4: Perform the Dictionary Attack: Create a method to perform the dictionary attack

STEP 5: Main Method: In your main method, call the above methods to read hashed passwords, load the dictionary, and perform the dictionary attack

STEP 6: Prepare Input Files: Create two text files, one for the hashed passwords (hashed_passwords.txt) and one for the dictionary (dictionary.txt). Populate them with the appropriate data.

STEP 7: Run the Program: Run the Java program, and it will attempt to match the hashed passwords from your input file with the words or phrases in the dictionary file.

PROGRAM:

Server side:

```
import java.io.*;

import java.net.*;

importjavax.net.ssl.*;

public class SecureServer {

public static void main(String[] args) throws Exception {

int port = 4433; // The port to listen on

SSLServerSocketFactoryserverSocketFactory = (SSLServerSocketFactory) SSLServerSocketFactory.getDefault();

SSLServerSocketserverSocket = (SSLServerSocket) serverSocketFactory.createServerSocket(port);

while (true) {

SSLSocketclientSocket = (SSLSocket) serverSocket.accept();

BufferedReader in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));

PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);

        String clientMessage = in.readLine();

System.out.println("Received: " + clientMessage);

        // Process data securely and send a response

out.println("Server: Message received - " + clientMessage);

clientSocket.close();

      }

   }

}
```

Client side:

```
import java.io.*;

import java.net.*;

importjavax.net.ssl.*;

public class SecureClient {

public static void main(String[] args) throws Exception {

    String serverAddress = "localhost"; // The address of the server

intserverPort = 4433; // The port to connect to

SSLSocketFactorysocketFactory= (SSLSocketFactory) SSLSocketFactory.getDefault();

SSLSocket socket = (SSLSocket) socketFactory.createSocket(serverAddress, serverPort);

BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

PrintWriter out = new PrintWriter(socket.getOutputStream(), true);

out.println("Hello, Server!");

    String serverResponse = in.readLine();

System.out.println("Server Response: " + serverResponse);

socket.close();

  }

}
```

Output:

Server side:

Server is running and listening on port 4433

Received from client: Hello, Server!

Client side:

Server Response: Server: Message received - Hello, Server!

RESULT:

Thus the java program to implement Eavesdropping is executed successfully and output is verified.

EX.NO.:6(b)	DICTIONARY ATTACKS
DATE:	

AIM:

To Implement dictionary attacks Using Java Program.

ALGORITHM:

Create a Java Project: Start by creating a Java project in your favorite integrated development environment (IDE) or a simple text editor.

STEP 1:

Import Required Libraries: You'll need libraries for hash functions and file handling. Import the necessary libraries.

STEP 2: Read the List of Hashed Passwords: Create a method to read the list of hashed passwords from a file. Assume the hashed passwords are stored in a file, one hash per line

STEP 3: Load the Dictionary: Create a method to load a dictionary file containing words or phrases you want to try as passwords

STEP 4: Perform the Dictionary Attack: Create a method to perform the dictionary attack

STEP 5: Main Method: In your main method, call the above methods to read hashed passwords, load the dictionary, and perform the dictionary attack

STEP 6: Prepare Input Files: Create two text files, one for the hashed passwords (hashed_passwords.txt) and one for the dictionary (dictionary.txt). Populate them with the appropriate data.

STEP 7: Run the Program: Run the Java program, and it will attempt to match the hashed passwords from your input file with the words or phrases in the dictionary file.

PROGRAM:

```
importjava.io.BufferedReader;
importjava.io.FileReader;
importjava.io.IOException;
public class DictionaryAttack {
public static void main(String[] args) {
        String username = "user123"; // The target username
        String dictionaryFile = "passwords.txt"; // Path to the dictionary file
booleanaccessGranted = false;
try (BufferedReader reader = new BufferedReader(new FileReader(dictionaryFile)) {
            String password;
while ((password = reader.readLine()) != null) {
if (authenticate(username, password)) {
System.out.println("Access granted for username: " + username + " with password: " +
password);
accessGranted = true;
break;
                }
            }
        } catch (IOException e) {
e.printStackTrace();
        }
if (!accessGranted) {
System.out.println("Dictionary attack failed. Access not granted.");
```

```
        }
    }
    // Simulate authentication logic
private static boolean authenticate(String username, String password) {
        // In a real system, this function would check if the provided username and password are correct.
        // For demonstration purposes, it only checks if the password matches "123456".
returnpassword.equals("123456");
    }
}
```

Output:

Access granted for username: user123 with password: 123456

If the program does not find a matching password in the dictionary, it will print:

Dictionary attack failed. Access not granted.

Result:

Thus the java program to implement Dictionary attackis executed successfully and output is verified.

EX.NO.:6(c)	MAN-IN-THE-MIDDLE
DATE:	

AIM :

To guide Man In The Middle Attack and resolved it.

INTRODUCTION :

A man-in-the-middle (MITM) attack is a malicious technique in which an attacker intercepts and possibly alters the communication between two parties by positioning themselves between them. By exploiting vulnerabilities in the communication channel, the attacker gains unauthorized access to sensitive information without the knowledge of the communicating parties.

Steps To Perform Advanced Man-in-the-Middle Attacks with Xerosploit

Step 1: Clone the official Xerosploit repository from Github using the git clone command.

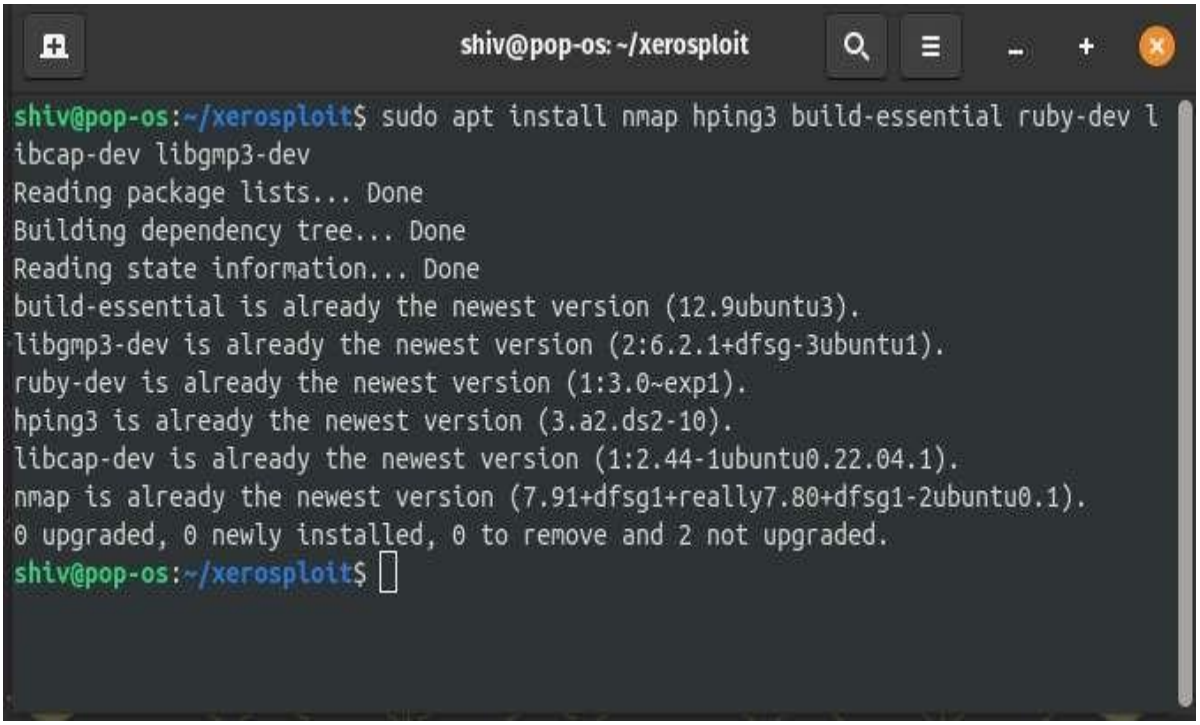
git clone https://github.com/LionSec/xerosploit

Step 2: Change the directory to xerosploit using the cd command. and then install the xerosploit.

sudo cd xerosploit
sudo python install.py

Step 3: Install it's all dependencies using the following command:

sudo apt install nmap hping3 build-essential ruby-devlibpcap-dev libgmp3-dev



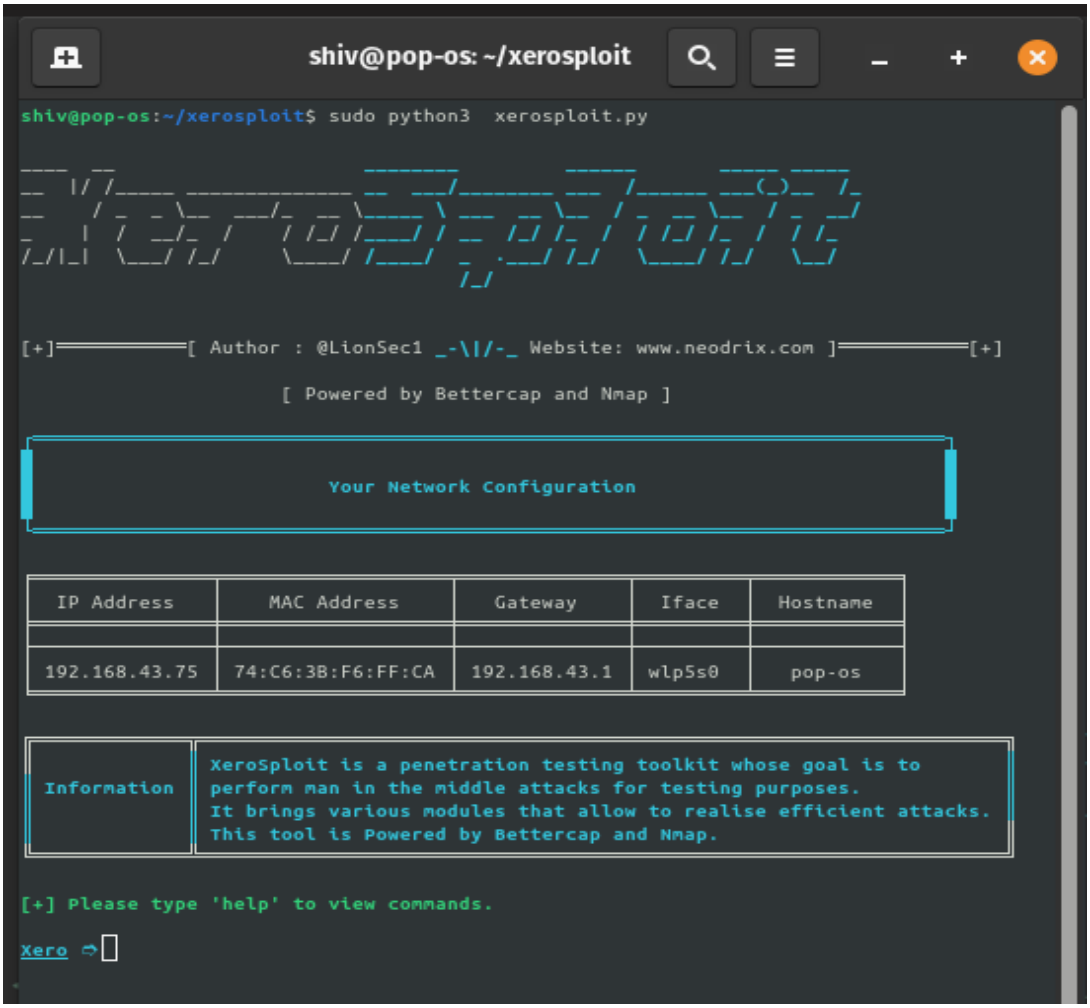
Step 4: Run the following command for setting up the terminaltables.

sudo pip3 tabulate terminaltables

```
shiv@pop-os:~/xerosploit$ sudo pip3 install tabulate terminaltables
Requirement already satisfied: tabulate in /usr/local/lib/python3.10/dist-package
s (0.8.10)
Collecting terminaltables
  Downloading terminaltables-3.1.10-py2.py3-none-any.whl (15 kB)
Installing collected packages: terminaltables
Successfully installed terminaltables-3.1.10
```

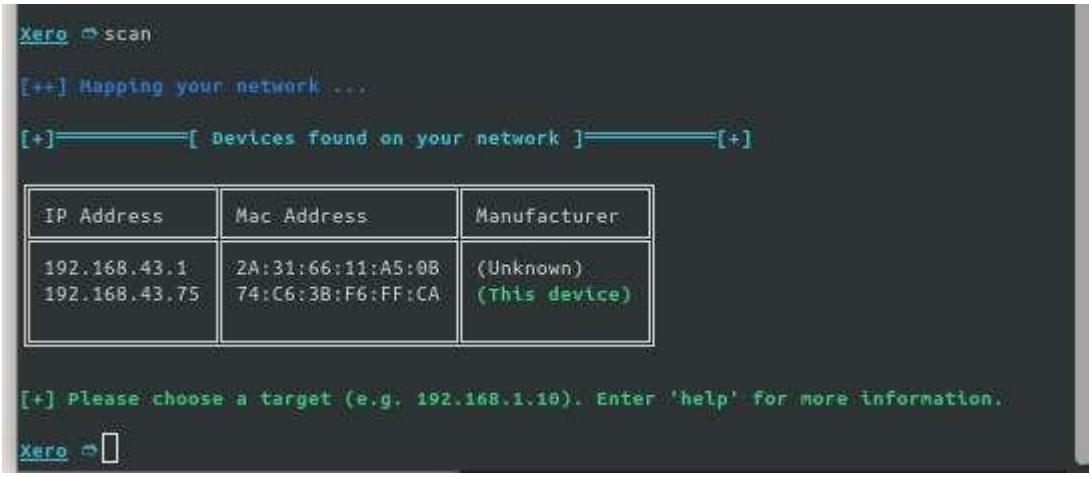
Step 5: Run Xerpsploit.

sudo python3 xerosploit.py

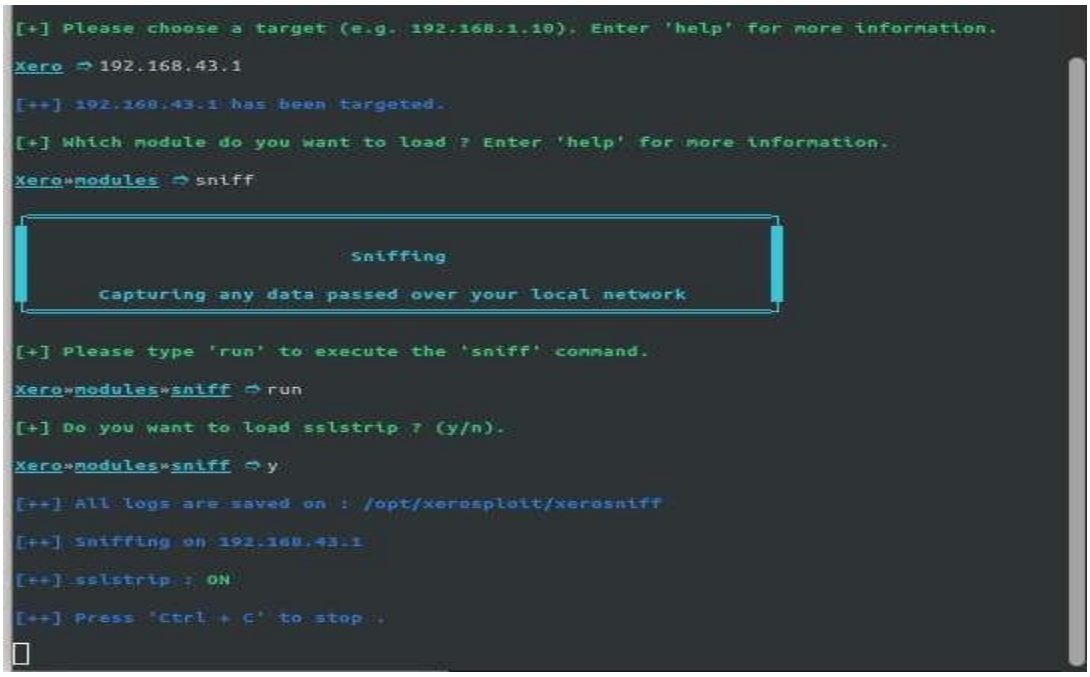


Step 6: Use the commands for various purposes like scanning the network use the belowcommand.

scan



Step 7: Select the target and run the module you want to execute.



Result:

Thus the man in the middle attacks discussed and all the modules are executed.

EX.NO:7	SNIFF TRAFFIC USING ARP POISONING
DATE:	

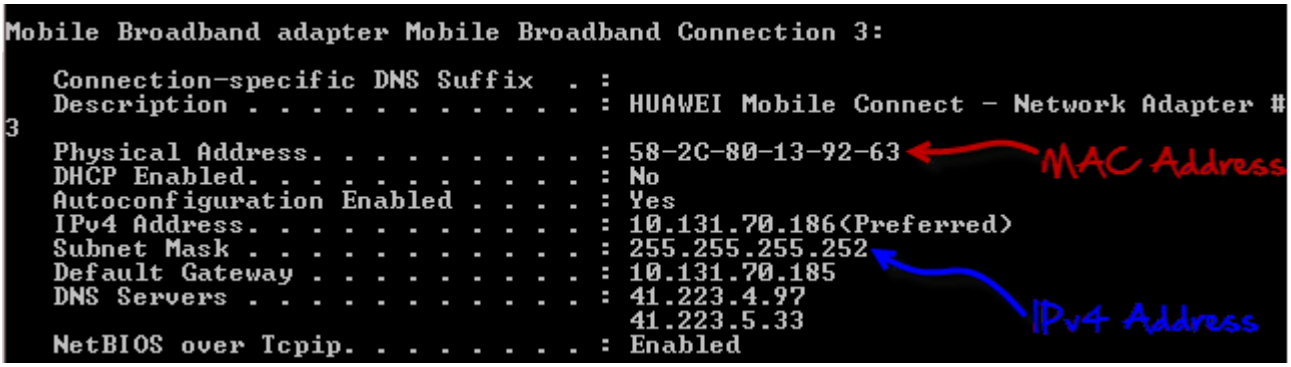
AIM:

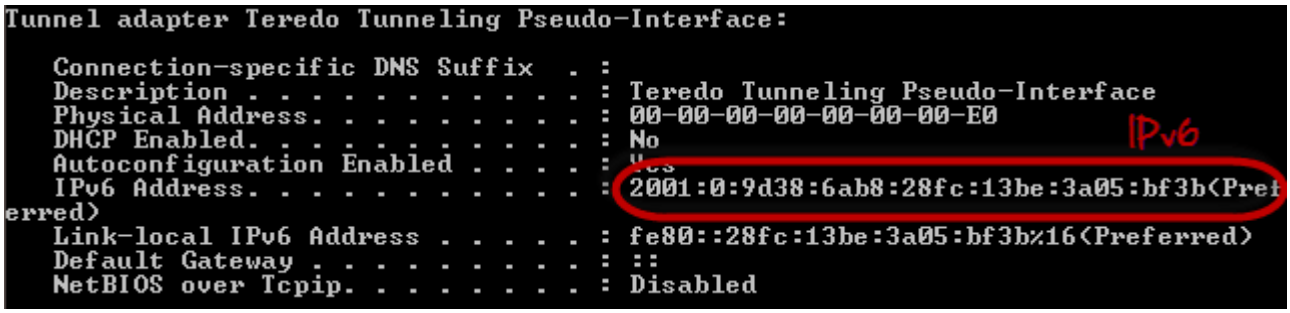
To guide Sniff Traffic Using Arp Poisoning and resolved it.

Enter the command

ipconfig /all

You will get detailed information about all the network connections available on your computer. The results shown below are for a broadband modem to show the MAC address and IPv4 format and wireless network to show IPv6 format.





What is ARP Poisoning?

ARP is the acronym for Address Resolution Protocol. It is used to convert IP address to physical addresses [MAC address] on a switch. The host sends an ARP broadcast on the network, and the recipient computer responds with its physical address [MAC Address]. The resolved IP/MAC address is then used to communicate. **ARP poisoning is sending fake MAC addresses to the switch so that it can associate the fake MAC addresses with the IP address of a genuine computer on a network and hijack the traffic**.

ARP Poisoning Countermeasures

Static ARP entries: these can be defined in the local ARP cache and the switch configured to ignore all auto ARP reply packets. The disadvantage of this method is, it's difficult to maintain on large networks. IP/MAC address mapping has to be distributed to all the computers on the network.

ARP poisoning detection software: these systems can be used to cross check the IP/MAC address resolution and certify them if they are authenticated. Uncertified IP/MAC address resolutions can then be blocked.

Operating System Security: this measure is dependent on the operating system been used. The following are the basic techniques used by various operating systems.

- **Linux based**: these work by ignoring unsolicited ARP reply packets.
- **Microsoft Windows**: the ARP cache behavior can be configured via the registry. The following list includes some of the software that can be used to protect networks against sniffing;
    - **AntiARP**– provides protection against both passive and active sniffing
    - **Agnitum Outpost Firewall**–provides protection against passive sniffing
    - **XArp**– provides protection against both passive and active sniffing
- **Mac OS**: ArpGuard can be used to provide protection. It protects against both active and passive sniffing.

Hacking Activity: Configure ARP entries in Windows

We are using Windows 7 for this exercise, but the commands should be able to work on other versions of windows as well.

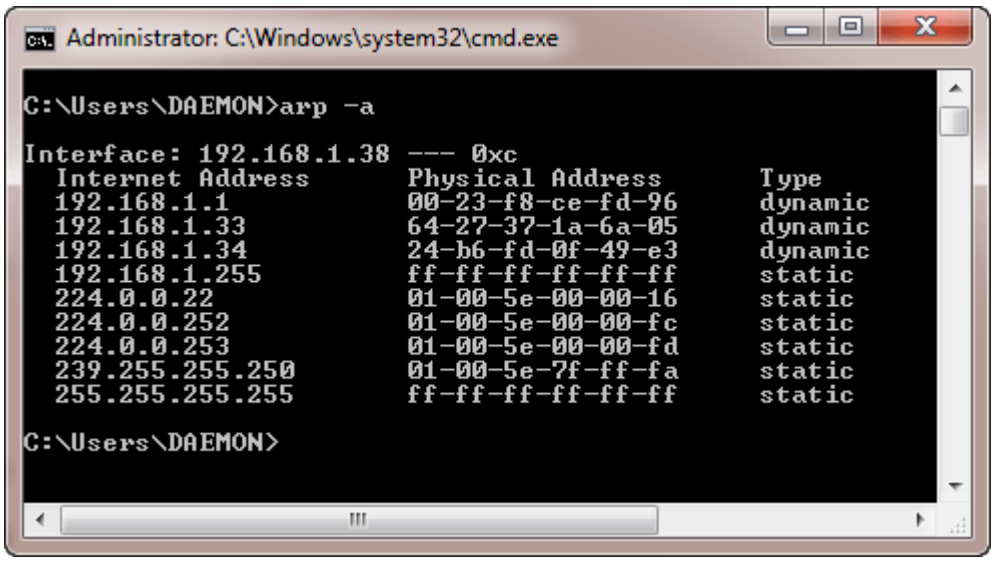
Open the command prompt and enter the following command

arp –a

HERE,

- **apr**calls the ARP configure program located in Windows/System32 directory
- **-a** is the parameter to display to contents of the ARP cache

You will get results similar to the following

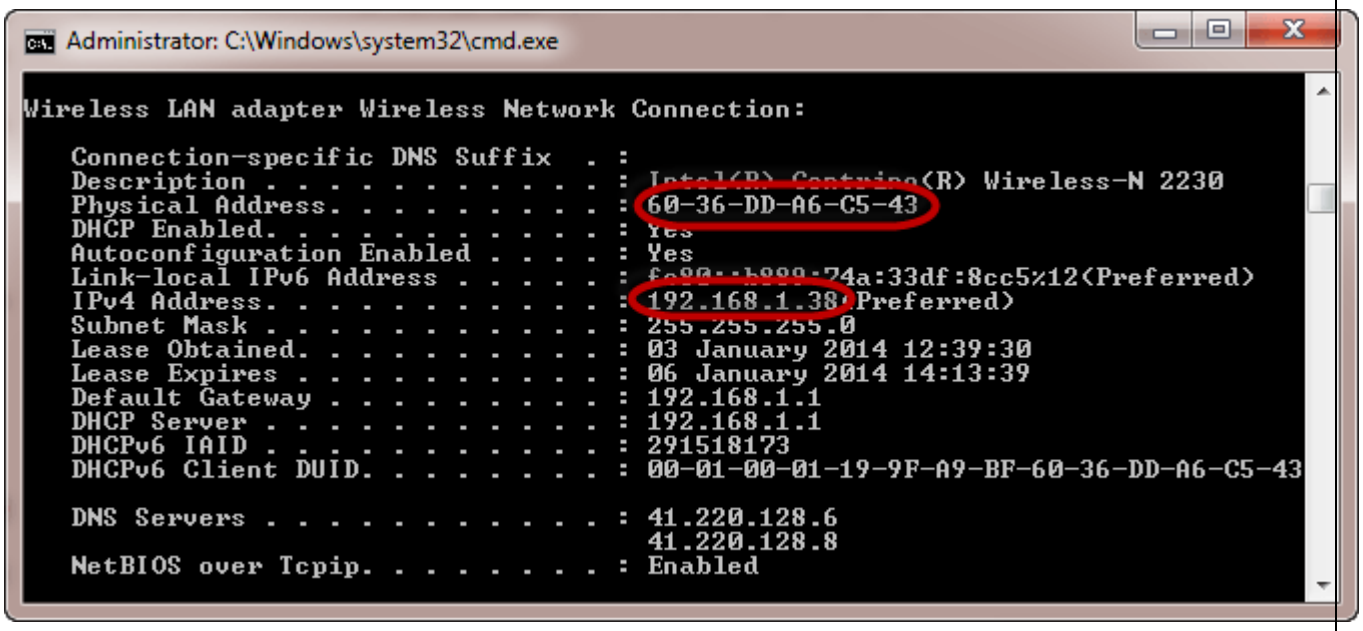


Note: dynamic entries are added and deleted automatically when using TCP/IP sessions with remote computers.

Static entries are added manually and are deleted when the computer is restarted, and the network interface card restarted or other activities that affect it.

Adding static entries

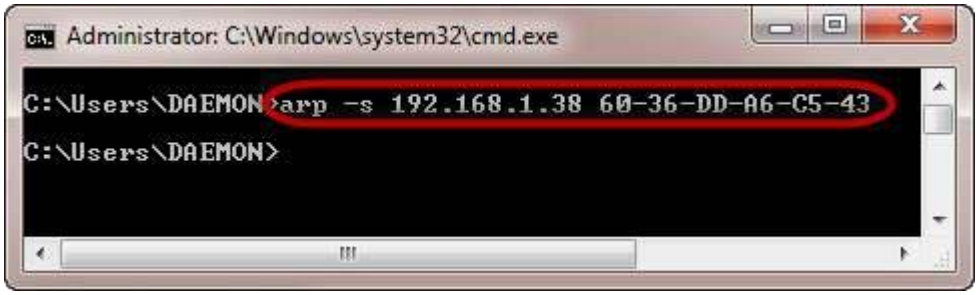
Open the command prompt then use the ipconfig /all command to get the IP and MAC address



The MAC address is represented using the Physical Address and the IP address is IPv4Address

Enter the following command

arp –s 192.168.1.38 60-36-DD-A6-C5-43

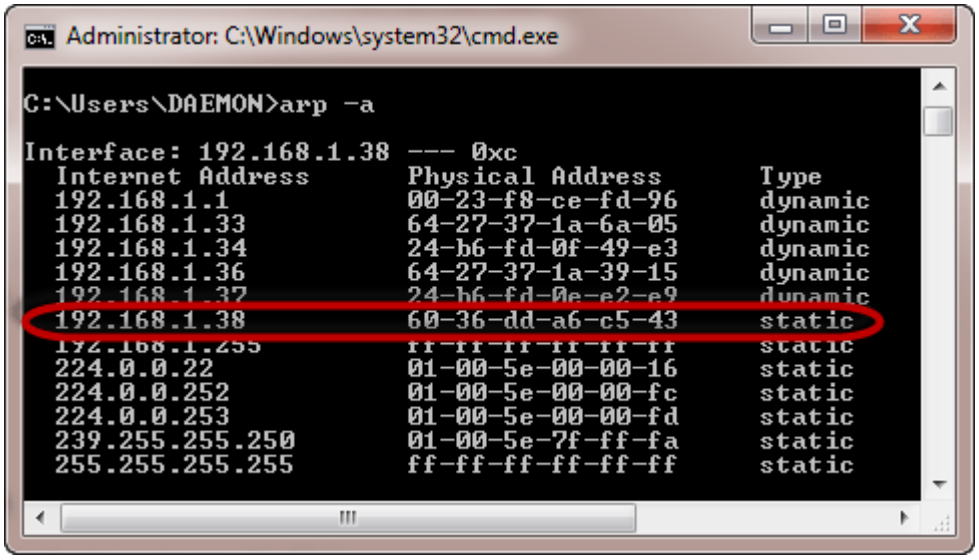


Note: The IP and MAC address will be different from the ones used here. This is because they are unique.

Use the following command to view the ARP cache

arp –a

You will get the following results

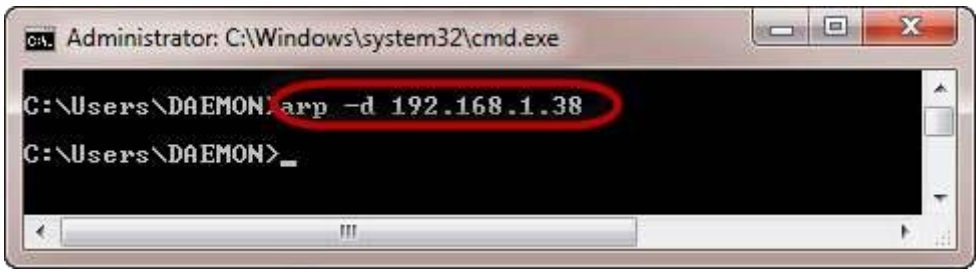


Note the IP address has been resolved to the MAC address we provided and it is of a static type.

Deleting an ARP cache entry

Use the following command to remove an entry

arp –d 192.168.1.38



P.S. ARP poisoning works by sending fake MAC addresses to the switch

RESULT:

The expected output is achieved with the command of Sniff Traffic Using Arp Poisoning

EX.NO:8	INTRUSION DETECTION
DATE:	

AIM:

Todemonstrate IntrusionDetectionSystem(IDS) usingSnortsoftware tool.

STEPSONCONFIGURINGANDINTRUSIONDETECTION:

1. DownloadSnort fromtheSnort.orgwebsite.(http://www.snort.org/snort-downloads)

2. DownloadRules(https://www.snort.org/snort-rules).Youmustregistertogettherules.(Youshoulddownloadtheseoften)

3. Double click on the .exe to install snort. This will install snort in the "C:\Snort"folder.ItisimportanttohaveWinPcap(https://www.winpcap.org/install/)installed

4. ExtracttheRulesfile.YouwillneedWinRARforthe.gzfile.

5. Copyallfilesfromthe"rules"folderoftheextractedfolder.Nowpastetherule sinto*"C:\Snort\rules"*folder.

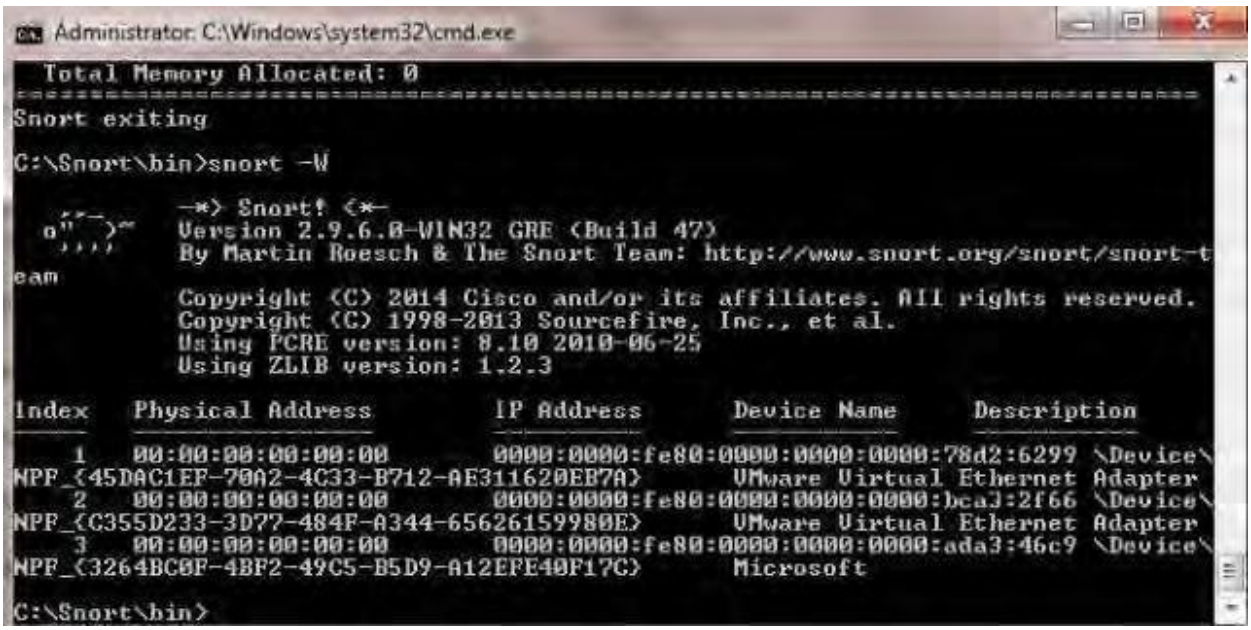
6. Copy"snort.conf"filefromthe"etc"folderoftheextractedfolder.Youmust pasteit into"C:\Snort\etc"folder.Overwriteany existingfile.Rememberifyoumodifyyoursnort.conffileanddownloadane wfile,youmustmodify itforSnort towork.

7. Openacommandprompt(cmd.exe)andnavigateto folder"C:\Snort\bin"folder.(atthePrompt,typecd\snort\bin)

8. Tostart(execute)snortinsniffer modeusefollowingcommand:snort-dev-i3

-iindicates theinterfacenumber.Youmustpickthecorrectinterfacenumber.Inmycase,itis3 .

-devis usedtorunsnorttocapture packetsonyournetwork. Tochecktheinterfacelist,usefollowingcomma nd:snort -W



Findinganinterface

Youcantellwhichinterfacetouse bylookingatthe IndexnumberandfindingMicrosoft. As you can see in the above example, the other interfaces are forVMWare.Myinterfaceis3.

9. TorunsnortinIDS mode,youwillneedtoconfigurethefile“snort.conf”accordingtoyournet work environment.

10. Tospecifythenetworkaddressthatyouwanttoprotect insnort.conffile, lookforthefollowingline.

varHOME_NET192.168.1.0/24(Youwillnormallysee anyhere)

11. YoumayalsowanttosettheaddressesofDNS_SERVERS,ifyouhavesome on yournetwork.

Example:

examplesnort

12. ChangetheRULE_PATHvariabletothepathofrulesfold er.var RULE_PATH c:\snort\rules pathtorules

13. Changethepathofalllibraryfiles withthenameandpathonyoursystem. andyoumustchange thepath of snort_dynamicpreprocessorvariable.C:\Snort\lib\snort_dynamiccpreproces sor

You need to do this to all library files in the “C:\Snort\lib” folder. The old

pathmightbe:“/usr/local/lib/…”.youwillneedto
replacethatpathwithyoursystempath.UsingC:\Snort\lib

14. Changethepathofthe“dynamicengine”variablevalueinthe“snort.conf”file..

Example:

dynamicengineC:\Snort\lib\snort_dynamicengine\sf_engine.dll

15. Addthepathsfor“includeclassification.config”and“includereference.config”files.

include
c:\snort\etc\classification.configinclude
c:\snort\etc\reference.config

16. Removethecomment(#)onthelinetoallowICMPrules,ifitiscommentedwitha#.

include$RULE_PATH/icmp.rules

17. YoucanalsoremovethecommentofICMP-inforulescomment,ifitiscommented.

include $RULE_PATH/icmp-info.rules

18. To add log files to store alerts generated by snort, search for the “output log”test insnort.confandaddthefollowingline:

outputalert_fast:snort-alerts.ids

19. Comment (add a #) the whitelist $WHITE_LIST_PATH/white_list.rules andtheblacklistChangethenested_ipinner,\tonested_ipinner#,\

20. Commentout(#)followinglines:#preprocessornormalize_ip4
#preprocessornormalize_tcp:ipsecnstream#preprocessornormalize_icmp4
#preprocessornormalize_ip6
#preprocessornormalize_icmp6

21. Savethe“snort.conf”file.

22. TostartsnortinIDSmode, runthefollowingcommand:

snort-cc:\snort\etc\snort.conf-lc:\snort\log-i3(Note:3isusedformyinterfacecard)

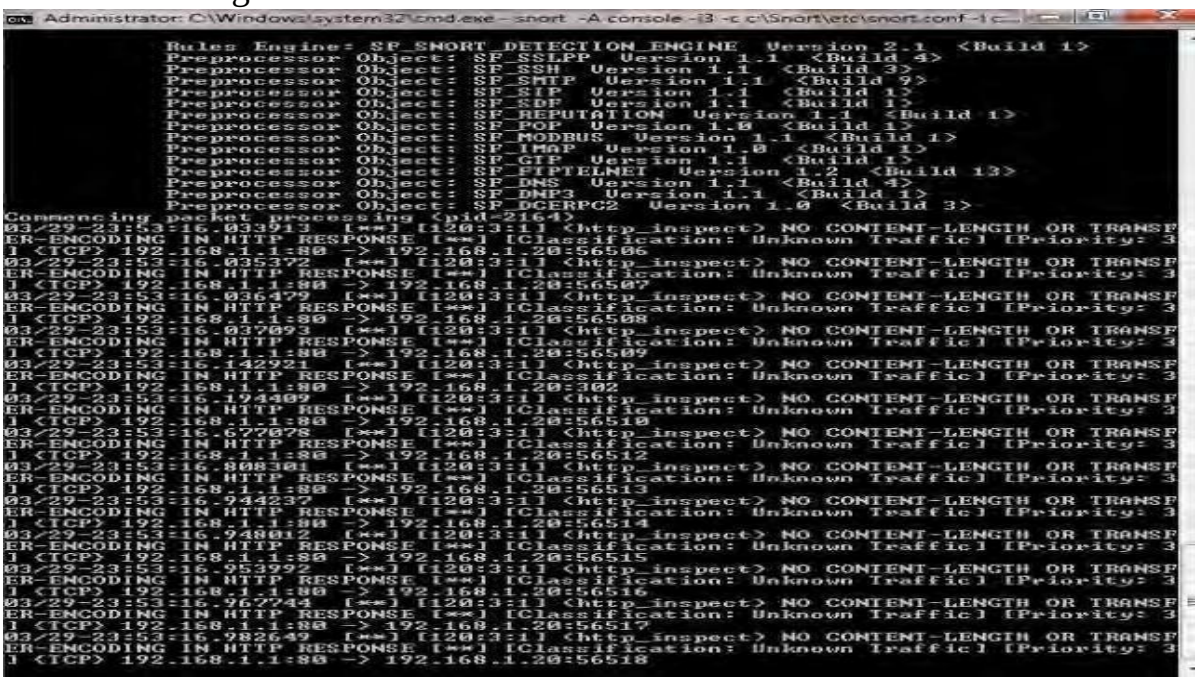
If a log is created, select the appropriate program to open it. You can useWordPard or NotePad++toread thefile.

TogenerateLogfilesinASCII mode,youcanusefollowingcommandwhilerunningsnortinIDSmode: snort-Aconsole-i3-c c:\Snort\etc\snort.conf-lc:\Snort\log-Kascii

23. Scan the computer that is running snort from another computer by using PINGorNMap(ZenMap).

Afterscanningorduringthescanyoucancheckthesnort-alerts.idsfileinthelogfoldertoinsureitisloggingproperly.YouwillseeIPaddressfolders appear.

Snortmonitoringtraffic–



RESULT:

ThustheIntrusionDetectionSystem(IDS) hasbeendemonstratedbyusingtheOpenSourceSnortIntrusionDetectionTool.

EX.NO:9	**NETWOTK MONITORING TOOLS**
DATE:	

AIM

To achieve the network monitoring tools with the required function and softwares

INTROUDCTION:

Networks are the fundamentals behind businesses worldwide. It plays a pivotal role in serving your employees for administrative purposes and your clients across the continents. The networks help you keep information in a centralized location - accessible to those who need and restrict every other inbound request. So how do you provide continuous top-notch end user experience and maintain your rapidly evolving network? Only by monitoring the availability, health, and performance of your networks over time with the help of reliable, real-time **network monitoring tools**.

Requirements of a network monitoring tool

While selecting a network monitortool for your IT environment, it is important to weigh in your current requirements and also your future needs because choosing the right one among various tools for network monitoring is crucial. Some of the essential elements that a **network monitoring tool** requires are:

Real time monitoring

Comprehensive monitoring capabilities

scalability

automation

user management

Real time monitoring

Learning that your network is down from your end users is the nightmare that every IT admin tries to avoid. The network monitoring application and its reporting tools must provide performance insights into your network in real time. This helps you identify performance hiccups early and avoid potential outages.

Comprehensive monitoring capabilities

Employing individual network monitoring tools for monitoring various network components such as switches/routers, servers, virtual environment, HCI, applications, storage devices, etc., and also employing network monitoring tools for Windows and Linux environments separately is more trouble than help as the tools themselves require constant management, additional resources, and there is also a certain learning curve associated with each network monitoring tool. Therefore, the network performance monitoring tools should support extensive monitoring in a single console.

Scalability

Network scalability is an important aspect to be considered when selecting enterprise network monitoring tools. A network management tool or **network monitoring software** can be called as scalable when it is more adaptable to the changing needs or demands of the business or users. Scalability helps a network to stay in par with increased productivity, trends, changing needs and new adaptations. Scalabiity ensures that the overall network performance does not significantly degrade, even if the size of the network is increasing. Businesses also need remote network monitoring tools for managing multiple geographical locations from one console.

Automation

In network monitoring, automation helps network performance monitoring tools to react based on threshold values or a set of framed rules/ criteria being met. With automation, the monitoring network tools can automatically detect and troubleshoot problems (proactive monitoring), send alert notifications, forecast storage growth, and much more. When you are monitoring and managing multiple devices in your environment, automation comes in super handy and saves you ample time and resources, making it an important usability aspect of your network monitoring solutions**.**

User Management

User Management helps organizations ensure network security by providing access to the designated users only. Apart from providing access to users with roles, the network monitoring tool should also define the scope for users. This helps IT teams with multiple staffs as it clearly defines their operational boundaries. Network monitoring tools with the above features are excepetionally benefiting to your business.

Why is ManageEngineOpManager considered one of the best network monitor tools in the market?

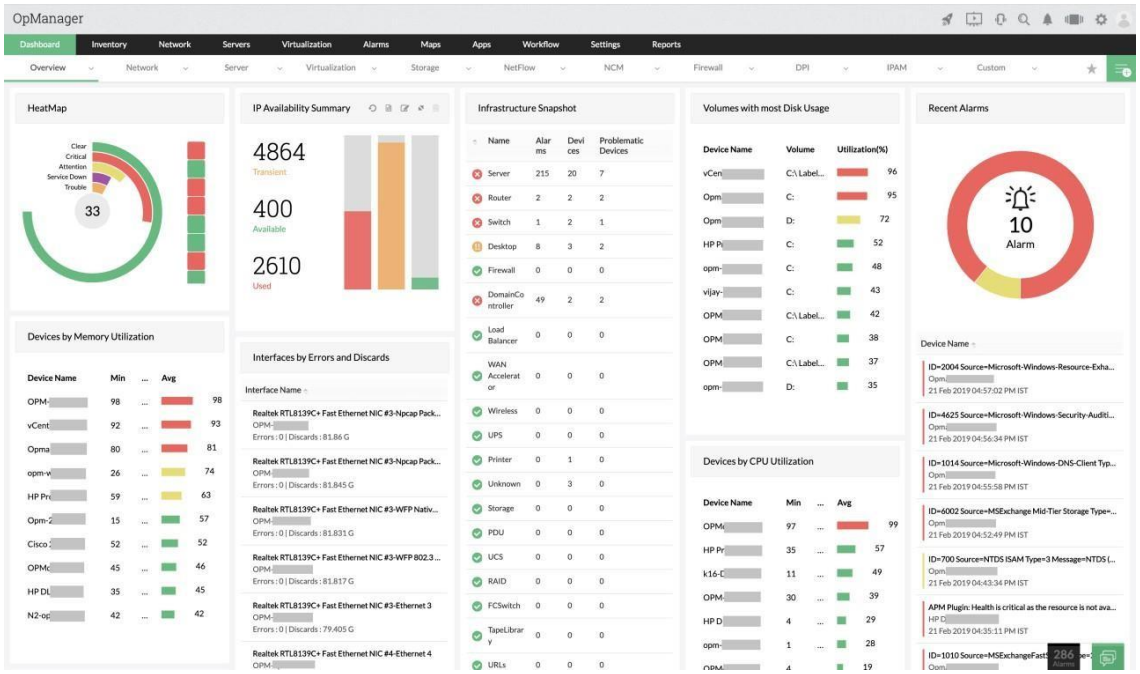
OpManager is a powerful network management and monitoring tool that monitors switches, routers, servers, WLC, load balancers, VPN, printers, firewalls, VMs, Nutanix environments, and anything that has an IP and connected to the network - in a single console.

1. top features

2. comprehensive monitoring capabilities

3. customer reviews

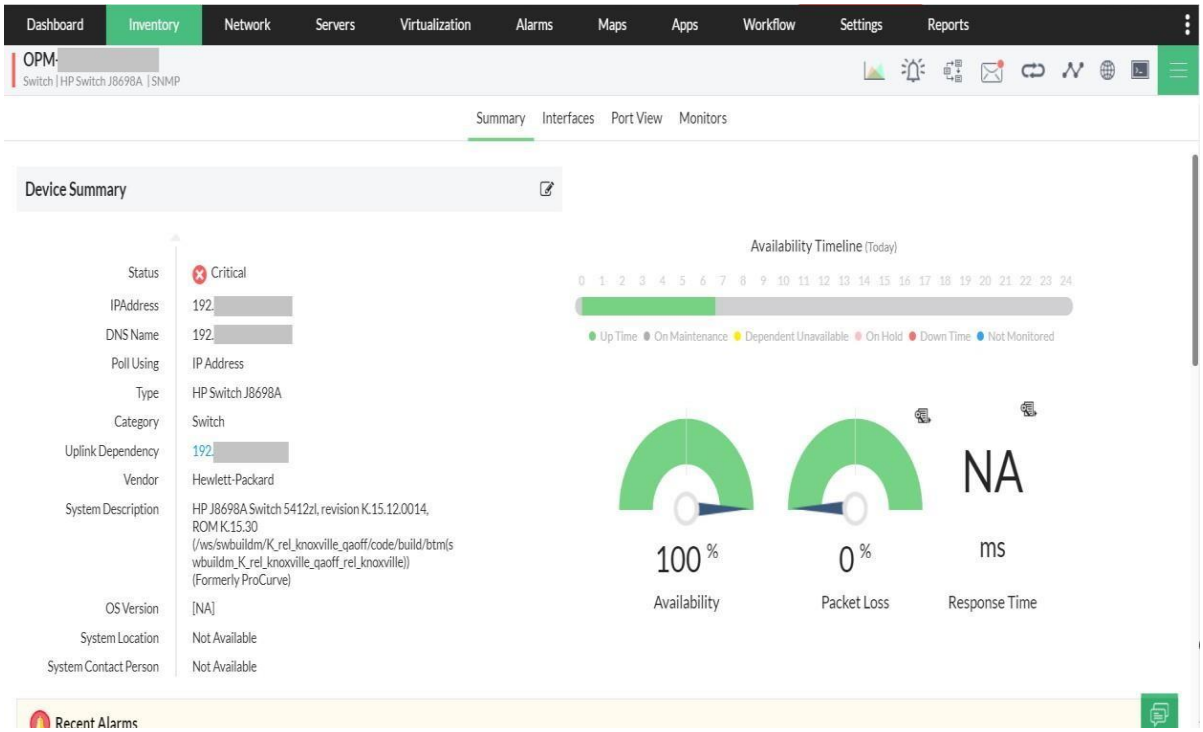
Top features:



Comprehensive monitoring capabilities

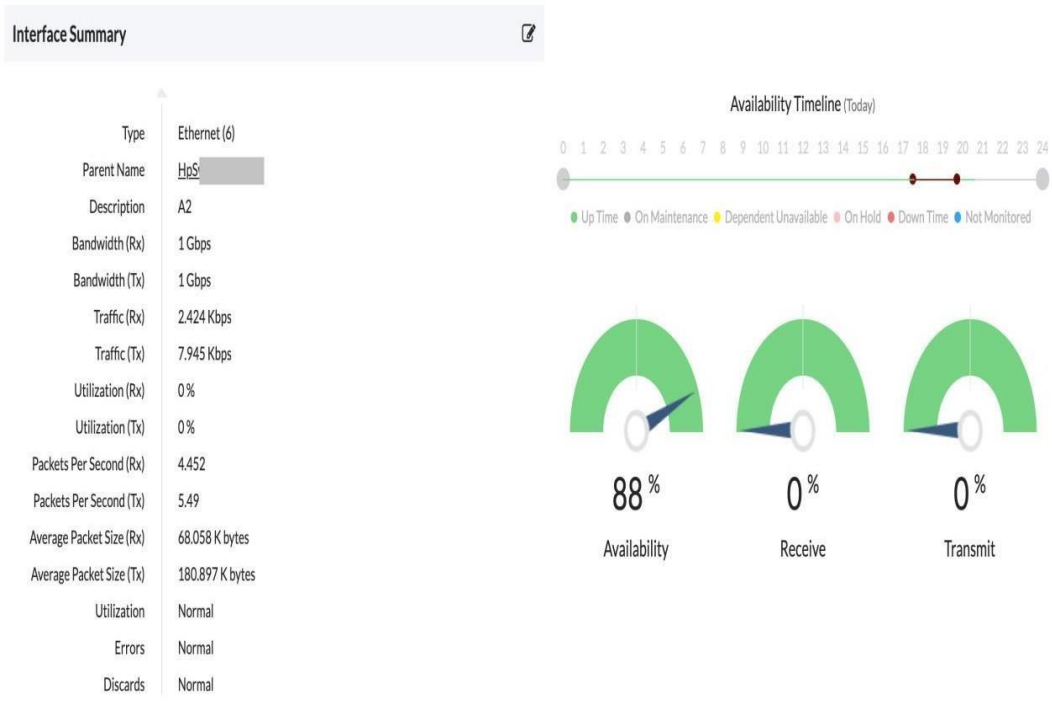
Switch/Router monitoring

Network switches and routers form the backbone of any IT infrastructure. Any issue with switch breaks the end user connectivity with the network. Using OpManager, you can monitor switches, and routers from the likes of Cisco, Juniper, Aruba, ZTE, and many other vendors for availability, health, and performance in real-time for 2,000+ parameters and avoid possible network pitfalls. Apart from monitoring switches, OpManager maps switch ports to devices and monitors the availability of the switch ports.



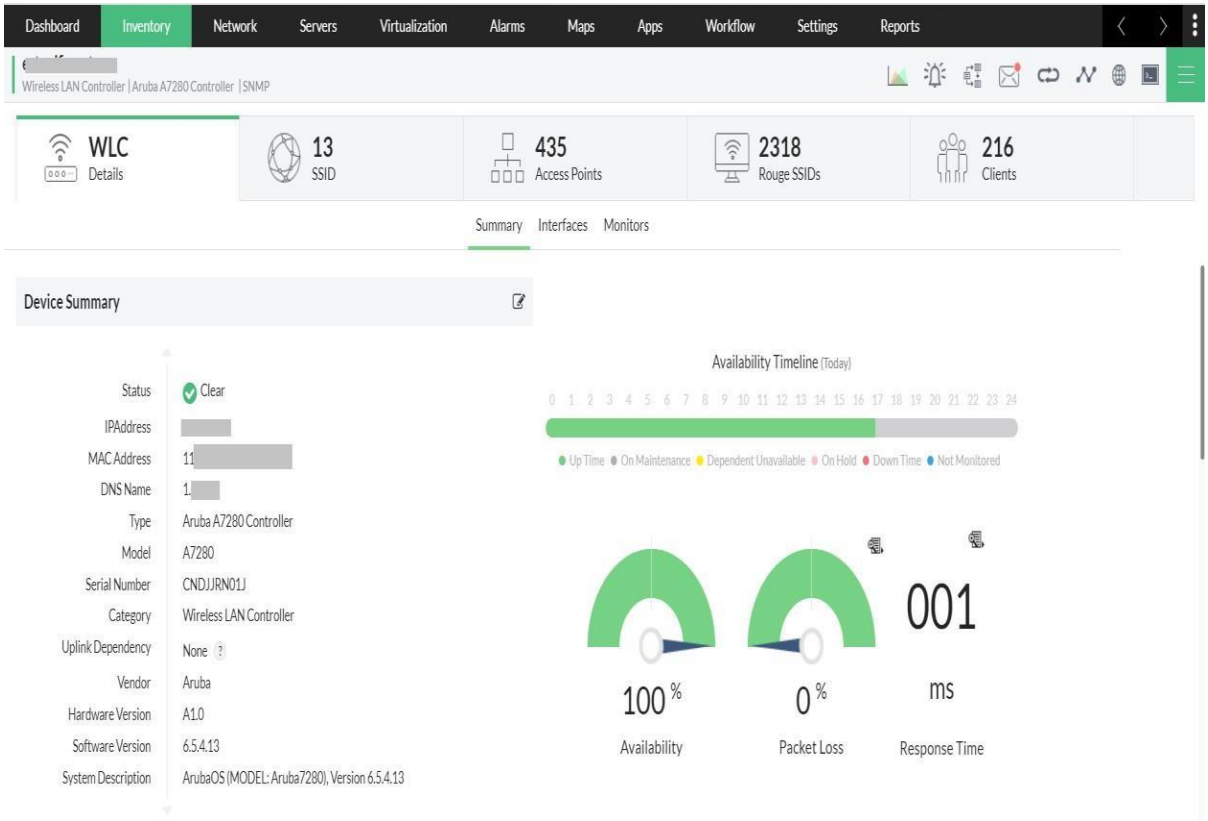
Network interface monitoring

Network interfaces are one of key performance indicators (KPI) as they help identify network performance degradation at the earliest. OpManager, the best network monitoring tool, monitors interfaces using SNMP and provides a single customizable dashboard to view and analyze bandwidth performance and network traffic for your IT network. You can monitor interfaces by checking the availability status of interfaces and monitor traffic speed on the interface, errors, discards, etc. using OpManager.



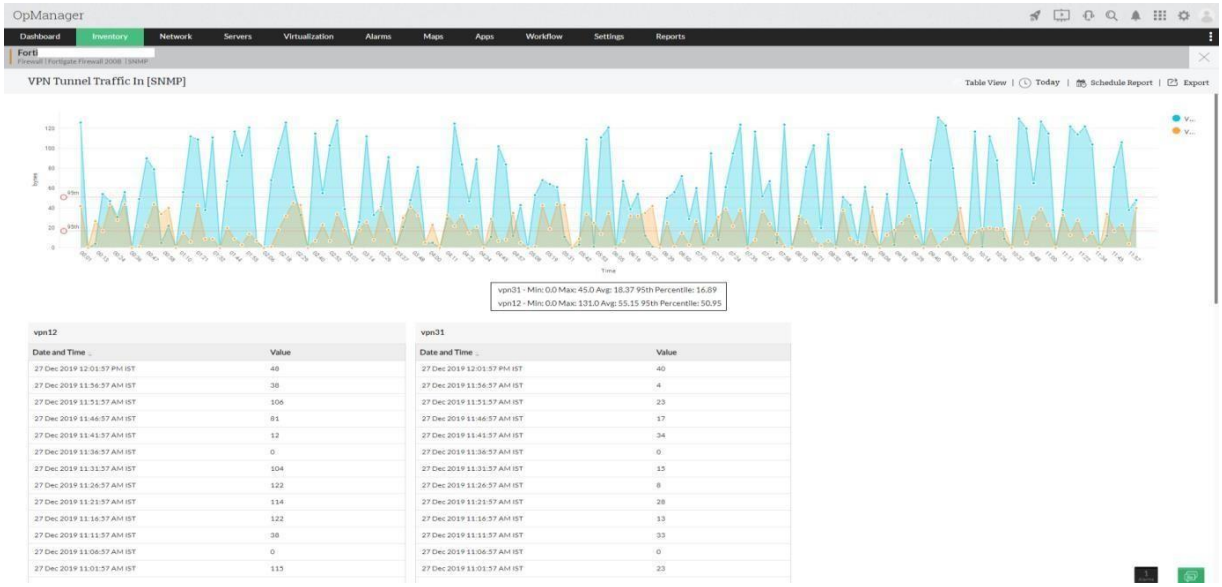
WLC monitoring

OpManager's multi-vendor WLC monitoring module allows you to keep your network intact by providing in-depth visibility of your wireless LAN controller (WLC), its associated service set identifiers (SSIDs) and access points (APs). Cisco's WLC monitoring tool in OpManager allows direct discovery of Cisco WLC and their associated SSIDs, APs and helps you monitor the overall performance of your wireless network with the help of Cisco WLC monitor. The WLC snapshot page provides inventory information, the device availability status, and other similar information. In addition, knowing the top five access points based on usage tells you who the top talkers are in your WLC environment, and custom dials display information on various parameters, including CPU and memory usage.



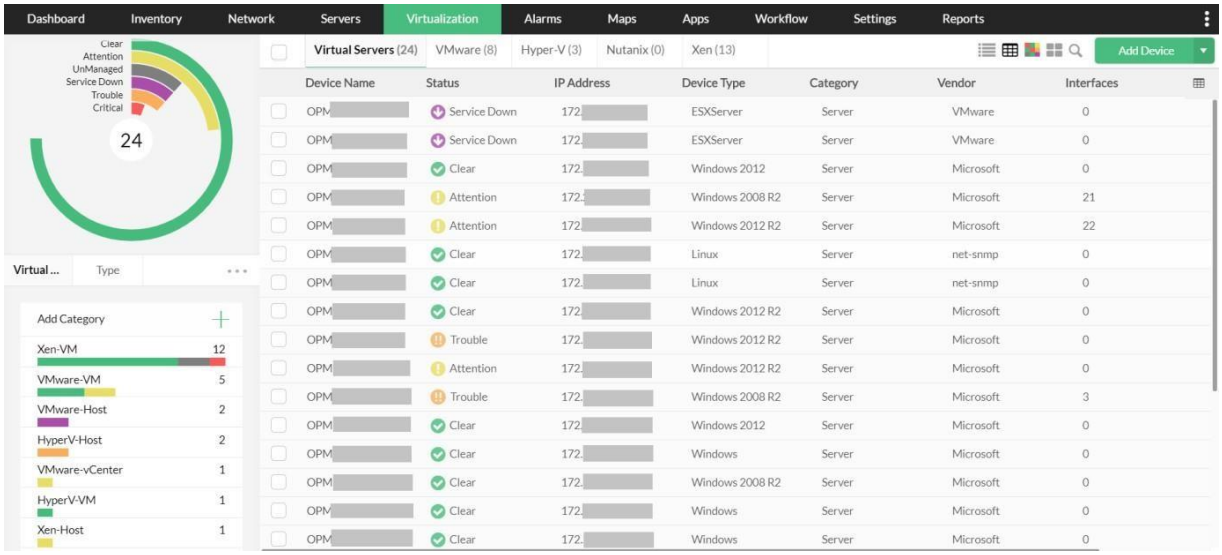
VPN monitoring

Organizations allow connections into their networks through VPNs for their remote workforce. These connections can sometimes be compromised, resulting in data theft or network attacks. With a monitoring tool like OpManager, you can monitor your VPN by tracking the number of active VPN sessions, VPN tunnel status, and VPN tunnels count in real time, and also receive instant alerts on VPN connection regularities making your network secure and keeping your remote productivity issues at bay.



Hybrid environment monitoring

Every organization's network has unique needs, so top-of-the-line networking technologies are employed to address them. This helps networks deliver business services but also poses a challenge with monitoring and managing the network. Using multiple network management tools is not efficient and cost effective. With OpManager, apart from monitoring switches, servers, etc., you can monitor VMware, Hyper-V, Hypervisors, Cisco UCS, Nutanix infrastructures, and more, all within a unified console, making it the best network monitor. Additionally, you can monitor your WAN with Cisco IP SLA using OpManager.



Mobile application

Access your OpManager's network monitoring and reporting anytime and anywhere using the new ManageEngineOpManager mobile application. Available for both Android and iOS, this

lets you visualize your infrastructure, act on the alerts, drill-down to the root cause of the problem without having to be physically present in your server room to resolve a fault!

OpManager makes your work easy

Apart from the above, OpManager, your comprehensive network monitoring solution monitors Windows servers, Linux servers, storage devices, Windows services, processes and scales upto 30,000 devices out of the box. This network software makes network monitoring effortless with intelligent automations, ML-based forecasting, and extensive protocol support. Here are some of OpManager'snetwork monitoring applications:

1. **Storage capacity forecasting:** With the help of ML-based forecasting techniques, this network monitoring and reporting software pinpoints when the device storage will reach 80 percent, 90 percent, and 100 percent of the allocated storage, and helps with planning purchase decisions.
2. **Notification profiles:**OpManager lets you notify network faults via Slack channels, trouble tickets, emails, SMS, and web alarms if they are not acknowledged, so no alarm goes unnoticed.
3. **Alarm Escalation:** Alarm escalation rules can be configured for mission-critical devices such as application servers, so any fault pertaining to availability, health, and performance is escalated to a higher authority via email or SMS based on user-defined criteria.
4. **Support for multiple vendors:**OpManager offers support for more than 53,000 vendor templates, so you can efficiently manage your network devices from vendors such as Cisco, Juniper, Fortigate, and many more. These templates can also be customized to address your organization's unique needs.
5. **Support for wide range of protocols:**OpManager supports communication protocols such as ICMP, and LAN management protocols such as SNMP, WMI, CLI, and more.
6. **Discovery Rule Engine:** Discovery Rule Engine automatically associates device templates and rules to network devices as defined by the user, thereby automating routine tasks, and saving valuable time and resources.
7. **In-built troubleshooting tools:**OpManager offers multiple tools such as Ping, SNMP Ping, Proxy Ping, Traceroute, WMI Query Tool, CLI Query Tool, and more that aid in troubleshooting network issues within OpManager.
8. **Dashboards:**OpManager provides intuitive dashboards that provide a 360-degree view of your entire IT infrastructure on one screen, and makes fault identification easier.
9. **Visualizations:** Determine the availability of crucial services in multiple branch offices with maps and business views. With OpManager, you can easily monitor remote locations visually, and get alerted in real time before network services are disrupted.
10. **Multi-level thresholds:**OpManager offers multi-level thresholds with color codes, so you can identify show-stopping network faults and promptly take action.

Result:

The modules are executed with the commands of network monitoring tools

EX.NO:10 (a)	FIREWALL
DATE:	

AIM

To Knows The Uses Of Firewall And Their Uses In Winodws

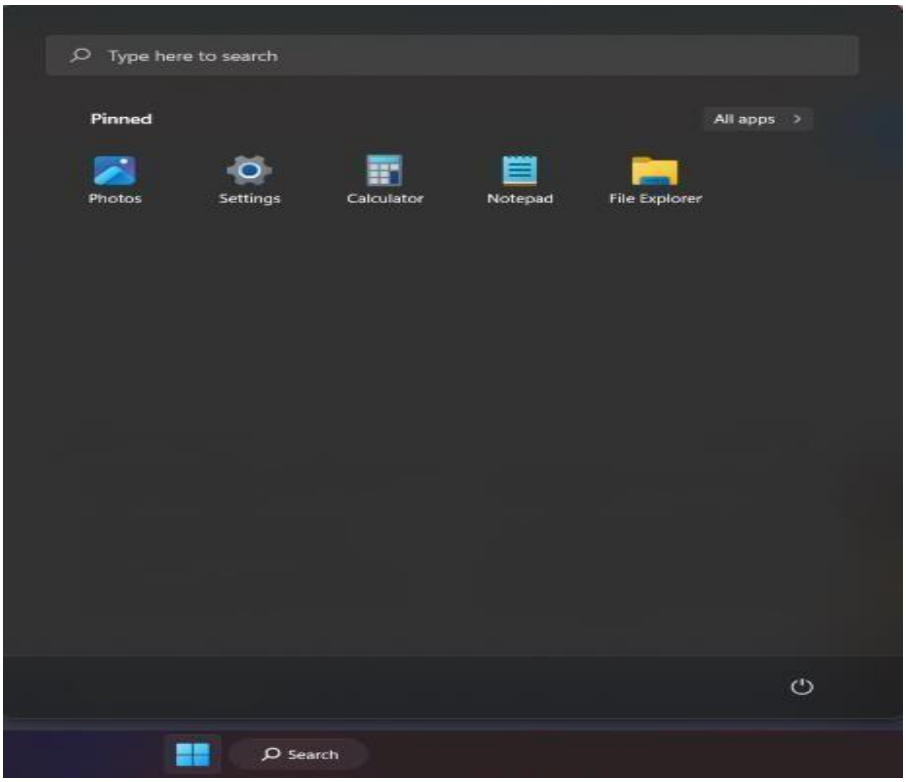
INTRODUCTION

A firewall is a network security device, either hardware or software-based, which monitors all incoming and outgoing traffic and based on a defined set of security rules it accepts, rejects or drops that specific traffic. Please refer to the article Introduction of Firewall in Computer Network for more details. We can configure 2 types of firewalls on Windows on the basis of firewall provider:

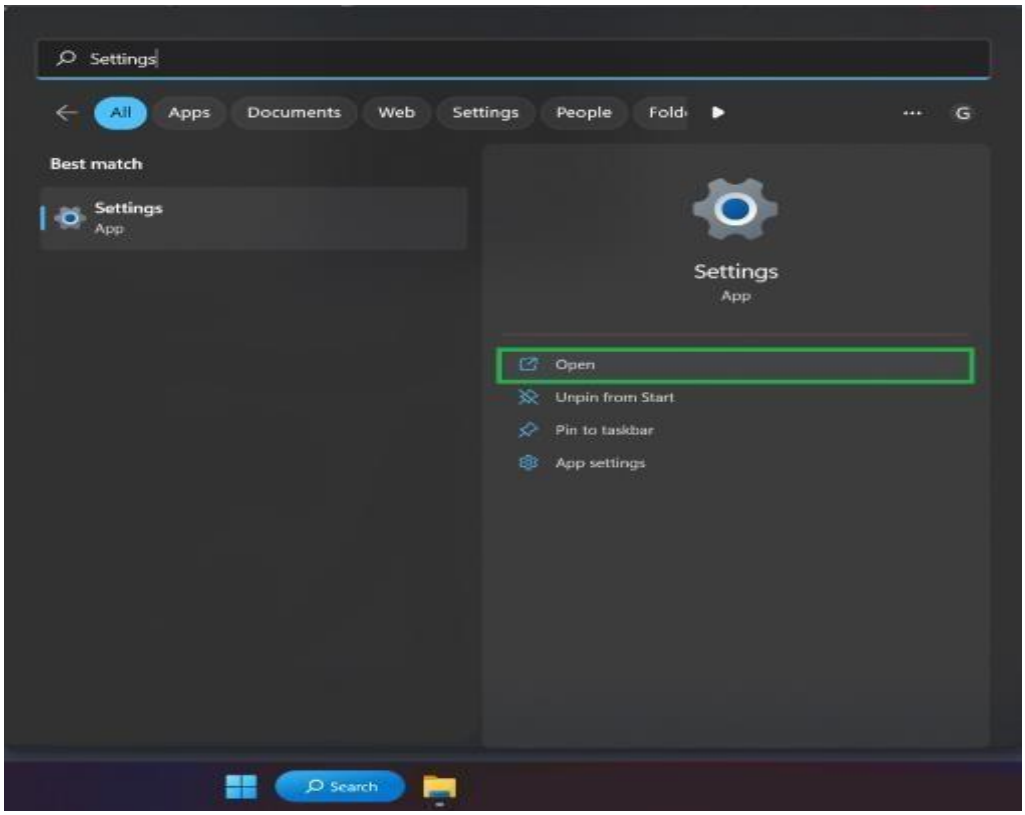
1. Windows Defender (Default firewall).
2. Third-party firewalls.

Configuring Firewall Defender on Windows:

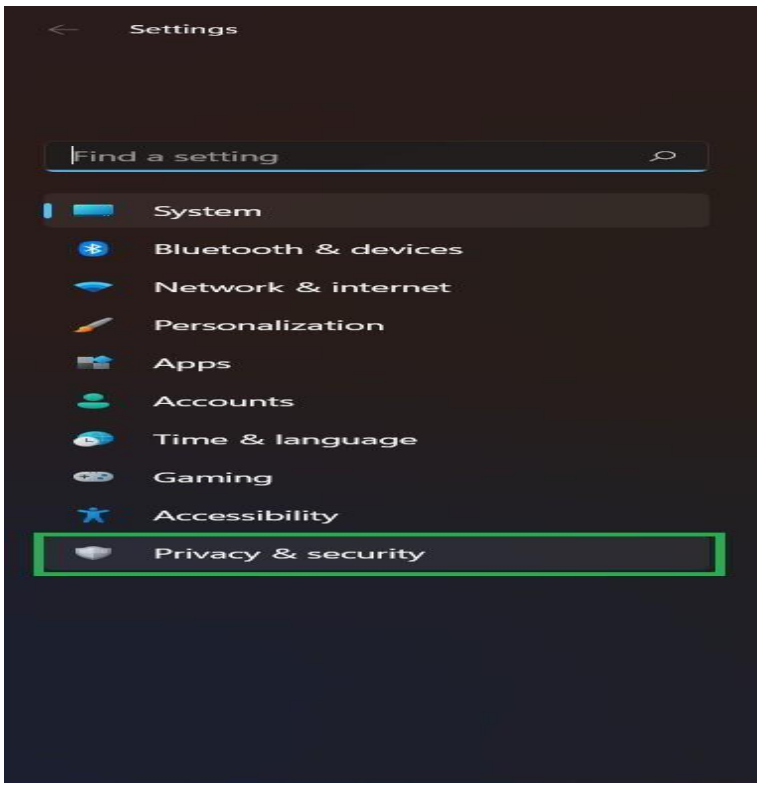
- **Step 1:** Launch **Start** from the taskbar.



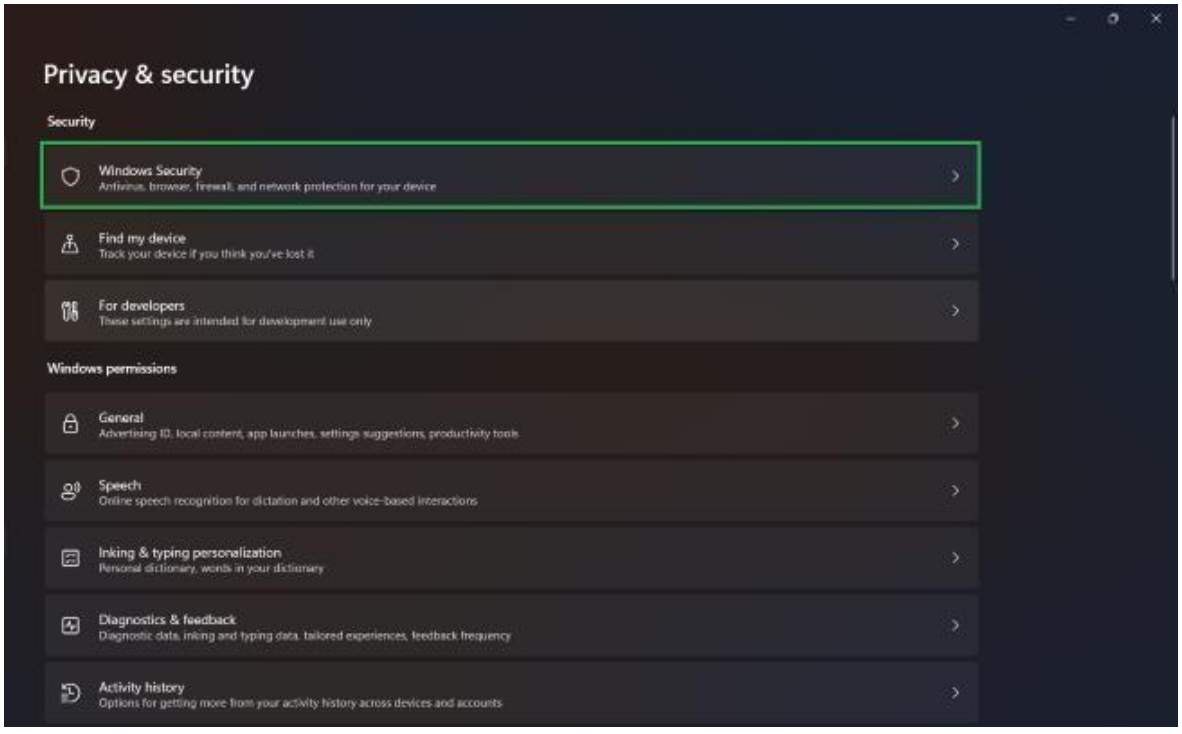
Step 2: Search **"Settings"** in the search bar if you do not find the Settings icon in Start menu.



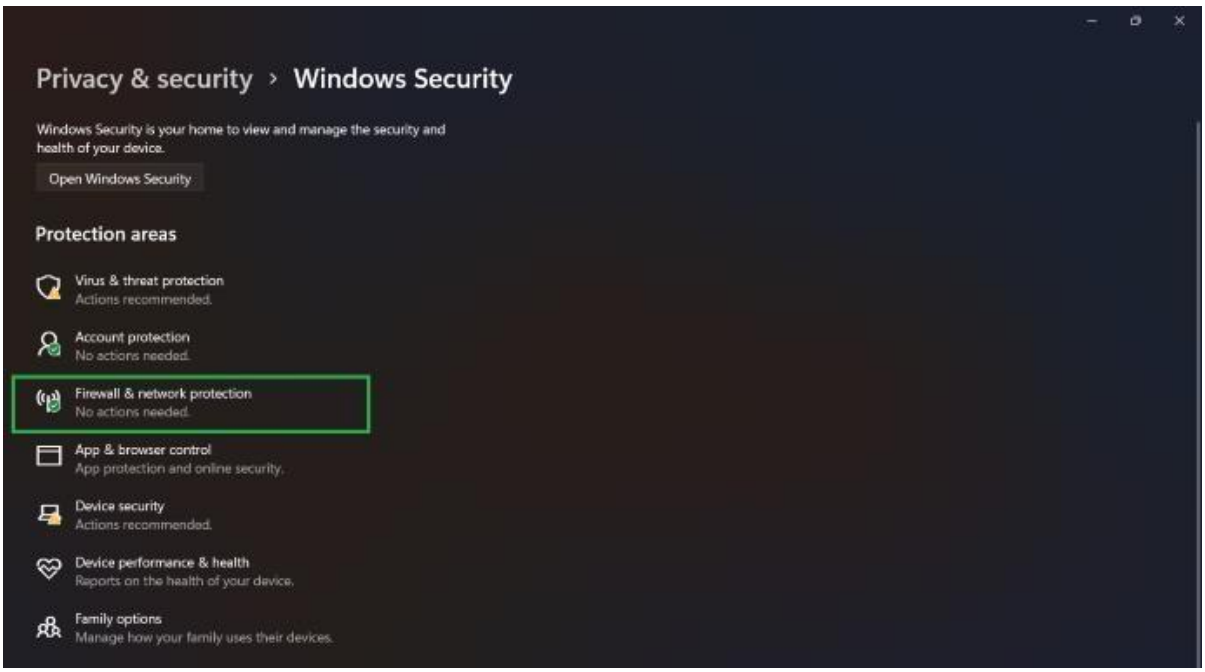
Step 3: In the left pane of Settings, click **Privacy & security**.



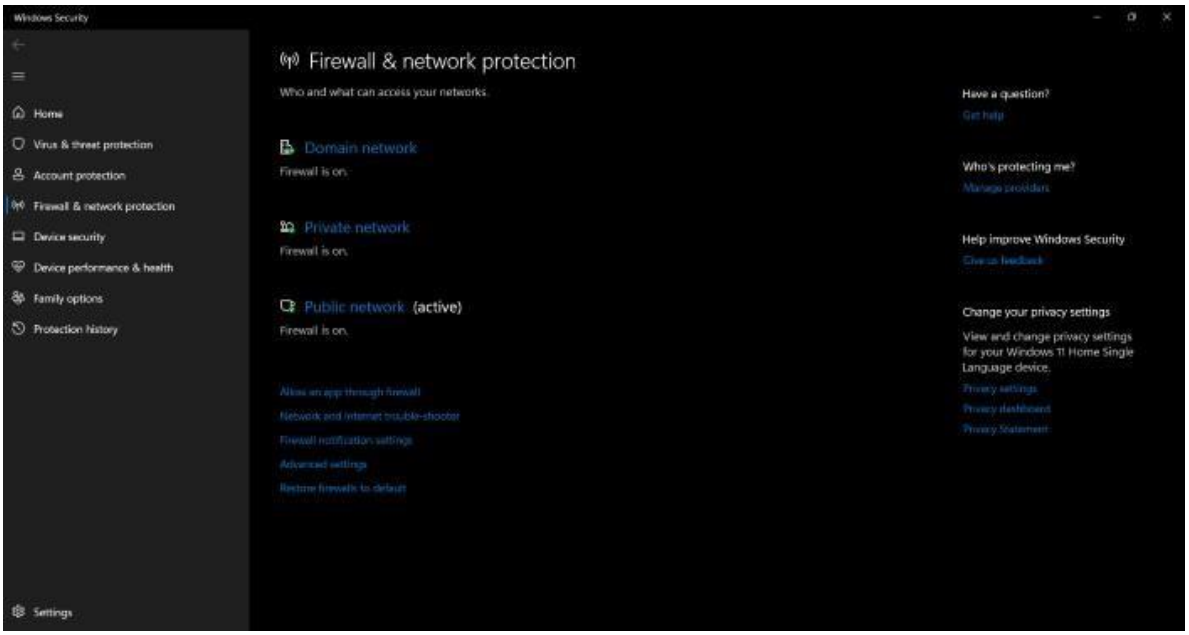
- **Step 4:** Click **Windows Security** option in Privacy & security menu.



Step 5: Select **Firewall & network protection**.

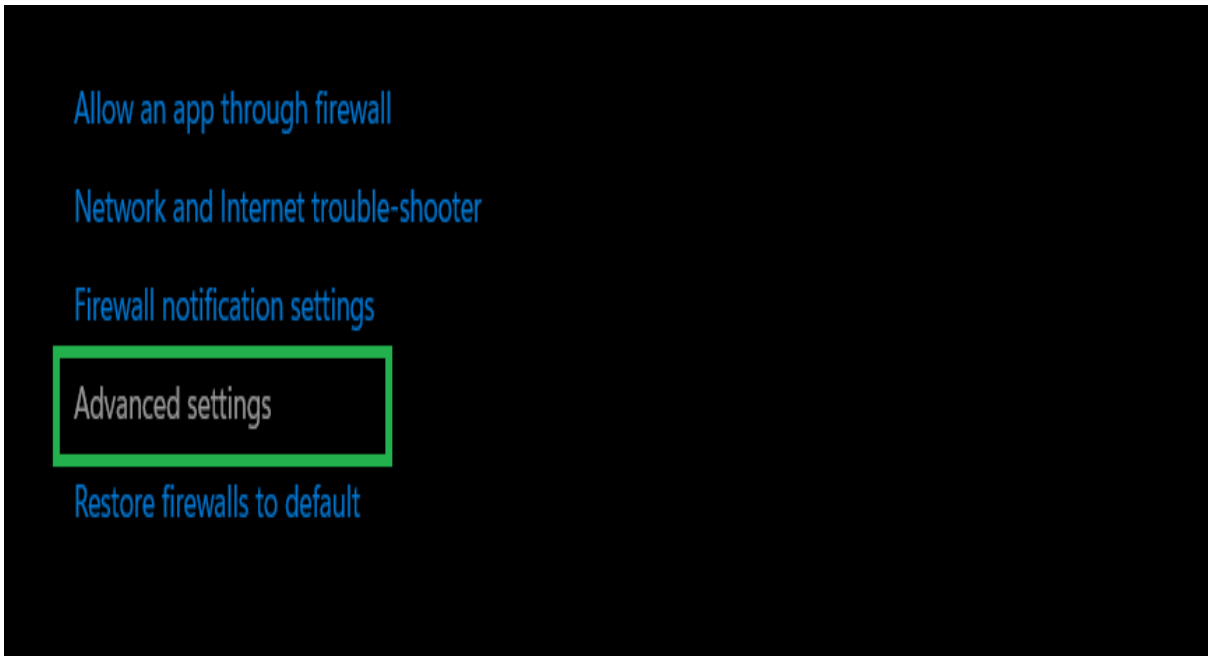


Step 6: Now **Window's Security** window will pop up window's. Here you can verify whether your Defender firewall is active or not.

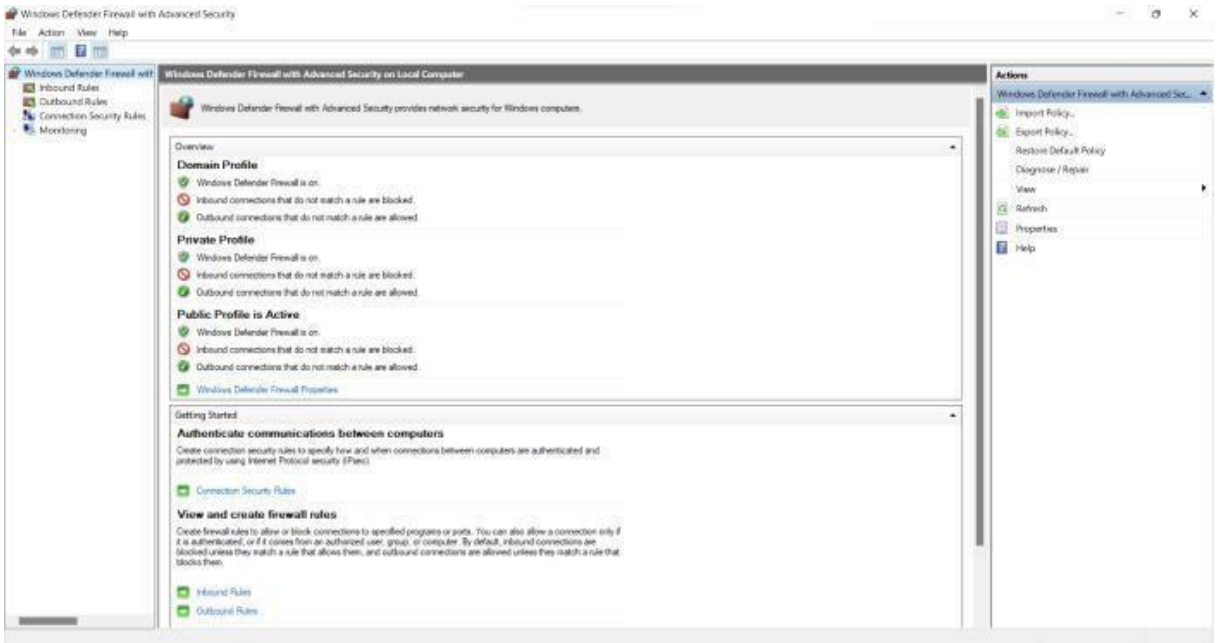


Step 7:

Now to configure the firewall according to your requirement, click **Advanced settings**. You will be prompted by User Account Control to give Administrative access to Windows Defender to make changes. Click **Yes** to proceed.



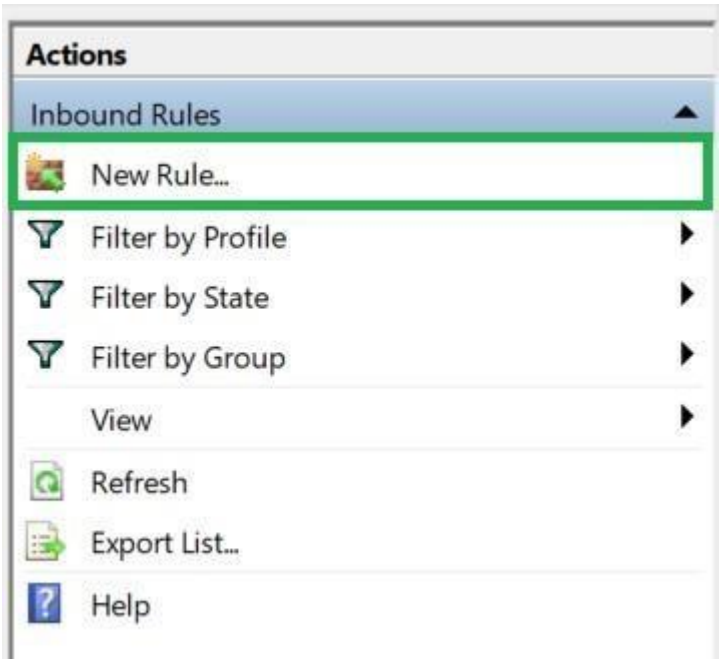
Step 8: Windows Defender Firewall with Advanced Security window will launch after giving administrative permission.



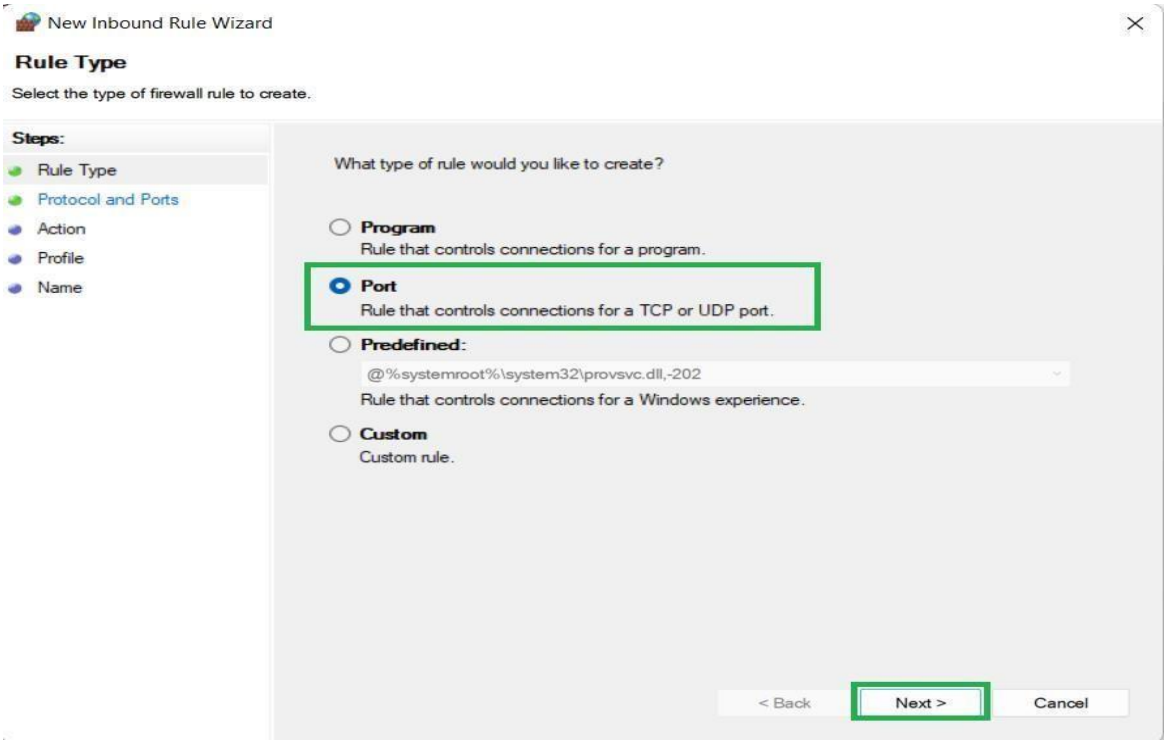
- **Step 9:** The left pane has several options:

- Inbound rules: Programs, processes, ports can be allowed or denied the incoming transmission of data within this inbound rules.
- Outbound rules: Here we can specify whether data can be sent outwards by that program, process, or port.

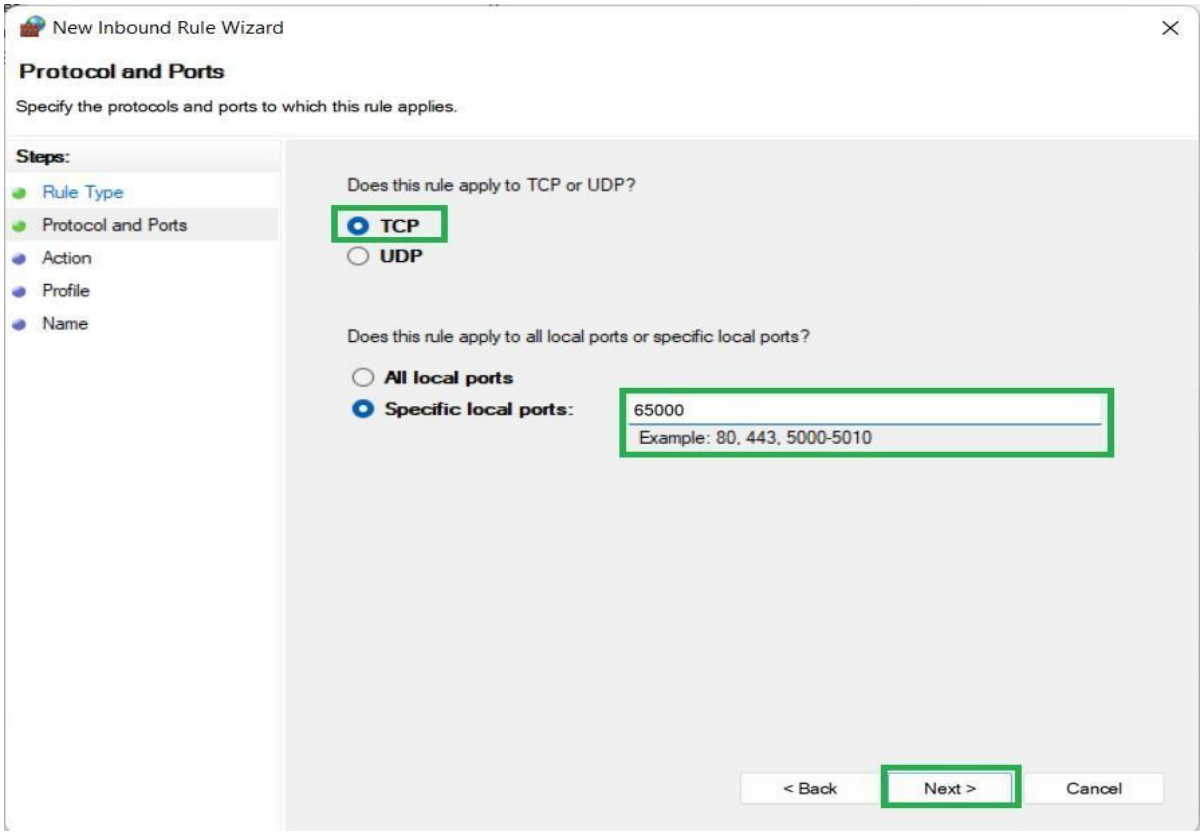
- **Step 10:** To add a new inbound rule, select **Inbound Rules** option, then click **New Rule…** from the right pane.



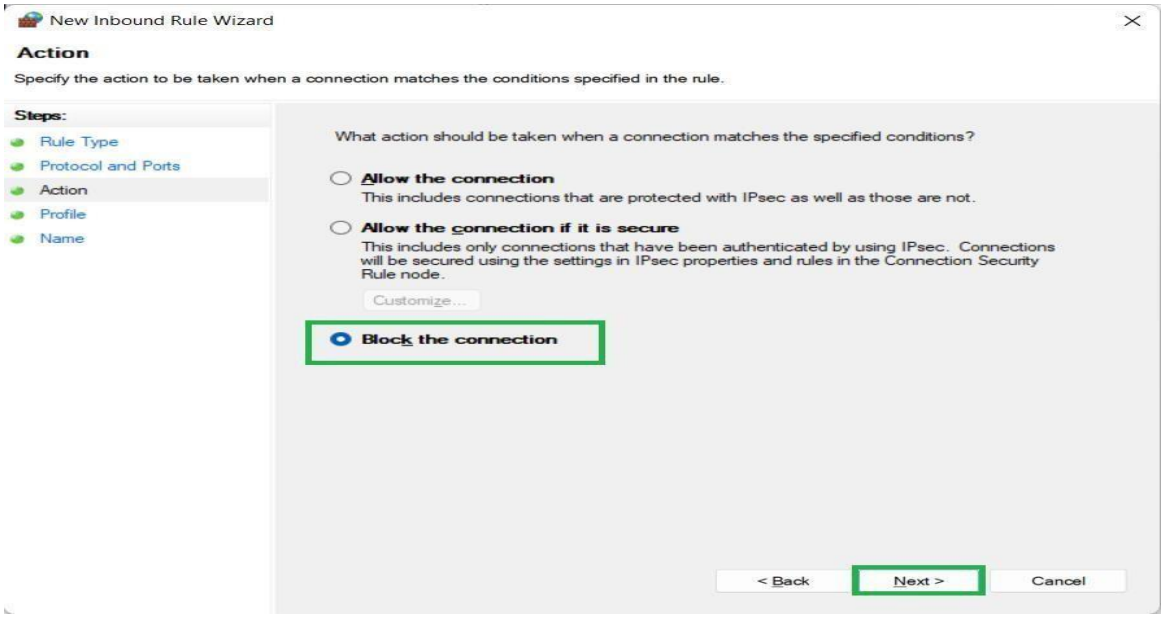
Step 11: Now we will configure an inbound rule for a network port. A **New Inbound Rule Wizard** window pops-up, select Port option and click next.



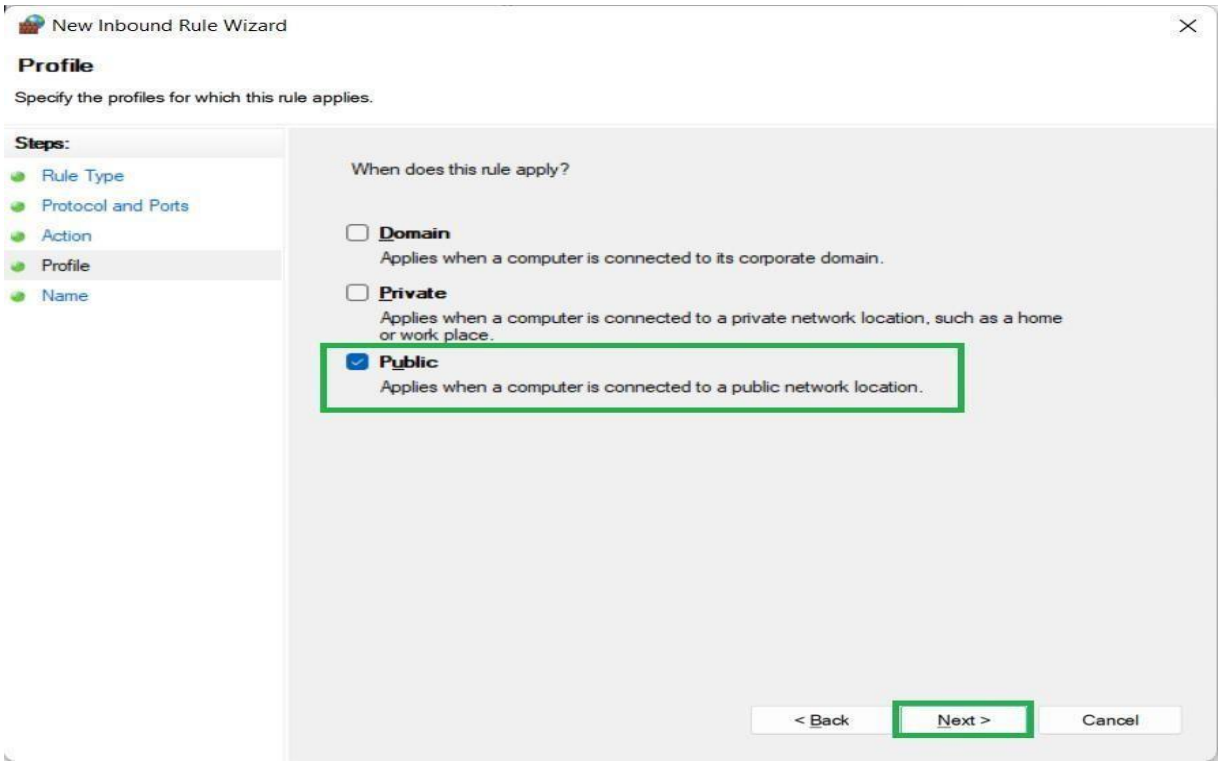
Step 12: Now select **TCP** and specify port number **65000**.



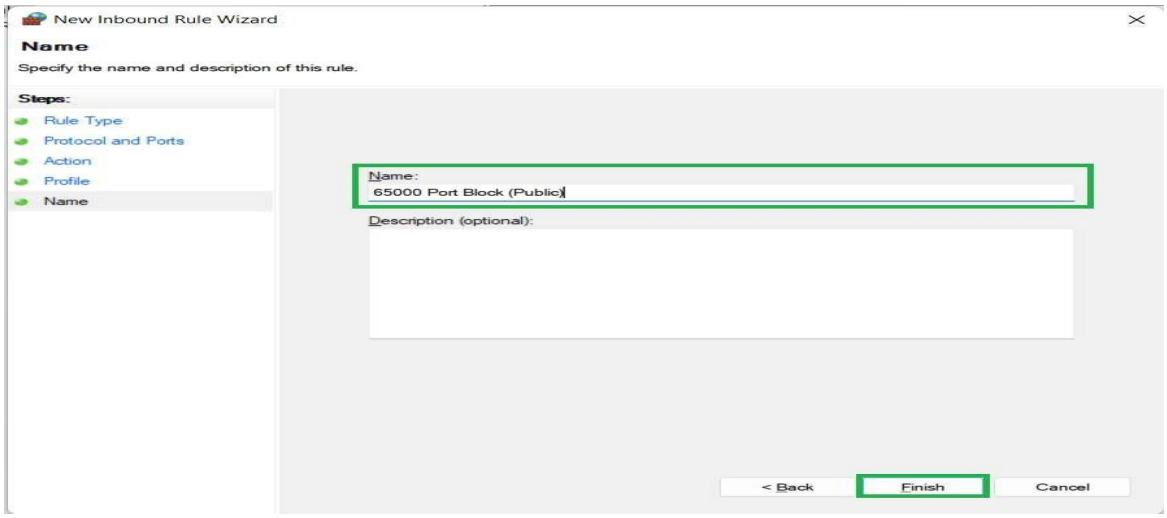
Step 13: Now we can select the action we need to take on this port. We will block the inbound connection by selecting **Block the connection** option then click **Next**.



Step 14: Here we can specify when should this rule come into action. We will keep only **Public** option selected and move **Next**.



Step 15: This is the last step. Here we provide a name to this rule so that we can keep track of it later in the Inbound rules list. Write the name "**65000 Port Block (Public)**". Click **Finish**.



Step 16: The inbound rule is successfully created. We can find "**65000 Port Block (Public)**" in the Inbound rules list.

Step 17: Right-click the rule we just created and there are multiple options with which it can be **Disabled** or **Deleted**.

Firewall can be configured on Windows in the above-mentioned way.

RESULT :

We protect our system with the properties of firewall in windows.

EX.NO:10 (b)	VPN
DATE:	

AIM:

To knows the uses of Vpn and functionality with application

1. Step 1: Line up key VPN components. ...
2. Step 2: Prep devices. ...
3. Step 3: Download and install VPN clients. ...
4. Step 4: Find a setup tutorial. ...
5. Step 5: Log in to the VPN. ...
6. Step 6: Choose VPN protocols. ...
7. Step 7: Troubleshoot. ...
8. Step 8: Fine-tune the connection.

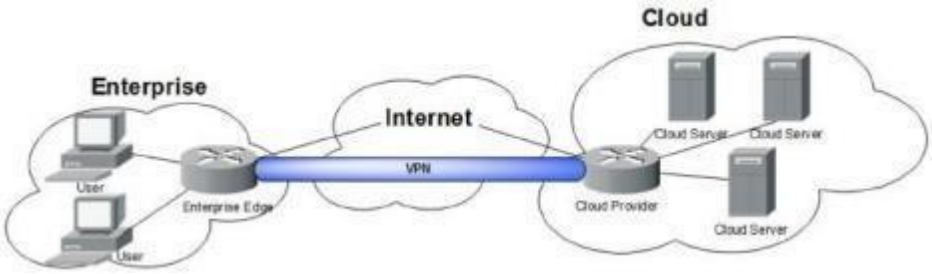
Configuring a Virtual Private Network (VPN) involves setting up a secure and encrypted connection between your device and a remote server. VPNs are commonly used to enhance online privacy, security, and anonymity. Below, I'll provide a general overview of how to configure a VPN:

1. **Choose a VPN Service:** Start by selecting a reputable VPN service provider. There are many options available, such as NordVPN, ExpressVPN, CyberGhost, and many more. Subscribe to the VPN service and follow their setup instructions.
2. **Download and Install the VPN Client:** Most VPN services offer dedicated apps for various platforms, including Windows, macOS, iOS, Android, and Linux. Download the appropriate VPN client for your device, and install it.
3. **Launch the VPN Client:** Open the VPN client on your device. You may need to log in using your VPN service credentials.
4. **Connect to a VPN Server:** In the VPN client, choose a server location to which you want to connect. VPN services typically offer servers in various countries. Your choice of server location may affect your internet speed and the websites or services you can access.
5. **Optional Configuration Settings:** VPN clients often provide advanced settings that you can configure to meet your specific needs. These settings might include the VPN protocol (e.g., OpenVPN, L2TP, or IKEv2), a kill switch to stop internet traffic if the VPN connection drops, split tunneling to specify which apps or websites use the VPN, etc. Configure these settings according to your requirements.
6. **Connect to the VPN:** Click or tap the "Connect" or "Start" button in the VPN client to establish a connection to the selected server. Once the connection is established, your internet traffic will be encrypted and routed through the VPN server.
7. **Verify Your Connection:** You can verify that your VPN connection is active by visiting a website like "WhatIsMyIP" to confirm that your IP address is now associated with the VPN server location.

8. **Use the Internet Securely:** Your internet activity is now encrypted and secure. You can access blocked content, enhance your privacy, and protect your data from eavesdropping on public Wi-Fi networks.
9. **Disconnect from the VPN:** When you're finished using the VPN, disconnect from the server by using the VPN client.
10. **Customize Your Experience:** Some VPN services offer additional features, such as ad-blocking, anti-malware, and more. Explore your VPN service's features to see how you can tailor your VPN experience.

It's essential to choose a trusted VPN service, as they have access to your internet traffic. Be sure to read their privacy policy and terms of service to understand their data handling practices. Additionally, keep your VPN software and device operating system up to date to ensure the highest level of security.

Architecture of VPN



Surfshark VPN Options and Features

1. CleanWeb

CleanWeb was able to block ads that managed to slip even past our regular ad blockers.

2. MultiHop

The MultiHop option provides some additional security but does have a negative impact on your internet speeds. After all, your traffic has to go through two VPN servers instead of one.

3. Whitelister

The Whitelister for apps and websites allows you to connect to specific apps and websites you trust using your own IP address instead of that of the VPN server (split tunneling).

4. Surfshark Alert

Surfshark Alert is an option you can use that will notify you when your email or passwords might be in danger of being compromised.

5. Surfshark Search

Surfshark Search is Surfshark's answer to the anonymous search engine DuckDuckGo. However, both Surfshark Alert and Surfshark Search are paid expansions of the normal Surfshark subscription.

6. Surfshark VPN for torrenting

Surfshark has specialized P2P servers that help you download torrents securely. You don't have to manually select these servers: they are automatically activated when you open a torrenting program.

7. Camouflage Mode

Camouflage Mode is Surfshark's version of "obfuscation technology." It's primarily designed to disguise your VPN traffic to bypass any content filtering and make it more difficult to monitor your activities. It runs automatically when you pick OpenVPN (TCP or UDP) as its protocol.

8. NoBorders Mode -

NoBorders Mode is a mode in which you can use Surfshark and access the internet freely even in restrictive regions. When Surfshark detects any kind of restrictions on your network, it automatically enables the NoBorders mode. This gives you a selected list of servers that perform well despite network restrictions.

9. GPS spoofing

Recently, Surfshark has added a GPS spoofing function to its service. GPS spoofing allows you to change your virtual location, even when it isn't based on your IP address. Surfshark's GPS spoofing option is only available on Android for now.

10. Kill Switch

A kill switch is an essential security feature for any VPN. It instantly "kills" your internet connection if your VPN stops working for whatever reason. This prevents data leaks (like your IP address) and keeps you protected when your VPN falters. Surfshark offers a kill

switch on Windows, macOS, iOS, and Android. You can turn it on by going into the settings in the Surfshark app.

Logging and privacy

Surfshark has a “no-logs’’ policy. They have made every effort to require as little personal information from you as possible for your VPN to work. Secondly, Surfshark — like most VPNs — requires certain information to monitor their service. For example, they use anonymized information to keep track of how busy their servers are and to see if there are any connection issues. As mentioned previously, they have a strict “no-logs” policy, and we couldn’t find anything in their regulations that contradict this.

COCLUSION

Furthermore, Surfshark recently switched to a RAM-only server network, which means that there is no longer any data on physical servers. This form of storage also ensures that data is automatically deleted when you disconnect. Your data will therefore only be stored temporarily and cannot be retrieved afterward. In addition, Surfshark is one of the first VPNs to offer two-factor authentication (2FA). This means logging in happens through two steps, which is an effective way to protect yourself against all kinds of online attacks. For Surfshark Alert users, this two-step authentication is mandatory, while the ‘normal’ Surfshark user can choose to use it for extra protection.

RESULT:

The modules are executed successfully with VPN application