**INDEX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr  No | Date | Title | Page | Sign |
| 1 |  | Implementing Substitution and Transposition Ciphers |  |  |
| 2 |  | RSA Encryption and Decryption |  |  |
| 3 |  | Message Authentication Codes |  |  |
| 4 |  | Digital Signatures |  |  |
| 5 |  | Key Exchange using Diffie-Hellman |  |  |
| 6 |  | IP Security (IPsec) Configuration |  |  |
| 7 |  | Malware Analysis and Detection |  |  |
| 8 |  | Firewall Configuration and Rule based Filtering |  |  |

**Practical 1**

**Aim:- Implementing Substitution and Transposition Ciphers:**

Design and implement algorithms to encrypt and decrypt messages using classical substitution and transposition techniques.

1. *Input:*

import java.io.\*;

class CaesarCipherProgram

{

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

{

String pt,ct="";

int key=3;

BufferedReader brk=new BufferedReader(new InputStreamReader(System.in));

System.out.print("Enter Plain Text");

pt=brk.readLine();

pt=pt.toLowerCase();

for(int i=0;i<pt.length();i++)

{

int acvalue=(int)pt.charAt(i);

acvalue=acvalue+key;

if(acvalue>122)

{

int diff=acvalue-122;

acvalue=96+diff;

}

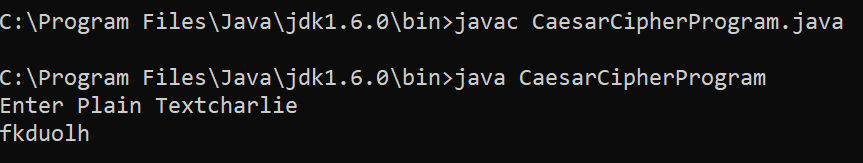
ct=ct+(char)acvalue;

}

System.out.println(ct);

}

}

Output:

1. *Input:*

import java.util.Scanner;

public class MonoalphabeticCipher

{

public static char p[]={'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z'};

public static char ch[]={'Q','W','E','R','T','Y','U','I','O','P','A','S','D','F','G','H','J','K','L','Z','X','C','V','B','N','M'};

public static String doEncryption(Strings)

{

char c[]=new char[(s.length())];

for (int i=0; i<s.length();i++)

{

for (int j=0;j<26;j++)

{

if (p[j]==s.charAt(i))

{

c[i]=ch[j];

break;

}

}

}

return (new String(c));

}

public static String doDecryption(String s)

{

char p1[]=new char[(s.length())];

for (int i=0;i<s.length();i++)

{

for (int j=0;j<26;j++)

{

if (ch[j]==s.charAt(i))

{

p1[i]=p[j];

break;

}

}

}

return (new String(p1));

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the message:");

String en = doEncryption(sc.next().toLowerCase());

System.out.println("Encryption message:"+en);

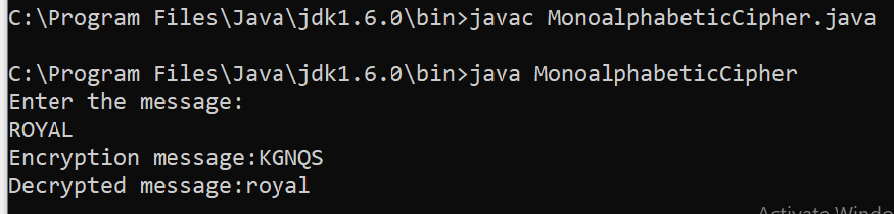
System.out.println("Decrypted message:"+doDecryption(en));

sc.close();

}

}

Output

: 

1. **Program that demonstrates Rail-Fence Technique.**

import java.io.\*; class Railfence

{

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

{

String pt,ct="";

BufferedReader brk=new BufferedReader(new InputStreamReader(System.in));

System.out.print("Enter Plain Text "); pt=brk.readLine();

pt=pt.toLowerCase();

for(int i=0;i<pt.length();i++)

{

if(i%2==0) ct=ct+pt.charAt(i);

}

for(int i=0;i<pt.length();i++)

{

if(i%2!=0) ct=ct+pt.charAt(i);

}

System.out.println("Cipher Text "+ct);

}

}

**Practical 2**

**Aim:- RSA Encryption and Decryption:**

Implement the RSA algorithm for public-key encryption and decryption, and explore its properties and security considerations.

*Input :*

import math

# step1

p=2

q=3

#step 2

n=p\*q

print("n=",n)

#step 3

phi = (p-1)\*(p-1)

#step 4

e=2

while(e<phi):

if (math.gcd(e, phi) == 1):

break

else:

e +=1

print("e=" ,e)

#step 5

k = 2

d=((k\*phi)+1)/e

print("d=", d)

print(f'Public Key: {e , n}' )

print (f' Private Key: {d , n}')

#plan text

msg = 11

print(f' original message:{msg}')

# encryption

C = pow(msg, e)

C = math.fmod(C, n)

print (f' Encrypted message:{C}')

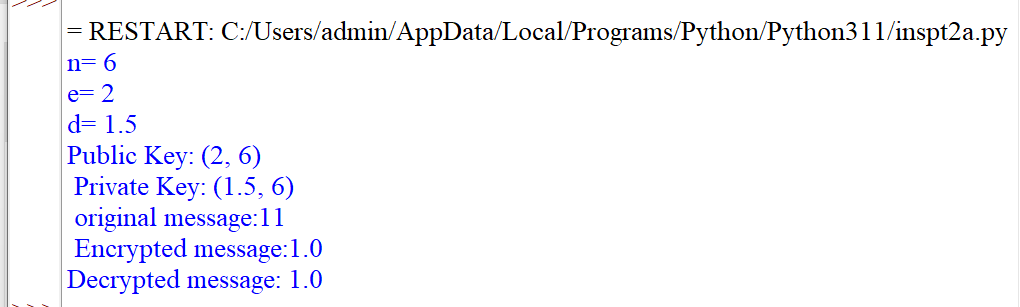
#Decryption

M=pow(C,d)

M=math.fmod(M,n)

print (f'Decrypted message: {M}' )

Output*:*



**Practical 3**

**Aim:- Message Authentication Codes:**

Implement algorithms to generate and verify message authentication codes (MACs) for ensuring data integrity and authenticity**.**

*Input:*

import hashlib

result = hashlib.md5(b'Ismile')

result1=hashlib.md5(b'Esmile')

#printing the equivalent byte value

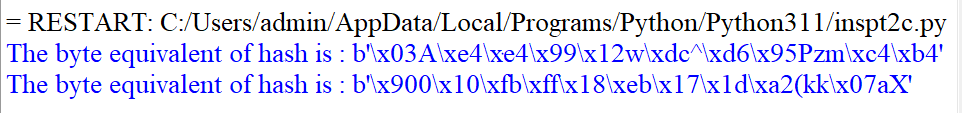
print("The byte equivalent of hash is : ",end="")

print(result.digest())

print("The byte equivalent of hash is : ",end="")

print(result1.digest())

output:



# B:- Pyhton code for implementing SHA Algorithm:

*Input:*

import hashlib

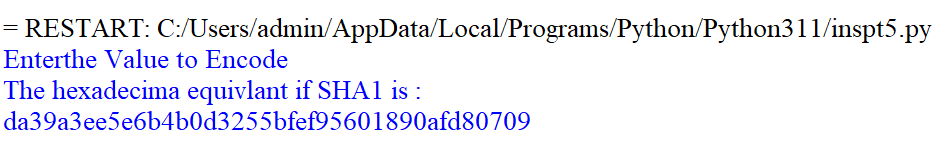
str = input ("Enterthe Value to Encode ")

result = hashlib.sha1 (str.encode())

print ("The hexadecima equivlant if SHA1 is :")

print(result.hexdigest())

output:



**Practical 4**

**Aim:- Digital Signatures**

*Input:*

from Crypto.Signature import PKCS1\_v1\_5

from Crypto.Hash import SHA256

from Crypto.PublicKey import RSA

from Crypto import Random

def generate\_signature(private\_key,message):

key = RSA.importKey(private\_key)

hashed\_message = SHA256.new(message.encode('utf-8'))

signer = PKCS1\_v1\_5.new(key)

signature = signer.sign(hashed\_message)

return signature

def verify\_signature(public\_key,message,signature):

key = RSA.importKey(public\_key)

hashed\_message = SHA256.new(message.encode('utf-8'))

verifier = PKCS1\_v1\_5.new(key)

return verifier.verify(hashed\_message,signature)

random\_generator = Random.new().read

key\_pair =RSA.generate(2048,random\_generator)

public\_key =key\_pair.publickey().export\_key()

private\_key = key\_pair.export\_key()

message = "Hello,World"

signature = generate\_signature(private\_key,message)

print("Generated Signature : " ,signature)

is\_valid = verify\_signature(public\_key,message,signature)

print("Signature Verification Result : ",is\_valid)

Output:



**Practical 5**

**Aim:- Key Exchange using Diffie-Hellman:**

Implement the Diffie-Hellman key exchange algorithm to securely exchange keys between two entities over an insecure network

*Input:*

from random import randint

if \_\_name\_\_== '\_\_main\_\_':

P=23

G=9

print('The value of P is: %d'%(P))

print('The value of G is: %d'%(G))

a=4

print('Secret Number for Alice is : %d'%(a))

x=int(pow(G,a,P))

b=6

print('Secret Number for Bob is : %d'%(b))

y = int(pow(G,a,P))

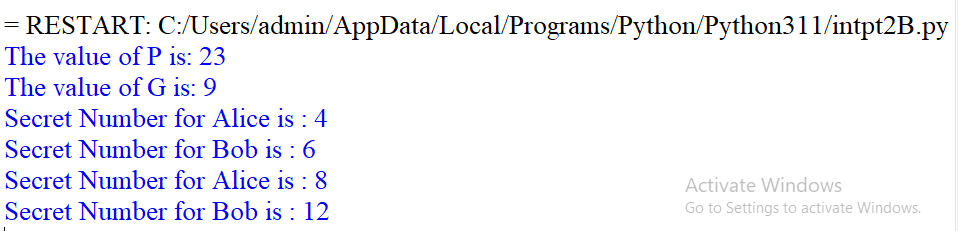
ka = int(pow(y,a,P))

kb = int(pow(x,b,P))

print('Secret Number for Alice is : %d'%(ka))

print('Secret Number for Bob is : %d'%(kb))

*Output*:



**Practical 6**

**Aim:-** To Configure IPSec on network devices to provide secure communication and protect against unauthorized access and attacks.

#### Theory:

Some theoretical aspects of IPSec and the concept of an IPSec VPN tunnel:

1. IPSec Overview:

* IPSec (Internet Protocol Security) is a comprehensive suite of protocols and standards used for securing communication over IP networks, such as the Internet.
* It ensures the confidentiality, integrity, and authenticity of data transmitted between devices or networks.

1. Security Goals of IPSec:

* Confidentiality: IPSec achieves data privacy through encryption.
* Integrity: It guarantees that data remains unaltered during transit.
* Authentication: IPSec verifies the identity of communicating parties to prevent unauthorized access and impersonation.

1. Components of IPSec:

- IPSec comprises multiple protocols and elements, including Authentication Header (AH), Encapsulating Security Payload (ESP), Security Associations (SAs), and key management protocols.

1. IPSec VPN Tunnel:

* An IPSec VPN tunnel is a secure, encrypted connection established between two endpoints or networks over the Internet or untrusted networks.
* It is created using the IPSec suite to provide a secure and private channel for data transmission.

1. Establishing a VPN Tunnel:

* The process begins with the negotiation and establishment of Security Associations (SAs) between the endpoints.
* These SAs define parameters like encryption methods, authentication, and shared keys.

1. Modes of Operation:

* VPN tunnels can operate in either Transport Mode (securing data payload) or Tunnel Mode (securing entire IP packets, including headers).
* Transport Mode is often used for host-to-host communication, while Tunnel Mode is suitable for network-to-network connections.

1. Data Encryption and Authentication:

* Data transmitted through the VPN tunnel is encrypted using algorithms specified in the SAs, ensuring data privacy.
* Authentication and data integrity checks prevent tampering or unauthorized access.

1. Routing and Secure Communication:

* Once established, the VPN tunnel allows secure data routing between the endpoints or network
* Applications and services on either side can communicate securely, even over untrusted networks like the Internet.

1. Use Cases:

- IPSec VPN tunnels are used for various purposes, including remote access VPNs, site-to-site VPNs, secure data transfer, and protecting real-time communication like VoIP and video conferencing.

1. Key Management:

* Secure key management is critical for the long-term security of IPSec VPN tunnels.
* Keys can be generated manually or through automated key exchange protocols like Internet Key Exchange (IKE).

1. Security Policies:

- Organizations define security policies that determine when and how IPSec should be applied to protect specific types of traffic or communication.

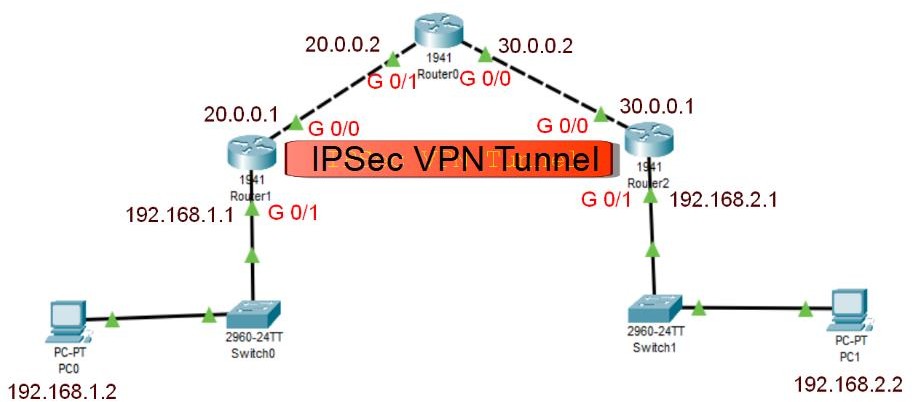
1. Interoperability:

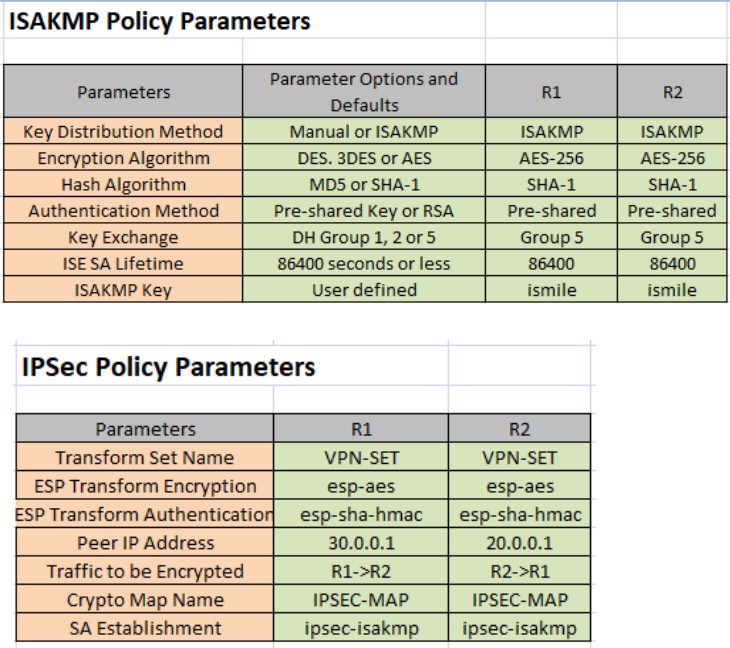
- IPSec is widely adopted, ensuring interoperability between different vendors' equipment and making it a versatile choice for securing networks and data.

Understanding the principles of IPSec and IPSec VPN tunnels is essential for designing, deploying, and managing secure communication in various network environments, ensuring data remains confidential, unaltered, and protected from unauthorized access.

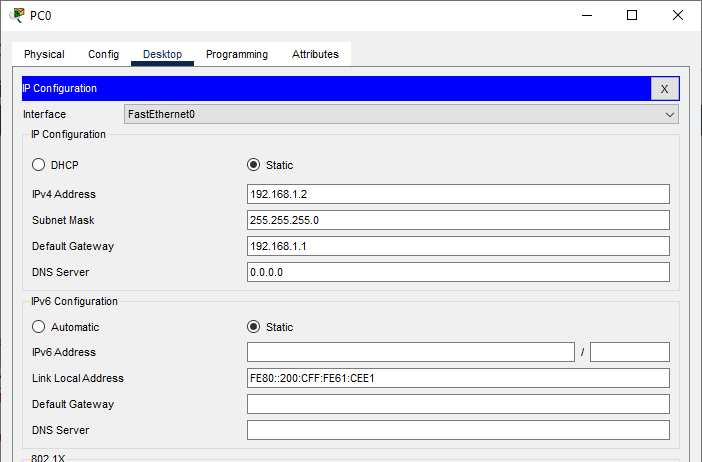
#### Topology:

We use the following topology for the present case

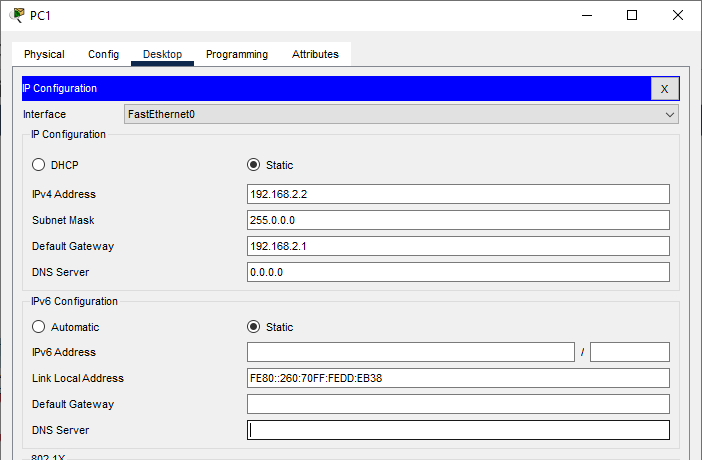




**Configuring PC0:**

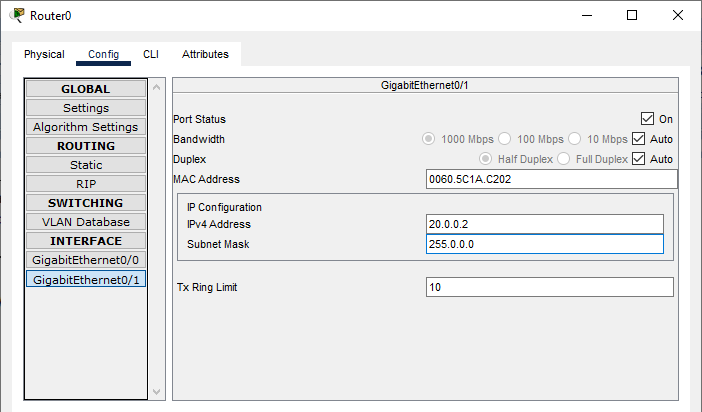


**Configuring PC1:**

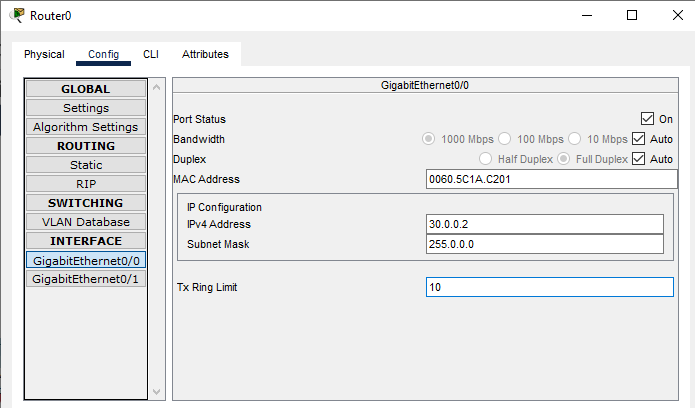


### Configuring Router0:

#### Interface GigabitEthernet0/1:

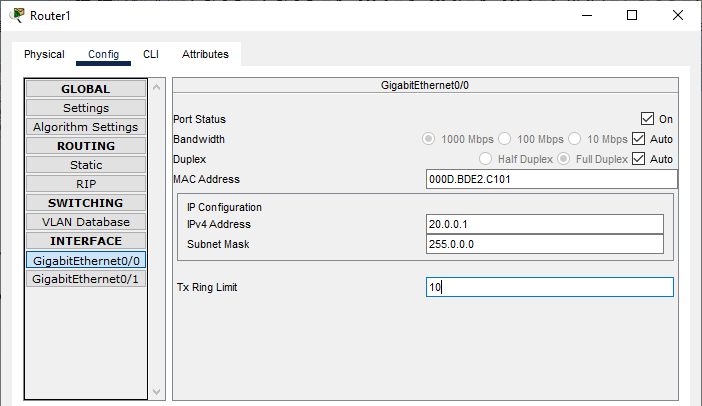


**Interface GigabitEthernet0/0:**

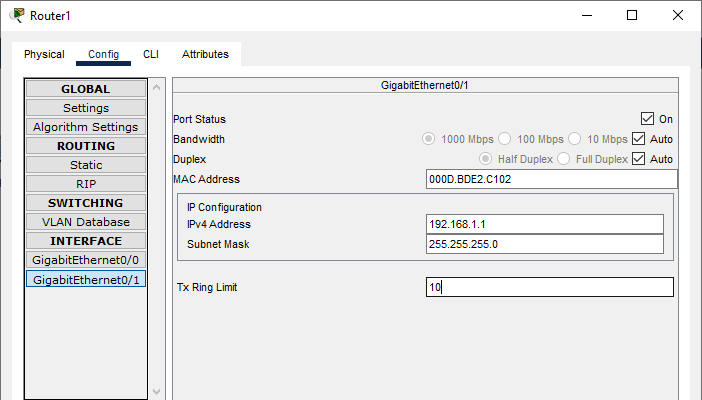


### Configuring Router1:

#### Interface GigabitEthernet0/0:

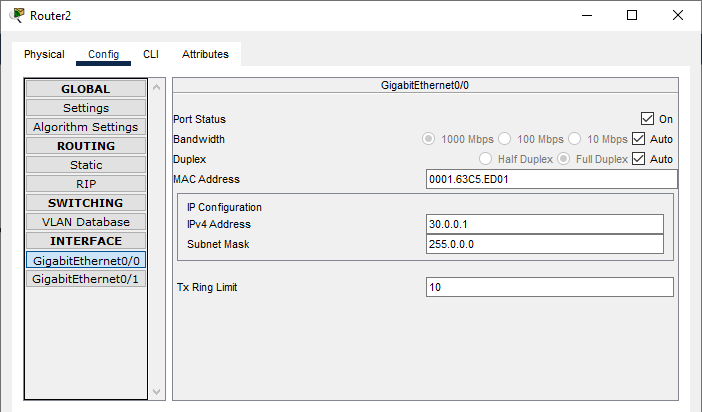


**Interface GigabitEthernet0/1:**

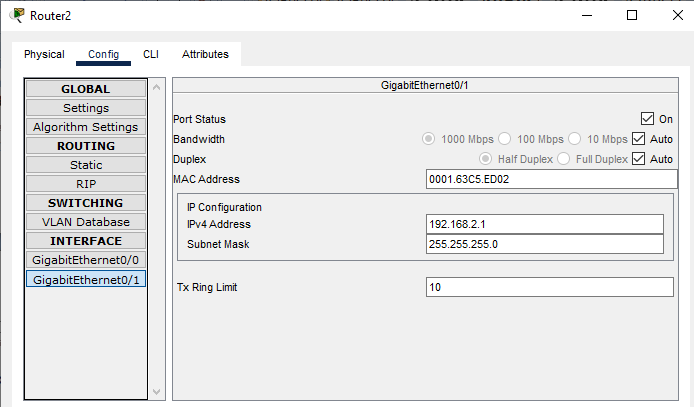


### Configuring Router2:

#### Interface GigabitEthernet0/0:



**Interface GigabitEthernet0/1:**

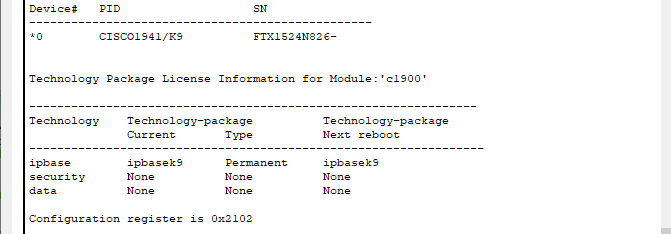


### Checking and Enabling the Security features in Router R1 and R2:

#### Enter the following command in the CLI mode of Router1

Router(config)#ip route 0.0.0.0 0.0.0.0 20.0.0.2 Router(config)#hostname R1

R1(config)#exit R1#show version



(We see that the security feature is not enabled, hence we need to enable the security package R1#

R1#configure terminal

Enter configuration commands, one per line. End with CNTL/Z. R1(config)#

R1(config)#license boot module c1900 technology-package securityk9 R1(config)#exit

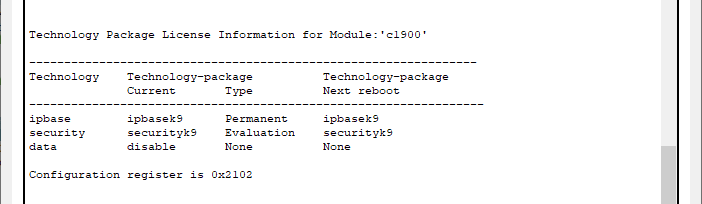
R1#

R1#copy run startup-config

R1#reload

R1>enable

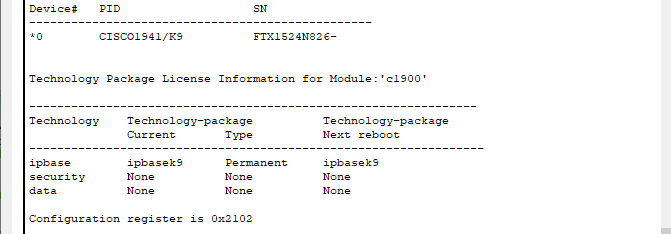
R1#show version



(The security package is enabled)

**Enter the following command in the CLI mode of Router2** Router(config)#ip route 0.0.0.0 0.0.0.0 30.0.0.2 Router(config)#hostname R2

R2(config)#exit R2#show version



(We see that the security feature is not enabled, hence we need to enable the security package R2#

R2#configure terminal

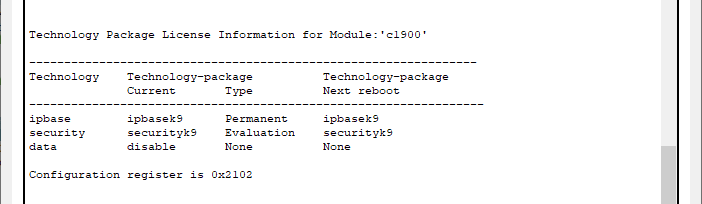
Enter configuration commands, one per line. End with CNTL/Z. R2(config)#

R2(config)#license boot module c1900 technology-package securityk9 R2(config)#exit

R2#

R2#copy run startup-config R2#reload

R2>enable R2#show version



(The security package is enabled)

#### Enter the following command in the CLI mode of Router0

Router>enable Router#configure terminal Router(config)#hostname R0 R0(config)#

#### Defining the Hostname for all Routers and Configuring the Routers R1 and R2 for IPSec VPN tunnel

R1#configure terminal

R1(config)#access-list 100 permit ip 192.168.1.0 0.0.0.255 192.168.2.0 0.0.0.255 R1(config)#crypto isakmp policy 10

R1(config-isakmp)#encryption aes 256 R1(config-isakmp)#authentication pre-share R1(config-isakmp)#group 5

R1(config-isakmp)#exit

R1(config)#crypto isakmp key ismile address 30.0.0.1

R1(config)#crypto ipsec transform-set R1->R2 esp-aes 256 esp-sha-hmac R1(config)#

R2#

R2#configure terminal

R2(config)#access-list 100 permit ip 192.168.2.0 0.0.0.255 192.168.1.0 0.0.0.255 R2(config)#crypto isakmp policy 10

R2(config-isakmp)#encryption aes 256 R2(config-isakmp)#authentication pre-share R2(config-isakmp)#group 5

R2(config-isakmp)#exit

R2(config)#crypto isakmp key ismile address 20.0.0.1

R2(config)#crypto ipsec transform-set R2->R1 esp-aes 256 esp-sha-hmac R2(config)#

R1>enable R1#configure terminal

R1(config)#crypto map IPSEC-MAP 10 ipsec-isakmp R1(config-crypto-map)#set peer 30.0.0.1

R1(config-crypto-map)#set pfs group5

R1(config-crypto-map)#set security-association lifetime seconds 86400 R1(config-crypto-map)#set transform-set R1->R2

R1(config-crypto-map)#match address 100 R1(config-crypto-map)#exit R1(config)#interface g0/0

R1(config-if)#crypto map IPSEC-MAP

R2>enable R2#configure terminal

R2(config)#crypto map IPSEC-MAP 10 ipsec-isakmp R2(config-crypto-map)#set peer 20.0.0.1

R2(config-crypto-map)#set pfs group5

R2(config-crypto-map)#set security-association lifetime seconds 86400 R2(config-crypto-map)#set transform-set R2->R1

R2(config-crypto-map)#match address 100 R2(config-crypto-map)#exit R2(config)#interface g0/0

R2(config-if)#crypto map IPSEC-MAP

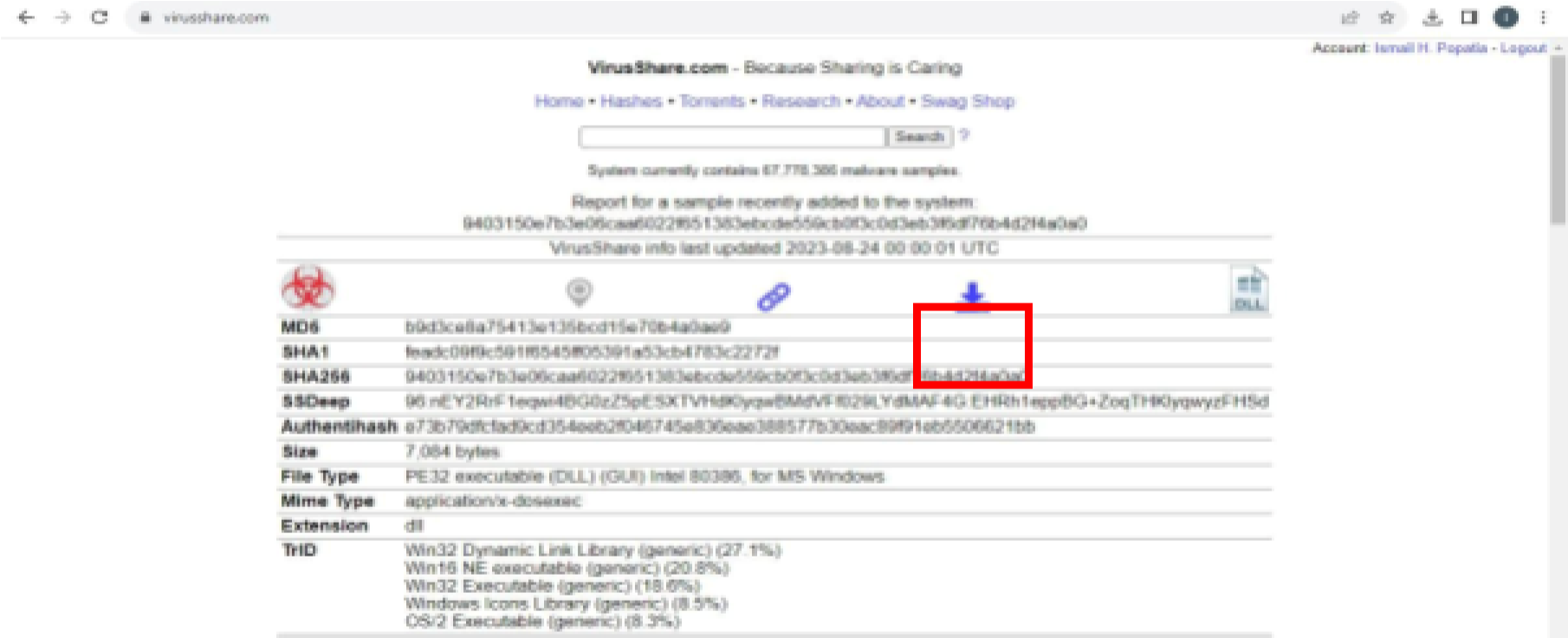
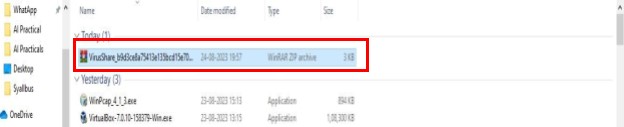
We verify the working of the IPSec VPN tunnel using the ping command as follows

|  |  |
| --- | --- |
| **Output:** | Pinging PC2(192.168.2.2) from PC1 and then PC1(192.168.1.2) from PC2 |
|  |  |

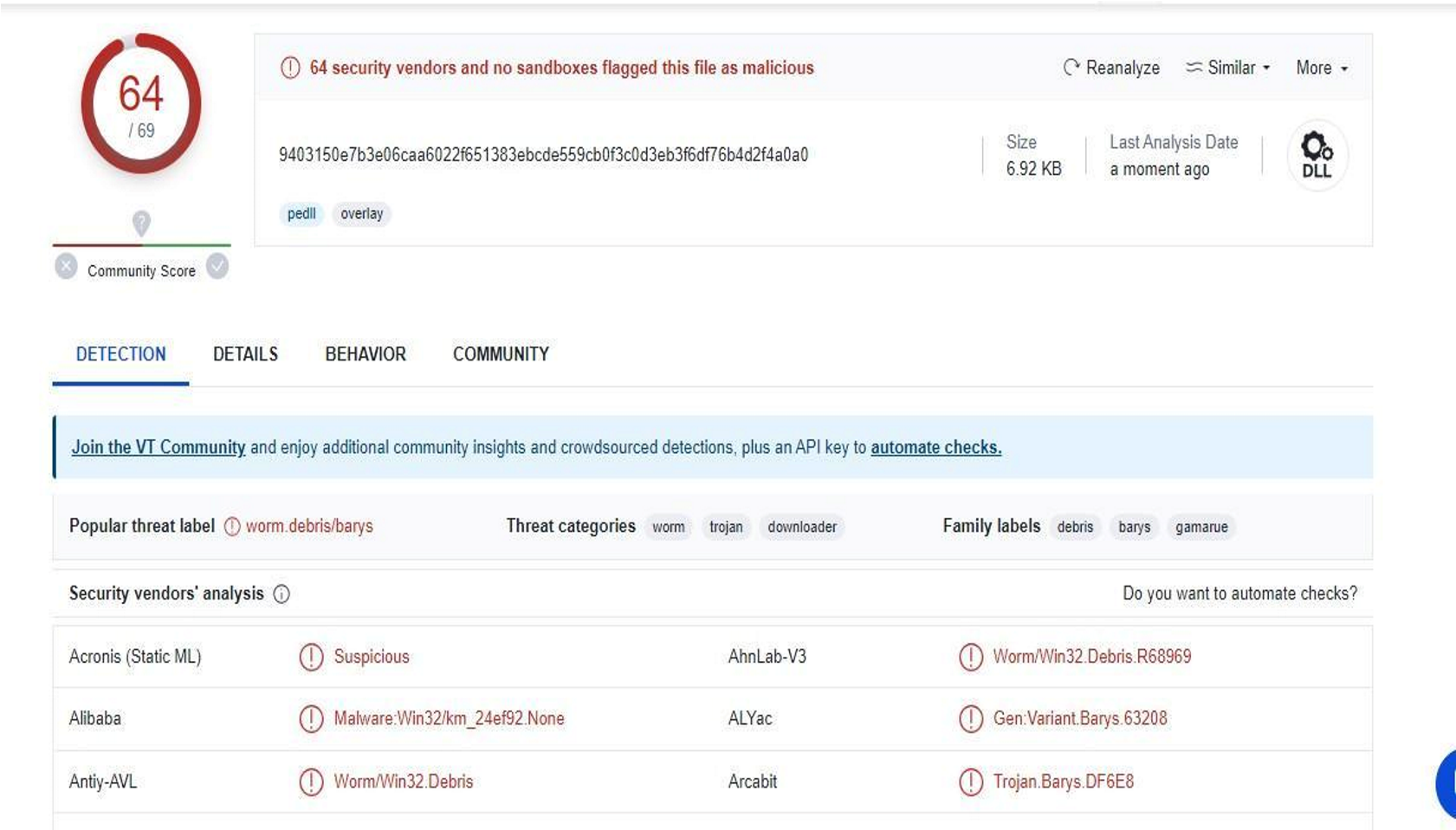
**Practical 7**

**Aim:-** **Malware Analysis and Detection**

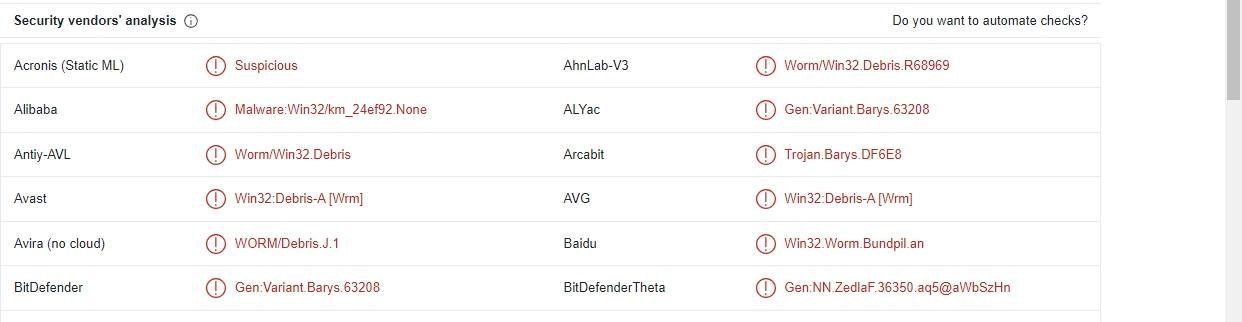
For analysing the Malware, we need one. A clean sample of the Malware needs to be downloaded from a trusted website, the downloading and analysis is demonstrated by the following steps

1. We select the website [www.virusshare.com](http://www.virusshare.com/) for downloading the clean sample of Malware (an account needs to be created for the same). Any other source can be selected to download the Malware (clean sample and authorised site)
2. By clicking the above download icon the Malware gets downloaded in ZIP format.
3. For unzip the password is “infected”, there is no need to unzip the file, we create a folder

“Malware” on desktop and save the file in the folder

1. In order to analyse the Malware, we select the website [www.virustotal.com](http://www.virustotal.com/)
2. Click on “Choose File” and select the file from the location (ZIP file will do, if asks for password enter infected)
3. We get the following after the upload is complete

We interpret the following findings

1. 64 security vendors out of 69 flagged this file as malicious
2. The detection tab shows the threats-type which were flagged by the vendors for

e.g

1. The details tab gives the following information
   1. Basic properties
   2. History
   3. Compiler products iv. Header
   4. Sections
   5. Imports
   6. Exports
   7. Overlays
2. The Behavior tab gives the following information
   1. Activity summary
   2. MITRE ATT&CK Tactics and Techniques
   3. Behavior Similarity Hashes
   4. Process and service actions

**Practical 8**

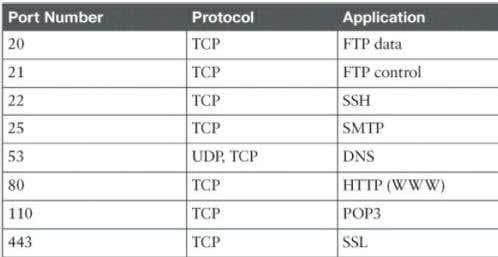
**Aim:- Firewall Configuration and Rule based Filtering**

We would use firewall to block

* 1. A Port
  2. A Program
  3. A Website

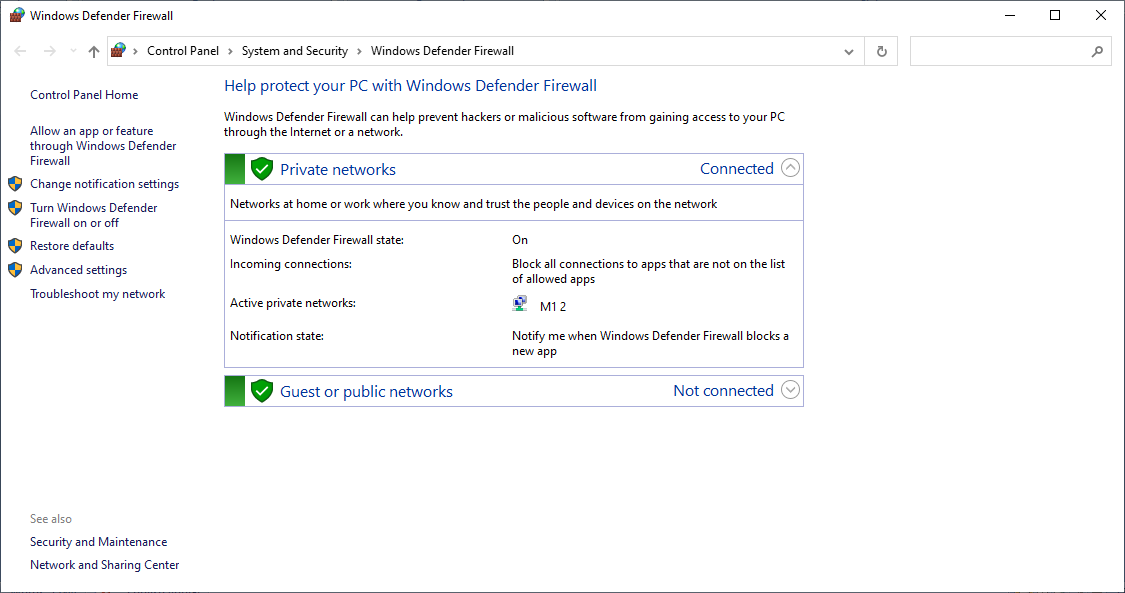
**Part 1: Blocking the HTTP and HTTPS (Port 80 and Port 443) using the Firewall**

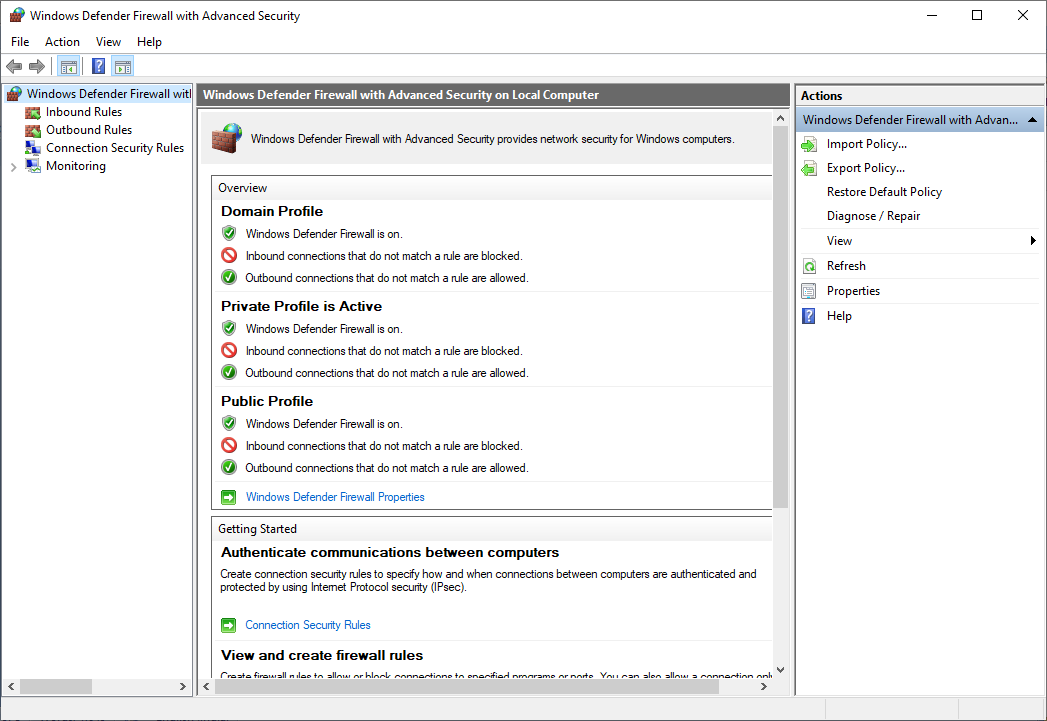
Before starting with the blocking port process, we note that the applications running at the server- end are identified with the well-known Port numbers, some of the commonly used are as follows

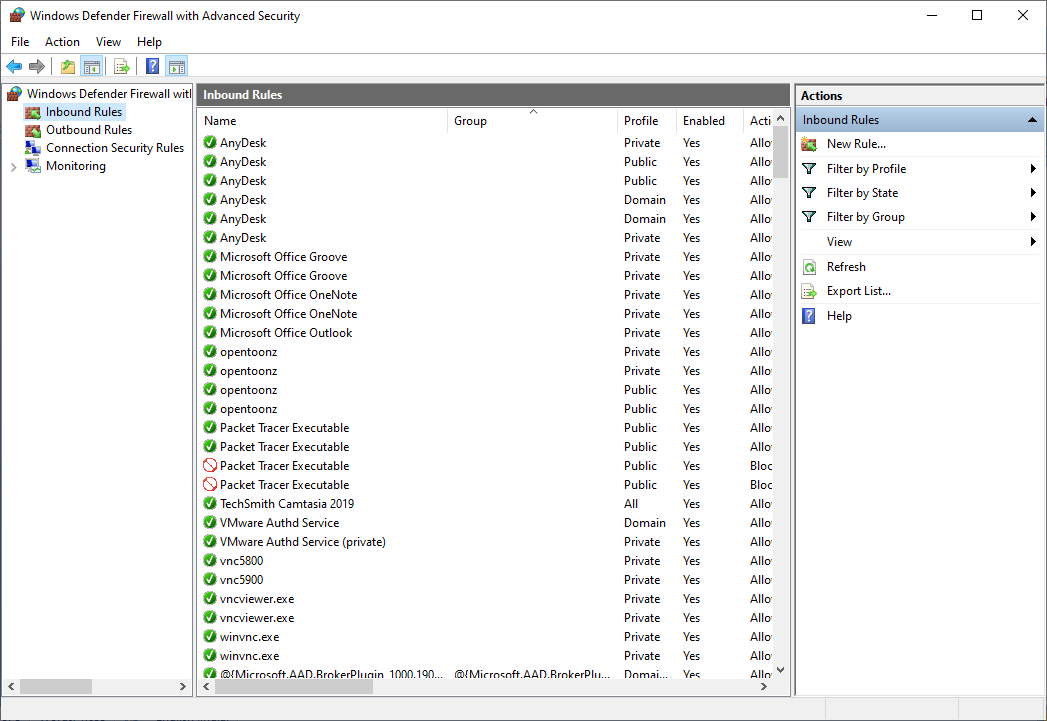


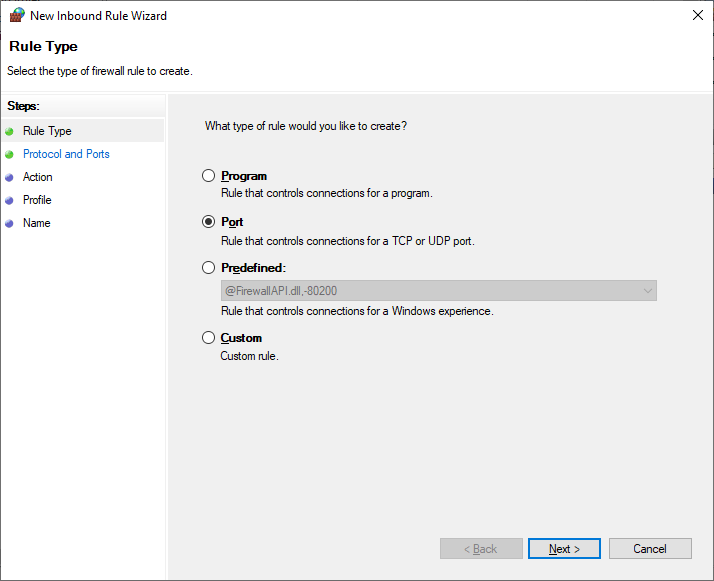
We perform the blocking Port operation as follows:

Step 1: We access any website through the browser and confirm that the HTTP/HTTPS protocols are working.

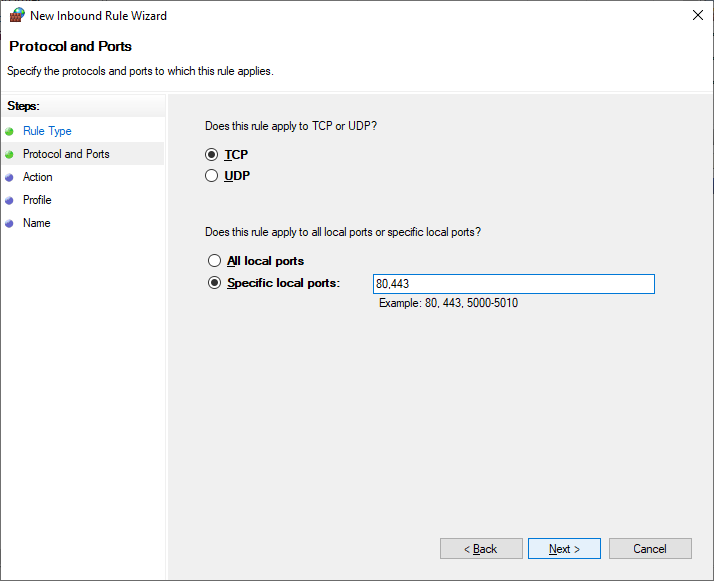
Step 2: We open ‘Windows Defender Firewall’

Next we click on ‘Advanced settings’

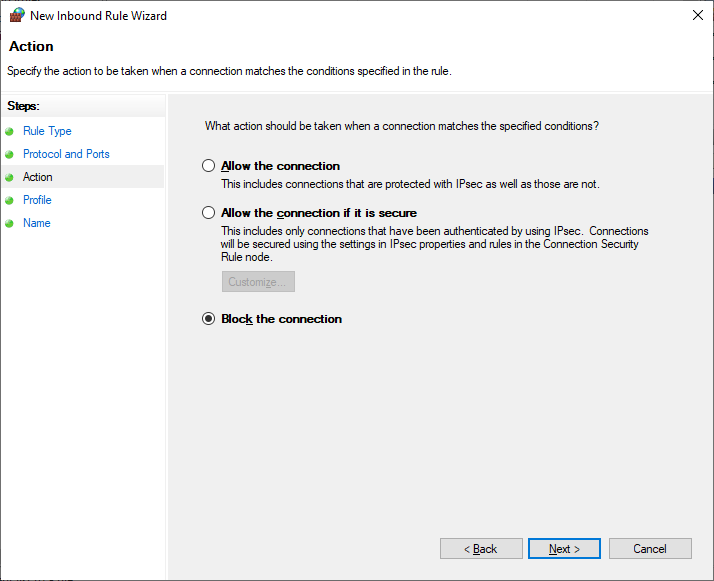
Next we click on ‘Inbound Rules’

Then click on ‘New Rule’

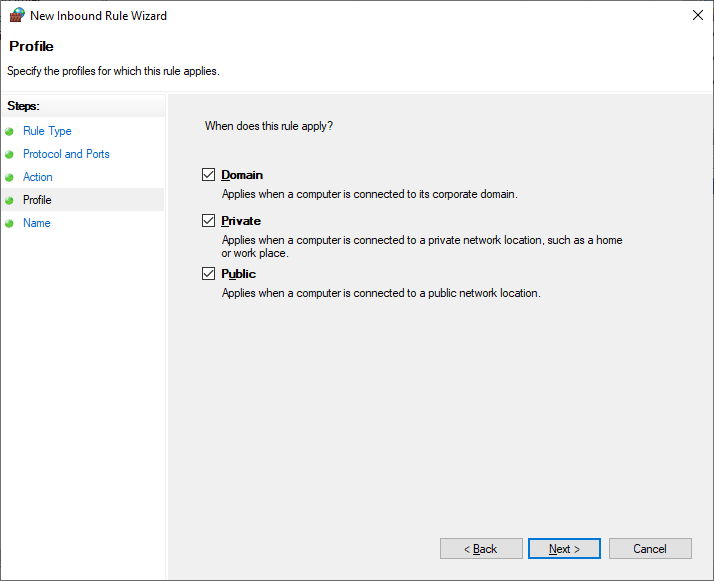
Select the radio button ‘Port’ and click ‘Next’ and enter the following

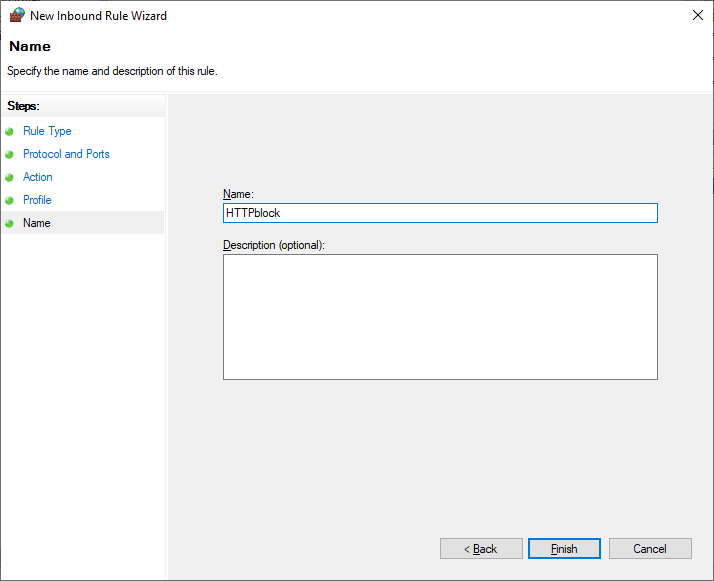


After next, we need to finalise the rule

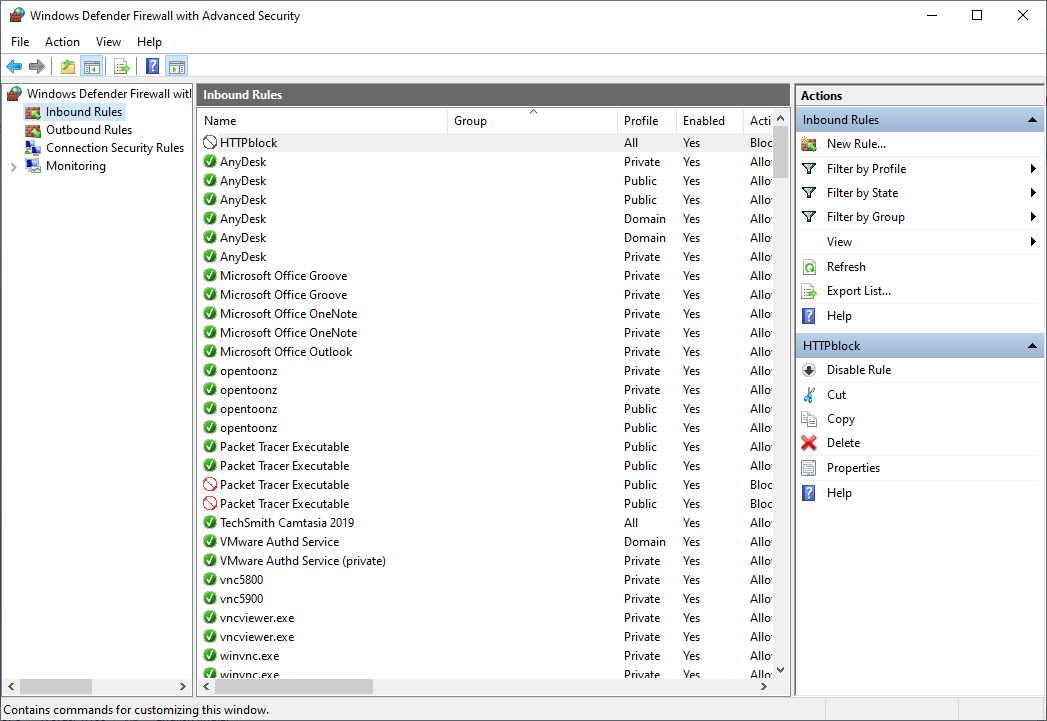


Click ‘Next’ and we get the following

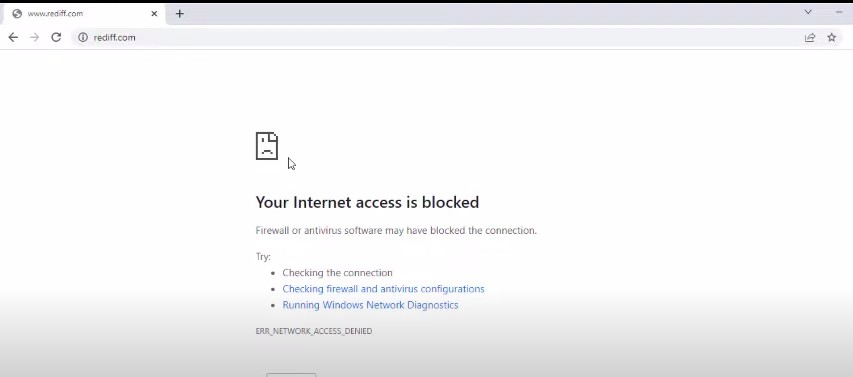


After clicking the ‘Next’ button we need to name the rule and click finish

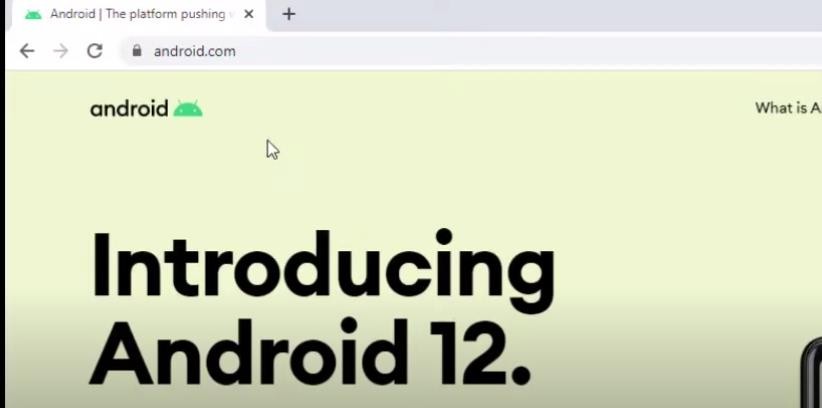
The Inbound rule is added

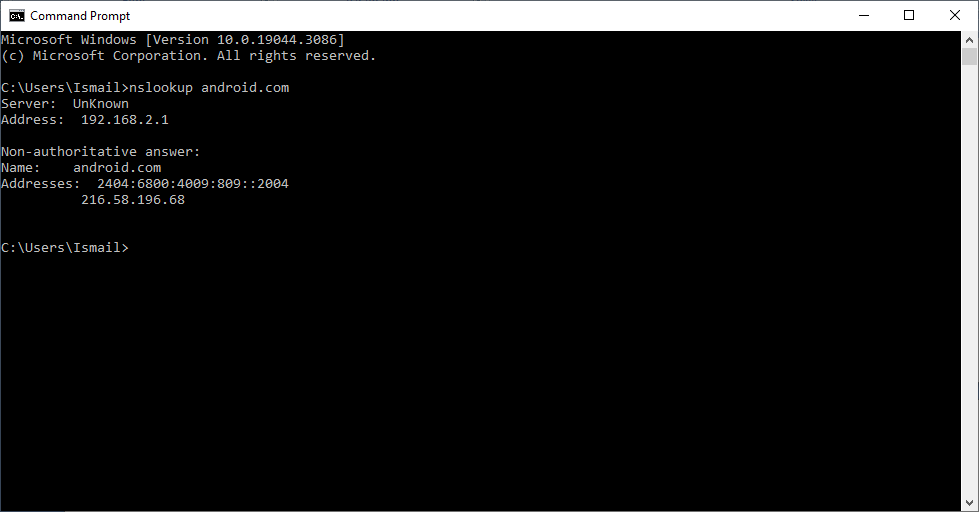


We repeat all the above steps for creating ‘Outbound Rules’, and then try to access the internet.

We see that the accessed is blocked

### Part 2: Blocking the website [www.android.com](http://www.android.com/)

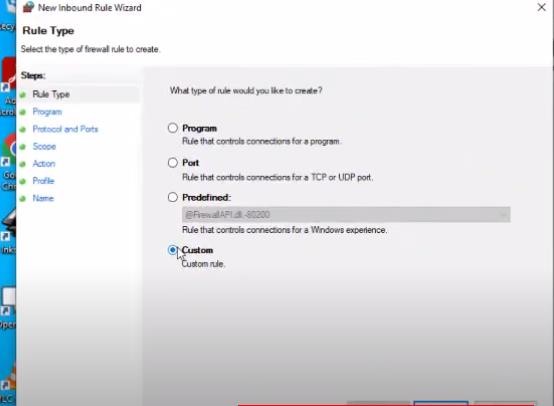
We open the browser and access the website, which is now accessible

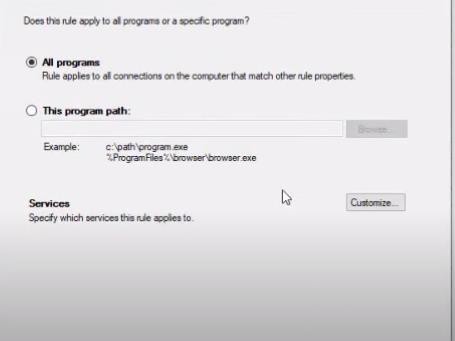
We find the IP addresses of the website using the following command

We save the IP addresses

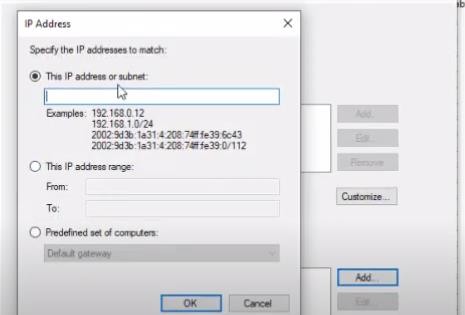
|  |  |
| --- | --- |
| IPv4 | 216.58.196.68 |
| IPv6 | 2404:6800:4009:809::2004 |

We open the windows Firewall settings and apply the Inbound Rule

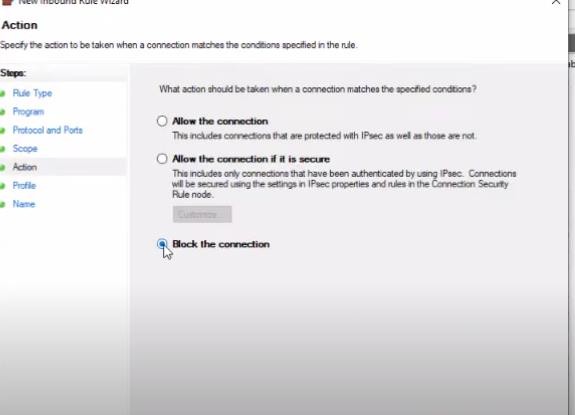




Insert the IP addresses both IPv4 and IPv6



Select Block connection



Provide a suitable name and finish



Repeat the above for Outbound Rules

Now if we try to access the website [www.android.com](http://www.android.com/) , it would be blocked

