SHRI G.P.M. DEGREE COLLEGE OF SCIENCE & COMMERCE

SHRI G.P.M. DEGREE COLLEGE OF SCIENCE & COMMERCE.

(COMMITED TO EXCELLENCE IN EDUCATION)

CERTIFICATE

This is to certify that Mr/Ms._____

a student of TY**.BSC-CS** Roll no: _____ has completed the required number of practicals in the subject of _____

as prescribed by the UNIVERSITY OF MUMBAI under my supervision during the academic year 2024-2025.

…….. ……

Prof. Incharge Principal

…… ……

External Examiner Course Co-ordinator

…….. ……

Date College Stamp

Professor Name: Prof. Sahil Wade	Class : TY.BSC-CS Semester : Sem IV (2024-2025)
Course code : USCSP502	Subject: Information & Network Security

Index

Sr. No.	Date		Page No.	Sign
1		**Theory-1 :** Implementing Substitution and Transposition Ciphers **Practical-1:** Design and implement algorithms to encrypt and decrypt messages using classical substitution and transposition techniques.	1-3	
2		**Theory-2 :** RSA Encryption and Decryption: **Practical-2**: Implement the RSA algorithm for public-key encryption and decryption, and explore its properties and security considerations.	4-6	
3		**Theory-3 :** Message Authentication Codes **Practical-3:** Implement algorithms to generate and verify message authentication codes (MACs) for ensuring data integrity and authenticity.	7-10	
4		**Theory-4** : Digital Signatures **Practical-4:**.Implement digital signature algorithms such as RSA-based signatures, and verify the integrity and authenticity of digitally signed messages.	11-13	
5		**Theory-5 :** Key Exchange using Diffie-Hellman **Practical-5:** Implement the Diffie-Hellman key exchange algorithm to securely exchange keys between two entities over an insecure network.	14-16	
6		**Theory-6 :** IP Security (IPsec) Configuration **Practical-6:** Configure IPsec on network devices to provide secure communication and protect against unauthorized access and attacks.	17-29	
7		**Theory-7 :** Web Security with SSL/TLS **Practical-7:** Configure and implement secure web communication using SSL/TLS protocols, including certificate management and secure session establishment.	30-33	
8		**Theory-8 :** Intrusion Detection System **Practical-8:** Set up and configure an intrusion detection system (IDS) to monitor network traffic and detect potential security breaches or malicious activities.	34-36	
9		**Theory-9** : Malware Analysis and Detection **Practical-9:** Analyze and identify malware samples using antivirus tools, analyze their behavior, and develop countermeasures to mitigate their impact.	37–41	
10		**Theory-10 :** Firewall Configuration and Rule-based Filtering **Practical-10:** Configure and test firewall rules to control network traffic, filter packets based on specified criteria, and protect network resources from unauthorized access.	42-49	

****

Theory-1

Implementing Substitution and Transposition Ciphers

Implementing Substitution and Transposition Ciphers

Caesar Cipher
The Caesar cipher is one of the simplest and most widely known substitution ciphers. It works by shifting each letter in the plaintext by a fixed number of positions down the alphabet.

Algorithm
- Encryption
- Choose a shift value k
- For each letter in the plaintext:
- If the letter is between 'A' and 'Z', replace it with (ascii value of letter − 65 + k) % 26+ 65
- If the letters are in between ‘a’ to ‘z’ then replace it with (ascii value of letter − 97+k) % 26 + 97
Non-alphabetic characters remain unchanged.
Decryption
Use the same shift value k
For each letter in the ciphertext:
Apply the inverse operation: (ascii value of letter − 65 − k) % 26 + ascii value of letter
For lowercase letter: (ascii value of letter − 97 − k) % 26 + ascii value of letter
For lowercase letters, adjust accordingly.

Rail Fence cipher
Description
The Rail Fence cipher is a simple transposition cipher that encrypts messages by writing them in a zigzag pattern across multiple "rails" and then reading off each line.

Algorithm
Encryption
- Choose the number of rails n.
- Write the message diagonally down and up across the rails.
- Read off each rail to create the ciphertext.
Decryption
- Reconstruct the zigzag pattern using the known number of rails.
- Read the characters in a zigzag manner to retrieve the original message.

Practical-1: Implementing Substitution and Transposition Ciphers

Aim-1: Implement Substitution Cipher or Stream Cipher or Caesar Cipher

Source code:

caesar-cipher.py

```
def caesar_cipher(x, k):
  c = ''
  for i in x:
    if i.isalpha():
      c += chr(ord(i) + k)
    else:
      c += i
  return c

p = 'Hello we are aliens!'
k = 3
e = caesar_cipher(p, k)
d = caesar_cipher(e, -k)
print("Plain text: ", p)
print("Encrypted text: ", e)
print("Decrypted text: ", d)
```

Output:

```
Plain text:  Hello we are aliens!
Encrypted text:  Khoor zh duh dolhqv!
Decrepted text:  Hello we are aliens!
```

Aim-2: Implement Block cipher or transposition cipher or rail fence cipher

Source code:

rail_fence_cipher.py

```
import math
def rail_fence_cipher(s, o):
    d, c1, c2 = '', '', ''
    if(o == 'encrypt'):
        for i in range(0, len(s)):
            if not(i & 1):
                c1+= s[i]
            else:
                c2 += s[i]
        return c1+c2
    else:
        l = len(s)
        split = math.ceil(l/2)
        s1, s2 = s[:split], s[split:]
        for i in range(0, len(s2)):
            d += s1[i]
            d+= s2[i]
        if(len(s1) > len(s2)):
            d+= s1[-1]
        return d

p = "We are Aliens!"
e = rail_fence_cipher(p, 'encrypt')
d = rail_fence_cipher(e, 'decrypt')
print(f"Plain Text: {p}\nEncrypted Text: {e}\nDecrypted Text: {d}")
```

Output:

```
Plain Text: Hello World
Encrypted Text: HloWrdel ol
Decrypted Text: Hello World
```

Learning Outcomes:

1. Successfully implemented encryption and decryption algorithms.
2. Understood the strengths and weaknesses of substitution and transposition ciphers.
3. Demonstrated the practical application of classical cryptography techniques.

Course Outcomes:

1. Apply cryptographic techniques to secure data.
2. Analyze the effectiveness of various encryption methods.
3. Implement basic cryptographic algorithms in Python.

Conclusion:

The experiment successfully demonstrated the implementation of classical ciphers, showcasing the encryption and decryption processes.

Viva Question:

1. What is cryptography?
2. What is substitution cipher technique?
3. What is encryption?
4. What is transposition cipher technique ?

For Faculty Use

Correction Parameters	Formative Assessment [ ]	Timely completion practical [ ]	Attendance Learning Attitude[ ]

Theory-2

RSA Encryption and Decryption

RSA is an acronym for Rivest-Shamir-Adleman

These are the surnames of the peoples that developed this algorithm

RSA is an asymmetric cryptographic algorithm

Asymmetric cryptographic means 2 keys are used for cryptography

One key for encryption and another one for decryption

The key for encryption is known to all users in a network and is called Public key

The key for decryption is known to only sender and receiver and is kept private from the public, so this key is also known as private key

The RSA scheme is a block cipher in which plain text and cipher text are integers between 0 to n-1 for some value n.

Steps:

1. Key generation:
   i) Select 2 large prime number 'p' and 'q'
   ii) Calculate n = p * q
   iii) Calculate n = (p-1) * (q-1)
   iv) Chose value of e such that $1 < e < n$ and gcd(n, e) = 1
   v) Calculate $d = e^{-1} \bmod(n)$

2. Encryption:
   i) $C = M^e \bmod(n)$
   Where C is cipher text,
   M is original message
   e is selected value
   n is p*q
   mod(n) means M^e / n but returns the remainder

3. Decryption:
   i) $M = C^e \bmod(n)$

Practical-2: RSA Encryption and Decryption

Aim:

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

Source code:

rsa.py

```
import rsa
public_key, private_key = rsa.newkeys(512)
message = input("Enter a message to encrypt: ")

enc_message = rsa.encrypt(message.encode(), public_key)

print("Original string: ", message)
print("Encrypted string: ", enc_message)

dec_message = rsa.decrypt(enc_message, private_key).decode()
print("Decoded string: ", dec_message)
```

Output:

```
Enter a message to encrypt: hellow
Original string:  hellow
Encrypted string:  b'''<z\x93\x8du\xfb\xdb\x10\xf3\x90\
                  x9dSHn\xe5\xb0\xef\x95g\xfc\x83\x9c
                  \xb3\xe2\x91\xbex\x90\x96\xad\xac
                  \x9e\xa8\xad\x08\x9d.\x9b\x12I\xbd
                  \xdc\x10\x83\x81\x9f\x86\xdc'\xf4
                  \x19t%\xd3\x13\x90\xfa\x14\xe0\x1a
                  \xb4\xdc\xc09'''
Decoded string:  hellow
```

Learning Outcomes:

1. Successfully implemented the RSA algorithm.
2. Understood key generation and the encryption/decryption process.
3. Explored the implications of key size on security.

Course Outcomes:

1. Apply RSA for secure communications.
2. Analyze the efficiency of public-key encryption.
3. Understand the vulnerabilities associated with RSA.

Conclusion:

The experiment successfully implemented RSA encryption and decryption, highlighting the importance of public-key cryptography in modern security.

Viva Question:

1. What is RSA algorithm?
2. What is Cryptography?
3. What is Symmetric cryptography?
4. What is Asymmetric cryptography?

For Faculty Use

Correction Parameters	Formative Assessment [ ]	Timely completion practical [ ]	Attendance Learning Attitude[ ]

Theory-3
Message Authentication Codes

MAC is a technique used to ensure the integrity and authenticity of messages exchanged between two parties. It involves the use of a secret key and a cryptographic hash function to generate a tag or code that can be appended to the message. The receiver can verify the integrity and authenticity of the message by recomputing the MAC using the same key and hash function and comparing it with the received MAC. MAC can be implement using various algorithms, we consider MD5 and SHA1

MD5 Algorithm: MD5 (Message Digest Algorithm 5) is a widely used cryptographic hash function. Although it hasbeen widely used historically, it is now considered to have vulnerabilities and is not recommendedfor security-critical applications. Nonetheless, it serves as an educational example for understandingMAC and cryptographic hash functions.

SHA1 (Secure Hash Algorithm): SHA is a family of cryptographic hash functions designed by the National Security Agency (NSA) in the United States. It provides secure one-way hashing and is widely used for various security applications. SHA1 is a 160-bit hash function. SHA-1 is an older hash function that has known vulnerabilities and is no longer considered secure for most applications. It produces a 160-bit hash value, which was considered relatively large for its time but is now considered insufficient for strong security.

SHA256: SHA-256 is a more modern and secure hash function belonging to the SHA-2 family. It produces a 256-bit hash value, making it significantly more resistant to collision attacks compared to SHA-1.

MAC Generation Process:

To generate a MAC using the MD5, SHA1 or SHA256 algorithm, follow these steps:

a) Both the sender and receiver must agree on a secret key, K, which is known only to them.

b) Concatenate the message, M, and the secret key, K: ConcatenatedData = M || K (|| denotes concatenation).

c) Apply the MD5 algorithm to the ConcatenatedData to obtain the MAC: MAC = MD5(ConcatenatedData).

d) MAC Verification Process:

To verify the integrity and authenticity of a received message using the MAC, follow these steps:

a) Receive the message, M, and the MAC, MAC.

b) Concatenate the received message, M, with the secret key, K: ConcatenatedData = M || K.

c) Apply the MD5 algorithm to the ConcatenatedData to compute the recalculated MAC: RecalculatedMAC = MD5(ConcatenatedData).

d) Compare the RecalculatedMAC with the received MAC. If they match, the message is considered authentic and intact

Practical-3: Message Authentication Codes
Aim: Implement algorithms to generate and verify message authentication codes (MACs) for ensuring data integrity and authenticity.

Aim-1: Implement MD5 hash function and verify the message with MAC
Source code:

sha256.py

```
import hashlib

sender_msg = "Hi I am sender"
secret_key = "my_secret_key"
concatenated_msg = sender_msg + secret_key

hash = hashlib.sha256(concatenated_msg.encode())
print(hash.hexdigest())

received_msg = "Hi I am sender"
received_msg = received_msg + secret_key
receiver_hash = hashlib.sha256(received_msg.encode())
print(receiver_hash.hexdigest())

if hash.hexdigest() == receiver_hash.hexdigest():
  print("Message is authentic and unchanged")
else:
  print("Message is different and not from the expected sender")
```

Output:

```
4c4d3d595463a2ae22e8abe9ea768cc9
4c4d3d595463a2ae22e8abe9ea768cc9
Message is authentic and unchanged
```

Aim-2: Implement SHA1 hash function and verify the message with MAC
Source code:

sha1.py

```
import hashlib

sender_msg = "Hi I am sender"
secret_key = "my_secret_key"
concatenated_msg = sender_msg + secret_key

hash = hashlib.sha1(concatenated_msg.encode())
print(hash.hexdigest())

received_msg = "Hi I am sender"
received_msg = received_msg + secret_key
receiver_hash = hashlib.sha1(received_msg.encode())
print(receiver_hash.hexdigest())

if hash.hexdigest() == receiver_hash.hexdigest():
  print("Message is authentic and unchanged")
else:
  print("Message is different and not from the expected sender")
```

Output:

```
0f446cc78f63659e84fbe10660eab4dc04c632e7
0f446cc78f63659e84fbe10660eab4dc04c632e7
Message is authentic and unchanged
```

Aim-3: Implement SHA256 hash function and verify the message with MAC

Source code:

main.py

```
import hashlib

sender_msg = "Hi I am sender"
secret_key = "my_secret_key"
concatenated_msg = sender_msg + secret_key

hash = hashlib.sha256(concatenated_msg.encode())
print(hash.hexdigest())

receiver_msg = "Hi I am sender"
receiver_msg = receiver_msg + secret_key
receiver_hash = hashlib.sha256(receiver_msg.encode())
print(receiver_hash.hexdigest())

if hash.hexdigest() == receiver_hash.hexdigest():
  print("Message is authentic and unchanged")
else:
  print("Message is different and not from the expected sender")
```

Output:

```
50fbf33aeaf59da2c0822a92ec90055b464702857cc3a5b5642dc7d30182bd92
50fbf33aeaf59da2c0822a92ec90055b464702857cc3a5b5642dc7d30182bd92
Message is authentic and unchanged
```

Learning Outcomes:

1. Successfully implemented MAC generation and verification.
2. Understood the importance of MACs in secure communications.
3. Demonstrated the practical use of cryptographic techniques.

Course Outcomes:

1. Apply MACs to secure data transmission.
2. Analyze the strengths of various MAC algorithms.
3. Understand the role of keys in data integrity.

Conclusion:

The experiment effectively demonstrated the implementation of MACs, showcasing their role in ensuring data integrity and authenticity.

Viva Question:

1. What is Message Authentication Code?
2. What is hash function?
3. Why do we use hash function?
4. Name any 2 hash functions?

For Faculty Use

Correction Parameters	Formative Assessment [ ]	Timely completion of practical [ ]	Attendance Learning Attitude[ ]

Theory-4
Digital Signatures

Digital signatures are introduced to verify that the message is sent by an authentic sender or not.
It plays a vital role in online transactions and e-commerce.
It is based on asymmetric key cryptography
It uses public key for decryption
It uses private key for encryption
It is used for message authentication, non-repudiation and message integrity
It is not used for confidentiality
It is introduced so that a person can not deny having sent a message or performing an action because it serves as a proof that he or she really has performed that action.

Process of Digital Signature:
Sender 'A' has a message, and wants to send it to 'B'
'A' uses hashing function such as SHA256 to create a hash value of the message
'A' generates public key and private using RSA algorithm
'A' uses the private key to encrypt the message and this creates the signature
This signature is nothing but encrypted message with the private key of 'A'
'A' sends the plain message and the digital signature to 'B'
'B' decrypt the signature using the public key of 'A' and gets the hash value of it
'B' then uses same hashing function as 'A' to create a hash value of the received message
'B" then compare the hash values of signature and message
If both are same then the message is authentic else not

Practical-4: Digital Signatures
Aim: Implement digital signature algorithms such as RSA-based signatures, and verify the integrity and authenticity of digitally signed messages
Source code:

Digital-Signature.py

```
#!pip install pycryptodome


from Crypto.PublicKey import RSA
from Crypto.Hash import SHA256
import binascii
from Crypto.Signature import pkcs1_15

message = b'I met aliens!'
key = RSA.generate(2048)
private_key = key.export_key()
public_key = key.publickey().export_key()
h = SHA256.new(message)
signature = pkcs1_15.new(RSA.import_key(private_key)).sign(h)

# Verify
h = SHA256.new(message)
try:
    pkcs1_15.new(RSA.import_key(public_key)).verify(h, signature)
    print("Signature is valid.")
except (ValueError, TypeError):
    print("Signature is invalid.")
```

Output:

```
Signature is valid.
```

Learning Outcomes:

1. Successfully implemented digital signature generation and verification.
2. Understood the importance of digital signatures in security protocols.
3. Demonstrated practical applications of digital signatures.

Course Outcomes:

1. Apply digital signatures to secure messages.
2. Analyze the legal and technical implications of digital signatures.
3. Understand how digital signatures differ from traditional signatures.

Conclusion:

The experiment successfully demonstrated the implementation of digital signatures, highlighting their importance in modern security practices.

Viva Question:

1. What is Digital Signature?
2. Why do we need digital signatures?
3. What is hashing?
4. What is cipher text?

For Faculty Use

Correction Parameters	Formative Assessment [ ]	Timely completion of practical [ ]	Attendance Learning Attitude[ ]

Theory-5
Key Exchange using Diffie-Hellman

The Diffie-Hellman algorithm is a widely used asymmetric key exchange algorithm. It enables two parties to securely establish a shared secret key over an insecure communication channel.

High-level working explanation of the Diffie-Hellman algorithm:
a) Select a large prime number, p, and a primitive root modulo p, g. These values are publicly known.
b) Each party, Alice and Bob, generates a private key, a and b, respectively.
c) Both Alice and Bob calculate their public keys:
d) Alice: $A = g^a \bmod p$
e) Bob: $B = g^b \bmod p$
f) Alice and Bob exchange their public keys over the insecure channel.

Practical 5: Diffie-Hellman key exchange algorithm

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

Source code:

digital-signature.py

```
import random
p = 23 # mod
g = 5 # base

def diffie_hellman():
  a = random.randint(1, 10) # client
  b = random.randint(1, 10) # server
  A = pow(g, a) % p
  B = pow(g, b) % p
  s_a = pow(B, a) % p  # shared A →  secret value
  s_b = pow(A, b) % p # shared B → same secret value
  if s_a == s_b:
    print("Shared secret:", s_a)
  else:
    print("Key exchange failed")

diffie_hellman()
```

Output:

```
Shared secret: 8
```

Learning Outcomes:

1. Successfully implemented the Diffie-Hellman key exchange.
2. Understood the process of deriving a shared key.
3. Explored the vulnerabilities associated with key exchange.

Course Outcomes:

1. Apply the Diffie-Hellman algorithm in secure communications.
2. Analyze the security of key exchange methods.
3. Understand the implications of public and private keys in Diffie-Hellman.

Conclusion:

The experiment effectively demonstrated the Diffie-Hellman key exchange algorithm, showcasing its importance in secure communication protocols.

Viva Question:

1. Why do we use diffie-hellman algorithm?
2. What is asymmetric cryptography?
3. What is public key?
4. What is private key?

For Faculty Use

Correction Parameters	Formative Assessment [ ]	Timely completion practical [ ]	Attendance Learning Attitude[ ]

Theory-6
IPSEC

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.
2. 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.
3. Components of IPSec:
    - IPSec comprises multiple protocols and elements, including Authentication Header (AH), Encapsulating Security Payload (ESP), Security Associations (SAs), and key management protocols.
4. 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.
5. 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.
6. 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.
7. 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.
8. Routing and Secure Communication:
    - Once established, the VPN tunnel allows secure data routing between the endpoints or networks.
    - Applications and services on either side can communicate securely, even over untrusted networks like the Internet.
9. 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.
10. 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).

11. Security Policies:
    - Organizations define security policies that determine when and how IPSec should be applied to protect specific types of traffic or communication.
12. Interoperability:
    - IPSec is widely adopted, ensuring interoperability between different vendors' equipment and making it a versatile choice for securing networks and data.

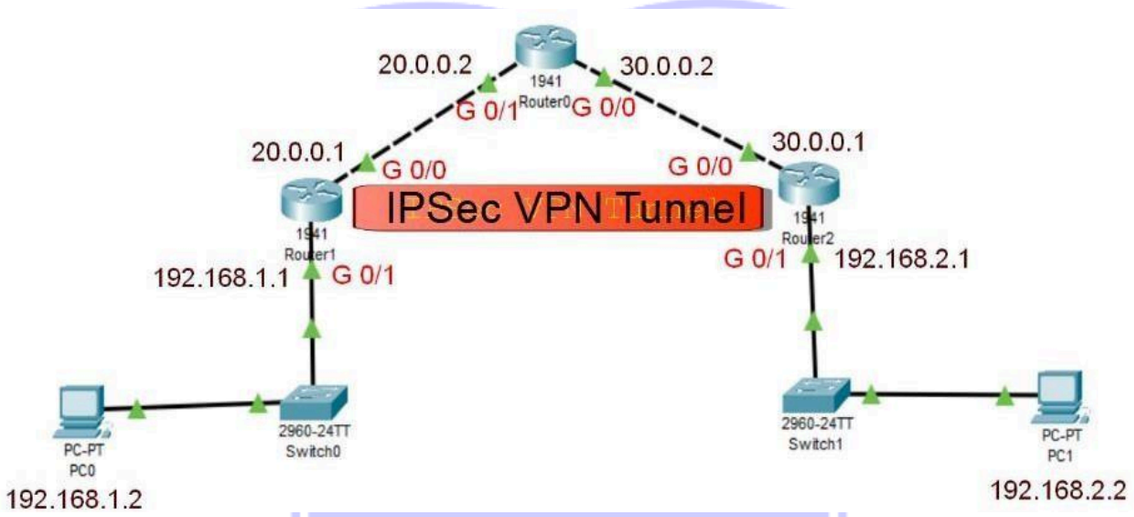
Understanding the principles of IPSec and IPSec VPN tunnels is essential for designing, deploying, and managing secure communication in various network environments, ensuring data remains confidential, unaltered, and protected from unauthorized access.

\

Practical-6: IP Security (IPsec) Configuration
Aim: Configure IPsec on network devices to provide secure communication and protect against unauthorized access and attacks.

Topology:

We use the following topology for the present case



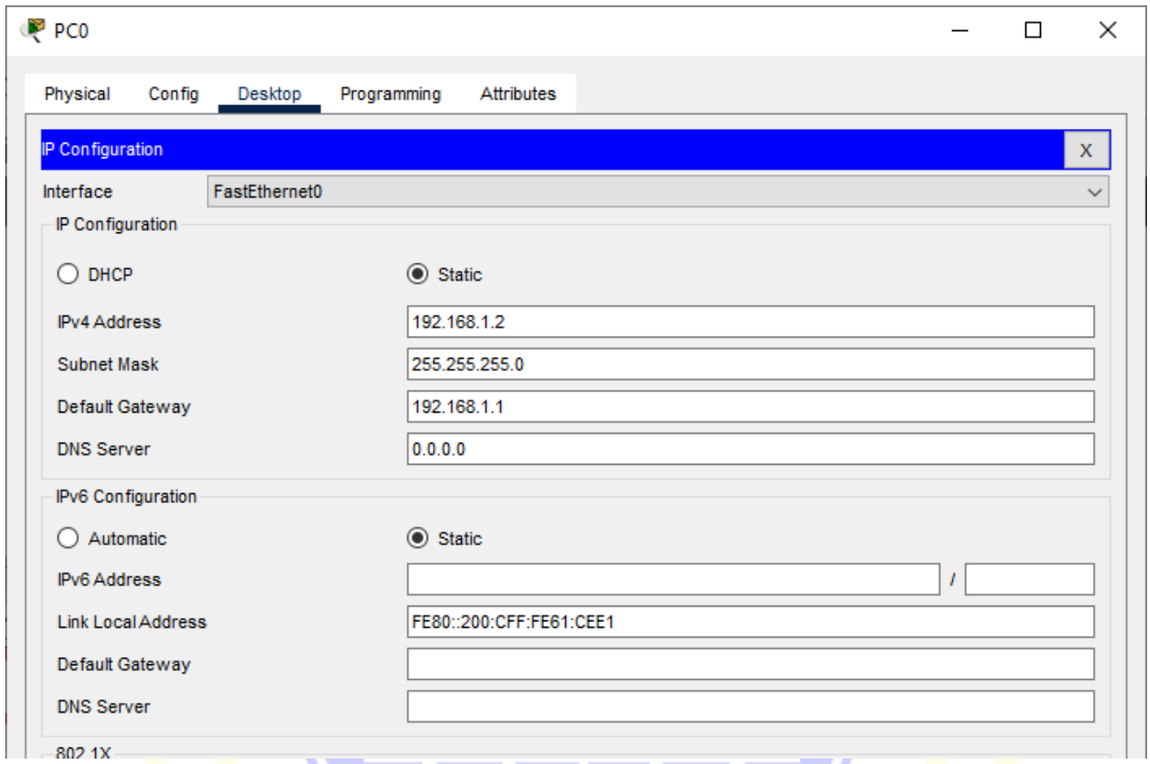
ISAKMP Policy Parameters

Parameters	Parameter Options and Defaults	R1	R2
Key Distribution Method	Manual or ISAKMP	ISAKMP	ISAKMP
Encryption Algorithm	DES. 3DES or AES	AES-256	AES-256
Hash Algorithm	MD5 or SHA-1	SHA-1	SHA-1
Authentication Method	Pre-shared Key or RSA	Pre-shared	Pre-shared
Key Exchange	DH Group 1, 2 or 5	Group 5	Group 5
ISE SA Lifetime	86400 seconds or less	86400	86400
ISAKMP Key	User defined	ismile	ismile

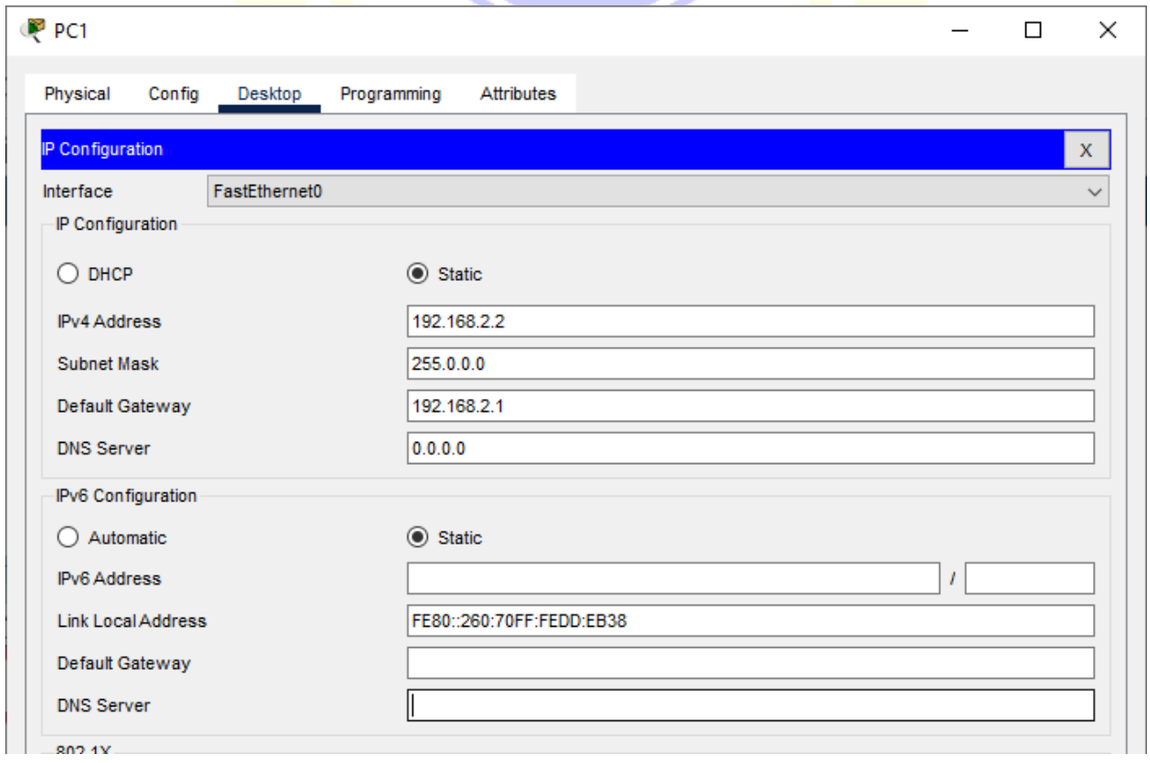
IPSec Policy Parameters

Parameters	R1	R2
Transform Set Name	VPN-SET	VPN-SET
ESP Transform Encryption	esp-aes	esp-aes
ESP Transform Authentication	esp-sha-hmac	esp-sha-hmac
Peer IP Address	30.0.0.1	20.0.0.1
Traffic to be Encrypted	R1->R2	R2->R1
Crypto Map Name	IPSEC-MAP	IPSEC-MAP
SA Establishment	ipsec-isakmp	ipsec-isakmp

Configuring PC0:

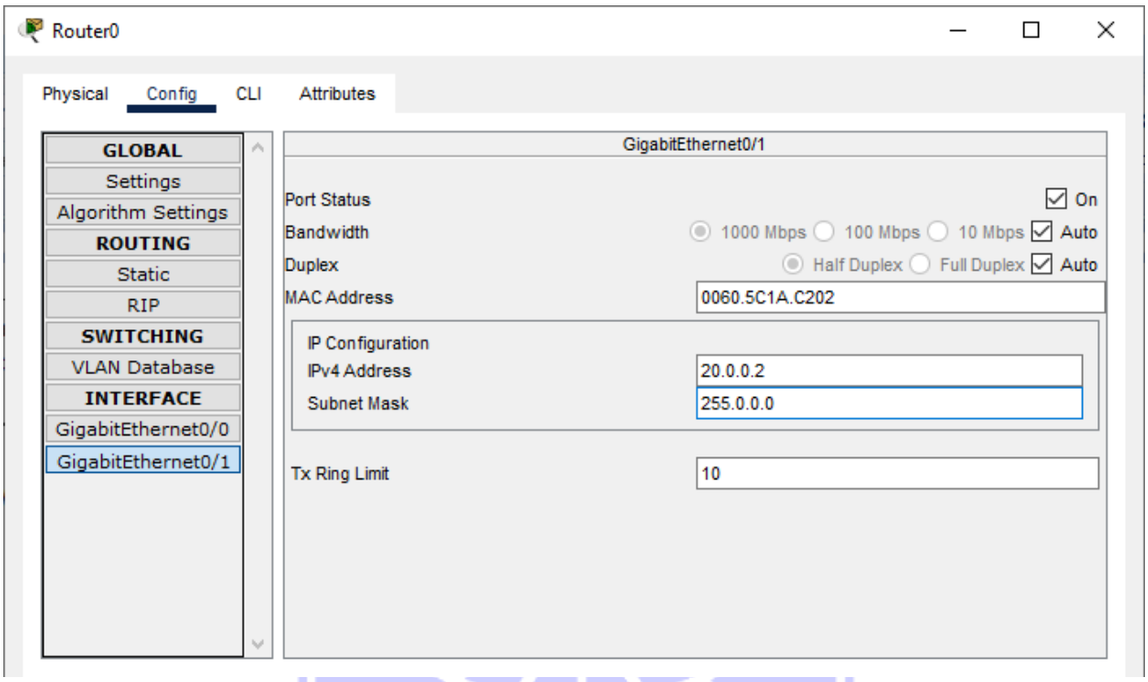


Configuring PC1:

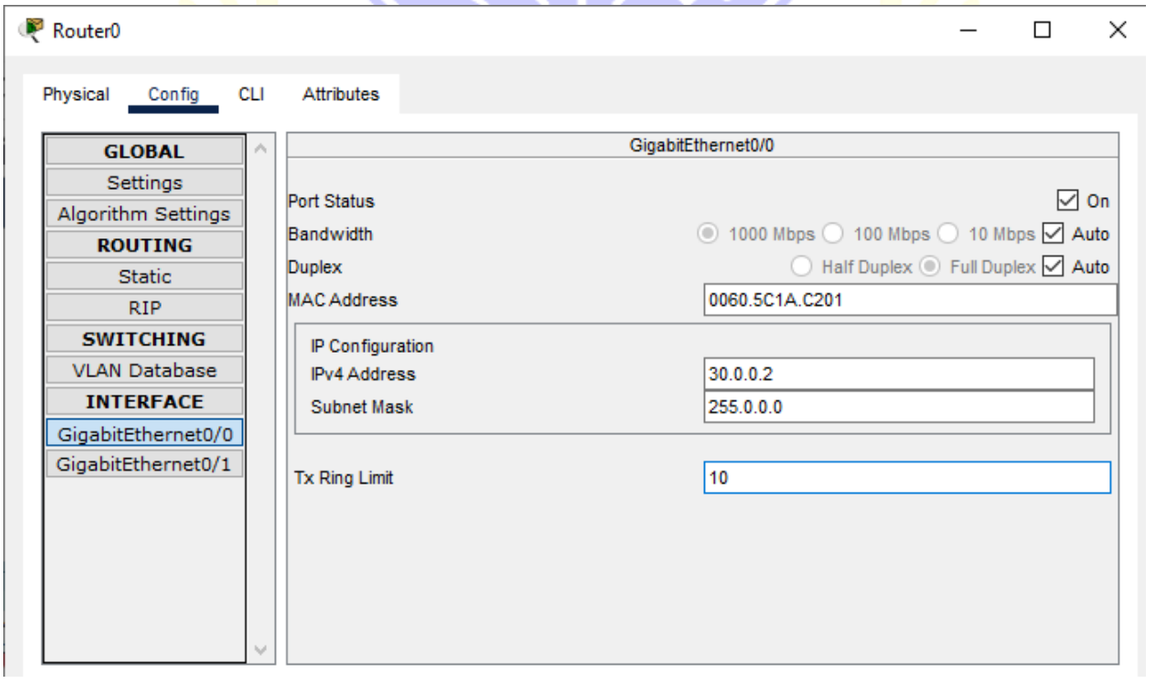


Configuring Router0:

Interface GigabitEthernet0/1:

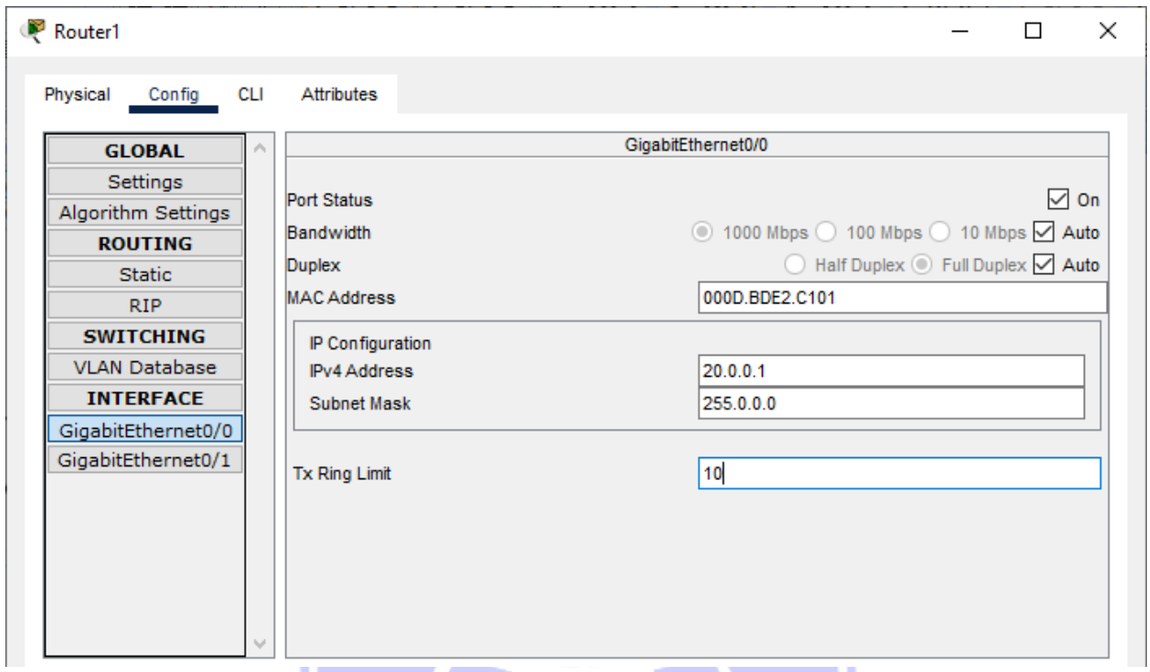


Interface GigabitEthernet0/0:

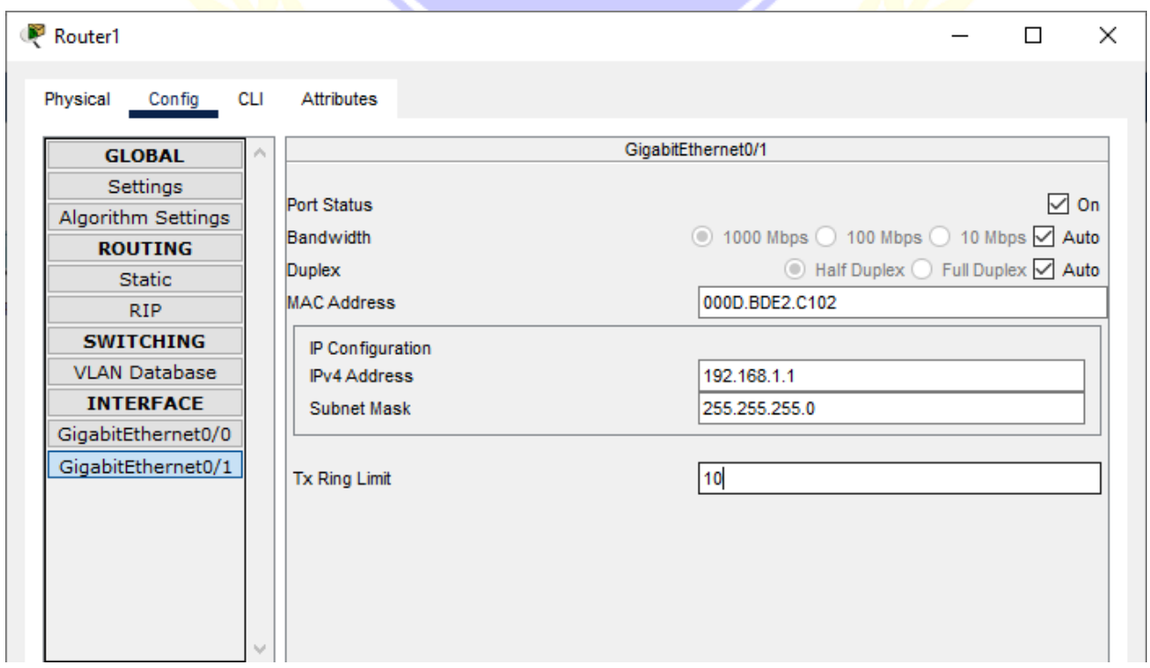


Configuring Router1:

Interface GigabitEthernet0/0:

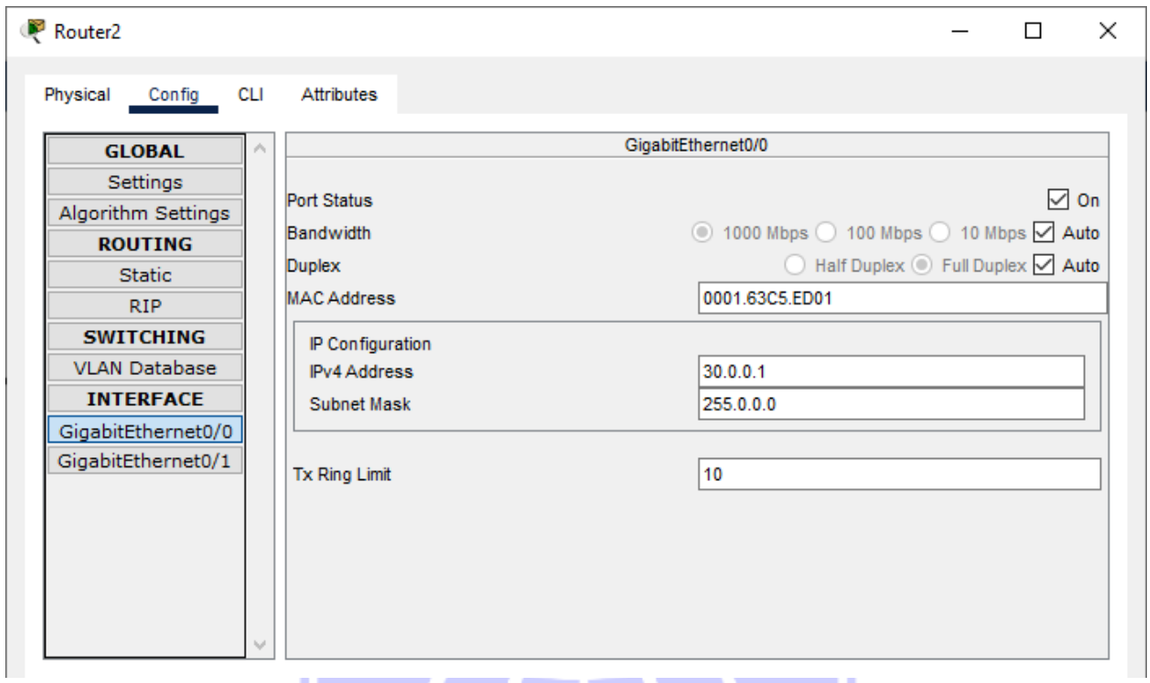


Interface GigabitEthernet0/1:

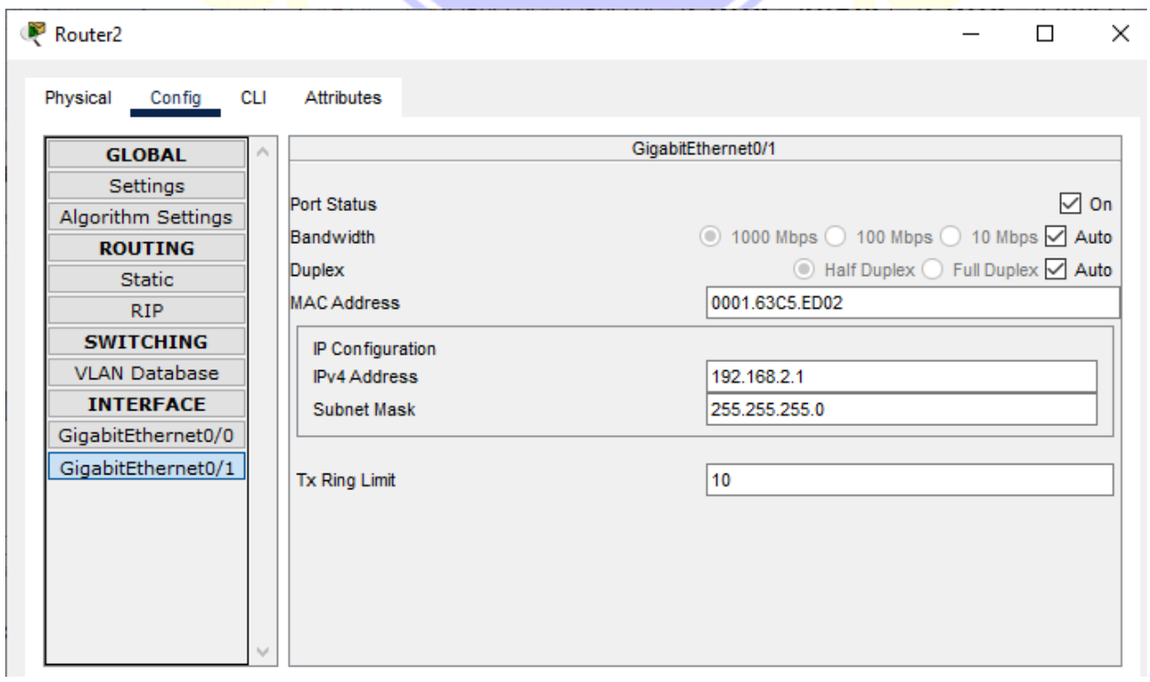


Configuring Router2:

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

```
Device#    PID                     SN
------------------------------------------------
*0         CISCO1941/K9            FTX1524N826-


Technology Package License Information for Module:'c1900'

-----------------------------------------------------------------
Technology    Technology-package          Technology-package
              Current       Type          Next reboot
-----------------------------------------------------------------
ipbase        ipbasek9      Permanent     ipbasek9
security      None          None          None
data          None          None          None

Configuration register is 0x2102
```

(We see that the security feature is not enabled, so we need to enable the security package)

R1#configure terminal

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

R1(config)#exit

R1#copy run startup-config

R1#reload

R1>enable

R1#show version

```


Technology Package License Information for Module:'c1900'

-----------------------------------------------------------------
Technology    Technology-package          Technology-package
              Current       Type          Next reboot
-----------------------------------------------------------------
ipbase        ipbasek9      Permanent     ipbasek9
security      securityk9    Evaluation    securityk9
data          disable       None          None

Configuration register is 0x2102
```

(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

```
Device#   PID                     SN
------------------------------------------------
*0        CISCO1941/K9            FTX1524N826-


Technology Package License Information for Module:'c1900'

-----------------------------------------------------------------
Technology    Technology-package          Technology-package
              Current       Type          Next reboot
-----------------------------------------------------------------
ipbase        ipbasek9      Permanent     ipbasek9
security      None          None          None
data          None          None          None

Configuration register is 0x2102
```

(We see that the security feature is not enabled, so we need to enable the security package)

R2#configure terminal

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

R2(config)#exit

R2#copy run startup-config

R2#reload

R2>enable R2#show version

```
Technology Package License Information for Module:'c1900'

-----------------------------------------------------------------
Technology    Technology-package          Technology-package
              Current       Type          Next reboot
-----------------------------------------------------------------
ipbase        ipbasek9      Permanent     ipbasek9
security      securityk9    Evaluation    securityk9
data          disable       None          None

Configuration register is 0x2102
```

(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 10R1(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 saad 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 saad 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

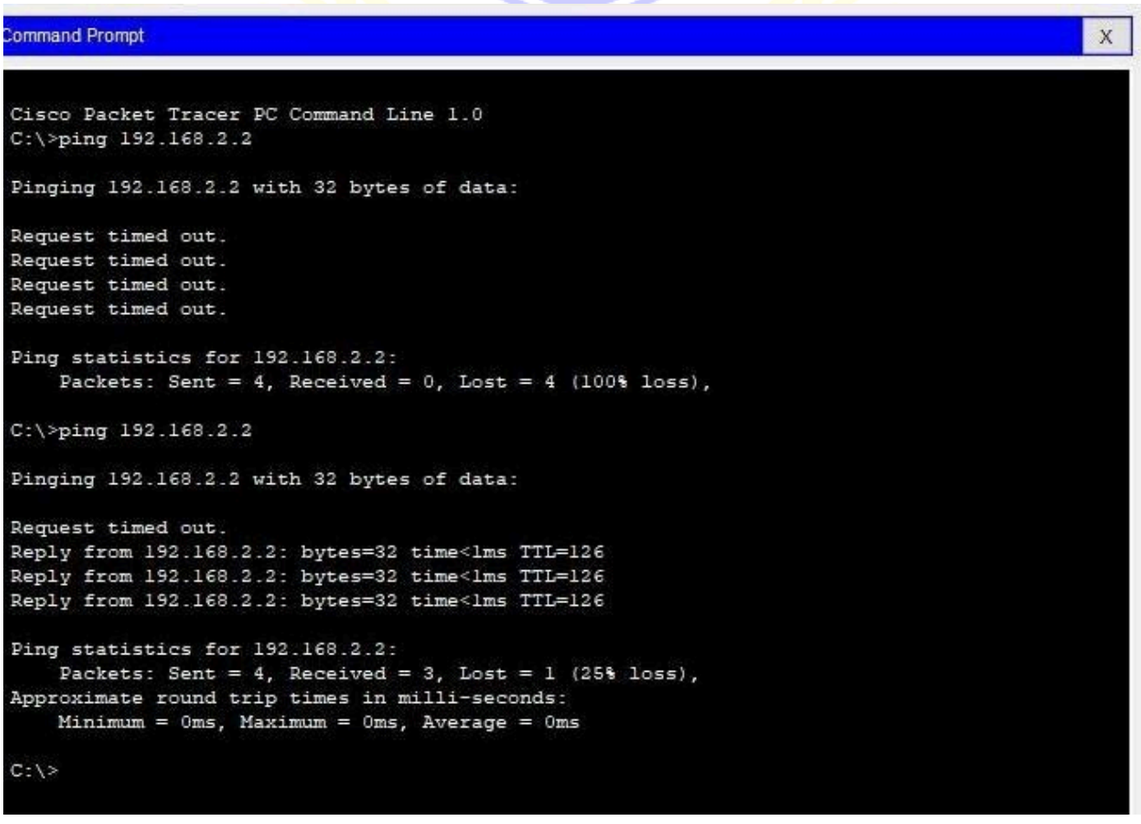
R2(config-crypto-map)#exit

R2(config)#interface g0/0

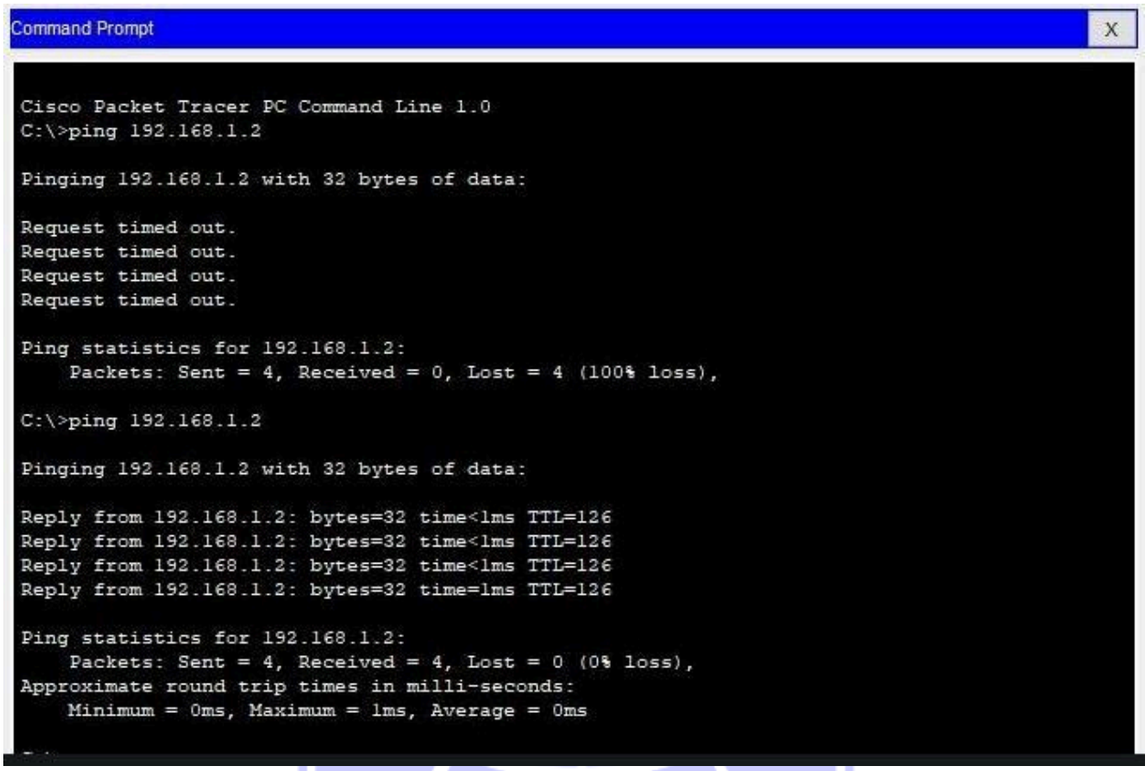
R2(config-if)#crypto map IPSEC-MAP

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



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



Learning Outcomes:

1. Successfully configured IPsec on network devices.
2. Understood the components and modes of IPsec.
3. Demonstrated the effectiveness of IPsec in securing network communications.

Course Outcomes:

1. Apply IPsec to secure network traffic.
2. Analyze the advantages of using IPsec in a network.
3. Understand the challenges of IPsec implementation.

Conclusion:

The experiment effectively demonstrated the configuration of IPsec, highlighting its role in securing network communications.

Viva Question:

1. What is the IPSEC?
2. What is VPN?
3. What is the full-form of VPN?
4. What is data encryption?

For Faculty Use

Correction Parameters	Formative Assessment [ ]	Timely completion practical [ ]	Attendance Learning Attitude[ ]

Theory-7
Web Security with SSL/TLS

Overview of SSL/TLS in Web Security:
SSL (Secure Sockets Layer) and TLS (Transport Layer Security) are cryptographic protocols designed to secure communications over a computer network. They play a crucial role in protecting sensitive information transmitted between clients and servers, such as credit card details and personal data.

Key Concepts of SSL/TLS:

1. Encryption
SSL/TLS employs encryption to safeguard data during transmission. This ensures that even if data is intercepted, it remains unreadable to unauthorized parties. The encryption process utilizes two types of keys:
- Asymmetric Keys: A pair consisting of a public key (shared with clients) and a private key (kept secret by the server) is used to establish a secure connection.
- Symmetric Session Keys: Once the connection is established, both parties use a symmetric key for encrypting and decrypting data during the session, which enhances performance and security.

2. Authentication
Authentication ensures that both parties in the communication are who they claim to be. This is achieved through a handshake process where the server presents its SSL certificate, which includes its public key. The client verifies this certificate against trusted root certificates to confirm the server's identity.

3. Data Integrity
Data integrity is maintained through cryptographic hash functions that verify that data has not been altered during transmission. This ensures that any tampering attempts can be detected, safeguarding the authenticity of the exchanged information.

The SSL/TLS Handshake Process

The handshake process is essential for establishing a secure connection. It involves several steps:
1. Client Hello: The client initiates the handshake by sending a request to the server.
2. Server Hello: The server responds with its SSL certificate and selected encryption methods.
3. Certificate Verification: The client verifies the server's certificate to ensure authenticity.
4. Key Exchange: The client generates a symmetric session key, encrypts it with the server's public key, and sends it to the server.
5. Session Establishment: Both parties now use the symmetric session key for secure communication.

Importance of SSL/TLS in Modern Security

With over 80% of internet traffic being encrypted, SSL/TLS has become vital for securing online communications, including web browsing, email, and e-commerce transactions. It helps prevent various cyber threats such as data interception, man-in-the-middle attacks, and data tampering. Additionally, the latest version, TLS 1.3, improves security by eliminating outdated algorithms and speeding up the handshake process.

In summary, SSL/TLS protocols are essential for ensuring secure web communications by providing encryption, authentication, and data integrity, thereby protecting users' sensitive information from unauthorized access and cyber threats.

Practical-7: Web Security with SSL/TLS

Aim:

Configure and implement secure web communication using SSL/TLS protocols

Source code:

```
#Server Code:

import socket

# Create a socket object
s = socket.socket()

# Get local machine name
host = socket.gethostname()

# Reserve a port
port = 12345

# Bind to the port
s.bind((host, port))

# Wait for one connection
s.listen(1)

print("Waiting for any incoming connections ...")

conn, addr = s.accept()

print(f"Received connection from {addr[0]}")

# Receive data from client
data = conn.recv(1024).decode()

print(f"Received data: {data}")


# Send data to client
data = "Thanks for connecting"
conn.send(data.encode())


# Close the connection
conn.close()
```

```python
#Client Code:

import socket

# Create a socket object
s = socket.socket()

# Get local machine name
host = socket.gethostname()

# Reserve a port
port = 12345

# Connect to server
s.connect((host, port))

# Send data to server
data = "Hello from Client!"
s.send(data.encode())

# Receive data from server
data = s.recv(1024).decode()

print(f"Received data: {data}")

# Close the connection
s.close()
```

Output:

```
~ $ python s3.py
Waiting for any incoming connections ...
```

```
~ $ python s3.py
Waiting for any incoming connections ...
Received connection from 127.0.0.1
Received data: Hello from Client!
```

```
~ $ python c3.py
Received data: Thanks for connecting
~ $
```

Learning Outcomes:

1. Successfully configured SSL/TLS on a web server.
2. Understood the process of certificate generation and management.
3. Demonstrated secure web communication.

Course Outcomes:

1. Apply SSL/TLS to secure web applications.
2. Analyze the benefits of using secure protocols in web security.
3. Understand the role of certificates in secure communication.

Conclusion:

The experiment effectively demonstrated the configuration of SSL/TLS, showcasing its importance in securing web communications.

Viva Question:

1. What is SSL?
2. What is socket?
3. What is authentication?
4. What is encryption?

For Faculty Use

Correction Parameters	Formative Assessment [ ]	Timely completion practical [ ]	Attendance Learning Attitude[ ]

Theory-8

Intrusion Detection System

SSL (Secure Sockets Layer) and TLS (Transport Layer Security) are protocols that provide secure communication over a computer network. They ensure data privacy and integrity between communicating applications

The socket module can be used to create server applications that can handle multiple client connections efficiently.

This module allows programs to create sockets, bind them to specific ports and listen for incoming connections. It also provides methods for sending and receiving data over the network.

The socket module in Python provides a set of methods and classes for working with TCP/IP sockets.

Socket():

Socket() is used to create a socket object that can be used to establish a connection between two machines. It takes the address family, socket type, and protocol type as arguments and returns a socket object.

Bind():

Bind() is used to bind a socket to a specific network address. It takes the socket object and a tuple containing an IP address and port number as arguments and binds the socket to that address.

Accept():

Accept() is used to accept incoming connections. It takes a socket object as an argument and waits for an incoming connection. When a connection is accepted, it returns a new socket object that can be used to communicate with the connected client.

Encode():

Encode() is used to convert a string or a sequence of bytes into a unicode string. It takes a string as an argument and returns a unicode string.

Decode():

Decode() is used to convert a unicode string into a string or a sequence of bytes. It takes a unicode string as an argument and returns a string.

Send():

Send() is used to send data over a socket. It takes a socket object and a string or a sequence of bytes as arguments and sends the data over the socket.

Close():

Close() is used to close a socket connection. It takes a socket object as an argument and closes the connection.

Practical-8: Intrusion Detection System

Aim: Set up and configure an intrusion detection system (IDS) to monitor network traffic and detect potential security breaches or malicious activities.

```
from scapy.all import *
import logging
# Set up logging to capture alerts in a log file
logging.basicConfig(filename="ids_alerts.log", level=logging.INFO, format="%(asctime)s - %(message)s")
# Function to process sniffed packets
def detect_malicious_activity(packet):
if packet.haslayer(IP):
ip_src = packet[IP].src
ip_dst = packet[IP].dst
# Check for suspicious activity
if packet.haslayer(TCP):
# Suspicious ports (example: commonly used by malware)
if packet[TCP].dport == 6667 or packet[TCP].sport == 6667: # IRC port, often used in botnets
logging.info(f"Suspicious activity detected: {ip_src} → {ip_dst} on port 6667 (Possible IRC traffic)")
print(f"ALERT: {ip_src} → {ip_dst} on port 6667 (Possible IRC traffic)")
# Detect port scanning (many SYN packets without ACK)
if packet[TCP].flags == "S":
logging.info(f"Port scan detected from {ip_src} to {ip_dst}")
print(f"ALERT: Port scan detected from {ip_src} to {ip_dst}")
# Check for unusual packet size (could indicate DDoS or other attacks)
if len(packet) > 1500: # Typical MTU is 1500
logging.info(f"Suspicious large packet detected: {ip_src} → {ip_dst} (size: {len(packet)})")
print(f"ALERT: Large packet detected: {ip_src} → {ip_dst} (size: {len(packet)})")
# Sniff network traffic
def start_sniffing():
print("Starting IDS to monitor network traffic...")
sniff(filter="ip", prn=detect_malicious_activity, store=0)
if __name__ == "__main__":
start_sniffing()
```

Output: If suspicious activity is detected, the terminal will print out alerts like:

```
Starting IDS to monitor network traffic...
ALERT: 192.168.1.5 -> 192.168.1.10 on port 6667 (Possible IRC traffic)
ALERT: Port scan detected from 192.168.1.6 to 192.168.1.10
ALERT: Large packet detected: 192.168.1.5 -> 192.168.1.10 (size: 2000)
```

Learning Outcomes:

1. Successfully configured an IDS for network monitoring.
2. Understood the types of intrusions and detection me

Course Outcomes:

1. Apply IDS techniques to enhance network security.
2. Analyze the data collected by the IDS for incident response.
3. Understand the challenges of intrusion detection.

Conclusion:

The experiment effectively demonstrated the setup and configuration of an IDS, highlighting its importance in proactive security measures.

Viva Question:

1. What is SSL/TLS??
2. How do SSL and TLS differ?
3. Why is certificate management important?
4. What are the steps to establish a secure session?

For Faculty Use

Correction Parameters	Formative Assessment [ ]	Timely completion practical [ ]	Attendance Learning Attitude[ ]

Theory-9
Malware Analysis and Detection

Malware, short for "malicious software," refers to a broad category of software programs or code specifically designed to infiltrate, damage, disrupt, or gain unauthorized access to computer systems, networks, and digital devices. Malware is created with malicious intent, often to steal sensitive information, gain control over systems, extort money, or cause harm to users or organizations.

There are various types of malware, each with distinct characteristics and purposes. Some common types of malware include:

1. Viruses: Viruses are malicious programs that attach themselves to legitimate files or programs and spread by infecting other files. When infected files are executed, the virus replicates and spreads further, potentially causing damage to data and systems.

2. Worms: Worms are standalone programs that replicate and spread across computer networks without needing to attach themselves to other files. They often exploit security vulnerabilities to self-propagate and can cause network congestion and data loss.

3. Trojans: Trojans are deceptive programs that disguise themselves as legitimate software, tricking users into installing them. Once installed, Trojans can perform a variety of malicious activities, suchas stealing sensitive information, opening backdoors, or launching attacks.

4. Ransomware: Ransomware encrypts a user's files or entire system and demands a ransom payment in exchange for providing the decryption key. It can lead to data loss and significant disruption if not properly managed.

5. Spyware: Spyware is designed to secretly gather information from a user's device, such as browsing habits, passwords, and personal data. This information is then sent to a remote attacker or entity.

6. Adware: Adware is software that displays unwanted advertisements, often in the form of pop-ups or banners, on a user's device. While not as malicious as other types of malware, it can be disruptive and invasive.

7. Keyloggers: Keyloggers record a user's keystrokes, allowing attackers to capture sensitive information like passwords, credit card numbers, and other confidential data.

8. Botnets: Botnets are networks of compromised computers or devices, known as "bots" or "zombies," controlled by a central attacker. Botnets are often used to launch coordinated attacks, distribute spam, or carry out other malicious activities.

9. Rootkits: Rootkits are designed to hide malicious activities from the user and security software. They can modify or replace core system files to gain unauthorized access and control over a system.

10. Backdoors: Backdoors provide unauthorized access to a compromised system. They can be used by attackers to maintain control over a system, often allowing them to return even after the initial breach is resolved.

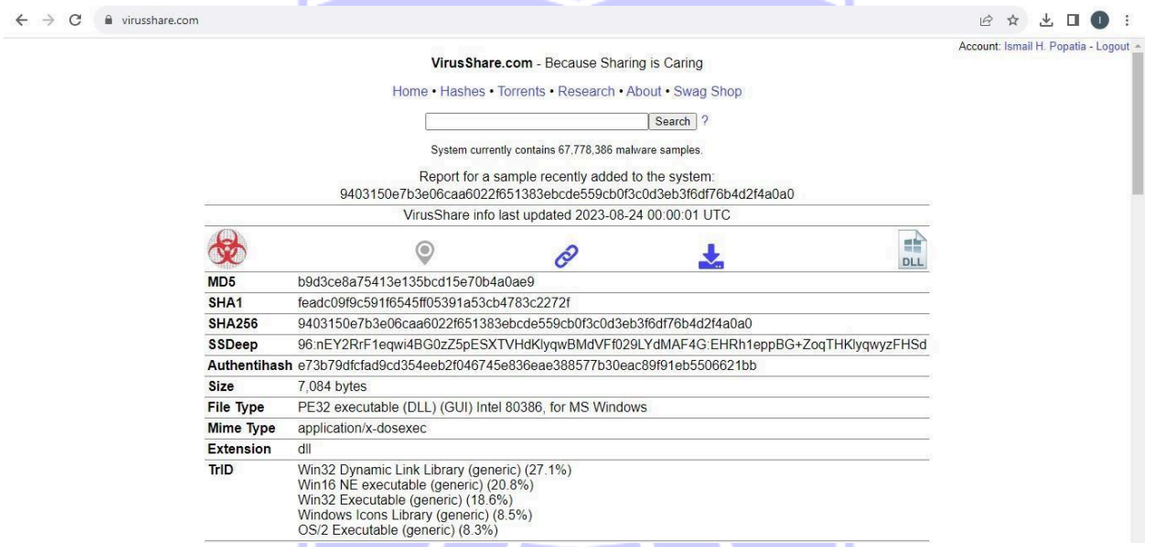
Malware can enter systems through various vectors, including malicious email attachments, compromised websites, infected software downloads, and even through physical devices like infected USB drives. To protect against malware, it's essential to maintain strong cybersecurity practices, including using reputable antivirus and anti-malware software, keeping software up-todate, avoiding suspicious links and downloads, and practicing safe browsing habits.

Practical-9: Malware Analysis and Detection

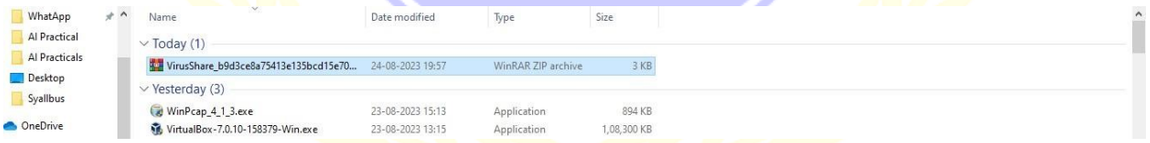
Aim: Analyze and identify malware samples using antivirus tools, analyze their behavior, and develop countermeasures to mitigate their impact.

Procedure:

We select the website 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)



By clicking the above download icon the Malware gets downloaded in ZIP format.



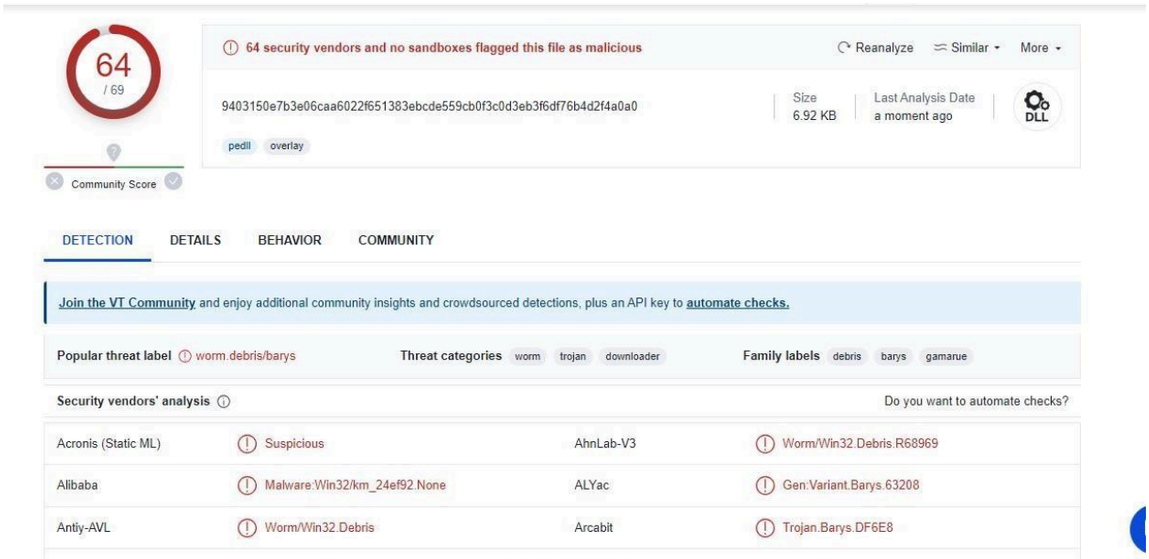
For unzip the password is “infected”, there is no need to unzip the file, we create a folder “Malware” on desktop and save the file in the folder

- In order to analyse the Malware, we select the website www.virustotal.com

- In order to analyse the Malware, we select the website www.virustotal.com

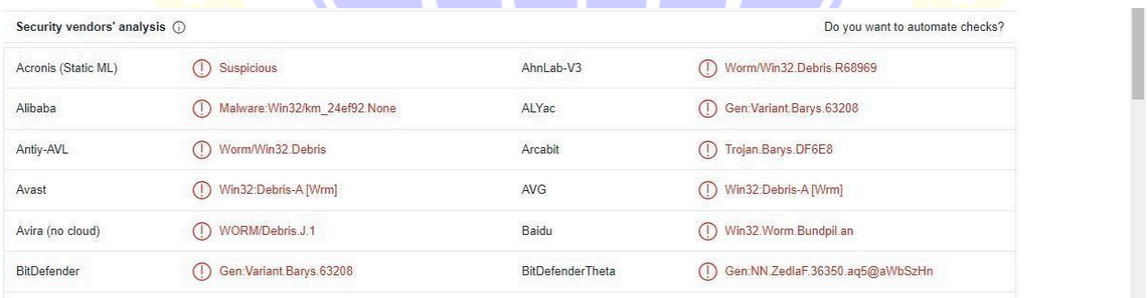
- Click on “Choose File” and select the file from the location (ZIP file will do, if asks for password enter infected)

- We get the following after the upload is complete



We interpret the following findings

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



- 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
- 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

Countermeasures:

Countermeasures are strategies, actions, or precautions taken to prevent or mitigate various risks, threats, or undesirable events. In the context of cyber-security and dealing with potential malware, viruses, and other online threats, here are some common countermeasures you can take:

Use Antivirus and Anti-Malware Software: Install reputable antivirus and anti-malware software on your devices. Keep the software updated to ensure you have the latest protection against known threats.

Keep Operating Systems and Software Updated: Regularly update your operating system, web browsers, plugins, and other software. Updates often include security patches that address vulnerabilities.

Use Strong and Unique Passwords: Use complex passwords that combine upper and lower case letters, numbers, and symbols. Avoid using common or easily guessable passwords. Consider using a password manager to securely store your passwords.

Enable Two-Factor Authentication (2FA): Whenever possible, enable two-factor authentication for your online accounts. This adds an extra layer of security by requiring a second form of verification in addition to your password.

Be Cautious with Email and Attachments: Be wary of unsolicited emails, especially those with attachments or links. Don't open attachments or click on links from unknown or suspicious sources. Verify the sender's authenticity before taking any action.

Use a Firewall: Enable firewalls on your devices and network. Firewalls help block unauthorized access and protect your system from external threats.

Regular Backups: Regularly back up your important data to an external source or a cloud storage service. In case of a malware attack or data loss, you'll have a copy of your important files.

Secure Wi-Fi Networks: Secure your home or office Wi-Fi network with a strong password and encryption. Avoid using public Wi-Fi networks for sensitive activities.

Use Ad-Blockers and Script Blockers: Install browser extensions that block ads and potentially malicious scripts. This can help prevent drive-by downloads and malvertising.

Disable Macros: Disable macros in office documents unless you're certain they are safe. Malicious macros are often used to deliver malware.

Download Software from Official Sources: Only download software from reputable and official sources. Be cautious of downloading software from unfamiliar websites.

Regularly Scan for Malware: Perform regular scans of your devices using reputable antivirus and anti-malware tools.

Use Virtual Private Networks (VPNs): When connecting to the internet, especially on public networks, use a VPN to encrypt your internet connection and enhance your privacy.

Implement Security Policies: If you're managing a network or a business, establish and enforce security policies for employees, including guidelines for safe browsing, email practices, and device usage.

Learning Outcomes:

1. Successfully analyzed malware samples.
2. Understood the characteristics and behaviors of different types of malware.
3. Developed effective strategies to counteract malware threats.

Course Outcomes:

1. Apply malware analysis techniques to enhance security.
2. Analyze malware behavior for incident response.
3. Understand the impact of malware on systems and networks.

Conclusion:

The experiment effectively demonstrated the analysis and detection of malware, highlighting the importance of proactive security measures.

Viva Question:

1. What is malware analysis?
2. How do you identify different types of malware?
3. What are common behaviors exhibited by malware?
4. How can organizations protect against malware attacks?

For Faculty Use

Correction Parameters	Formative Assessment [ ]	Timely completion practical [ ]	Attendance Learning Attitude[ ]

Theory-10

Firewall Configuration and Rule-based Filtering

Firewalls are network security devices or software applications designed to monitor, filter, and control incoming and outgoing network traffic based on a set of predetermined security rules. The primary purpose of a firewall is to act as a barrier between a trusted internal network (such as a company's internal network) and untrusted external networks (like the internet), in order to prevent unauthorized access, data breaches, and other potential cyber threats.

Firewalls operate by inspecting network packets (small units of data) as they pass through the firewall and making decisions about whether to allow or block the traffic based on a set of predefined rules. These rules can be configured to specify which types of traffic are permitted and which are denied. Firewalls can be implemented in various locations within a network, including at the network perimeter, on individual devices, or even within cloud environments.

There are several types of firewalls, including:

Packet Filtering Firewalls: These examine network packets and decide whether to allow or block them based on criteria like source and destination IP addresses, port numbers, and protocols. While they are relatively simple, they lack the ability to inspect the actual content of the data packets.

Stateful Inspection Firewalls: Also known as dynamic packet filtering firewalls, these maintain a state table to keep track of active connections and only allow incoming traffic that matches an existing, legitimate connection. This approach is more secure than basic packet filtering.

Proxy Firewalls: Proxy firewalls act as intermediaries between the internal network and the external network. They receive requests from internal users, then initiate and manage connections to external resources on behalf of those users. This can provide an additional layer of security by hiding internal network details.

Application Layer Firewalls: These operate at the application layer of the OSI model and can understand the context of the traffic, such as the specific application or service being accessed. They are capable of making more fine-grained decisions based on the actual content of the traffic.

Next-Generation Firewalls (NGFW): NGFWs combine traditional firewall functionality with advanced features such as intrusion prevention, deep packet inspection, and application-aware filtering. They are designed to provide more comprehensive protection against modern threats.

Firewalls play a crucial role in network security by helping organizations establish a strong defence against unauthorized access, malware, and various cyber attacks. However, it's important to note that while firewalls are an essential component of a comprehensive cyber- security strategy, they are not a standalone solution. They should be used in conjunction with other security measures such as antivirus software, intrusion detection systems, regular software updates, and user training to ensure a robust defence against evolving threats.

Using a firewall to block unauthorized access is an important aspect of securing your network and systems. Firewalls act as barriers between your network and potential threats from the internet or other external sources.

Practical-10: Firewall Configuration and Rule-based Filtering

Aim:

Configure and test firewall rules to control network traffic, filter packets based on specified criteria, and protect network resources from unauthorized access.

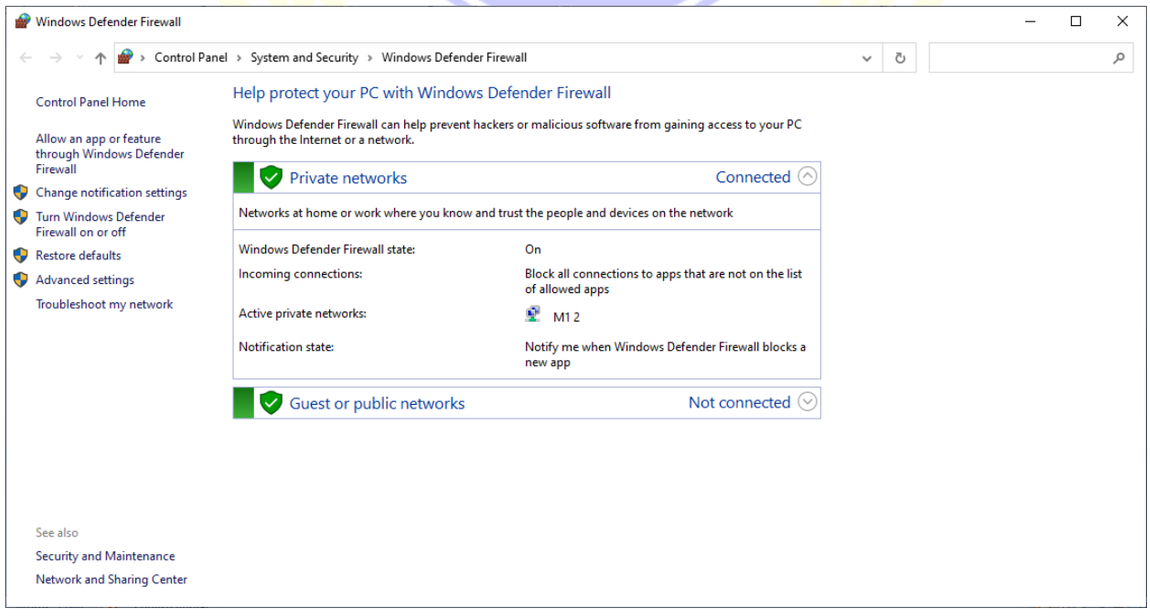
Procedure: Blocking the HTTP and HTTPS (Port 80 and Port 443) using the Firewall

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

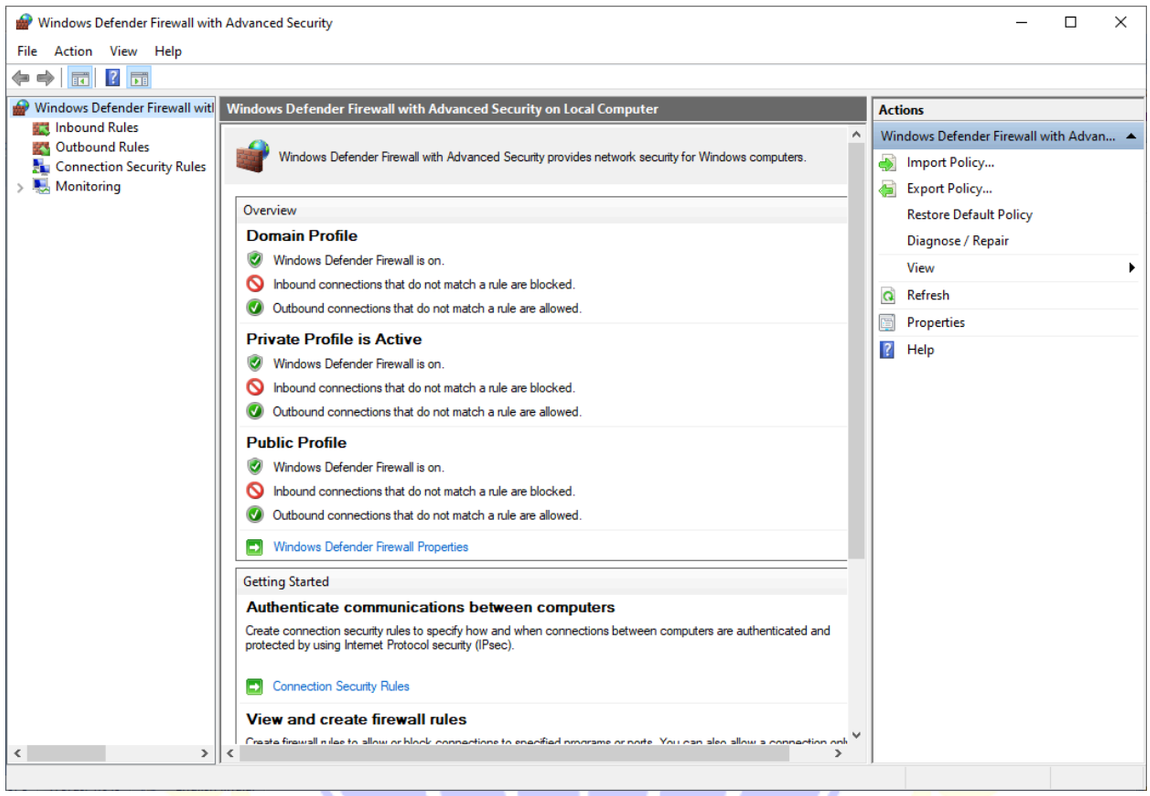
Port Number	Protocol	Application
20	TCP	FTP data
21	TCP	FTP control
22	TCP	SSH
25	TCP	SMTP
53	UDP, TCP	DNS
80	TCP	HTTP (WWW)
110	TCP	POP3
443	TCP	SSL

We perform the blocking Port operation as follows:

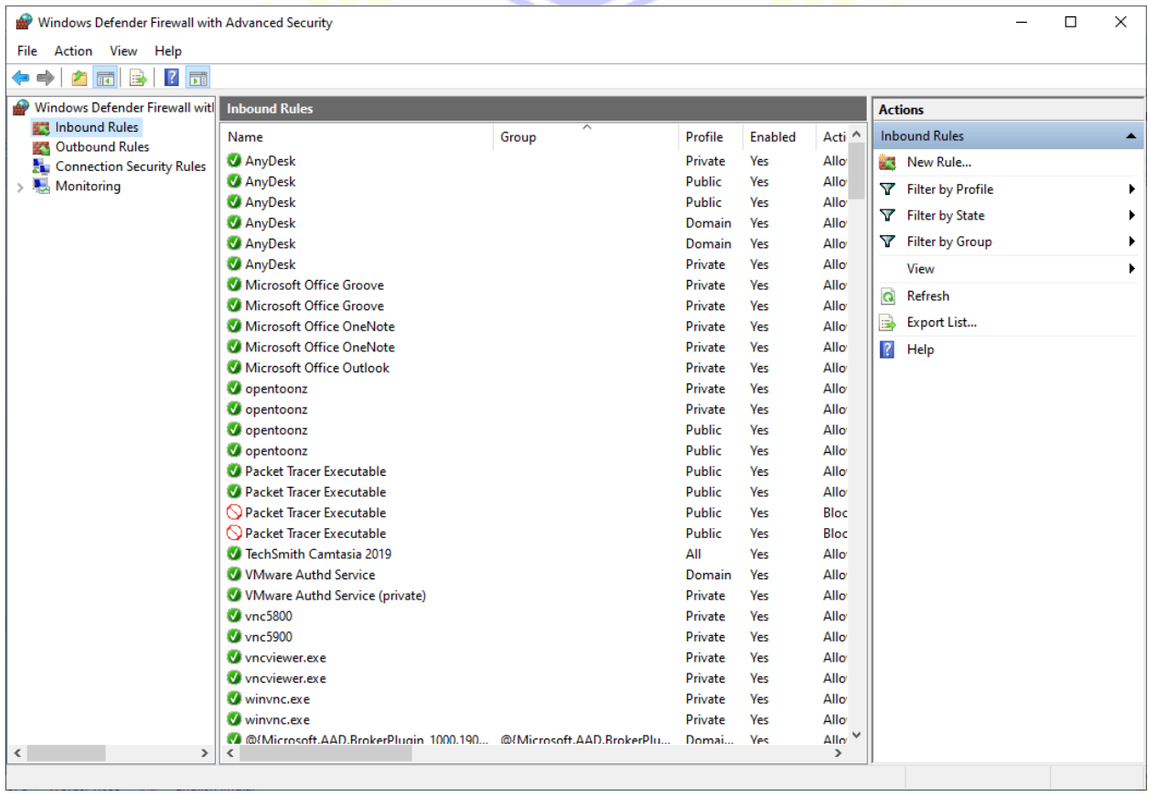
Step 1: We access any website through the browser and confirm that the HTTP/HTTPS protocols are working.



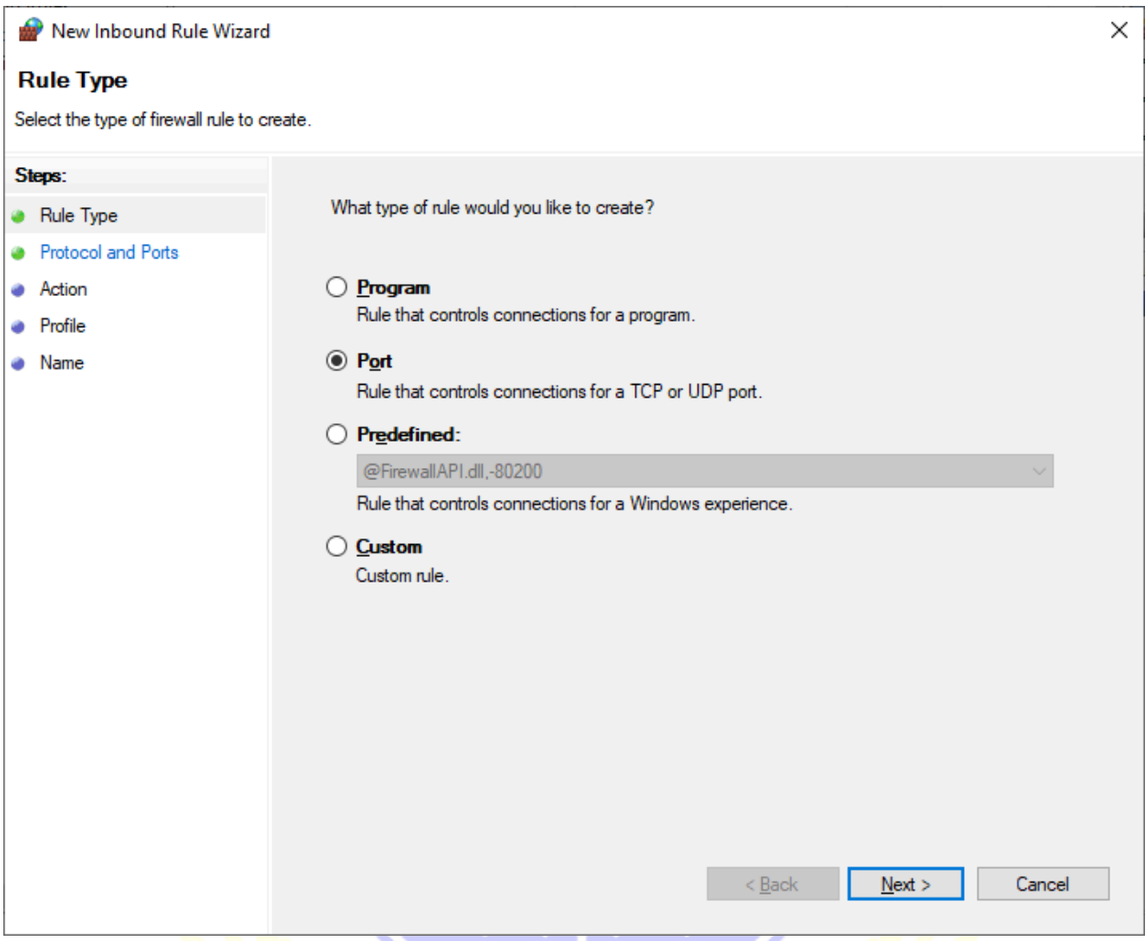
Step 2: We open 'Windows Defender Firewall'



Next we click on 'Advanced settings'

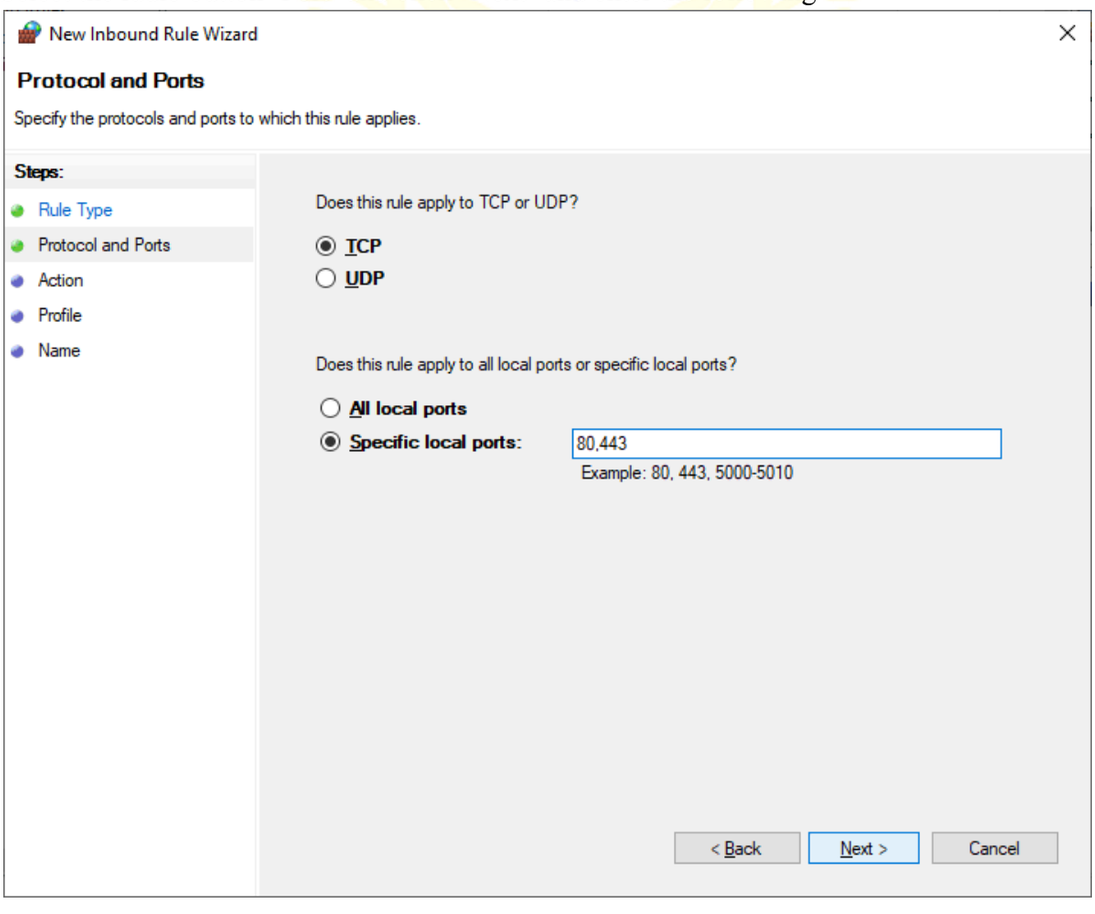


Next we click on ‘Inbound Rules’

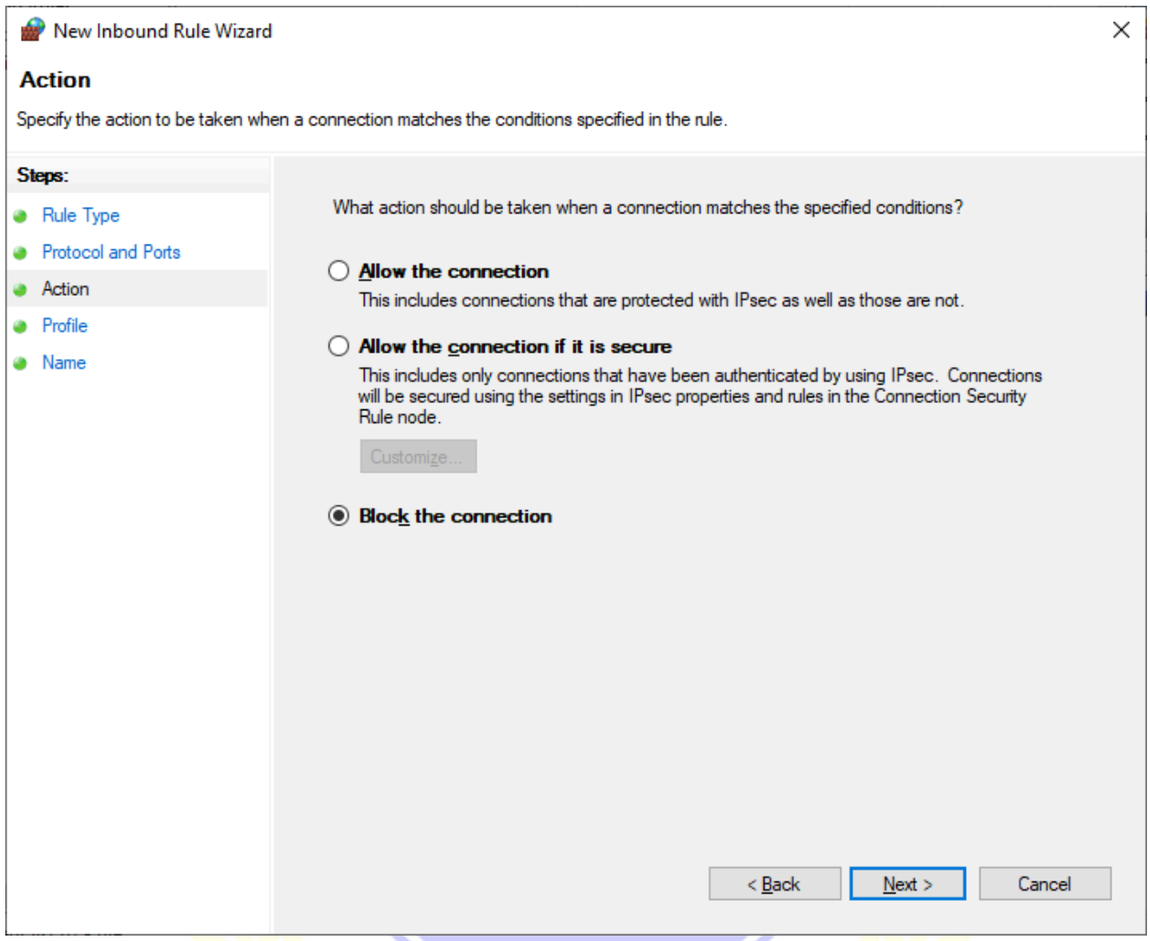


Then click on ‘New Rule’

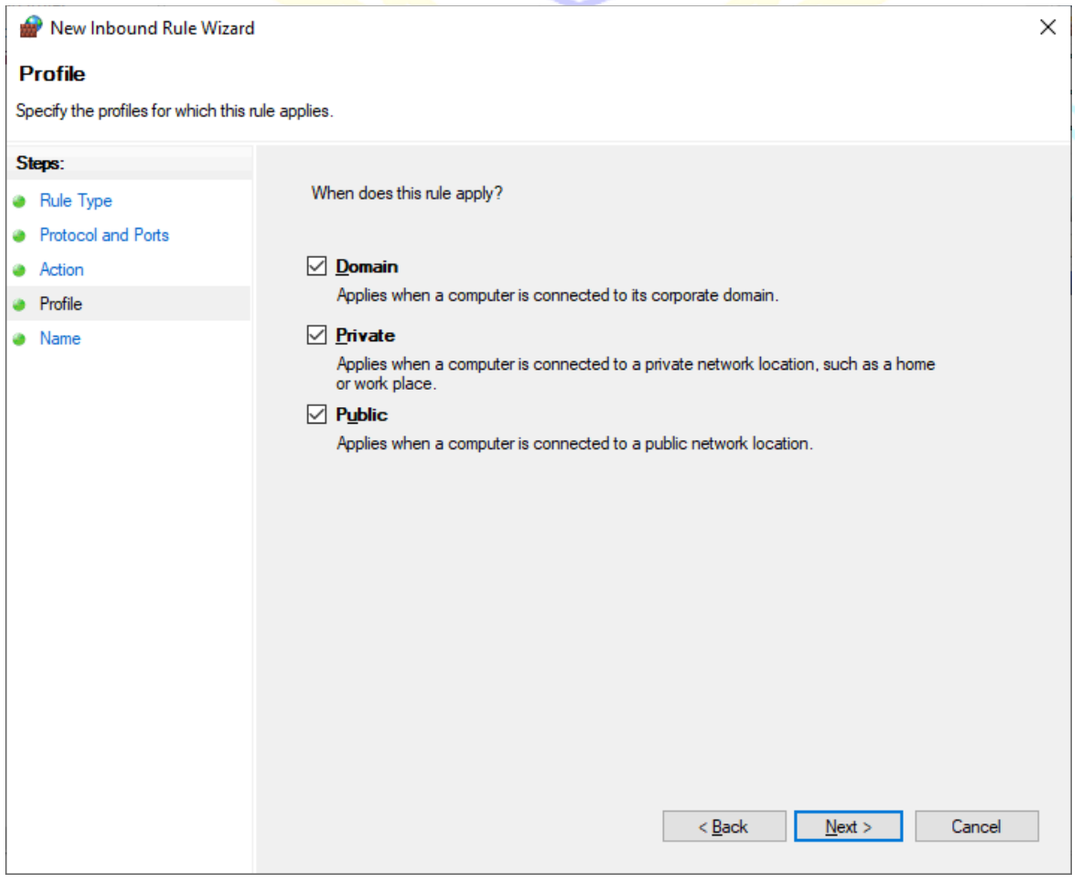
Select the radio button ‘Port’ and click ‘Next’ and enter the following

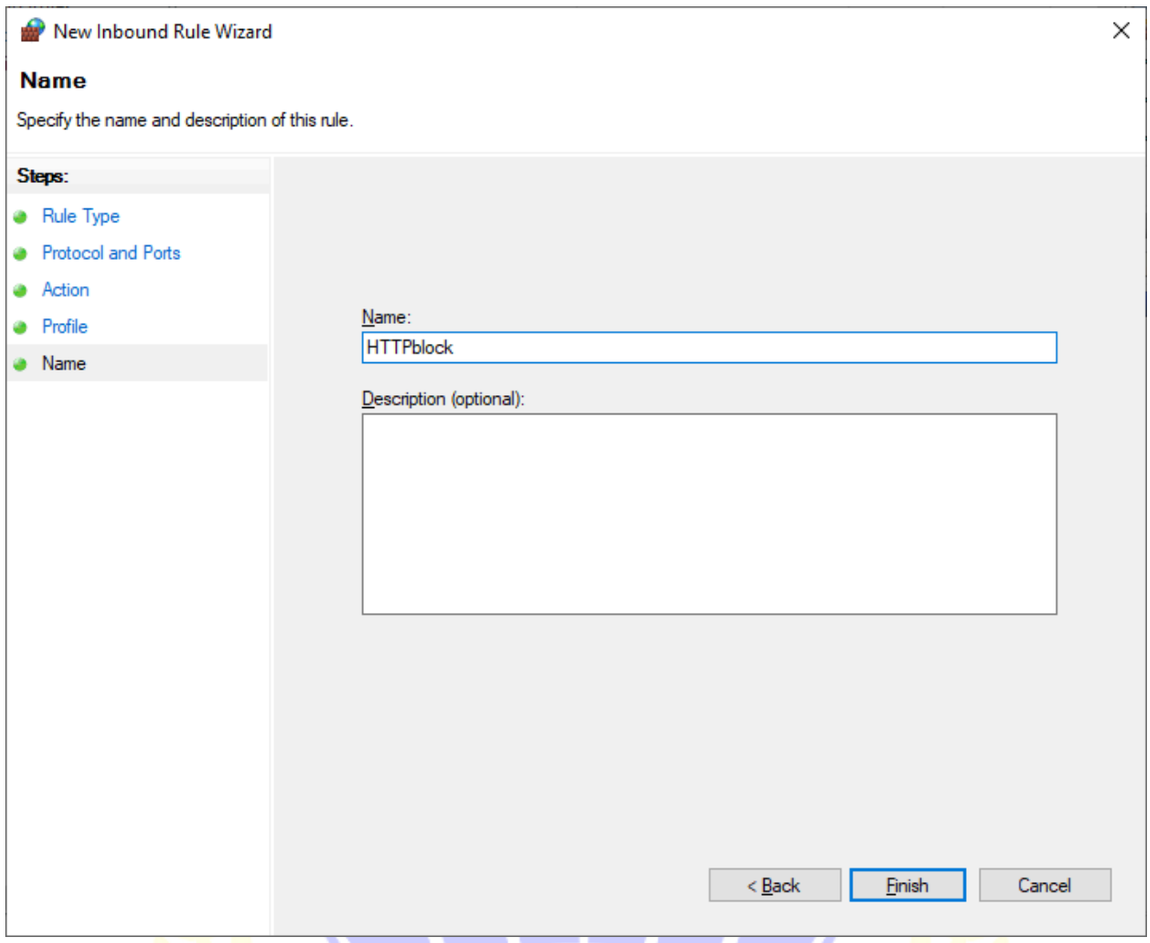


After next, we need to finalise the rule



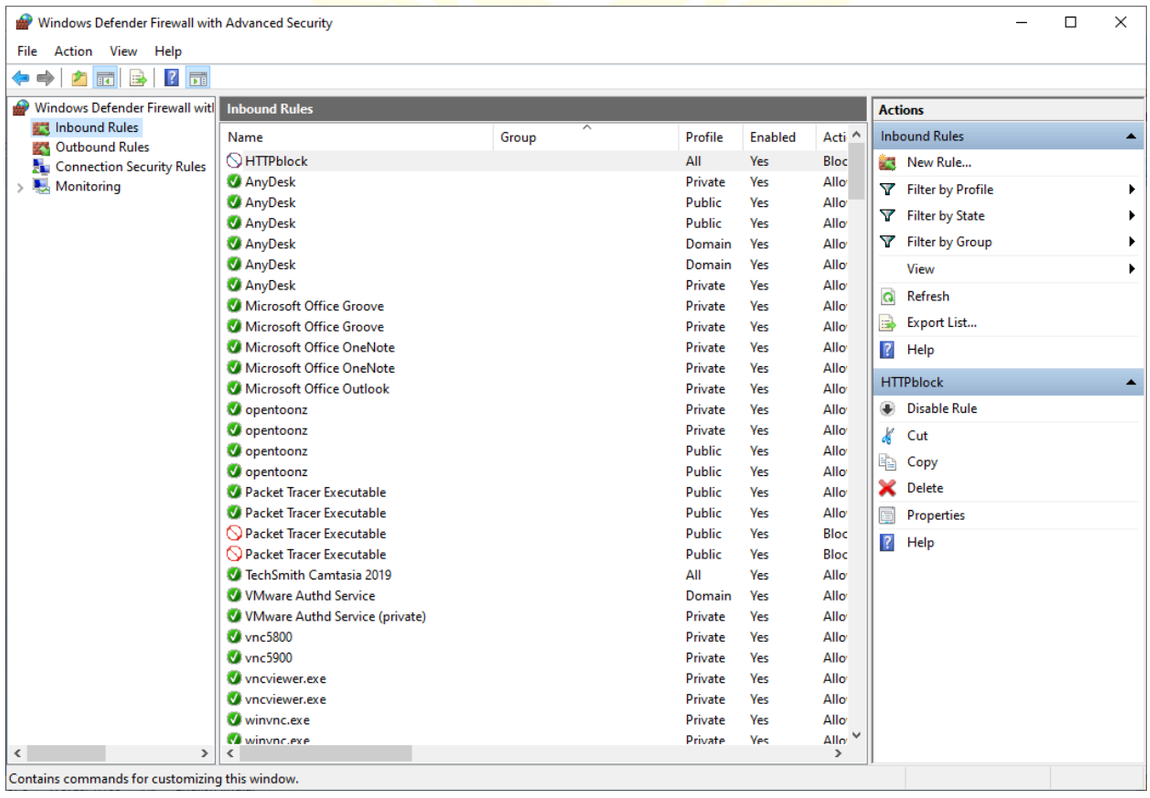
Click 'Next' and we get the following



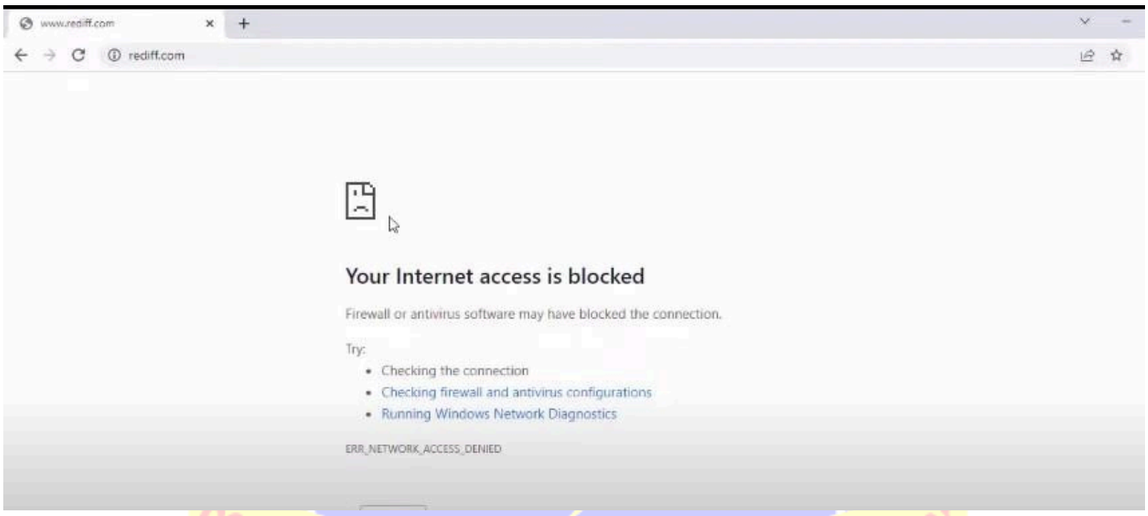


After clicking the 'Next' button we need to name the rule and click finish

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



Learning Outcomes:

1. Successfully configured firewall rules for traffic management.
2. Understood the concepts of packet filtering and access control.
3. Demonstrated the effectiveness of firewalls in securing networks.

Course Outcomes:

.1. Apply firewall techniques to enhance network security.
2. Analyze the data collected by firewalls for incident response.
3. Understand the challenges of firewall management.

Conclusion:

The experiment effectively demonstrated the configuration and testing of firewall rules, highlighting their importance in securing network resources.

Viva Question:

1. What is a firewall?
2. How does packet filtering work?
3. What are the types of firewalls?
4. How can you improve firewall security?

For Faculty Use

Correction Parameters	Formative Assessment [ ]	Timely completion practical [ ]	Attendance Learning Attitude[ ]