

SCHEME :K

Name : _____
Roll No.: _____ Year : 20 ____ 20 ____
Exam Seat No. : _____

LABORATORY MANUAL FOR NETWORK AND INFORMATION SECURITY (316317)



COMPUTER ENGINEERING GROUP



**MAHARASHTRA STATE BOARD OF
TECHNICAL EDUCATION, MUMBAI**
(Autonomous)(ISO21001:2018)(ISO/IEC27001:2013)

Vision

To ensure that the Diploma level Technical Education constantly matches the latest requirements of Technology and industry and includes the all-round personal development of students including social concerns and to become globally competitive, technology led organization.

Mission

We, at MSBTE are committed to offer the best in class academic services to the students and institutes to enhance the delight of industry and society. This will be achieved through continual improvement in management practices adopted in the process of curriculum design, development, implementation, evaluation and monitoring system along with adequate faculty development programmes

Core Values

MSBTE believes in the following:

- Skill development in line with industry requirements
- Industry readiness and improved employability of Diploma holders
- Synergistic relationship with industry
- Collective and Cooperative development of all stake holders
- Technological interventions in societal development
- Access to uniform quality technical education

A Practical Manual
For
Network and Information
Security
(316317)

Semester-VI
CM/ CO/ CW / IF/ BD/ SE



Maharashtra State Board of Technical
Education, Mumbai

(Autonomous) (ISO 21001:2018)(ISO/IEC 27001:2013)

‘K’ Scheme Curriculum



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

(Autonomous) (ISO 21001:2018)(ISO/IEC 27001:2013)

Address: 4th floor, Govt. Polytechnic Building, 49,
Kherwadi, Bandra (E), Mumbai- 400 051

Tel: 022 62542100

Email: secretary@msbte.com



Maharashtra State Board of Technical Education

Certificate

This is to certify that Mr./ Ms
Roll No..... of.....Semester of Diploma
in.....of **Institute Name**
.....(**Inst. Code:.....**) has
completed the term work satisfactorily in Course **Network and Information**
Security (Course Code:316317) for the academic year 20..... to 20..... as
prescribed in the curriculum.

Place

Enrollment No.....

Date:

Exam Seat No.

Course Teacher

Head of the Department

Principal



Preface

The primary focus of any engineering laboratory/field work in the technical education system is to develop the much needed industry relevant competencies and skills. With this in view, MSBTE embarked on this innovative 'K' Scheme curricula for engineering Diploma programmes with outcome-based education as the focus and accordingly, relatively large amount of time is allotted for the practical work. This displays the great importance of laboratory work making each teacher, instructor and student to realize that every minute of the laboratory time need to be effectively utilized to develop these outcomes, rather than doing other mundane activities. Therefore, for the successful implementation of this outcome-based curriculum, every practical has been designed to serve as a 'vehicle' to develop this industry identified competency in every student. The practical skills are difficult to develop through 'chalk and duster' activity in the classroom situation. Accordingly, the “K” scheme laboratory manual development team designed the practical's to focus on outcomes, rather than the traditional age old practice of conducting practical's to verify the theory (which may become a byproduct along the way).

This laboratory manual is designed to help all stakeholders, especially the students, teachers and instructors to develop in the student the pre-determined outcomes. It is expected from each student that at least a day in advance, they have to thoroughly read the concerned practical procedure that they will do the next day and understand minimum theoretical background associated with the practical. Every practical in this manual begins by identifying the competency, industry relevant skills, course outcomes and practical outcomes which serve as a key focal point for doing the practical. Students will then become aware about the skills they will achieve through procedure shown there and necessary precautions to be taken, which will help them to apply in solving real-world problems in their professional life.

This manual also provides guidelines to teachers and instructors to effectively facilitate student-centered lab activities through each practical exercise by arranging and managing necessary resources in order that the students follow the procedures and precautions systematically ensuring the achievement of outcomes in the students.

Operating systems are an essential part of any computer system. Similarly, a course on operating systems is an essential part of any computer group. We hope that students will also find it useful. It provides a clear description of practical performance, execution and working of Operating System.

Although all care has been taken to check for mistakes in this laboratory manual, yet it is impossible to claim perfection especially as this is the first edition. Any such errors and suggestions for improvement can be brought to our notice and are highly welcome.

Program Outcomes (POs) to be achieved through Course:

PO1	Basic and Discipline specific knowledge: Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
PO2	Problem analysis: Identify and analyses well-defined engineering problems using codified standard methods.
PO3	Design/ development of solutions: Design solutions for well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
PO4	Engineering Tools, Experimentation and Testing: Apply modern engineering tools and appropriate technique to conduct standard tests and measurements.
PO5	Engineering practices for society, sustainability and environment: Apply appropriate technology in context of society, sustainability, environment and ethical practices.
PO6	Project Management: Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
PO7	Life-long learning: Ability to analyses individual needs and engage in updating in the context of technological changes.

List of Relevant Skills

The following industry-relevant skills of the competency “Implement Network and Information Security Mechanisms” are expected to be developed in you by performing the practicals of this laboratory manual:

1. Configure and manage antivirus software and apply privacy and security settings on operating systems.
2. Implement user authentication techniques such as password recovery, access control, and multi-factor authentication.
3. Apply cryptographic algorithms (e.g., Caesar, Vernam, Rail Fence, Columnar Transposition) for data encryption and decryption.
4. Generate and verify digital signatures, hash codes, and digital certificates using open-source tools.
5. Configure and analyze firewall settings and implement secure email communication using encryption tools.
6. Use steganography techniques to securely embed and extract messages within digital media.
7. Identify and mitigate network and database security threats using security tools and protocols (e.g., IDS, Kerberos, IPsec).

Practical Course Outcome Matrix

Course Outcomes (COs)

CO1	Identify types of Cyber attacks and threats.
CO2	Apply multi-factor user authentication and access control.
CO3	Implement encryption/decryption techniques.
CO4	Use tools and techniques to prevent cyber attacks.
CO5	Apply security on Network and Database.

Sr. No.	Title of the Experiment	CO1	CO2	CO3	CO4	CO5
1	* i. Install and configure Antivirus software on system (Licensed copy) ii. Use privacy and security settings on operating system	✓				
2	* i. Set up single level authentication for computer system ii. Recover the password of computer system using any freeware password recovery tool (Example- John the ripper).		✓			
3	* i. Grant security to file, folder or application using access permissions and verify it ii. Grant access permission while sharing file and folder		✓			
4	* Write a utility using C/Shell programming to create strong password authentication (Password should be more than 8 characters, and combination of digits, letters and special characters #, %, &, @)		✓			
5	* i. Write a C program to implement caesar cipher technique to perform encryption and decryption of text ii. Apply Caesar cipher technique to perform encryption and decryption of text using any open-source tool (Example - Cryptool)			✓		
6	i. Implement Vernam cipher encryption technique to perform encryption of text using C programming language ii. Apply Vernam cipher technique to perform encryption and decryption of text using any open-source tool (Example - Cryptool)			✓		

7	Implement railfence encryption technique to perform encryption of text using C programming language			✓		
8	Implement simple Columnar Transposition encryption technique to perform encryption of text using C programming language			✓		
9	* Create and verify Hash Code for given message using any Open-source tool (Example-Cryptool)			✓		
10	i. Write a C program to implement Diffie-Hellman key exchange algorithm to perform encryption of text ii. Use Diffie-Hellman key exchange algorithm to perform encryption and decryption of text using any open-source tool (Example - Cryptool)				✓	
11	* Use Steganography to encode and decode the message using any open-source tool (Example-OpenStego)				✓	
12	* Create and verify digital signature using any Open- source tool (Example-Cryptool)				✓	
13	Create and verify digital Certificate using any Open- source tool (Example-Cryptool).				✓	
14	Configure firewall settings on any operating system.				✓	
15	* Send a test mail securely using any open-source tool (Example- Pretty Good Privacy with GnuPG)					✓
16	Find the origin of email using email tracker pro.					✓

Guidelines to Teachers

1. Teachers should align the explanation of the topic to teaching learning outcome (TLOs).
2. Refer to laboratory learning outcome (LLOs) for the execution of the practical to focus on the defined objectives.
3. Promote life-long learning by training the students to equip themselves with essential knowledge, skills and attitudes.
4. If required, provide demonstration for the practical emphasizing on the skills that the student should achieve.
5. Teachers should give opportunity to the students for exhibiting their skills after the demonstration.
6. Provide feedback and/or suggestions and share insights to improve effectiveness.
7. Assess students' skill achievement related to COs of each unit.

Instructions for Students

1. 100% attendance is compulsory for all practical sessions.
2. Students must adhere to ethical practices.
3. All the students must follow the schedule of practical sessions, complete the assigned work/activity and submit the assignment in stipulated time as instructed by the course teacher.
4. Students shall listen carefully the lecture given by teacher about importance of subject, learning structure, course outcomes.
5. Students shall understand the purpose of experiment and its practical implementation.
6. Students shall write the answers of the questions during practical.
7. Student should feel free to discuss any difficulty faced during the conduct of practical.
8. Student shall attempt to develop related hands-on skills and gain confidence.
9. Students should develop habit to submit the write-ups on the scheduled dates and time.

Content Page**List of Practical and Formative Assessment Sheet**

Sr. No	Practical Title	Date of Performance	Date of Submission	Assessment Marks (25)	Teacher Sign	Remark
1	* i. Install and configure Antivirus software on system (Licensed copy) ii. Use privacy and security settings on operating system					
2	* i. Set up single level authentication for computer system ii. Recover the password of computer system using any freeware password recovery tool (Example- John the ripper).					
3	* i. Grant security to file, folder or application using access permissions and verify it ii. Grant access permission while sharing file and folder					
4	* Write a utility using C/Shell programming to create strong password authentication (Password should be more than 8 characters, and combination of digits, letters and special characters #, %, &, @)					
5	* i. Write a C program to implement caesar cipher technique to perform encryption and decryption of text ii. Apply Caesar cipher technique to perform encryption and decryption of text using any open-source tool (Example - Cryptool)					
6	i. Implement Vernam cipher encryption technique to perform encryption of text using C programming language ii. Apply Vernam cipher technique to perform encryption and decryption of text using any open-source tool (Example - Cryptool)					

7	Implement railfence encryption technique to perform encryption of text using C programming language					
8	Implement simple Columnar Transposition encryption technique to perform encryption of text using C programming language					
9	* Create and verify Hash Code for given message using any Open-source tool (Example-Cryptool)					
10	i. Write a C program to implement Diffie-Hellman key exchange algorithm to perform encryption of text ii. Use Diffie-Hellman key exchange algorithm to perform encryption and decryption of text using any open-source tool (Example - Cryptool)					
11	* Use Steganography to encode and decode the message using any open-source tool (Example-OpenStego)					
12	* Create and verify digital signature using any Open- source tool (Example-Cryptool)					
13	Create and verify digital Certificate using any Open- source tool (Example-Cryptool).					
14	Configure firewall settings on any operating system.					
15	* Send a test mail securely using any open-source tool (Example- Pretty Good Privacy with GnuPG)					
16	Find the origin of email using email tracker pro.					
Total						

***Total marks to be transferred to proforma published by MSBTE**

Note:

- '*' Marked Practical's (LLOs) Are mandatory.
- Minimum 80% of above list of lab experiment are to be performed.
- Judicial mix of LLOs are to be performed to achieve desired outcomes.

**Practical No. 01: i. Install and configure Antivirus software on system (Licensed copy)
ii. Use privacy and security settings on operating system****I. Practical Significance**

This practical demonstrates installation, activation, configuration, and testing of antivirus software, along with applying operating system security updates and settings. Antivirus and patching form the first line of defence against malware, viruses, worms, and exploits. By combining endpoint protection with OS security updates, organizations can significantly reduce vulnerabilities.

II. Industry / Employer Expected Outcome

Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO1 - Identify types of Cyber attacks and threats.

IV. Laboratory Learning Outcome (LLO)

LLO 1.1 Install Antivirus software on system.

LLO 1.2 Apply privacy and security settings to protect operating system.

V. Relevant Affective domain related Outcomes(s)

1. Responsibility: following institutional security policy when configuring systems.
2. Attention to detail: verifying update settings and scheduled scans.
3. Ethical awareness: only using licensed software and not circumventing controls.

VI. Theoretical Background

Antivirus software is a security application designed to detect, prevent, and remove malware such as viruses, worms, Trojans, spyware, ransomware, and rootkits. It works by scanning files, monitoring system behaviour, and comparing data against a database of known malware signatures.

Since operating systems and applications are constantly at risk of malicious attacks, antivirus provides a first line of defence. It prevents unauthorized code from executing, safeguards sensitive data, and ensures the system remains operational.

Operating system security is the process of implementing measures to protect OS resources such as files, memory, processes, and user accounts from unauthorized access and misuse. Privacy and security settings ensure confidentiality, integrity, and availability (CIA triad) of the system and user data.

Every OS has default settings that may expose vulnerabilities. By customizing and strengthening privacy/security configurations, we minimize risks like malware infection, identity theft, unauthorized data access, and insider misuse.

VII. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	Antivirus software,(Quick Heal)		

VIII. Procedure

Install and Configure Antivirus Software in Your System

We will install **Quick Heal Antivirus** using the Quick Heal CD or by downloading the Quick Heal Installer from www.quickheal.co.in.

A. Using Quick Heal CD

1. Insert the Quick Heal antivirus CD/DVD in the DVD drive.
2. Click **Install**.
3. Follow the on-screen instructions.

B. Using Quick Heal Installer

1. Download the Quick Heal antivirus from:
<https://www.quickheal.co.in/quick-heal-product-installer>
2. Double-click the downloaded product setup.
3. Follow the on-screen instructions to complete the installation.



Part B: Register Quick Heal Antivirus

To enjoy full benefits and configurations of Quick Heal, you must register the product online.

1. Open **Quick Heal Antivirus**.
2. On the Quick Heal Dashboard, click the **Register Now** button.
3. On the Registration Wizard, enter the **20-digit Product Key** and click **Next**.
4. The Registration Information window will appear.

5. Enter details in the **Purchased From** and **Register For** text boxes, then click **Next**.
6. Provide your **Name, Email Address, and Contact Number**.
 - Select your **Country, State, and City**.
 - If your State/Province or City is not in the list, type it manually in the respective box.
7. Click **Next** to continue.
8. A confirmation screen will appear with the details entered.
 - If corrections are needed, click **Back** to edit.

Part C: Activate and Configure Scanning

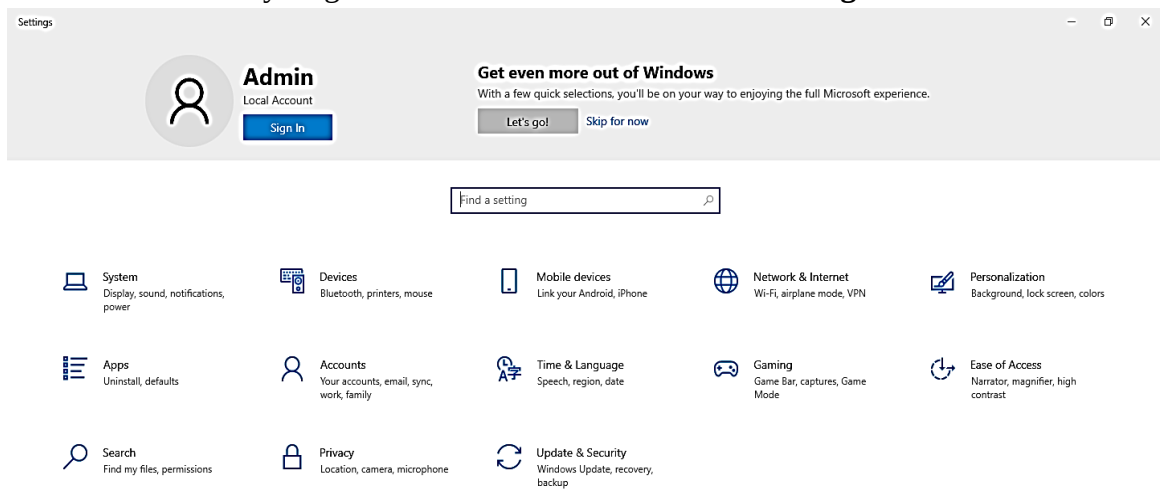
1. Click **Next** to continue.
2. The product will be activated successfully. The **expiry date of your license** will be displayed.
3. Click **Finish** to close the Registration Wizard.
4. Quick Heal will automatically perform a **full system scan**.
5. A window will prompt you to:
 - Select how frequently you want your system to be scanned.
 - Select the type of scan.
6. Click **Finish** to apply the settings.

Use Privacy and Security Settings on Operating System

Privacy and security settings protect your system, data, and online activity from unauthorized access or malicious attacks. Modern operating systems like **Windows 10/11** and **Windows 7/8** provide built-in tools to configure these settings.

Steps for Windows 10/11

1. **Open Settings**
 - Press **Windows + I** keys together or click the **Start** menu → **Settings**.



2. **Navigate to Privacy Settings**
 - In the Settings window, click **Privacy**.
 - Review options such as:
 - **Location** – Turn off if you don't want apps to track your location.
 - **Camera & Microphone** – Choose which apps can access these.
 - **Activity History** – Disable if you don't want Windows to collect usage data.
 - **Background Apps** – Turn off unnecessary apps to reduce tracking.

3. Navigate to Security Settings

- In the Settings window, go to **Update & Security** → **Windows Security**.
- Configure:
 - **Virus & Threat Protection** – Ensure Windows Defender or another antivirus is active.
 - **Firewall & Network Protection** – Keep the firewall turned ON for all networks.
 - **App & Browser Control** – Enable SmartScreen for safer browsing.
 - **Device Security** – Check hardware security features.

4. Set Login and Account Security

- Go to **Accounts** → **Sign-in Options**.
- Set a **Strong Password** or use **Windows Hello (PIN, fingerprint, or face recognition)**.
- Enable **Two-Factor Authentication (2FA)** for your Microsoft Account.

Steps for Windows 7/8/8.1

1. Open **Control Panel** → **Action Center**.
2. Click **Security** tab and check:
 - **Windows Update** – Turn ON automatic updates.
 - **Windows Firewall** – Ensure it is enabled.
 - **Virus Protection** – Make sure antivirus is installed and active.
 - **Internet Security Settings** – Use recommended levels for safer browsing.
 - **User Account Control (UAC)** – Keep it ON for permission prompts when apps make changes.
3. Open **Privacy Settings**:
 - Disable location services if not needed.
 - Manage app permissions through installed applications.

IX. Precautions

1. Use legitimate license keys only.
2. Do not interrupt updates mid-way.
3. If system fails to boot after update, revert snapshot/restore.
4. Use isolated test environment when experimenting.
5. During lab, do not perform other simultaneous installations to avoid conflicts.

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. State the purpose of antivirus software?
2. Difference between a quick scan and a full scan in antivirus software.
3. Demonstrate how to check and install pending OS updates on your system.
4. Compare antivirus protection with firewall defence. How do they complement each other?
5. Design a basic security policy for a workstation that includes antivirus, firewall, and OS updates

Space for Answer

This image shows a full page of a document template designed for handwriting practice or general note-taking. It consists of approximately 28 evenly spaced horizontal dotted lines across the entire width of the page. The background is plain white, and there are no margins, headers, footers, or other markings present.

XI. Exercise

1. Install and configure NP (Net Protector) on a system and schedule a scan.
2. Enable and disable Windows Firewall; analyze the changes in inbound/outbound connections.
3. Apply OS updates and generate a report/log of update history.

XII. References / Suggestions for further reading

1. Stallings, W. & Brown, L., 2014. Computer Security: Principles and Practice. Pearson. Publisher link
2. Administrator Guide Quick Heal Endpoint Security 5.2. Available at: https://dlupdate.quickheal.com/documents/manual/eps5.2_userguide.pdf
3. YouTube, 2020. Firewall Configuration Tutorial. Available at: <https://www.youtube.com/watch?v=T9c5ZpT2FV0>
4. NPTEL, 2022. Introduction to Information Security. Available at: <https://archive.nptel.ac.in/courses/106/106/106106129/>
5. Virtual Labs, IIIT Hyderabad, n.d. Cryptography Experiments. Available at: <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Following correct installation procedure of Antivirus	20 %
2	Configuring scans, updates, and OS security settings	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Quality of Output (Antivirus installed, updated, OS patched, firewall active)	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 02: i. Set up single level authentication for computer system
ii. Recover the password of computer system using any freeware password recovery tool (Example- John the ripper)

I. Practical Significance

This practical demonstrates the importance of authentication and password management in computer systems. Single-level authentication ensures that only authorized users can access the system, while password recovery tools help administrators recover lost credentials. Understanding authentication mechanisms and recovery methods is crucial for maintaining system availability and security.

II. Industry / Employer Expected Outcome

Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO2 – Apply multi-factor user authentication and access control.

IV. Laboratory Learning Outcome (LLO)

LLO 2.1 Setup and recover password of computer system.

V. Relevant Affective domain related Outcomes(s)

1. Responsibility: applying authentication policies securely.
2. Attention to detail: ensuring correct configuration of authentication and recovery tools.
3. Ethical awareness: using password recovery tools only for authorized and legitimate purposes.

VI. Theoretical Background

Authentication is the process of verifying the identity of a user or system. Single-level authentication typically involves using a username and password. Weak or lost passwords can make systems vulnerable, hence recovery mechanisms are needed.

Password recovery tools such as John the Ripper or Ophcrack use techniques like dictionary attacks, brute force, or rainbow tables to crack passwords. While they can be misused, they are essential for administrators in recovery and auditing scenarios.

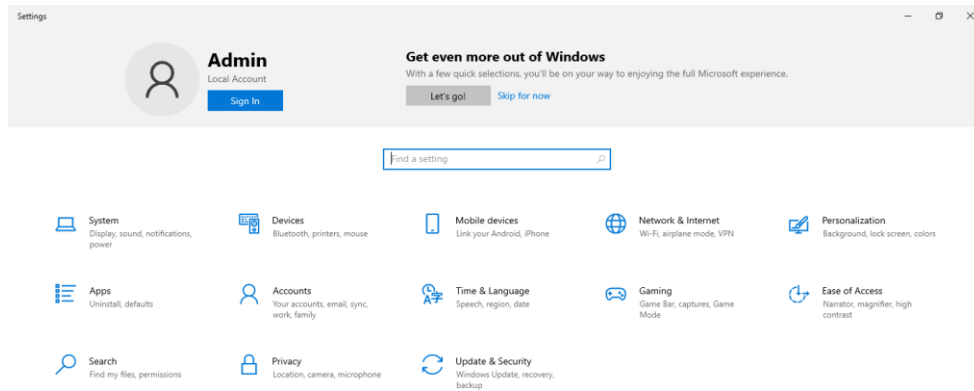
VII. Resources required:

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	John the Ripper		

VIII. Procedure

Steps to Set Up Single Level Authentication (Windows 10/11)

1. **Open Settings**
 - Press **Windows Key + I**
 - OR click **Start → Settings**
2. **Go to Accounts**
 - In the Settings window, click **Accounts**
3. **Access Sign-in Options**
 - From the left menu, select **Sign-in options**
4. **Create a Password**
 - Under **Password**, click **Add** (if no password is set)
 - Enter a **new password**
 - Confirm the password
 - Add a **Password Hint** (to help if you forget it)
 - Click **Next → Finish**
5. **Require Sign-in at Startup**
 - In **Sign-in options**, under **Require sign-in**, choose **When PC wakes up from sleep**
 - This ensures authentication is required every time the system starts or wakes.
6. **Lock and Test**
 - Press **Windows + L** to lock the computer.
 - Try logging in with the new password to confirm setup.



Steps for Windows 7 / 8 / 8.1

1. Open **Control Panel → User Accounts → Create a password for your account**
2. Enter a **strong password** and confirm it.
3. Click **Create Password**.
4. Restart or lock (**Win + L**) to test login authentication.

Recover the password of computer system using any freeware password recovery tool (Example- John the ripper)

A — Prepare (common prerequisites)

1. Use an isolated **lab VM** (snapshot before you begin).
2. Download **John the Ripper** (jumbo build recommended) for Windows:
<https://www.openwall.com/john/> and unzip it.

3. Obtain a wordlist (e.g. rockyou.txt or other wordlists). Place it in the John run folder or note its path.

PASSWORD CRACKING WITH JOHN-THE-RIPPER TOOL:

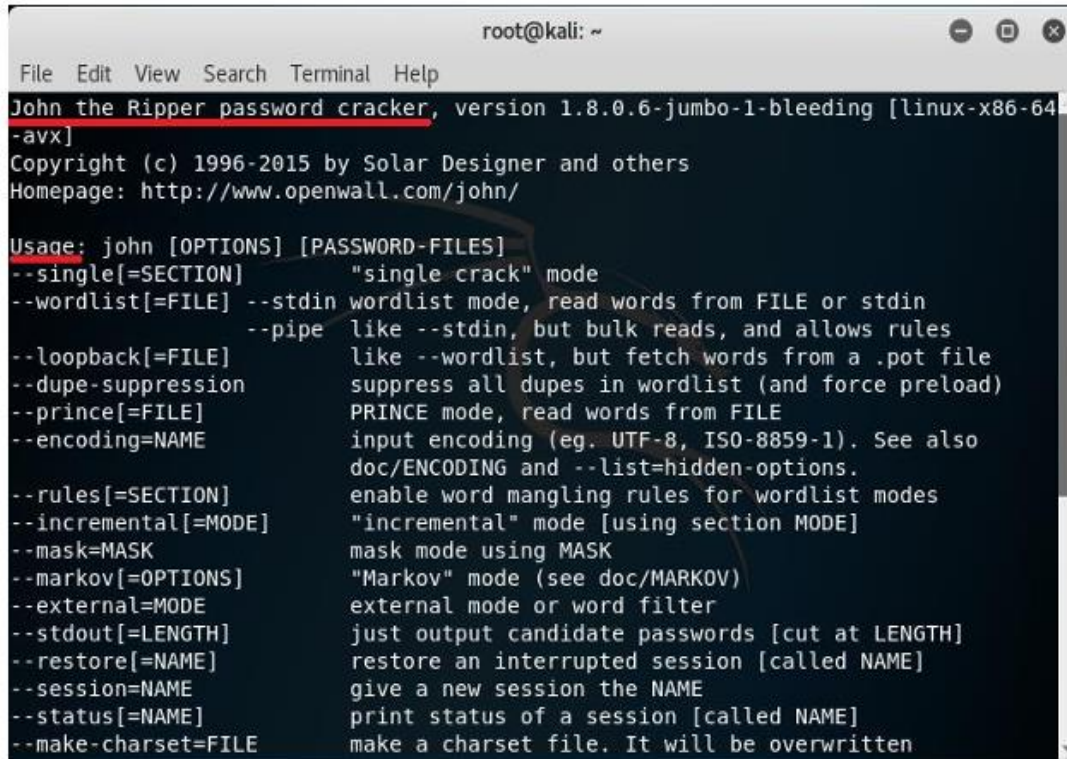
The password in plaintext from hash can be recovered with John-the-ripper tool with the following steps:

Step 1: Open Kali Linux operating system

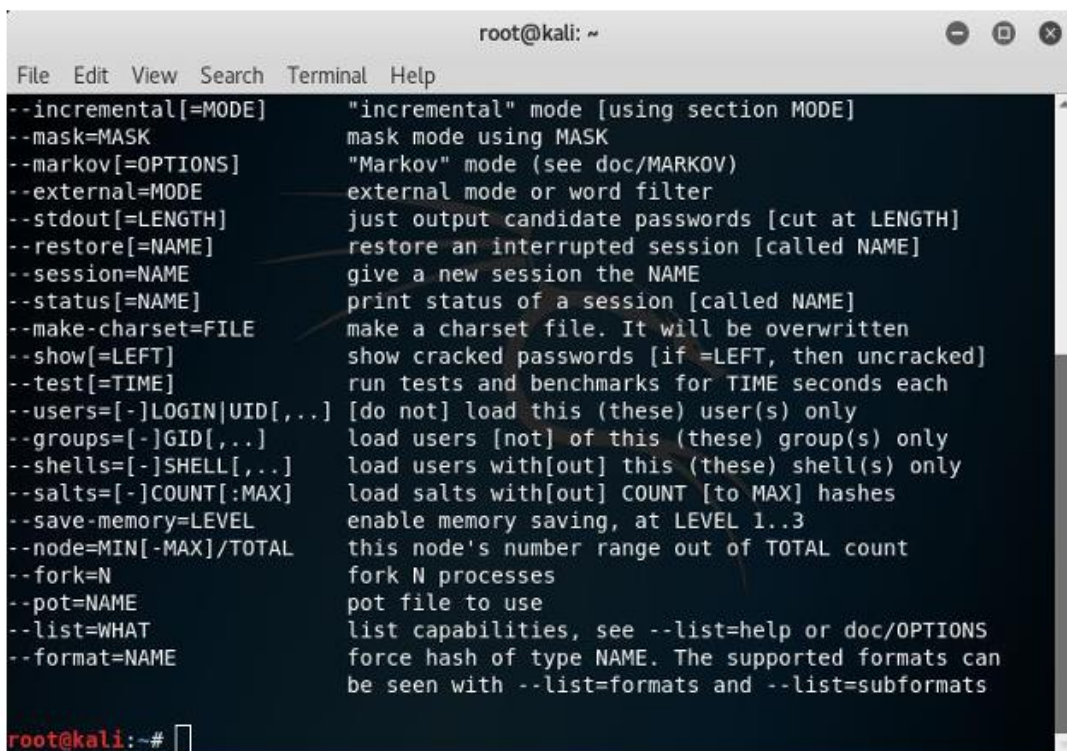


Step 2: In Kali Linux operating system, open John-the-ripper tool. Go to Applications-> Password attacks-> john



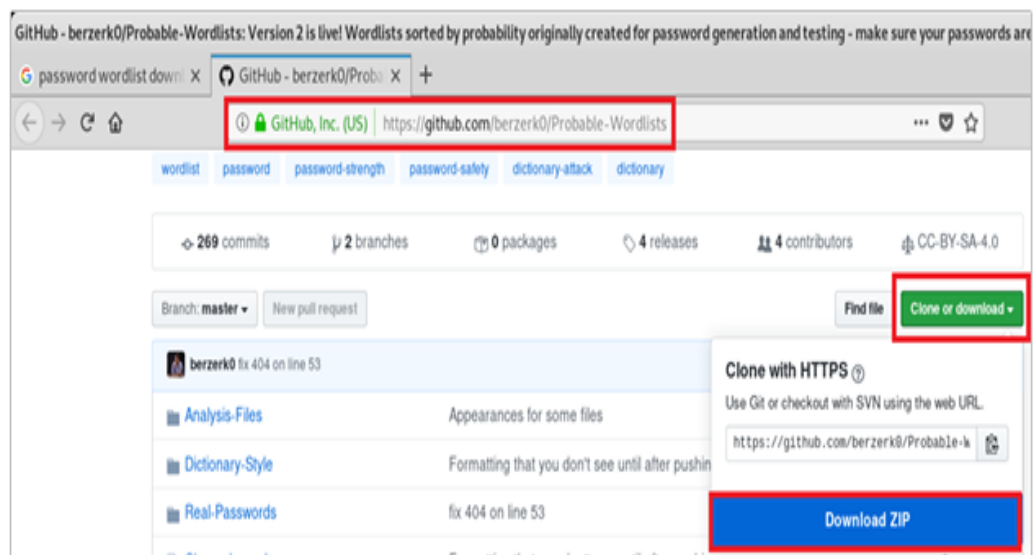
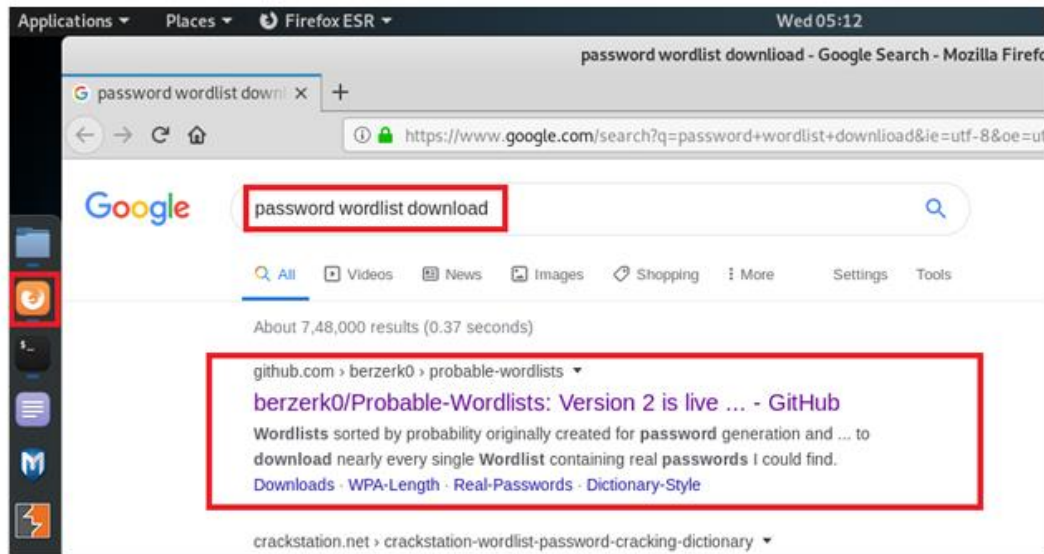
Step 3: A terminal with usage of John-the-ripper tool will open

```
root@kali: ~  
File Edit View Search Terminal Help  
John the Ripper password cracker, version 1.8.0.6-jumbo-1-bleeding [linux-x86-64  
-avx]  
Copyright (c) 1996-2015 by Solar Designer and others  
Homepage: http://www.openwall.com/john/  
  
Usage: john [OPTIONS] [PASSWORD-FILES]  
--single[=SECTION]      "single crack" mode  
--wordlist[=FILE] --stdin wordlist mode, read words from FILE or stdin  
                        --pipe  like --stdin, but bulk reads, and allows rules  
                        like --wordlist, but fetch words from a .pot file  
--loopback[=FILE]      suppress all dupes in wordlist (and force preload)  
--dupe-suppression      PRINCE mode, read words from FILE  
--prince[=FILE]         input encoding (eg. UTF-8, ISO-8859-1). See also  
--encoding=NAME         doc/ENCODING and --list=hidden-options.  
--rules[=SECTION]      enable word mangling rules for wordlist modes  
--incremental[=MODE]    "incremental" mode [using section MODE]  
--mask=MASK             mask mode using MASK  
--markov[=OPTIONS]      "Markov" mode (see doc/MARKOV)  
--external=MODE         external mode or word filter  
--stdout[=LENGTH]      just output candidate passwords [cut at LENGTH]  
--restore[=NAME]        restore an interrupted session [called NAME]  
--session=NAME          give a new session the NAME  
--status[=NAME]         print status of a session [called NAME]  
--make-charset=FILE     make a charset file. It will be overwritten
```

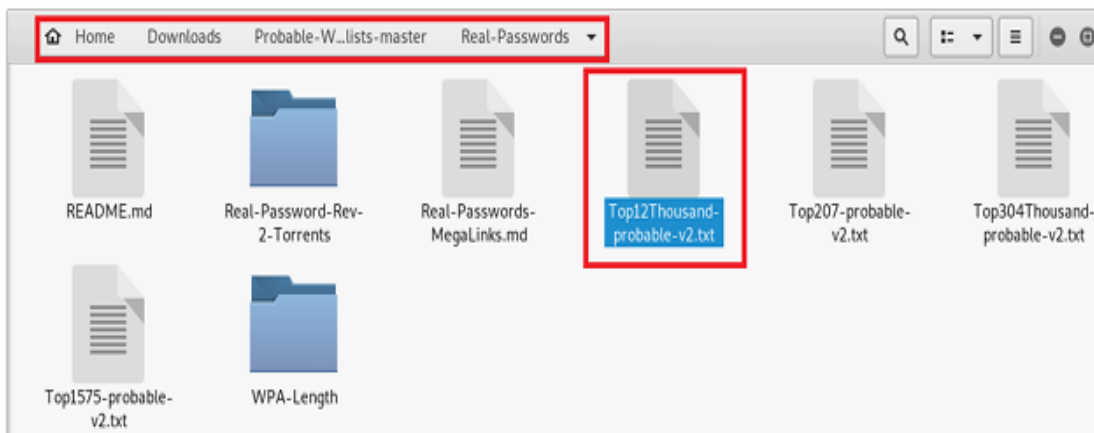
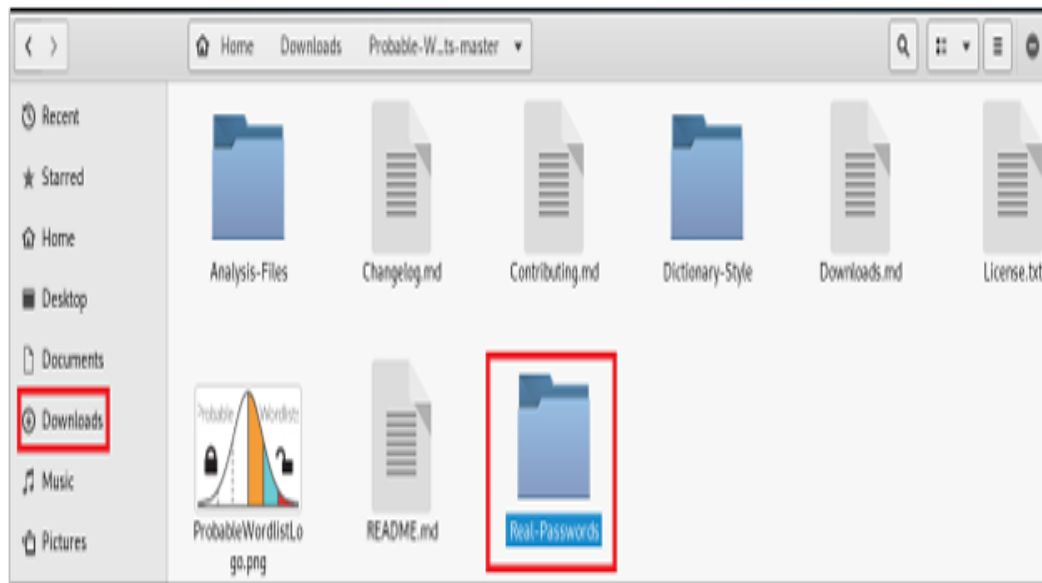


```
root@kali: ~  
File Edit View Search Terminal Help  
--incremental[=MODE]    "incremental" mode [using section MODE]  
--mask=MASK             mask mode using MASK  
--markov[=OPTIONS]      "Markov" mode (see doc/MARKOV)  
--external=MODE         external mode or word filter  
--stdout[=LENGTH]      just output candidate passwords [cut at LENGTH]  
--restore[=NAME]        restore an interrupted session [called NAME]  
--session=NAME          give a new session the NAME  
--status[=NAME]         print status of a session [called NAME]  
--make-charset=FILE     make a charset file. It will be overwritten  
--show[=LEFT]          show cracked passwords [if =LEFT, then uncracked]  
--test[=TIME]           run tests and benchmarks for TIME seconds each  
--users=[-]LOGIN|UID[,..] [do not] load this (these) user(s) only  
--groups=[-]GID[,...]   load users [not] of this (these) group(s) only  
--shells=[-]SHELL[,...] load users with[out] this (these) shell(s) only  
--salts=[-]COUNT[:MAX] load salts with[out] COUNT [to MAX] hashes  
--save-memory=LEVEL     enable memory saving, at LEVEL 1..3  
--node=MIN[-MAX]/TOTAL this node's number range out of TOTAL count  
--fork=N               fork N processes  
--pot=NAME             pot file to use  
--list=WHAT            list capabilities, see --list=help or doc/OPTIONS  
--format=NAME          force hash of type NAME. The supported formats can  
                        be seen with --list=formats and --list=subformats  
  
root@kali:~#
```

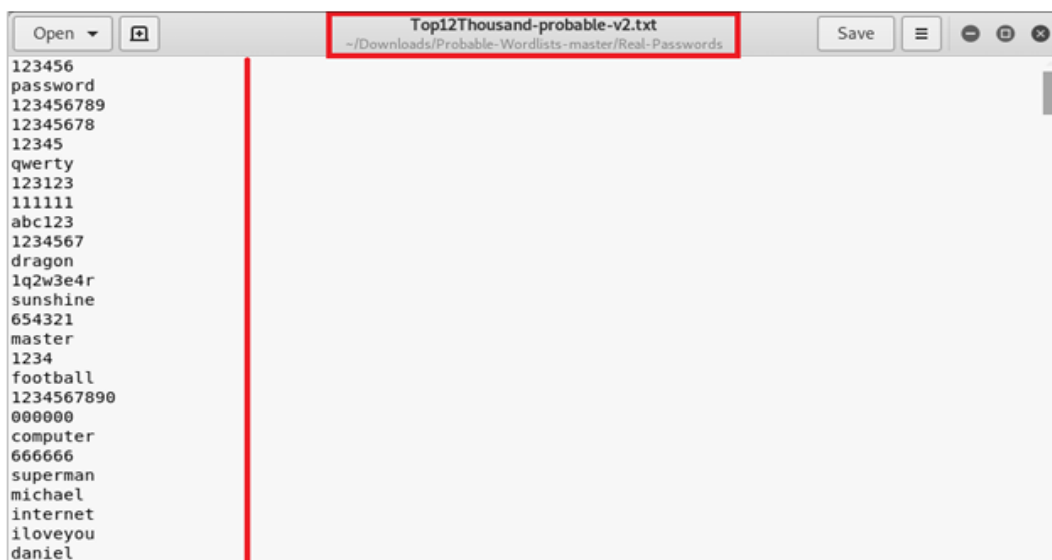
Step 4: Search the password wordlist by browsing Google search engine. Open the GitHub website and download the ZIP file

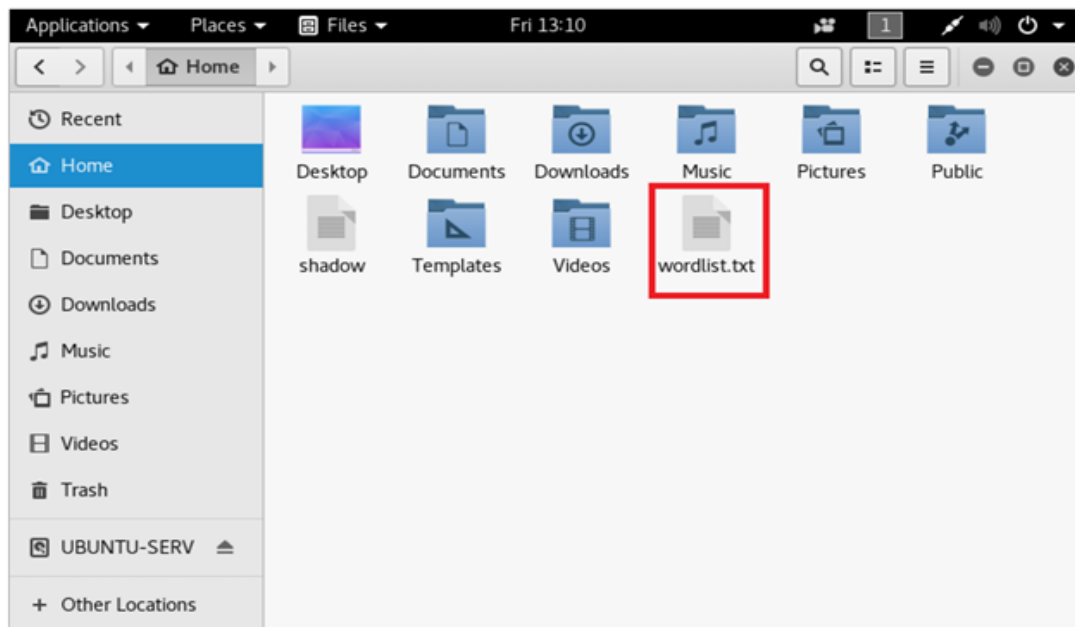


Step 5: Save and open the downloaded file. Open the “Real-Passwords” folder to see the passwords wordlist.



Step 6: Open any password wordlist (e.g., Top12Thousand probable-v2.txt file), Copy this file in Home directory and rename as “wordlist.txt”.





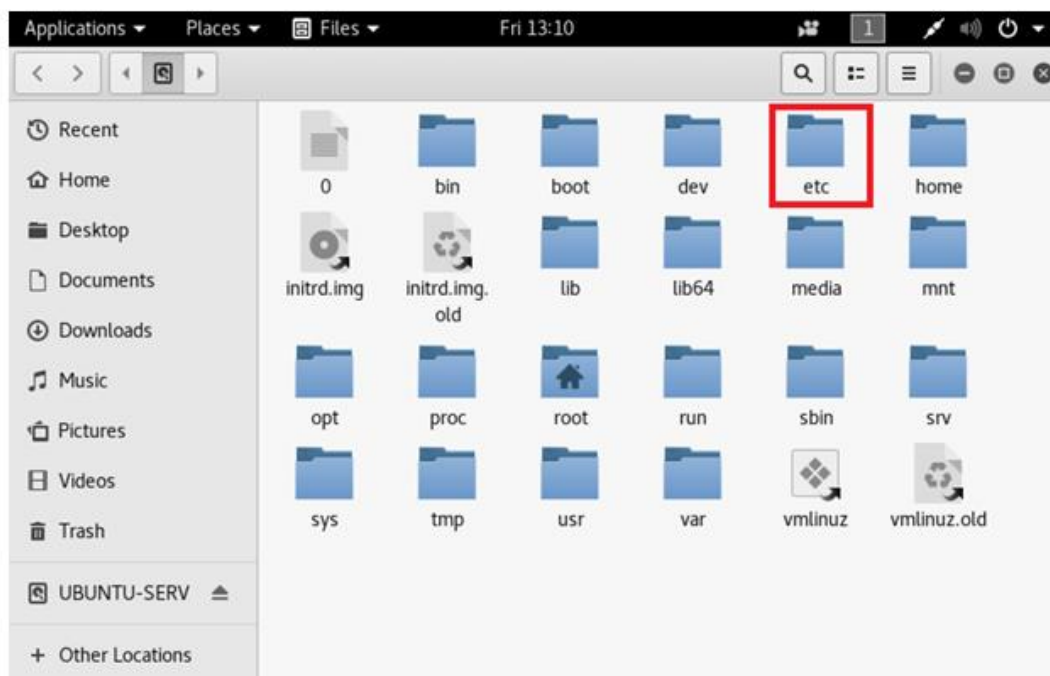
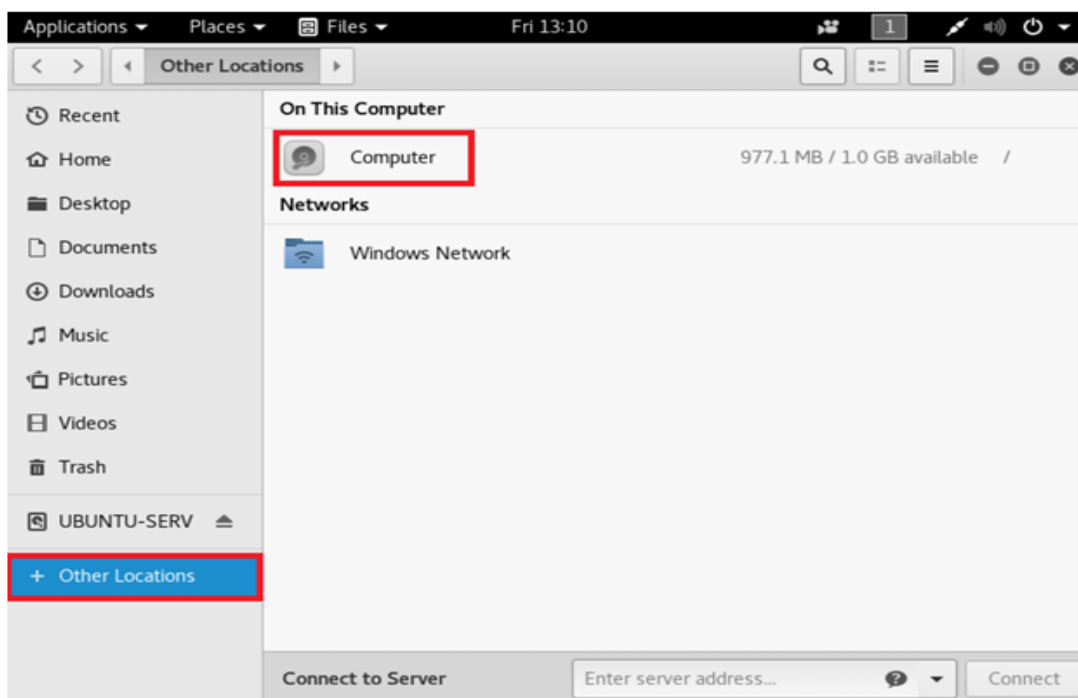
Step 7: Add new users in kali Linux operating system. Set a password and press 'Y' while creating new users.

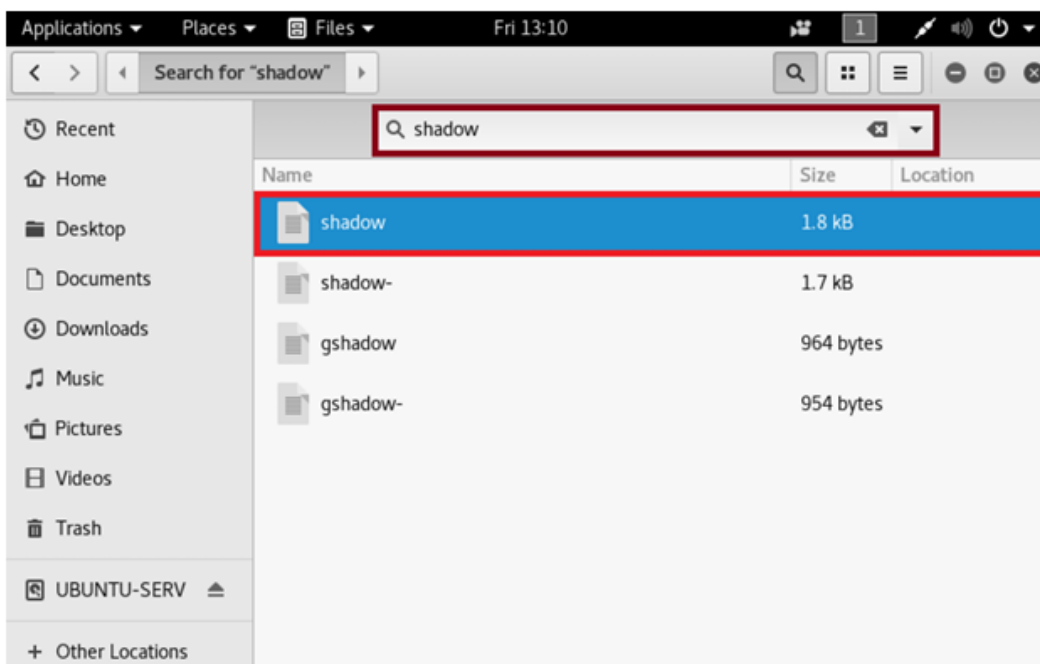
```
root@kali: ~  
File Edit View Search Terminal Help  
--session=NAME          give a new session the NAME  
--status[=NAME]         print status of a session [called NAME]  
--make-charset=FILE      make a charset file. It will be overwritten  
--show[=LEFT]           show cracked passwords [if =LEFT, then uncracked]  
--test[=TIME]           run tests and benchmarks for TIME seconds each  
--users=[-]LOGIN|UID[,..] [do not] load this (these) user(s) only  
--groups=[-]GID[,..]    load users [not] of this (these) group(s) only  
--shells=[-]SHELL[,..] load users with[out] this (these) shell(s) only  
--salts=[-]COUNT[:MAX] load salts with[out] COUNT [to MAX] hashes  
--save-memory=LEVEL     enable memory saving, at LEVEL 1..3  
--node=MIN[-MAX]/TOTAL this node's number range out of TOTAL count  
--fork=N               fork N processes  
--pot=NAME             pot file to use  
--list=WHAT            list capabilities, see --list=help or doc/OPTIONS  
--format=NAME          force hash of type NAME. The supported formats can  
                        be seen with --list=formats and --list=subformats  
  
root@kali:~# adduser shweta  
Adding user `shweta' ...  
Adding new group `shweta' (1000) ...  
Adding new user `shweta' (1000) with group `shweta' ...  
Creating home directory `/home/shweta' ...  
Copying files from `/etc/skel' ...  
Enter new UNIX password: 
```

```
root@kali: ~  
File Edit View Search Terminal Help  
--fork=N          fork N processes  
--pot=NAME        pot file to use  
--list=WHAT       list capabilities, see --list=help or doc/OPTIONS  
--format=NAME     force hash of type NAME. The supported formats can  
                  be seen with --list=formats and --list=subformats  
  
root@kali:~# adduser shweta  
Adding user `shweta' ...  
Adding new group `shweta' (1000) ...  
Adding new user `shweta' (1000) with group `shweta' ...  
Creating home directory `/home/shweta' ...  
Copying files from `/etc/skel' ...  
Enter new UNIX password:  
Retype new UNIX password:  
passwd: password updated successfully  
Changing the user information for shweta  
Enter the new value, or press ENTER for the default  
    Full Name []:  
    Room Number []:  
    Work Phone []:  
    Home Phone []:  
    Other []:  
Is the information correct? [Y/n] Y  
root@kali:~#
```

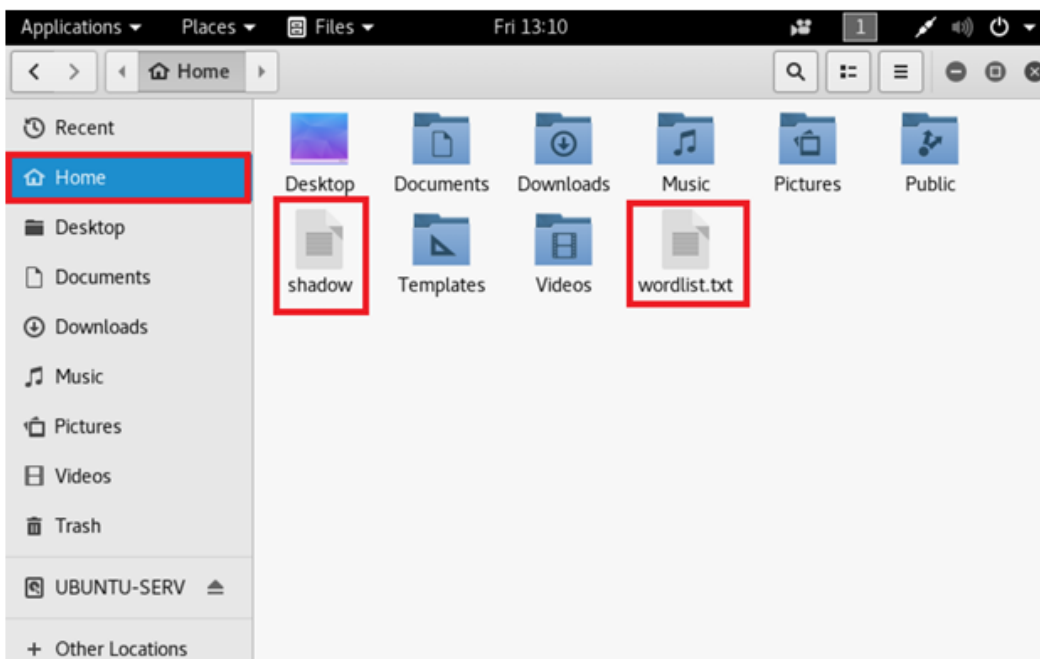
```
root@kali: ~  
File Edit View Search Terminal Help  
    Full Name []:  
    Room Number []:  
    Work Phone []:  
    Home Phone []:  
    Other []:  
Is the information correct? [Y/n] Y  
root@kali:~# adduser elite  
Adding user `elite' ...  
Adding new group `elite' (1001) ...  
Adding new user `elite' (1001) with group `elite' ...  
Creating home directory `/home/elite' ...  
Copying files from `/etc/skel' ...  
Enter new UNIX password:  
Retype new UNIX password:  
passwd: password updated successfully  
Changing the user information for elite  
Enter the new value, or press ENTER for the default  
    Full Name []:  
    Room Number []:  
    Work Phone []:  
    Home Phone []:  
    Other []:  
Is the information correct? [Y/n] Y  
root@kali:~#
```

Step 8: Go to Other Locations->Computer->etc folder to find the shadow file.





Step 9: Copy the shadow file and paste in Home directory.



Step 10: Rename the shadow file as shadow1 and open the file to find the usernames and password in the form of hash values.

```

Applications ▾ Places ▾ Text Editor ▾ Fri 13:12
Open ▾ shadow1 Save
root:X014elvznJq7E:18362:0:99999:7:::
daemon:!:17426:0:99999:7:::
bin:!:17426:0:99999:7:::
sys:!:17426:0:99999:7:::
sync:!:17426:0:99999:7:::
games:!:17426:0:99999:7:::
man:!:17426:0:99999:7:::
lp:!:17426:0:99999:7:::
mail:!:17426:0:99999:7:::
news:!:17426:0:99999:7:::
uucp:!:17426:0:99999:7:::
proxy:!:17426:0:99999:7:::
www-data:!:17426:0:99999:7:::
backup:!:17426:0:99999:7:::
list:!:17426:0:99999:7:::
irc:!:17426:0:99999:7:::
gnats:!:17426:0:99999:7:::
nobody:!:17426:0:99999:7:::
systemd-timesync:!:17426:0:99999:7:::
systemd-network:!:17426:0:99999:7:::
systemd-resolve:!:17426:0:99999:7:::
systemd-bus-proxy:!:17426:0:99999:7:::
_apt:!:17426:0:99999:7:::
mysql:!:17426:0:99999:7:::
epmd:!:17426:0:99999:7:::
Debian-exim:!:17426:0:99999:7:::
uuidd:!:17426:0:99999:7:::
rwhod:!:17426:0:99999:7:::

Applications ▾ Places ▾ Text Editor ▾ Fri 13:12
Open ▾ shadow1 Save
usbmux:!:17426:0:99999:7:::
miredo:!:17426:0:99999:7:::
Debian-snmp:!:17426:0:99999:7:::
ntp:!:17426:0:99999:7:::
stunnel4:!:17426:0:99999:7:::
rtkit:!:17426:0:99999:7:::
postgres:!:17426:0:99999:7:::
dnsmasq:!:17426:0:99999:7:::
messagebus:!:17426:0:99999:7:::
iodine:!:17426:0:99999:7:::
arpwatch:!:17426:0:99999:7:::
sslh:!:17426:0:99999:7:::
gluster:!:17426:0:99999:7:::
couchdb:!:17426:0:99999:7:::
avahi:!:17426:0:99999:7:::
sshd:!:17426:0:99999:7:::
colord:!:17426:0:99999:7:::
saned:!:17426:0:99999:7:::
speech-dispatcher:!:17426:0:99999:7:::
pulse:!:17426:0:99999:7:::
Debian-gdm:!:17426:0:99999:7:::
king-phisher:!:17426:0:99999:7:::
dradis:!:17426:0:99999:7:::
beef-xss:!:17426:0:99999:7:::
shweta:$6$3vszJprt$LYzRq02e6VWPRaQaegxVjKLvo70LMerPVBxE/
B3fbdo41VGyz76FLGwdU1Txqx8gWRxQpjPK0k53YKUzWQ87V1:18362:0:99999:7:::
elite:$6$UBGII4uM$DGLiZkHW0gttXEHf3Hkkt/FznYy5wWX4DVUy6.3spHcm0rI.0UDBU7Vcnjm5wYUXdrqA/
rBuEVBikM0rvDqLE.:18362:0:99999:7:::

```

Step 11: Write the command “john --wordlist=/root/ wordlist.txt” to recover the hash of root and “john --show shadow1” to display the passwords in plaintext.

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# john --wordlist=/root/wordlist.txt shadow1  
Warning: only loading hashes of type "descript", but also saw type "sha512crypt"  
Use the "--format=sha512crypt" option to force loading hashes of that type instead  
Warning: only loading hashes of type "descript", but also saw type "crypt"  
Use the "--format=crypt" option to force loading hashes of that type instead  
Using default input encoding: UTF-8  
Loaded 1 password hash (descript, traditional crypt(3) [DES 128/128 AVX-16])  
Press 'q' or Ctrl-C to abort, almost any other key for status  
toor (root)  
lg 0:00:00:00 DONE (2020-04-10 13:18) 50.00g/s 1580Kp/s 1580Kc/s 1580Kc/s trippi  
n..seismic  
Use the "--show" option to display all of the cracked passwords reliably  
Session completed  
root@kali:~# john --show shadow1  
root:toor:18362:0:99999:7:::  
1 password hash cracked, 2 left
```

Step 12: Write the command “john --wordlist= /root/wordlist.txt --format=sha512crypt” to recover the hash of other users and “john --show shadow1” to display the passwords in plaintext. The passwords in plaintext are displayed and highlighted in red rectangular box.

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# john --wordlist=/root/wordlist.txt shadow1 --format=sha512crypt  
Using default input encoding: UTF-8  
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3) $6$ [SHA5  
12 128/128 AVX 2x])  
Press 'q' or Ctrl-C to abort, almost any other key for status  
password123 (shweta)  
test4echo (elite)  
2g 0:00:06:44 DONE (2020-04-10 13:27) 0.004941g/s 593.8p/s 595.4c/s 595.4C/s tet  
rameric..tesseract  
Use the "--show" option to display all of the cracked passwords reliably  
Session completed  
root@kali:~# john --show shadow1  
root:toor:18362:0:99999:7:::  
shweta:password123:18362:0:99999:7:::  
elite:test4echo:18362:0:99999:7:::  
3 password hashes cracked, 0 left  
root@kali:~#  
root@kali:~#
```

IX. Precautions

1. Use password recovery tools only on authorized systems.
2. Do not share recovered passwords.
3. Ensure system backup before applying password recovery.
4. Avoid weak passwords to reduce the risk of unauthorized access.

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Difference between single-level and multi-level authentication.
2. Steps to recover passwords using John the Ripper.
3. Describe the risks of weak passwords.
4. Why should password recovery be restricted to authorized administrators?

Space for Answer

[illegible]

.....

.....

.....

.....

.....

.....

.....

.....

.....

XI. Exercise

1. Configure password complexity policy in Windows/Linux.
2. Use a different open-source tool (e.g., Ophcrack) for password recovery.
3. Compare recovery time between dictionary attack and brute force attack.

XII. References / Suggestions for further reading

1. Stallings, W. & Brown, L., 2014. Computer Security: Principles and Practice. Pearson. Publisher link
2. John the Ripper Documentation: <https://www.openwall.com/john>
3. NPTEL: Introduction to Information Security.

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Following correct authentication setup procedure	20 %
2	Recovering password using open-source tool	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Quality of output (System secured & password recovered)	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 03: i. Grant security to file, folder or application using access permissions and verify it
ii. Grant access permission while sharing file and folder

I. Practical Significance

This practical demonstrates how to secure files, folders, and applications by assigning proper access permissions. Access control ensures that only authorized users can read, modify, or execute resources. Proper sharing and permission management prevents unauthorized access, accidental modifications, and data breaches in enterprise and multi-user environments.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO2: Apply multi-factor user authentication and access control.

IV. Laboratory Learning Outcome (LLO)

LLO 3.1: Grant read, write, and execute permission on file and folder.

V. Relevant Affective domain related Outcomes(s)

- Responsibility: Maintaining integrity of organizational data.
- Attention to Detail: Carefully assigning and verifying permissions.
- Ethical Awareness: Avoiding misuse of administrative rights.

VI. Theoretical Background

Access control mechanisms in operating systems allow administrators to define which users or groups can access specific resources. In both Windows and Linux, permissions are fundamental to maintaining data confidentiality, integrity, and availability.

In Windows, every file and folder has its own set of permissions defined through Access Control Lists (ACLs). These permissions can be inherited from parent folders, typically involving users such as SYSTEM, the logged-in account, and the Administrators group. Windows supports six primary permission types: Full Control, Modify, Read & Execute, List Folder Contents, Read, and Write. Together, NTFS permissions and sharing permissions regulate both local and network access.

In Linux, file permissions follow a similar principle but are represented using symbolic notation (r, w, x for read, write, and execute) or octal values (e.g., 755). These permissions apply to three categories of users: owner, group, and others.

By combining local permissions with shared folder access control, operating systems ensure secure collaboration while preventing unauthorized access and safeguarding system resources.

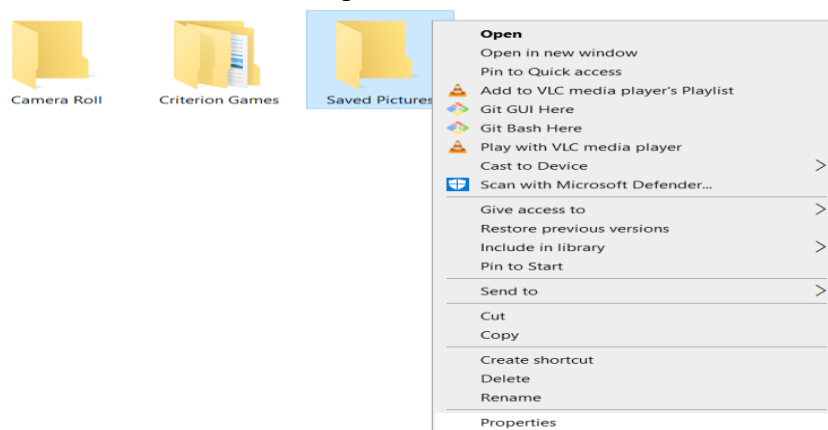
VII. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	File Explorer (Windows),		

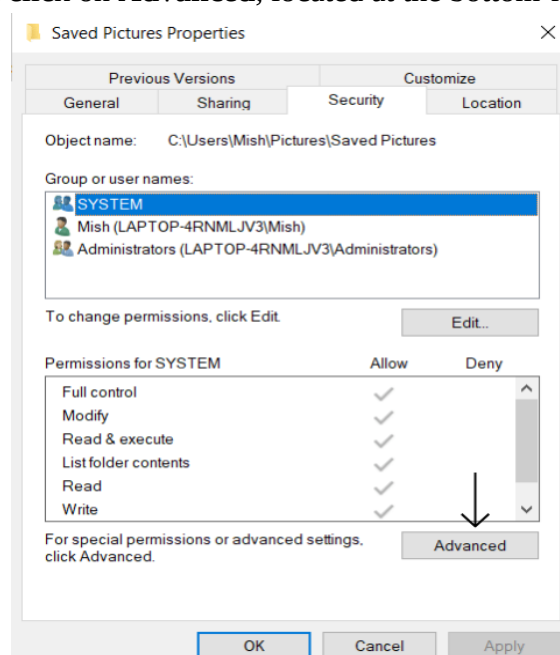
VIII. Procedure

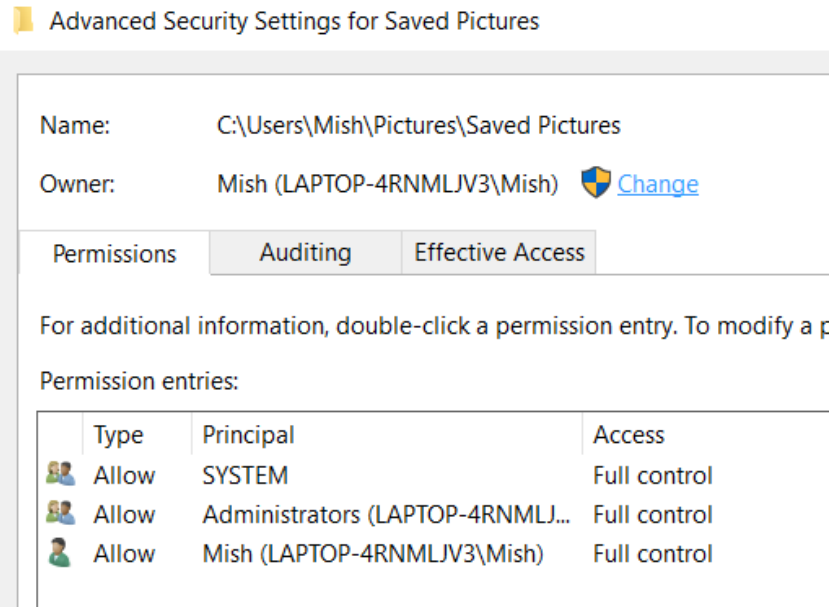
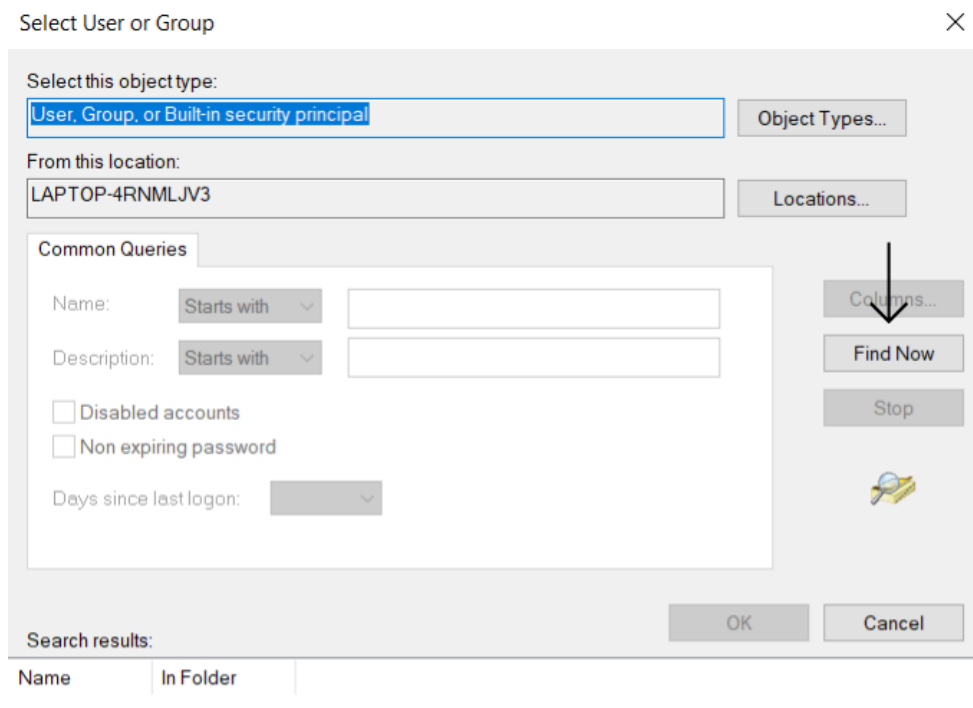
Apply security to file folder

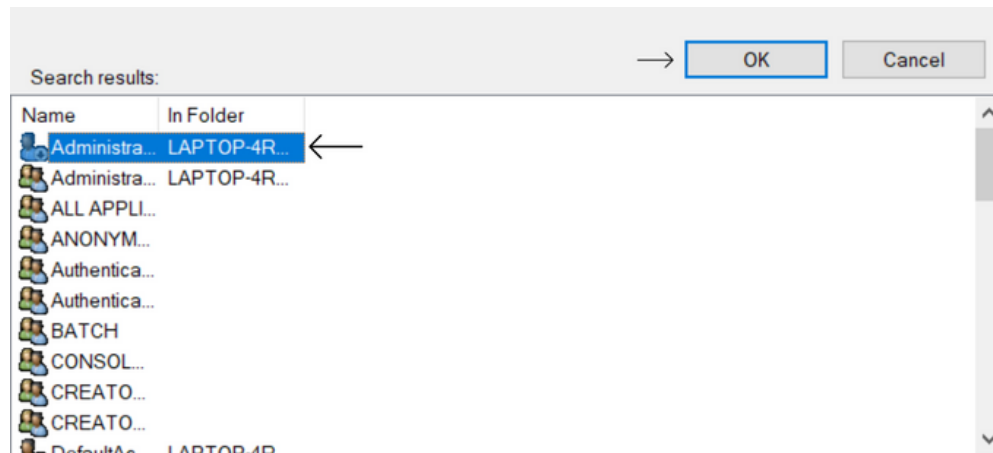
1. Right click the folder or file and click “Properties” in the context menu.



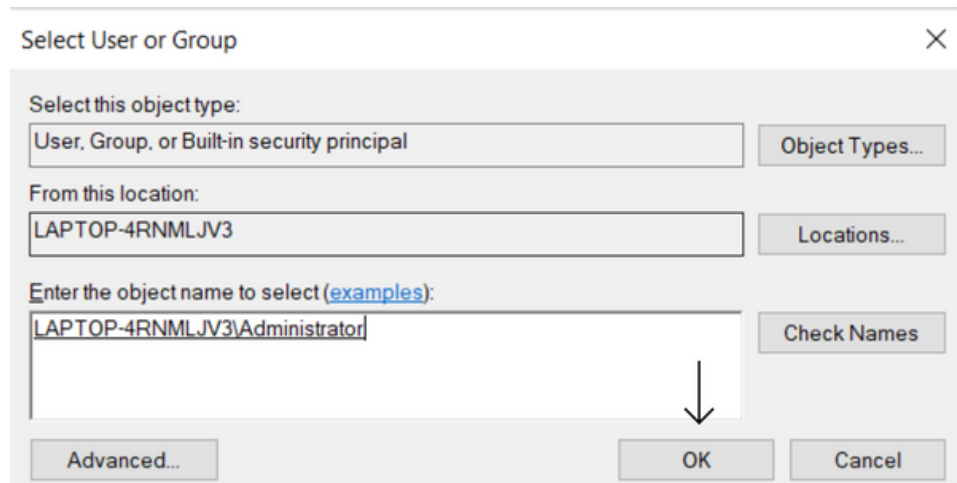
2. Go to the **Security** tab and click on **Advanced**, located at the bottom-right corner of your screen.



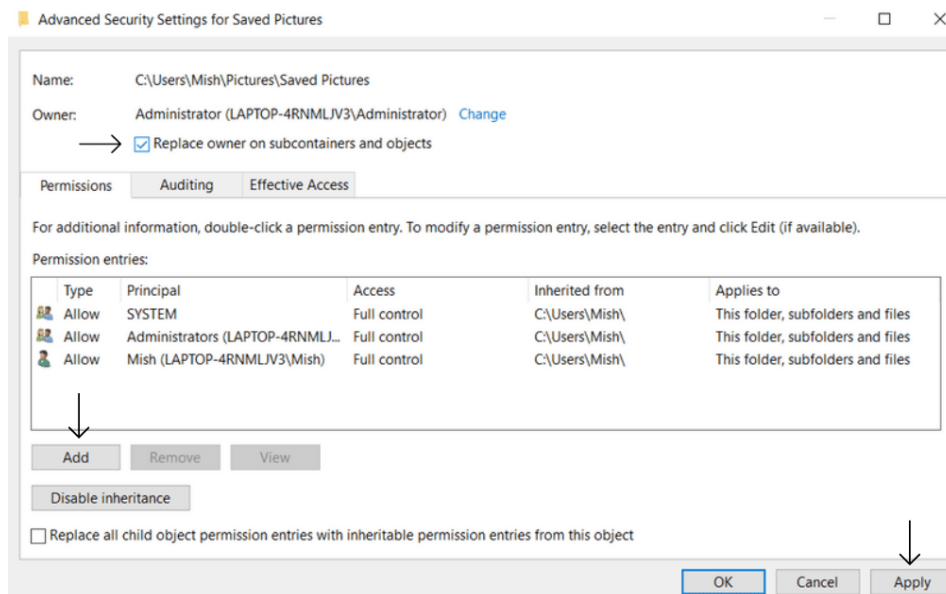
3. Click on **Change**.4. Click on **Find Now**.5. Select your username and hit **OK**.



6. Click **OK**.



- Check the box that says **Replace owner on sub containers and objects** and click on **Apply**. Now, you need to repeat **steps 1-3** and reach this page again.
- This time around, click on the **Add** button on your screen.



7. Click on **Select a principal**.

Permission Entry for Saved Pictures

Principal: [Select a principal](#) ←

Type: Allow

Applies to: This folder, subfolders and files

Basic permissions:

- ☐ Full control
- ☐ Modify
- ☒ Read & execute
- ☒ List folder contents
- ☒ Read
- ☐ Write
- ☐ Special permissions

☐ Only apply these permissions to objects and/or containers within this container

8. Check the box that says **Full control**. Then, hit **OK**.

Permission Entry for Saved Pictures

Principal: Administrator (LAPTOP-4RNMLJ3\Administrator) [Select a principal](#)

Type: Allow

Applies to: This folder, subfolders and files

Basic permissions:

- ☐ Full control
- ☐ Modify
- ☒ Read & execute
- ☒ List folder contents
- ☒ Read
- ☐ Write
- ☐ Special permissions

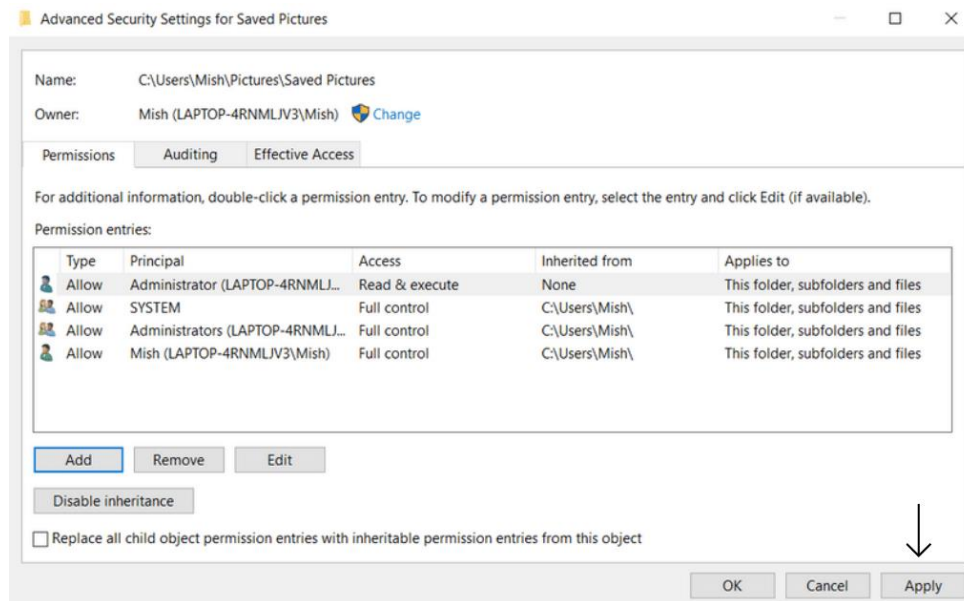
☐ Only apply these permissions to objects and/or containers within this container

[Show advanced permissions](#)

[Clear all](#)

[OK](#) [Cancel](#)

9. Click on **Apply**.



Procedure of Folder sharing:

Step 1: Select the Folder or File

Click the folder you want to share.

If you want to share a subfolder or a file instead, open its parent folder and then select the subfolder or file.

Step 2: Launch the File Sharing Wizard

Click the **Share** button in the task pane.

Windows Vista launches the **File Sharing Wizard**, which asks you to choose the user accounts you want to share the item with.

Step 3: Add Users

Type the username and click **Add**.

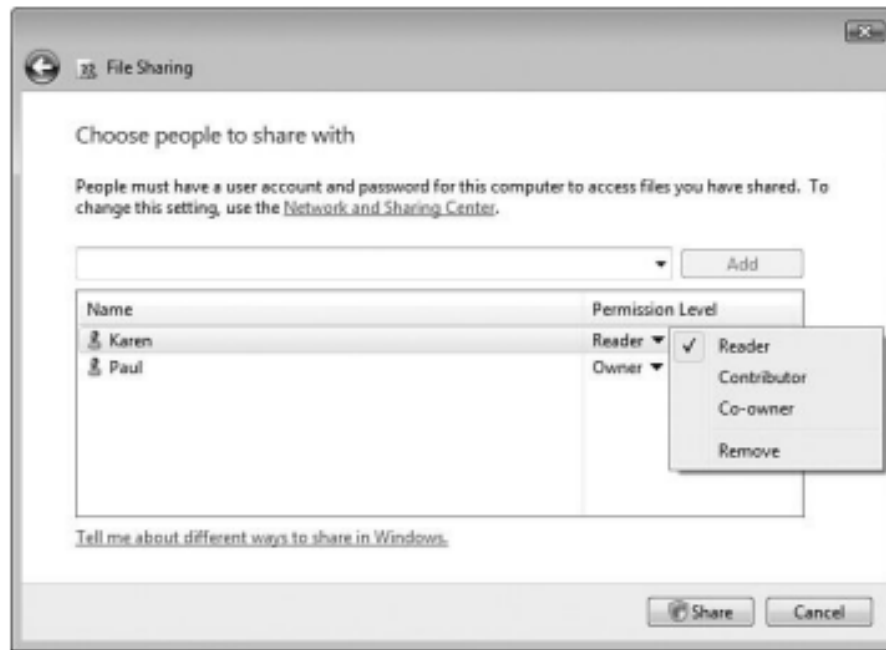
Step 4: Repeat for Additional Users

Repeat step 4 to share the folder or file with other users.

Step 5: Assign Permission Levels

For each user added, assign a permission level by clicking the **downward arrow** and selecting one of the following:

- **Reader** – Default level; user can view and open contents but cannot create, change, or delete anything.
- **Contributor** – User can add new files to the shared folder, and modify or delete only the files they added.
- **Co-owner** – User can create new items, and modify or delete any items in the folder.



IX. Precautions

1. Always assign the minimum required permissions (principle of least privilege).
2. Do not provide full control to unauthorized users.
3. Test permissions after configuration to avoid access errors.
4. Avoid sharing sensitive files without encryption.

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Differentiate between NTFS permissions and Share permissions.
2. How do Linux file permissions differ from Windows file permissions?
3. Why is the principle of least privilege important in access control?

Space for Answer

.....

.....

.....

.....

.....

.....

.....

.....

This image shows a full page of white paper with horizontal dotted lines. The lines are evenly spaced and run across the width of the page, providing a guide for handwriting practice. There are no margins, text, or other markings on the page.

XI. Exercise

1. Assign read-only permission to a folder for a user and verify.
2. Share a folder on a Windows system and access it from another system.

XII. References / Suggestions for further reading

1. Stallings, W. & Brown, L., 2014. Computer Security: Principles and Practice. Pearson. Publisher link
2. Administrator Guide Quick Heal Endpoint Security 5.2. Available at: https://dlupdate.quickheal.com/documents/manual/eps5.2_userguide.pdf
3. YouTube, 2020. Firewall Configuration Tutorial. Available at: <https://www.youtube.com/watch?v=T9c5ZpT2FV0>
4. NPTEL, 2022. Introduction to Information Security. Available at: <https://archive.nptel.ac.in/courses/106/106/106106129/>
5. Virtual Labs, IIIT Hyderabad, n.d. Cryptography Experiments. Available at: <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Following correct procedure to assign file/folder/application permissions	20 %
2	Configuring and verifying share permissions on network	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Quality of output (Permissions correctly applied, verified, and secure sharing working)	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 04: Write a utility using C/Shell programming to create strong password authentication (Password should be more than 8 characters, and combination of digits, letters and special characters #, %, &, @)

I. Practical Significance

These practical helps students understand the importance of password complexity in authentication. Weak passwords make systems vulnerable to attacks such as brute-force, dictionary attacks, and social engineering. By writing a C program that enforces password policies, learners gain practical knowledge of implementing authentication systems that meet security standards.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO2: Apply multi-factor user authentication and access control.

IV. Laboratory Learning Outcome (LLO)

LLO 4.1: Implement password authentication.

V. Relevant Affective domain related Outcomes(s)

- Responsibility: Creating secure password authentication utilities.
- Attention to Detail: Enforcing rules correctly.
- Ethical Awareness: Protecting users from weak security practices.

VI. Theoretical Background

A password is a secret string of characters (letters, numbers, or symbols) used to verify the identity of a user and grant access to a computer system, application, or network resource. It is the most widely used method of authentication in operating systems and network security.

Authentication is the process of verifying the identity of a user or system. Password-based authentication is the most common method, but weak passwords compromise security. A **strong password** should:

1. Be more than 8 characters long.
2. Include uppercase and lowercase letters.
3. Include digits.
4. Contain at least one special character (#, %, &, @).

In **C programming**, we can enforce these rules using standard library functions like `strlen()`, `isupper()`, `islower()`, `isdigit()`, and by scanning for special characters.

VII. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	Turbo C		

VIII. Procedure**Pseudo code to create strong password authentication****Step 1:** START**Step 2:** Initialize variable N to store the desired password length.**Step 3:** Seed the random number generator using the current system time (srand(time(NULL))) to ensure different output each time.**Step 4:** Define four character sets:

- numbers[] = "0123456789"
- lowercase_letters[] = "abcdefghijklmnopqrstuvwxyz"
- uppercase_letters[] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
- symbols[] = "!@#\$%^&*?"

Step 5: Initialize a variable randomizer with a random number between 0 and 3.**Step 6:** Repeat for each position i from 0 to N-1:

- a. If randomizer == 0, select a random character from lowercase_letters.
- b. Else if randomizer == 1, select a random character from numbers.
- c. Else if randomizer == 2, select a random character from symbols.
- d. Else if randomizer == 3, select a random character from uppercase_letters.
- e. Append the selected character to the password string.
- f. Generate a new randomizer (rand() % 4) for the next iteration.

Step 7: Display the generated password to the user.**Step 8:** STOP.**IX. Precautions**

1. Do not use visible echo for sensitive password input in real systems.
2. Avoid storing plain-text passwords.
3. Use secure hashing (e.g., SHA) in real applications.
4. Test with multiple sample passwords.

XI. Exercise

1. Enhance the program to allow password re-entry until conditions are met.
2. Implement a password policy that disallows common passwords (like "123456").
3. Modify program to hide input characters while typing.

XII. References / Suggestions for further reading

1. Stallings, W. & Brown, L., 2014. Computer Security: Principles and Practice. Pearson. Publisher link
2. Atul Kahate, *Cryptography and Network Security*. McGraw-Hill.
3. GeeksforGeeks: Password validation programs in C.
4. NPTEL, 2022. Introduction to Information Security. Available at: <https://archive.nptel.ac.in/courses/106/106/106106129/>
5. Virtual Labs, IIIT Hyderabad, Cryptography Experiments. Available at: <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Writing correct C program structure for password authentication	20 %
2	Implementing password validation rules (length, uppercase, lowercase, digit, special char)	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Quality of output (Password utility works correctly for strong/weak inputs)	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

- Practical No. 05: i. Write a C program to implement caesar cipher technique to perform encryption and decryption of text**
- ii. Apply Caesar cipher technique to perform encryption and decryption of text using any open-source tool (Example - Cryptool)**

I. Practical Significance

This practical demonstrates how classical substitution ciphers, specifically the Caesar cipher, provide a foundation for understanding modern cryptography. Although insecure by today's standards, it introduces the concepts of encryption, decryption, and key management. By applying the cipher both programmatically and via open-source tools, students gain insight into manual implementation and automated cryptographic utilities.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO3 - Implement encryption/decryption techniques.

IV. Laboratory Learning Outcome (LLO)

LLO 5.1: Implement caesar cipher encryption technique.

V. Relevant Affective domain related Outcomes(s)

1. Ethical awareness – Recognizing that weak ciphers should not be used in practice.
2. Analytical thinking – Understanding how shifts in the alphabet affect ciphertext.
3. Attention to detail – Ensuring correct key application in both encryption and decryption.

VI. Theoretical Background

Caesar Cipher : It is a mono-alphabetic cipher wherein each letter of the plaintext is substituted by another letter to form the ciphertext. It is the simplest form of a substitution cipher scheme.

This cryptosystem is generally referred to as the **Shift Cipher**. The concept is to replace each alphabet by another alphabet which is **shifted** by some fixed number between 0 and 25.

For this type of scheme, both sender and receiver agree on a *secret shift number* for shifting the alphabet. This number, between 0 and 25, becomes the **key** of encryption.

The name *Caesar Cipher* is occasionally used to describe the Shift Cipher when the **shift of three** is used.

Process of Shift Cipher:

- To encrypt a plaintext letter, the sender positions the sliding ruler underneath the first set of plaintext letters and slides it to the **LEFT** by the number of positions of the secret shift.
- The plaintext letter is then encrypted to the ciphertext letter on the sliding ruler underneath.

Encryption formula:

$$C = (P + K) \bmod 26$$

Decryption formula:

$$P = (C - K) \bmod 26$$

Where,

P = Plaintext letter,

C = Ciphertext letter,

K = Key (shift value).

Example:**Encryption:**

Plaintext = **AISSMS**

Let's choose a **shift key K = 3** (Caesar's classic).

- A $\rightarrow (0+3) \bmod 26 = 3 \rightarrow D$
- I $\rightarrow (8+3) \bmod 26 = 11 \rightarrow L$
- S $\rightarrow (18+3) \bmod 26 = 21 \rightarrow V$
- S $\rightarrow (18+3) \bmod 26 = 21 \rightarrow V$
- M $\rightarrow (12+3) \bmod 26 = 15 \rightarrow P$
- S $\rightarrow (18+3) \bmod 26 = 21 \rightarrow V$

Ciphertext = DLVVPV

Decryption:

We now take the ciphertext **DLVVPV** and apply the decryption formula:

- D $\rightarrow (3 - 3) \bmod 26 = 0 \rightarrow A$
- L $\rightarrow (11 - 3) \bmod 26 = 8 \rightarrow I$
- V $\rightarrow (21 - 3) \bmod 26 = 18 \rightarrow S$
- V $\rightarrow (21 - 3) \bmod 26 = 18 \rightarrow S$
- P $\rightarrow (15 - 3) \bmod 26 = 12 \rightarrow M$
- V $\rightarrow (21 - 3) \bmod 26 = 18 \rightarrow S$

Recovered **Plaintext = AISSMS**

VII. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	Turbo C Compiler, Open source-crytool 1		

VIII. Procedure

Caesar Cipher Implementation using C program

Step 1: START

Step 2: Display menu and get user choice

- Option 1: Encryption
- Option 2: Decryption

Step 3: Accept input text from user

Step 4: Accept shift value (key) from user

Step 5: Validate and normalize shift value

- $\text{shift} = \text{shift} \% 26$
- If shift is negative, add 26

Step 6: Check user choice:

- If choice = 1, go to ENCRYPTION STEPS
- If choice = 2, go to DECRYPTION STEPS

Step 7: Display the result (encrypted or decrypted text)

Step 8: STOP.

**Caesar cipher technique to perform encryption and decryption of text using any open-source tool
(Example – CrypTool 1.4.42)**

PART A: ENCRYPTION PROCESS

Step 1: Launch CrypTool

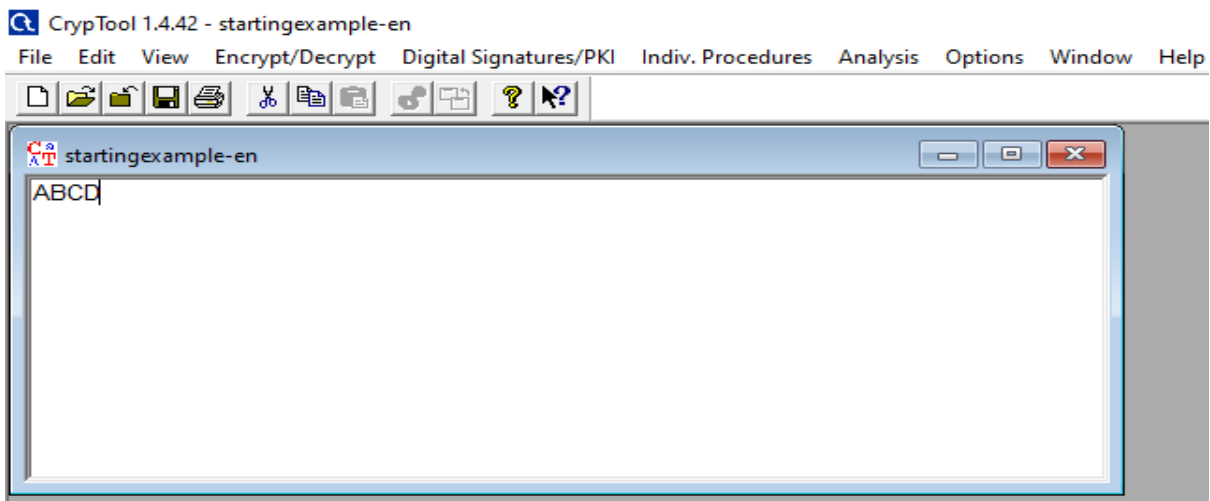
- Open **CrypTool 1.4.42** application on your computer
- Wait for the main interface to load

Step 2: Create New Document

- Click on **File** menu in the menu bar
- Select **New** to create a blank document
- A new text editor window will open

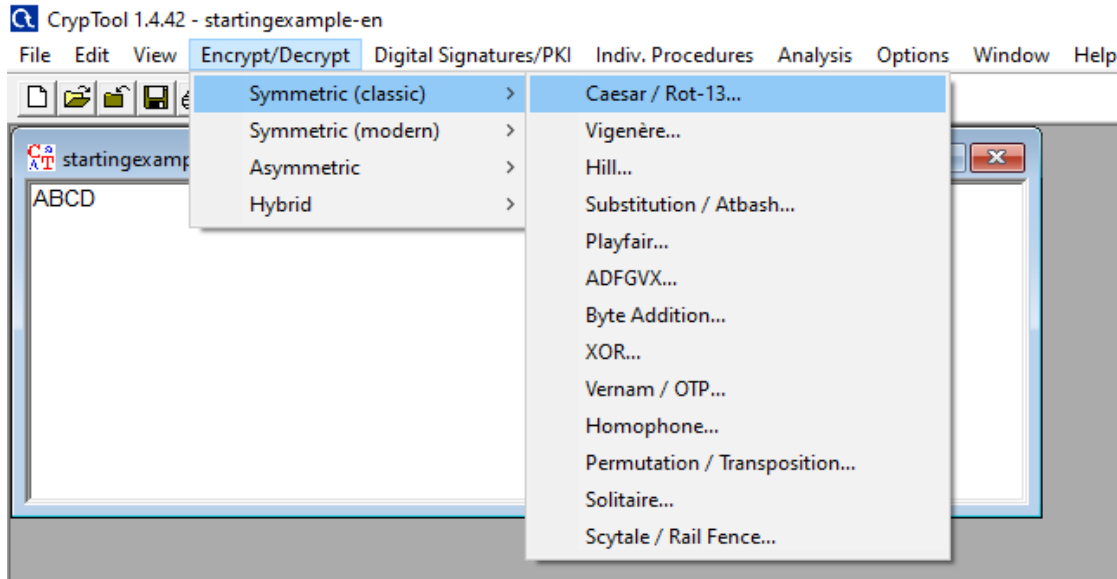
Step 3: Enter Plain Text

- In the blank document window, type your message (plain text)
- Example: "ABCD" or any text you want to encrypt
- The text can include letters, spaces, and special characters



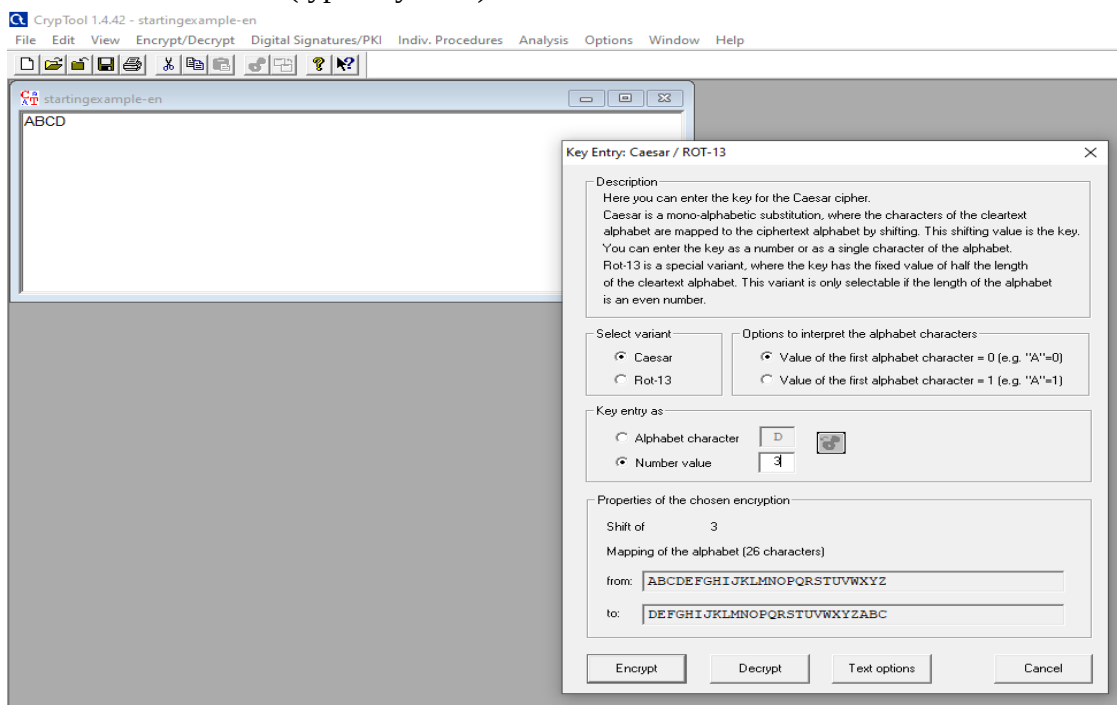
Step 4: Access Caesar Encryption

- Click on **Encrypt/Decrypt** menu from the top menu bar
- Navigate to **Symmetric (classic)**
- Select **Caesar** from the submenu
- A Caesar cipher dialog box will appear



Step 5: Configure Encryption Settings

- In the Caesar dialog box, you will see:
 - **Alphabet:** Select the character set (usually "A-Z" for English)
 - **Key (Shift Value):** Enter the shift value (e.g., 3, 5, 13, etc.)
 - **Options:** Choose encryption/decryption mode
- Set your desired shift value (typically 1-25)



Step 6: Set Encryption Mode

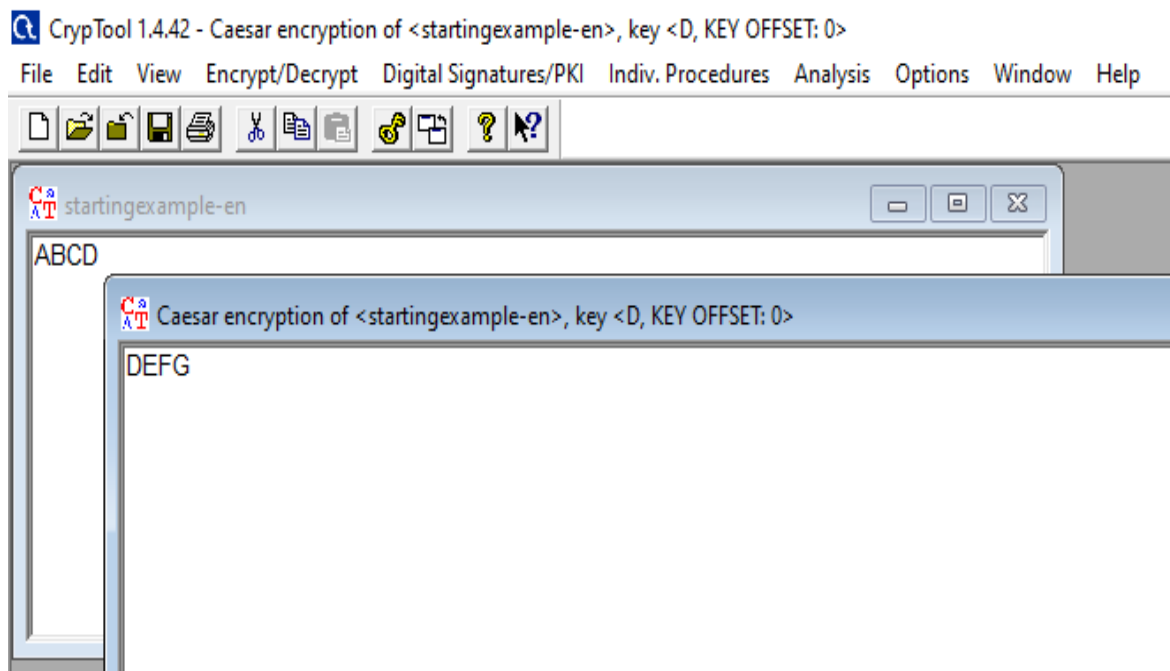
- Ensure **Encrypt** option is selected (radio button)
- Verify your shift value is correct
- Check the alphabet settings

Step 7: Execute Encryption

- Click **OK** or **Encrypt** button
- CrypTool will process your text
- A new window will open showing the encrypted text (ciphertext)

Step 8: View Encrypted Result

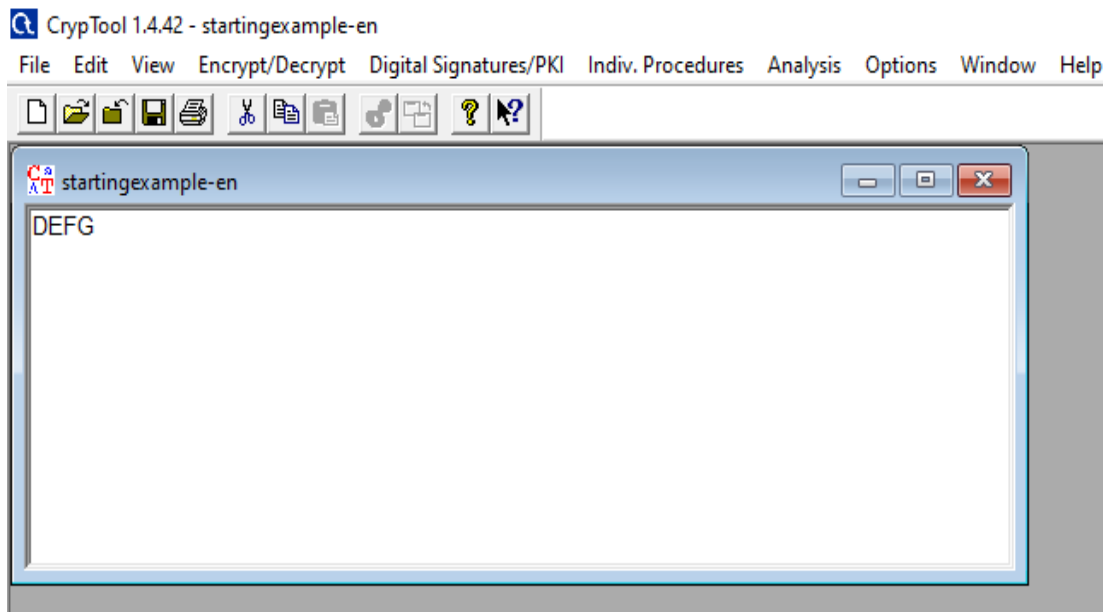
- The encrypted text will be displayed in a new document window
- The original plain text remains in the first window
- You can see the transformation applied to each character

**PART B: DECRYPTION PROCESS****Step 1: Open Encrypted Text**

- If continuing from encryption, keep the ciphertext window active
- OR open a saved encrypted file: **File** → **Open**

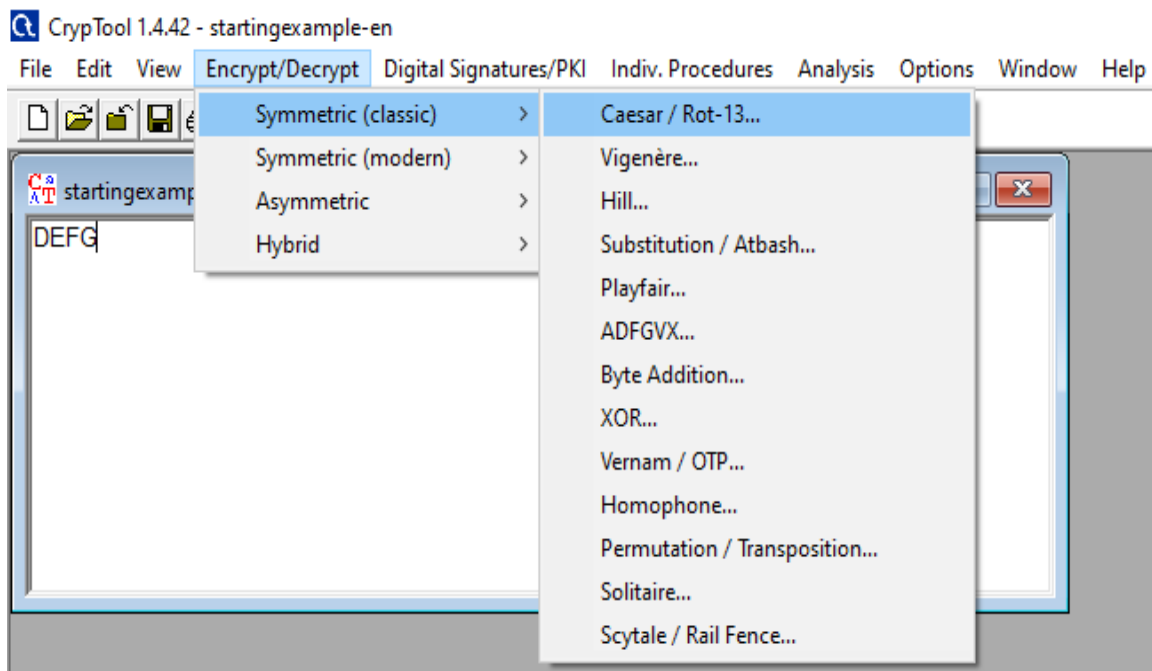
Step 2: Access Caesar Decryption

- With the encrypted text window active
- Click on **Encrypt/Decrypt** menu
- Navigate to **Symmetric (classic)** → **Caesar**
- Caesar dialog box appears again



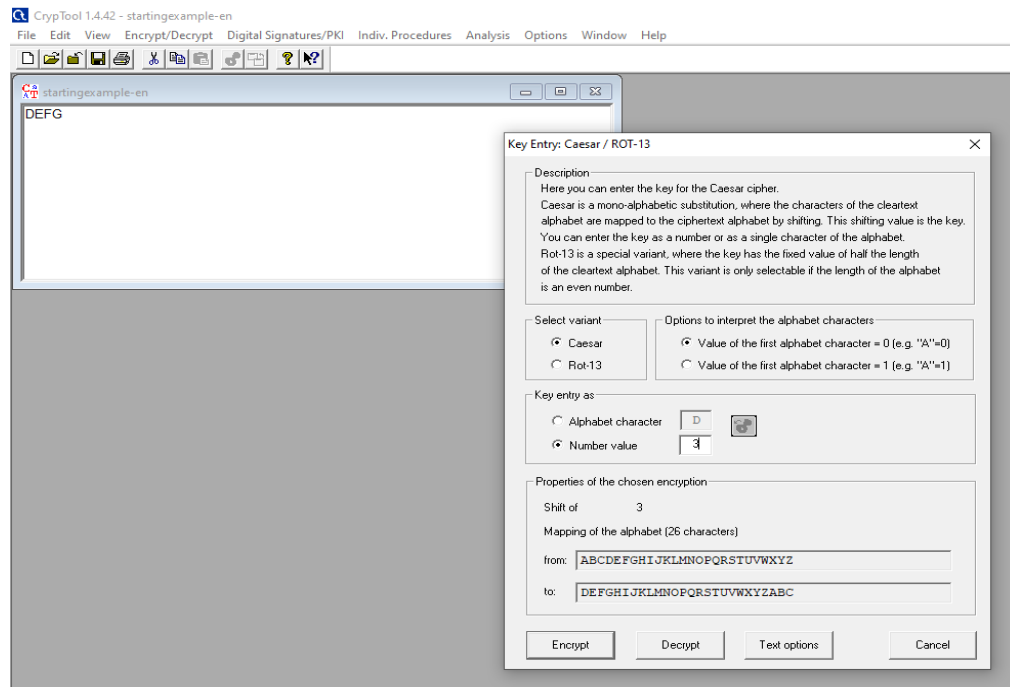
Step 3: Configure Decryption Settings

- In the Caesar dialog box:
 - **Alphabet:** Use the same alphabet as encryption
 - **Key (Shift Value):** Enter the SAME shift value used for encryption
 - **Mode:** Select **Decrypt** option (radio button)



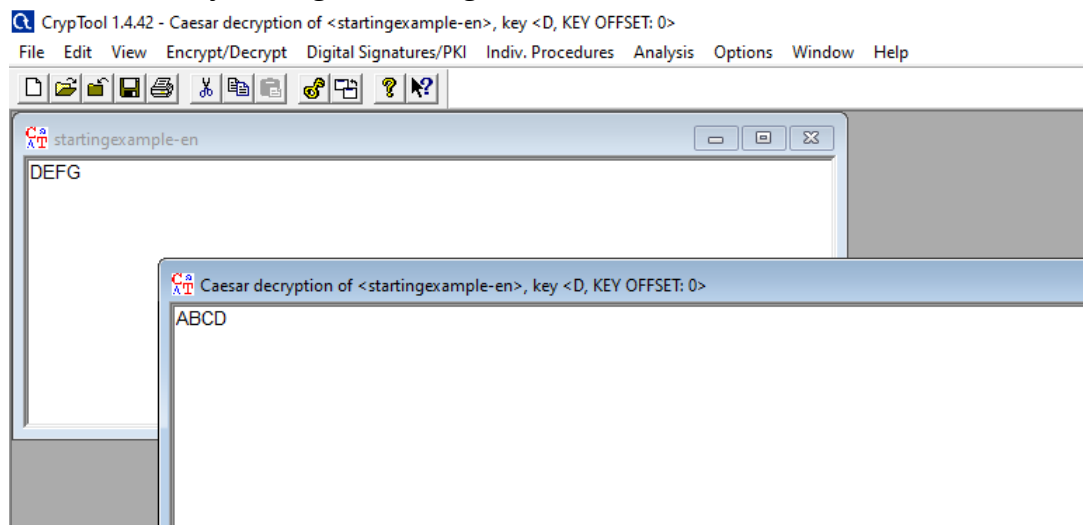
Step 4: Execute Decryption

- Verify the shift value matches the encryption key
- Click **OK** or **Decrypt** button
- CrypTool will reverse the Caesar cipher



Step 5: View Decrypted Result

- A new window opens displaying the decrypted text
- Compare with original plain text to verify correctness
- The text should match your original message



Step 6: Verify Results

- Check that decrypted text = original plain text
- If incorrect, verify the shift value was correct

IX. Precautions

1. Always use correct key during encryption and decryption.
2. Ensure input text contains only alphabets for predictable results.
3. Save original plaintext before testing to avoid data loss.
4. Do not confuse Caesar cipher with modern strong encryption (AES, RSA).

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Describe working of Caesar cipher with an example.
2. Why is Caesar cipher considered insecure?
3. Consider plain text “NETWORK” and convert given plain into cipher text.
4. Implement Caesar cipher for numeric digits in addition to alphabets
5. Compare Caesar cipher with Columnar Transposition cipher in terms of strength and weaknesses.

Space for Answer

[illegible]

.....

.....

.....

.....

.....

.....

XI. Exercise

1. Modify the C program to handle both uppercase and lowercase input correctly.
2. Extend the program to allow brute-force decryption without a key.
3. Compare Caesar cipher with Modified Caesar cipher in terms of security.

XII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. CrypTool Official Website: <https://www.cryptool.org>
4. NPTEL Course: Introduction to Cryptography and Network Security.
5. Virtual Labs, IIIT Hyderabad, n.d. Cryptography Experiments. Available at: <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct implementation of Caesar cipher in C	20 %
2	Successful use of CrypTool for encryption/decryption	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Correctness of ciphertext and plaintext recovery	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 06: i. Implement Vernam cipher encryption technique to perform encryption of text using C programming language
ii. Apply Vernam cipher technique to perform encryption and decryption of text using any open-source tool (Example - Cryptool)

I. Practical Significance

This practical introduces the Vernam cipher, a stream cipher based on the XOR operation. Unlike substitution ciphers, Vernam cipher provides a stronger encryption approach when combined with a truly random key (known as a One-Time Pad). Students will implement Vernam cipher in C and also apply it using open-source tools such as Cryptool, helping them understand the significance of randomness and key length in cryptographic security.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO3 – Implement encryption/decryption techniques.

IV. Laboratory Learning Outcome (LLO)

LLO 6.1: Implement Vernam cipher encryption technique.

V. Relevant Affective domain related Outcomes(s)

1. Ethical awareness: Understanding limitations of using short keys versus one-time pad.
2. Analytical skills: Linking bitwise operations to cryptographic principles.
3. Responsibility: Correct handling of encryption keys.

VI. Theoretical Background

Vernam Cipher:

The Vernam Cipher is a symmetric key stream cipher invented in 1917 by Gilbert Vernam.

It operates by applying the XOR (Exclusive OR) logical operation between each bit of the plaintext and a corresponding bit of the key.

This makes it different from simple substitution ciphers because encryption and decryption use the same XOR function.

Process Vernam Cipher:

1. Key Selection

- Choose a **key** of the **same length as the plaintext**.
- If the key is truly random and never reused, the cipher becomes a **One-Time Pad (OTP)**.

2. Convert Plaintext and Key to Binary/Numbers

- Represent plaintext letters and key letters as **ASCII binary codes** (or A=0, B=1, ..., Z=25 if working mod 26).

3. Encryption (XOR Operation)

- Apply **bitwise XOR ($\oplus$)** between each bit of the plaintext and key.

$$C = P \oplus K$$

4. Ciphertext Formation

- Combine all XOR results to get the **ciphertext in binary**.
- Convert the binary back into characters (may look random).

5. Decryption

- Receiver applies XOR again with the same key:

$$P = C \oplus K$$

- Since XOR is reversible, applying the same operation restores the original plaintext.

Where,

P = Plaintext letter,

C = Ciphertext letter,

K = Key (shift value).

$\oplus$ = XOR

Example:

Encryption:

Plaintext = **AISSMS**

- **Plaintext $\rightarrow$ Numbers (A=0, B=1, ..., Z=25)**

- A = 0
- I = 8
- S = 18
- S = 18
- M = 12
- S = 18

So, Plaintext = **0 8 18 18 12 18**

- **Choose a Key (same length as plaintext)**

Let's take Key = **XMCKLQ**

- X = 23
- M = 12
- C = 2
- K = 10
- L = 11
- Q = 16

So, Key = **23 12 2 10 11 16**

- **Encryption Formula**

$$C = (P + K) \bmod 26$$

Now compute:

- A (0) + X (23) = 23 $\rightarrow$ **X**
- I (8) + M (12) = 20 $\rightarrow$ **U**
- S (18) + C (2) = 20 $\rightarrow$ **U**
- S (18) + K (10) = 28 mod 26 = 2 $\rightarrow$ **C**

- $M(12) + L(11) = 23 \rightarrow X$
- $S(18) + Q(16) = 34 \bmod 26 = 8 \rightarrow I$
- **Ciphertext**

Ciphertext = XUUCXI

VII. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB)	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	Turbo C Compiler, Open source-crytool 1		

VIII. Procedure

Vernam Cipher Implementation using C program

Step 1: START

Step 2: INPUT PLAINTEXT

- Accept plaintext from user
- Store in plaintext array

Step 3: PREPROCESS PLAINTEXT

- Convert all characters to uppercase
- Remove spaces and non-alphabetic characters
- Store in processedPlaintext array

Step 4: CALCULATE LENGTH

- Determine length of processed plaintext
- Store as variable n

Step 5: VALIDATE PLAINTEXT

- Check if length > 0
- If length = 0, display error and EXIT
- Otherwise, continue

Step 6: INPUT KEY

- Accept key from user
- Key length must be $\geq$ plaintext length

Step 7: PREPROCESS KEY

- Convert all characters to uppercase
- Remove spaces and non-alphabetic characters
- Store in processedKey array

Step 8: VALIDATE KEY LENGTH

- Check if key length $\geq$ plaintext length

- If key is too short, display error and EXIT
- Otherwise, trim key to match plaintext length

Step 9: CONVERT TO NUMBERS

- Convert plaintext characters to numbers (A=0, B=1, ..., Z=25)
- Convert key characters to numbers (A=0, B=1, ..., Z=25)

Step 10: PERFORM ENCRYPTION

- For each position i (0 to $n-1$):
 - Apply formula: $C[i] = (P[i] + K[i]) \bmod 26$
- Store results in ciphertext array

Step 11: CONVERT BACK TO CHARACTERS

- Convert numeric ciphertext to characters (0=A, 1=B, ..., 25=Z)

Step 12: DISPLAY RESULTS

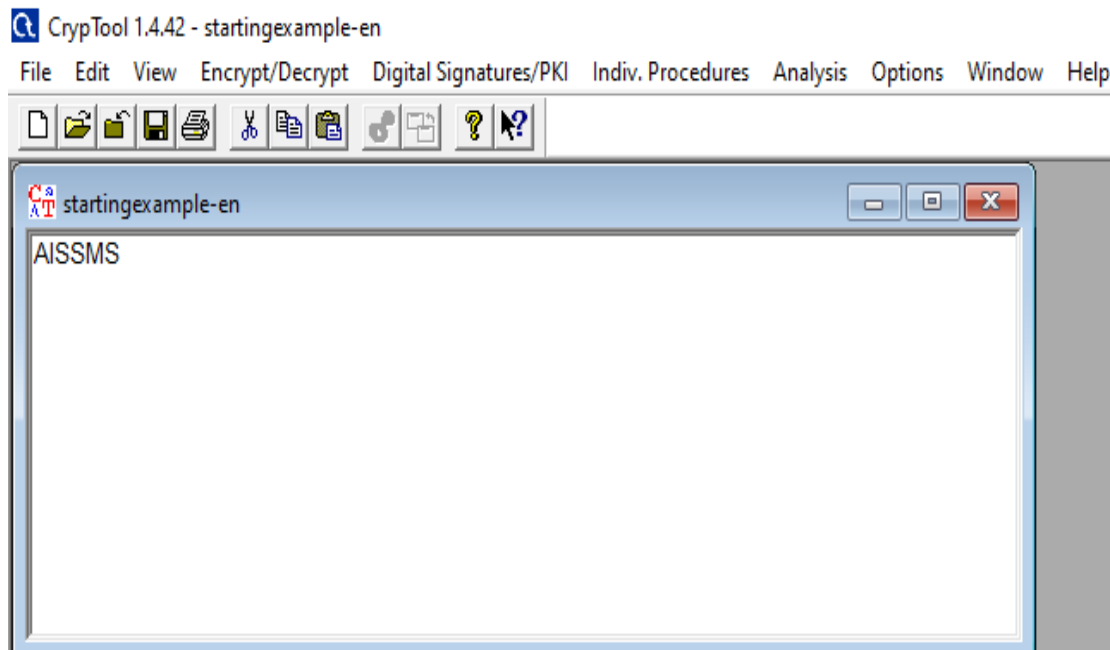
- Show plaintext, key, and ciphertext
- Display numeric representations

Step 13: STOP

Caesar cipher technique to perform encryption and decryption of text using any open-source tool (Example – CrypTool 1.4.42)

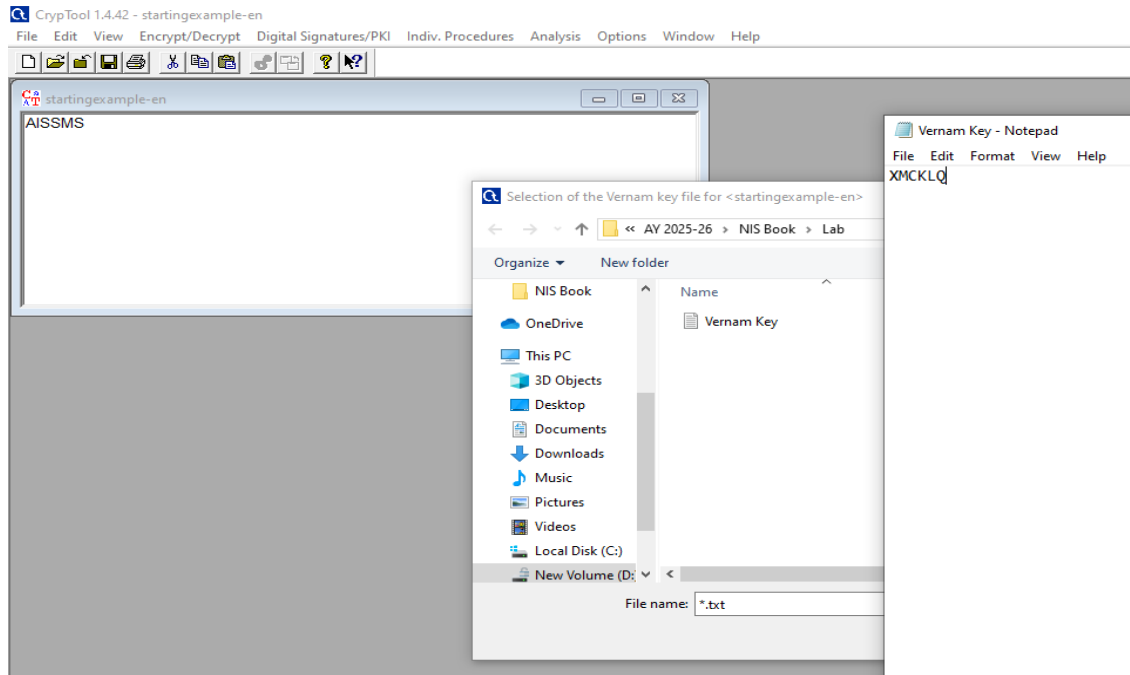
PART A: ENCRYPTION PROCESS**Step 1: Launch CrypTool**

- Open CrypTool 1.4.42 application on your computer
 - Create a new text window in CrypTool (File → New → Textfile or click the new-document button) and type your plaintext (example: AISSMS as in your screenshot). Leave that CrypTool editor window active.
- Step 2: Create New Document

**Step 2: Create & save the key file**

- Open **Notepad** (or Notepad++). Type your key characters.

- The key must be at least as long as the plaintext (preferably exactly the same length for a demo or truly random one-time pad in real OTP use).
 - Example key you used: XMKCLQ.
 - Save the key file: **File** → **Save As** → choose a folder, give it a name like Vernam Key.txt.
- Important:** set Encoding = **ANSI** (or "UTF-8 without BOM") to avoid a byte-order-mark (BOM) being inserted. Click Save.

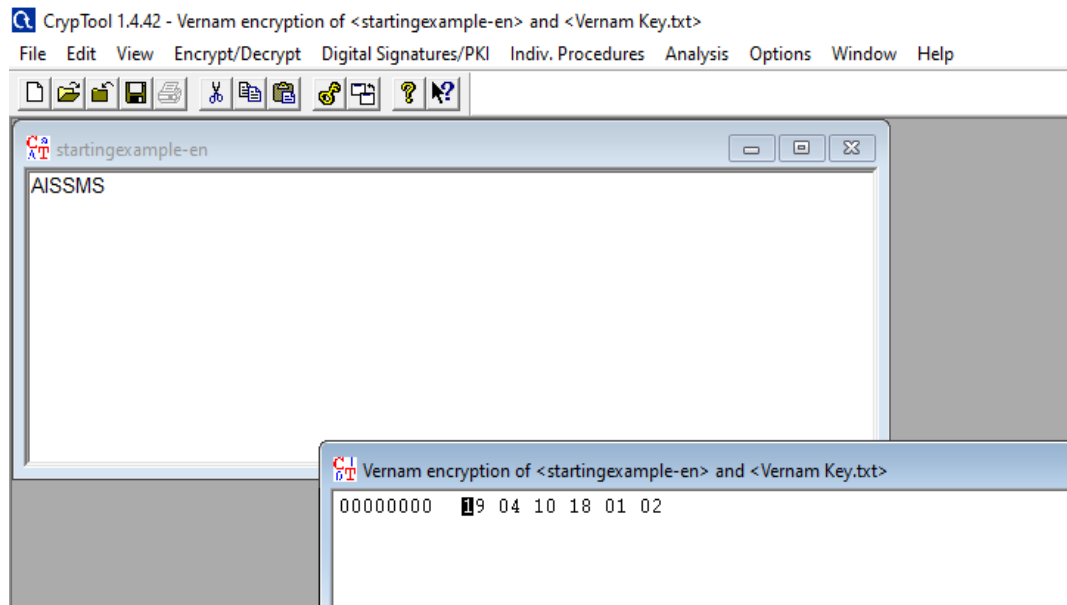


Step 3: Encrypt with Vernam (XOR)

- In CrypTool, with the plaintext window active go to:

Encrypt/Decrypt → **Symmetric (Classic)** → **Vernam / OTP...**

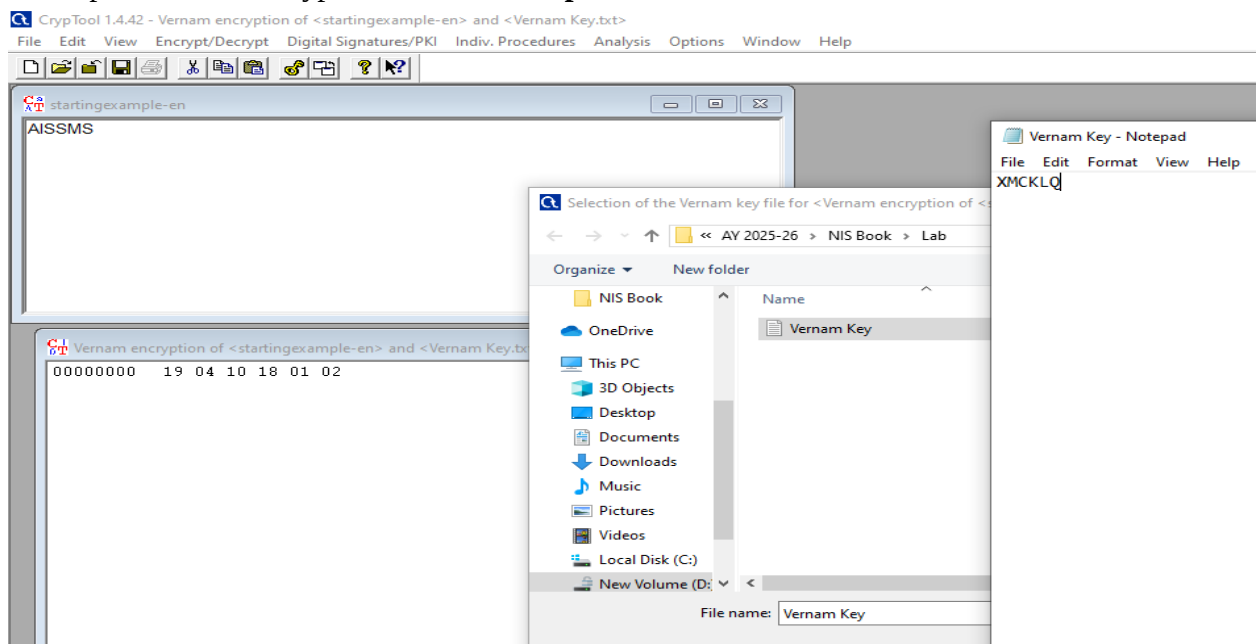
- A file-selection dialog will open asking you to select the Vernam key file. Navigate to and select the Vernam Key.txt you saved. (If CrypTool offers options like reading as text/hex, choose **text** / default.)
- CrypTool will perform the Vernam operation (byte-wise XOR of plaintext bytes with key bytes). It will open a small hex output window showing the XOR bytes and will create a new text window containing the ciphertext (or show ciphertext as a binary/hex view). You can save that ciphertext window (File → Save As) if desired.



PART B: DECRYPTION PROCESS

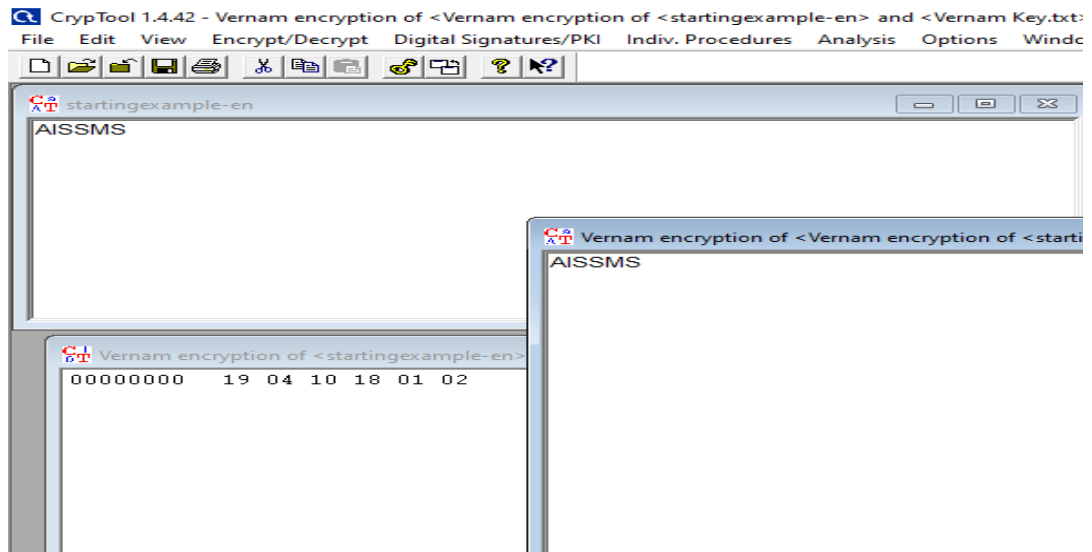
Step 1: Open Encrypted Text

- If continuing from encryption, keep the ciphertext window active
- OR open a saved encrypted file: **File** → **Open**



Step 2: Decrypt

- To decrypt: open (or keep) the ciphertext in a CrypTool text window and repeat Encrypt/Decrypt → Symmetric (Classic) → Vernam / OTP... and select the same key file.
- Vernam/OTP is self-inverse: applying the same XOR key to ciphertext returns the original plaintext, so the same menu action decrypts it.
- The result should appear as the original plaintext in a new window.



IX. Precautions

1. Key length must equal plaintext length.
2. Keys must remain secret and used only once.
3. Save test data to verify correctness.

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. State the encryption and decryption formulas for Vernam cipher.
2. Differentiate between Caesar cipher and Vernam cipher.
3. Encrypt your name using CrypTool with Vernam cipher and attach a screenshot.
4. Implement Vernam cipher to encrypt the string "NETWORK"

Space for Answer

This image shows a full page of white paper with horizontal dashed lines, typical of primary school writing paper. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

This image shows a full page of white paper with horizontal dotted lines, typical of primary school writing paper. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

XI. Exercise

1. Modify the C program to accept keys in binary format.
2. Demonstrate encryption of a paragraph of text using Vernam cipher.
3. Compare Vernam cipher with modern stream ciphers (RC4, Salsa20).

XII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. CrypTool Official Website: <https://www.cryptool.org>
4. NPTEL Course: Introduction to Cryptography and Network Security.
5. Virtual Labs, IIIT Hyderabad, n.d. Cryptography Experiments. Available at: <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct implementation of Vernam cipher in C	20 %
2	Successful use of CrypTool for encryption/decryption	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Correctness of ciphertext and plaintext recovery	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical Number: 07: Implement railfence encryption technique to perform encryption of text using C programming language.

I. Practical Significance:

This practical demonstrates the Rail Fence cipher, a transposition cipher that rearranges characters in a zig-zag pattern across multiple “rails.” Unlike substitution ciphers, Rail Fence does not alter the characters themselves but changes their order, introducing the concept of transposition techniques in cryptography.

II. Industry / Employer Expected Outcome:

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s):

CO3 – Implement encryption/decryption techniques.

IV. Laboratory Learning Outcome (LLO):

LLO 7.1: Implement railfence encryption technique.

V. Relevant Affective domain related Outcomes(s):

1. Analytical skills: Recognizing how rearrangement of characters alters ciphertext.
2. Attention to detail: Correctly implementing row and column arrangement.
3. Responsibility: Verifying encryption/decryption results with different keys (rails).

I. Theoretical Background

Rail Fence Cipher: The Rail Fence cipher is a transposition cipher that arranges plaintext letters diagonally in a zigzag pattern across multiple rails (rows). Ciphertext is obtained by reading characters row by row.

Example:

Choose a key = number of rails.

Write plaintext in a zigzag manner on rails.

Read text row-wise → Ciphertext.

For decryption, reconstruct zigzag pattern using ciphertext and number of rails.

Example (2-rail cipher):

Plaintext: HELLO WORLD

Arrangement:

H . L . O . W . R . D

. E . L . . O . L .

Ciphertext: HLOWRDELOL

II. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	Turbo C Compiler		

III. Procedure**Rail Fence Cipher Implementation using C program****Step 1:** START**Step 2:** Input the plaintext message.

Example: "HELLO WORLD"

Step 3: Remove spaces (or store their positions if you want to restore them later).

Result: "HELLOWORLD"

Step 4: Input the key (number of rails).

Example: key = 2

Step 5: Initialize a 2D array rail[key][length_of_text] and fill all positions with a marker (e.g., '\n').**Step 6:** Set direction flag down = false and row = 0.**Step 7:** For each character in plaintext:

- Place the character in rail[row][col].
- If row == 0, set down = true.
- Else if row == key - 1, set down = false.
- Move to next column.
- If down == true, row++; else row--.

Step 8: After filling the matrix, read all characters row by row to form ciphertext.**Step 9:** Display ciphertext.**Step 10:** STOP.**IV. Precautions**

- The number of rails must be less than the length of plaintext.
- Spaces and punctuation should be handled consistently (removed or preserved).
- Encryption and decryption must use the same number of rails (key).

V. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

- State the difference between substitution and transposition techniques.
- Explain the working of the Rail Fence cipher with an example

3. Encrypt the text "CRYPTOGRAPHY" using 3-rail cipher.
4. Design a Rail Fence cipher program that supports both encryption and decryption modes in one implementation.

Space for Answer

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

.....

.....

.....

.....

.....

.....

.....

VI. Exercise

1. Modify program to preserve spaces in ciphertext.
2. Implement Rail Fence cipher with a GUI interface in C or Python.
3. Compare results of encryption for 2 rails, 3 rails, and 4 rails with the same plaintext.

VII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. NPTEL Course: Introduction to Cryptography and Network Security.
4. Virtual Labs, IIIT Hyderabad, n.d. Cryptography Experiments. Available at: <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

VIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct implementation of Rail Fence cipher in C	20 %
2	Demonstration of working with different keys	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Correctness of ciphertext and plaintext recovery	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 08: Implement simple Columnar Transposition encryption technique to perform encryption of text using C programming language.

I. Practical Significance

This practical demonstrates the Columnar Transposition Cipher, one of the classical transposition techniques where the order of plaintext characters is rearranged according to a predefined key. It introduces students to the concepts of columnar arrangement, permutation, and reading ciphertext column-wise. By implementing this cipher in C, students understand the importance of positional rearrangement in cryptography.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO3 – Implement encryption/decryption techniques.

IV. Laboratory Learning Outcome (LLO)

LLO 8.1: Implement simple columnar transposition technique.

V. Relevant Affective domain related Outcomes(s)

1. Analytical skills: Rearranging data systematically according to a given key.
2. Attention to detail: Handling incomplete rows and padding characters.
3. Responsibility: Applying correct key order for both encryption and decryption.

VI. Theoretical Background

Columnar Transposition Cipher:

A Columnar Transposition Cipher is a classical encryption technique that rearranges the letters of the plaintext into a fixed column structure according to a key. Unlike substitution ciphers that replace letters with others, this method changes only the positions of characters, making the ciphertext a permutation of the original message.

The Columnar Transposition Cipher works on the principle of transposition, where the sequence of characters is altered while preserving their original identities. To encrypt, a key (word or number) is chosen, and the plaintext is written into a grid with as many columns as the length of the key. The columns are then rearranged according to the alphabetical order of the key characters (or numerical order if digits are used). Finally, the ciphertext is generated by reading the letters column by column. This cipher is relatively simple but offers stronger security compared to direct substitution since frequency analysis becomes less effective. However, it can still be broken using known-plaintext or brute force attacks. Columnar transposition is often combined with substitution methods to create product ciphers for enhanced security.

Example:**Plaintext:** POLYTECHNIC**Key:** COMP**Step 1: Number the Key**We order the letters in **alphabetical order**:Key = **C O M P**

- C → 1
- M → 2
- O → 3
- P → 4

So key = **1 3 2 4****Step 2: Arrange Plaintext in Grid**

Plaintext length = 11 letters

Key length = 4 → grid has 4 columns

We write row by row and add padding X to fill:

Key: 1 3 2 4

C	O	M	P
1	3	2	4
P	O	L	Y
T	E	C	H
N	I	C	X

Step 3: Read Columns in Order

Rearrange columns according to key order (1 → 2 → 3 → 4):

- Column Name 1 → P T N
- Column Name 2 → L C C
- Column Name 3 → O E I
- Column Name 4 → Y H X

Step 4: Ciphertext

Now read column by column:

PTNLCCOEIYHX

Final **Ciphertext** = **PTNLCCOEIYHX****VII. Resources required**

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	Turbo C Compiler		

VIII. Procedure

Columnar Transposition Cipher implementation using C program

Step 1: START

Step 2: Take input from the user:

- Plaintext
- Key

Step 3: Remove all spaces from the plaintext.

Convert both the plaintext and key to **uppercase** letters.

Step 4: Determine the length of the key → keyLen

Determine the length of the plaintext → textLen

Step 5: Number the letters of the key:

- a. Arrange key letters in **alphabetical order**.
- b. Assign sequential numbers 1, 2, 3, ... according to their order.
Example: Key = ZEBRA → Order = 5 2 1 3 4

Step 6: Create a table (matrix) with:

- **Columns = keyLen**
- **Rows = ceil(textLen / keyLen)**

Step 7: Fill the table **row by row** with letters of the plaintext.

If any empty boxes remain, **fill them with 'X'** as padding.

Step 8: Read the letters **column by column** according to the order of key numbers

(1 → 2 → 3 → ... → keyLen).

Step 9: Concatenate all letters read in Step 8 to form the **Ciphertext**.

Step 10: Display the Ciphertext.

Step 11: STOP

IX. Precautions

1. Ensure key contains no repeated letters (or handle them carefully).
2. Always pad plaintext so matrix is completely filled.
3. Encryption and decryption must use exactly the same key.

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Explain the encryption process of Columnar Transposition with an example.
2. Encrypt the text "CRYPTOGRAPHY IS FUN" using the key "CIPHER".
3. Compare Columnar Transposition with Rail Fence cipher.
4. Design a hybrid encryption method combining Caesar cipher and Columnar Transposition.

Space for Answer

This image shows a full page of white paper with horizontal dotted lines. The lines are evenly spaced and run across the width of the page, providing a guide for handwriting practice. There are no margins, text, or other markings on the page.

.....

.....

.....

.....

.....

.....

.....

XI. Exercise

1. Modify program to allow decryption as well.
2. Extend Columnar Transposition cipher to support numeric digits.
3. Implement the cipher using both row-major and column-major ordering and compare results.

XII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. NPTEL Course: Introduction to Cryptography and Network Security.
4. Virtual Labs, IIIT Hyderabad, n.d. Cryptography Experiments. Available at: <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct implementation of Columnar Transposition cipher in C	20 %
2	Handling of key ordering	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Correctness of ciphertext	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 09: Create and verify Hash Code for given message using any Open-source tool (Example-Cryptool).

I. Practical Significance

This practical introduces hash functions, which are mathematical algorithms that transform input data into a fixed-size hash value (digest). Hash codes are widely used in data integrity verification, password storage, and digital signatures. Students will learn to generate and verify hash codes for a given message using open-source tools such as CrypTool.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO3 – Implement encryption/decryption techniques.

IV. Laboratory Learning Outcome (LLO)

LLO 9.1: Generate hash code.

V. Relevant Affective domain related Outcomes(s)

1. Responsibility: Correctly applying tools to verify message integrity.
2. Ethical awareness: Understanding risks of weak hash algorithms (e.g., MD5 collisions).
3. Analytical skills: Comparing outputs of different hash algorithms.

VI. Theoretical Background

A **hash function** is a special mathematical function that converts an input of any size (called a *message*) into a fixed-size string of characters (called a *hash value*, *digest*, or *checksum*). It is commonly used in cryptography, computer security, and data integrity verification.

A hash function takes arbitrary input data and produces a fixed-length output, usually represented as a sequence of hexadecimal characters. The key property of a hash function is that even a tiny change in the input produces a drastically different output (called the **avalanche effect**).

In computer science, hash functions are widely used in:

- **Cryptography** (e.g., SHA-256, MD5, SHA-3) to protect passwords and digital signatures.
- **Data integrity** (detecting if a file or message has been tampered with).
- **Hash tables** (for efficient data retrieval in constant time).

A **cryptographic hash function** must satisfy:

1. **Deterministic** – Same input always gives the same output.
2. **Pre-image resistance** – Hard to reverse-engineer the input from the hash.
3. **Collision resistance** – Hard to find two different inputs that produce the same hash.
4. **Avalanche effect** – Small input change gives a very different hash output.

Common Hash Algorithms:

- **MD5 (128-bit digest)** – Fast but insecure due to collisions.
- **SHA-1 (160-bit digest)** – Stronger but broken for cryptographic purposes.
- **SHA-256 (256-bit digest)** – Strong, widely used in blockchain and digital signatures.

MD5 Hash Function

- **MD5** (Message Digest 5) is a widely known hash function.
- It produces a **128-bit (16-byte) hash value**, usually represented as a **32-digit hexadecimal number**.
- It is **fast** and still used for checksums, but it is **not secure** for cryptography (because collisions can be found).

Example:**Input (Plaintext):**

POLYTECHNIC

Output (MD5 Hash Value):

361dd8f32c3b59f9a5c0d6a5c1f4f40

If the input changes slightly, the hash output changes completely:**Input (Plaintext):**

polytechnic

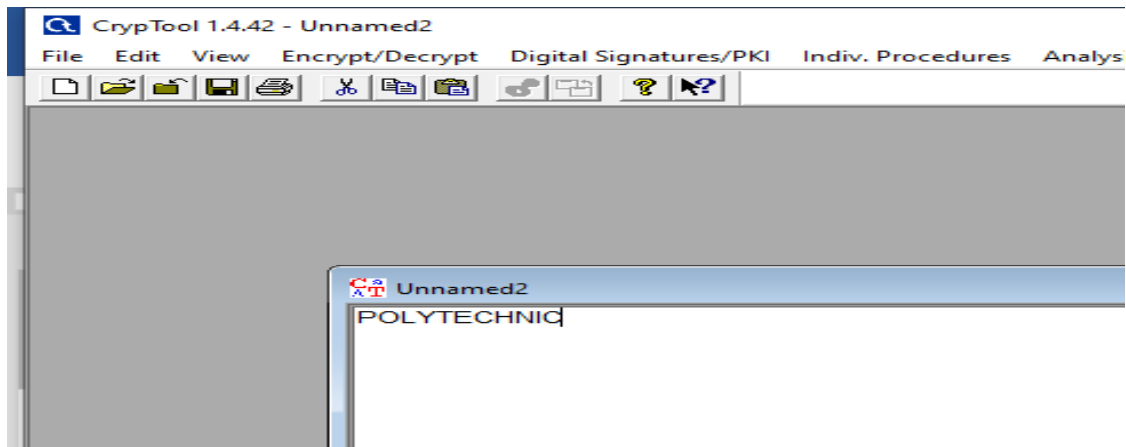
Output (MD5 Hash Value):

268d5f6e72d0d6f471a96a1740b5e5e4

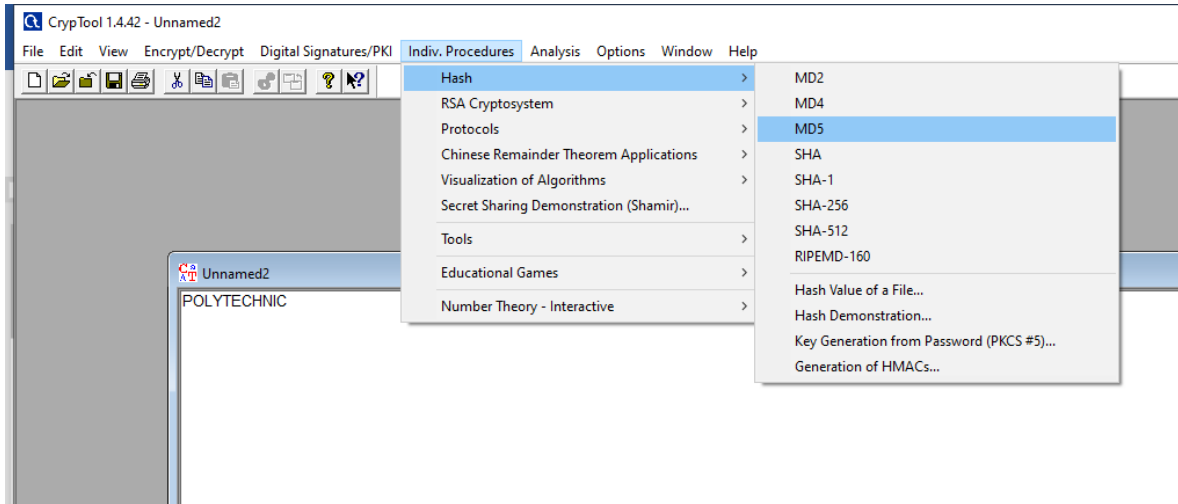
VII. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	CrypTool 1.4.42		

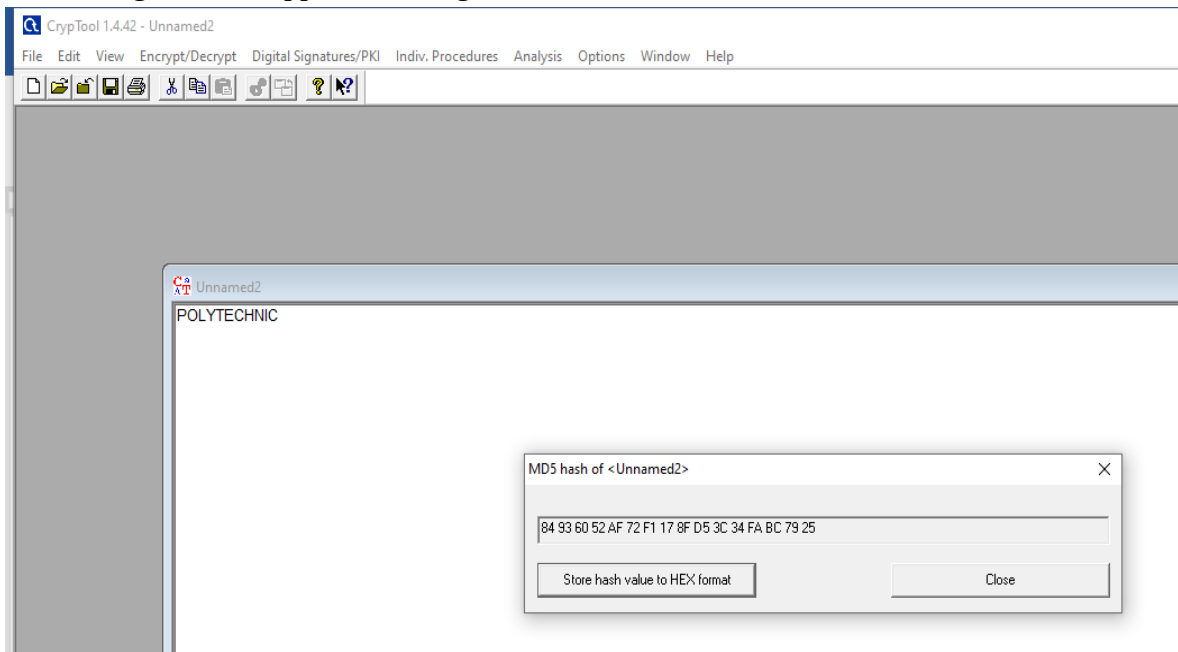
VIII. Procedure**Create and verify Hash Code (MD-5) for given message using CrypTool****Step 1: Start CrypTool 1.4.2.****Step 2: Select File → New** to create a new document.**Step 3: Type the plaintext** (example: POLYTECHNIC) into the editor window.



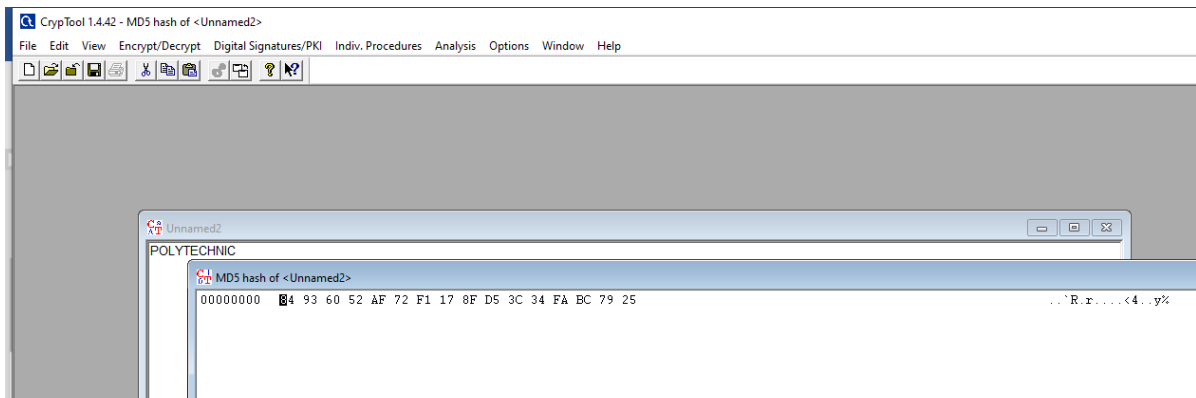
Step 4: From the menu, go to **Indiv. Procedures** → **Hash** → **MD5**.



Step 5: A dialog box will appear showing the **MD5 hash value** in hexadecimal form.



Step 6: If required, click **Store hash value to HEX format** to open the hash in a new window.



Step 7: Copy the hex values (remove spaces if needed) to form the **32-character MD5 string**.

IX. Precautions

1. Always use strong hash functions (SHA-256 or higher) for secure applications.
2. Do not rely on MD5 or SHA-1 for cryptographic security.
3. Ensure consistent input (case-sensitive, whitespace matters).

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. List any four properties of a good hash function.
2. Generate SHA-1, and SHA-256 hashes for the string "NETWORK SECURITY" using CrypTool.
3. Compare outputs of MD5 and SHA-256 for the same message.
4. Design a simple program (in Python/C) to compute SHA-256 hash of user input and verify integrity.

Space for Answer

[illegible]

This image shows a full page of white paper with horizontal dotted lines. The lines are evenly spaced and run across the width of the page, providing a guide for handwriting practice. There are no margins, text, or other markings on the page.

XI. Exercise

1. Demonstrate avalanche effect by computing SHA-256 hash of HELLO and HELLo.
2. Compare hash outputs using CrypTool and online generators.
3. Research and report on MD-5 algorithm.

XII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. NPTEL Course: Introduction to Cryptography and Network Security.
4. CrypTool Official Website: <https://www.cryptool.org>
5. Virtual Labs, IIIT Hyderabad, n.d. Cryptography Experiments. Available at: <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct generation of hash values using CrypTool	20 %
2	Verification of integrity using hash values	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Correctness of hash outputs	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 10: i. Write a C program to implement Diffie-Hellman key exchange algorithm to perform encryption of text
ii. Use Diffie-Hellman key exchange algorithm to perform encryption and decryption of text using any open-source tool (Example - Cryptool)

I. Practical Significance

This practical introduces the Diffie-Hellman key exchange protocol, the first widely used public key exchange method. It allows two parties to establish a shared secret key over an insecure channel without transmitting the key itself. This secret key can then be used for symmetric encryption. Students will implement the algorithm in C and apply it using Cryptool to understand how secure key exchange is achieved in real-world cryptography.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO4 – Use tools and techniques to prevent cyber attacks.

IV. Laboratory Learning Outcome (LLO)

LLO 10.1: Implement Diffie-Hellman key exchange encryption technique.

V. Relevant Affective domain related Outcomes(s)

1. Ethical awareness: Understanding that secure key management is essential for data privacy.
2. Analytical skills: Evaluating modular arithmetic and prime number role in cryptography.
3. Attention to detail: Correct implementation of exponentiation and modular operations.

I. Theoretical Background

Diffie–Hellman Key Exchange (DHKE):

The Diffie–Hellman Key Exchange (DHKE) is a method in cryptography that allows two parties to securely establish a shared secret key over an insecure communication channel. This shared key can then be used to encrypt further communications.

It was introduced in 1976 by Whitfield Diffie and Martin Hellman, and is considered one of the first practical implementations of public key cryptography.

The DH algorithm is based on the difficulty of the **Discrete Logarithm Problem (DLP)** in modular arithmetic.

1. Two parties (say **A** and **B**) agree on public parameters:
 - A large **prime number** p
 - A **primitive root** (or generator) g modulo p

These values (p, g) are **publicly known**.

2. Each party chooses a **private key**:

- Alice chooses a (secret)
- Bob chooses b (secret)
- 3. Each computes a **public key**:
 - Alice computes $A = g^a \bmod p$
 - Bob computes $B = g^b \bmod p$

Then they exchange these values.

- 4. Each party computes the **shared secret**:

- Alice computes $S = B^a \bmod p$
- Bob computes $S = A^b \bmod p$

Both values are mathematically equal:

$$S = g^{ab} \bmod p$$

- 5. Now, both Alice and Bob have the **same secret key S**, without ever sending it directly.

Example (Small Numbers for Simplicity)

1. Public parameters:
 - Prime $p=23$
 - Generator $g=5$
2. Private keys:
 - Alice chooses $a=6$
 - Bob chooses $b=15$
3. Public keys:
 - Alice computes: $A = g^a \bmod p = 5^6 \bmod 23 = 15625 \bmod 23 = 8$
 - Bob computes: $B = g^b \bmod p = 5^{15} \bmod 23 = 30517578125 \bmod 23 = 19$
 - So Alice sends **8**, Bob sends **19**.
4. Shared secret:
 - Alice computes: $S = B^a \bmod p = 19^6 \bmod 23 = 47045881 \bmod 23 = 2$
 - Bob computes: $S = A^b \bmod p = 8^{15} \bmod 23 = 35184372088832 \bmod 23 = 2$
 - Both Alice and Bob now share the secret key **2**, which can be used for encryption.

II. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	Turbo C Compiler, CrypTool 1.4.42		

III. Procedure

Diffie–Hellman Key Exchange (DHKE): Implementation using C program

Step 1: START**Step 2: Declare variables**

- int p // large prime modulus
- int g // primitive root (generator) modulo p
- int a, b // private keys (chosen by Alice and Bob respectively)
- int A, B // public keys ($A = g^a \text{ mod } p$, $B = g^b \text{ mod } p$)
- int s1, s2 // shared secrets computed by Alice and Bob

Step 3: Implement helper function modularExponentiation(base, exp, mod)

- Input: base, exp, mod
- Output: $(\text{base}^{\text{exp}}) \% \text{mod}$ computed efficiently using binary (fast) exponentiation
- Pseudocode for helper:
 1. result $\leftarrow$ 1
 2. base $\leftarrow$ base % mod
 3. while exp > 0 do
 - a. if (exp % 2 == 1) then result $\leftarrow$ (result * base) % mod
 - b. exp $\leftarrow$ exp / 2 (integer division)
 - c. base $\leftarrow$ (base * base) % mod
 4. return result

Step 4: Input or set public parameters p and g

- Option A: Prompt user to enter a prime p and generator g
- Option B: Use example/standard values for demonstration (ensure p is prime and g is a primitive root mod p)

Step 5: Alice chooses private key a (random integer, $1 < a < p-1$)**Step 6: Bob chooses private key b (random integer, $1 < b < p-1$)****Step 7: Alice computes her public key A**

- $A \leftarrow \text{modularExponentiation}(g, a, p)$

Step 8: Bob computes his public key B

- $B \leftarrow \text{modularExponentiation}(g, b, p)$

Step 9: Alice sends A to Bob; Bob sends B to Alice (over insecure channel)**Step 10: Alice computes shared secret s1 using Bob's public key**

- $s1 \leftarrow \text{modularExponentiation}(B, a, p)$

Step 11: Bob computes shared secret s2 using Alice's public key

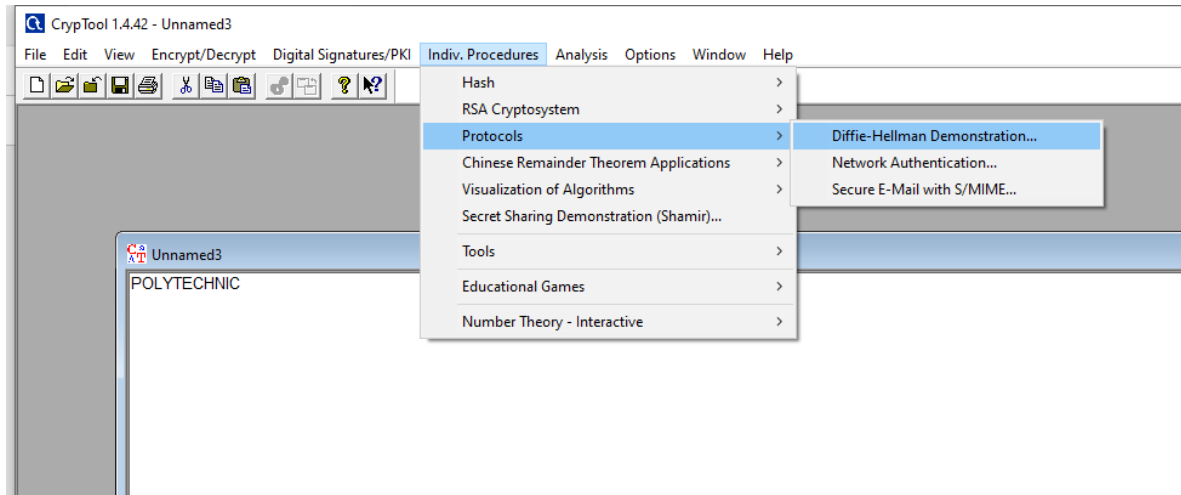
- $s2 \leftarrow \text{modularExponentiation}(A, b, p)$

Step 12: Verify shared secret equality

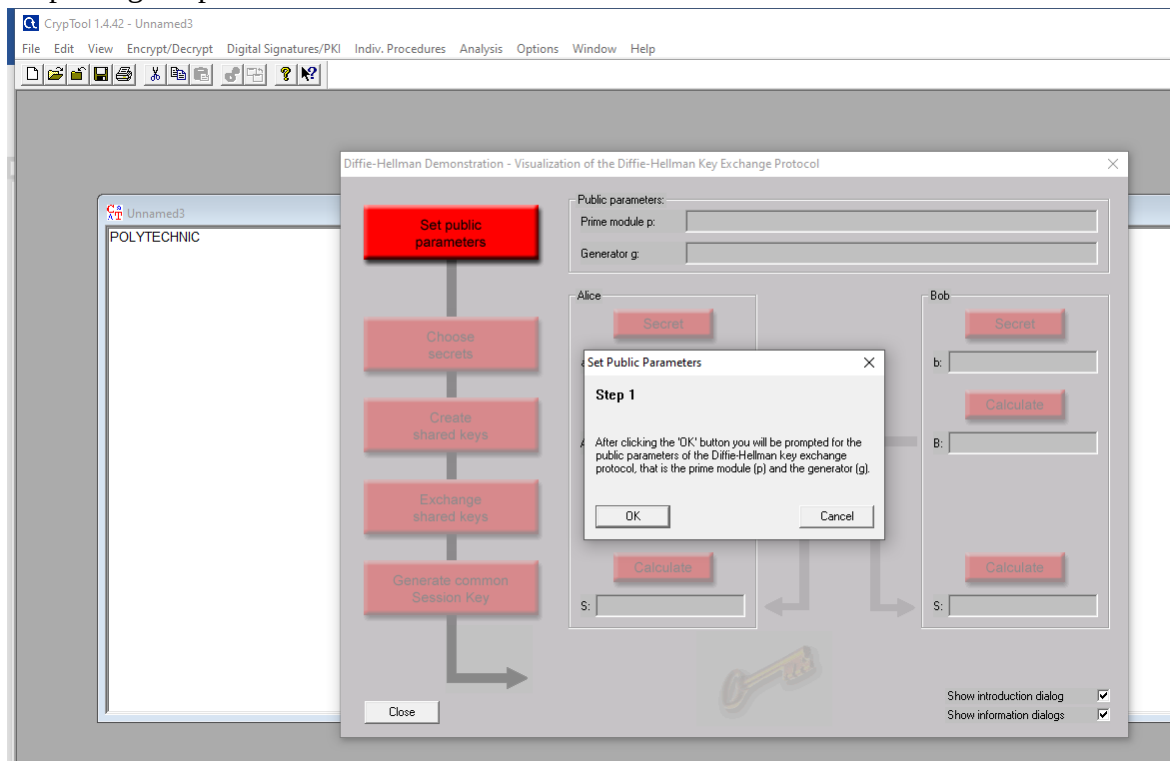
- If $s1 == s2$ then
 - a. Shared secret established: shared_secret $\leftarrow$ s1
 - b. Else: error — keys do not match

Step 13: Display values (optional / for demonstration):

- Show p, g, a (private of Alice — usually not displayed), b (private of Bob — not displayed), A, B, and shared_secret

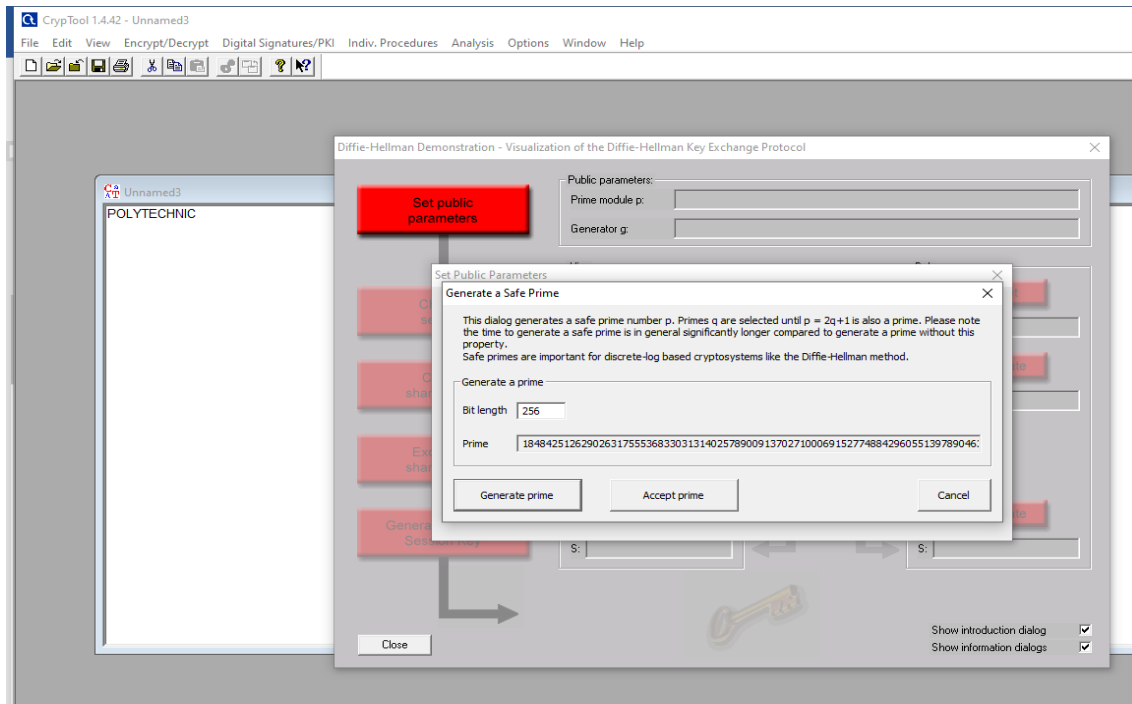
Step 14: STOP**Diffie-Hellman Key Exchange (DHKE) using CrypTool 1.4.42****Step 1: START****Algorithm / Steps:****Step 1:** Open CrypTool 1.4.2.**Step 2:** From the menu, select **Indiv. Procedures** → **Protocols** → **Diffie-Hellman Demonstration**.**Step 3:** Set public parameters:

- Generate a large prime number (p) and accept it.
- Generate a generator (g) and accept it.
- Both p and g are public.

**Step 4:** Choose secrets:

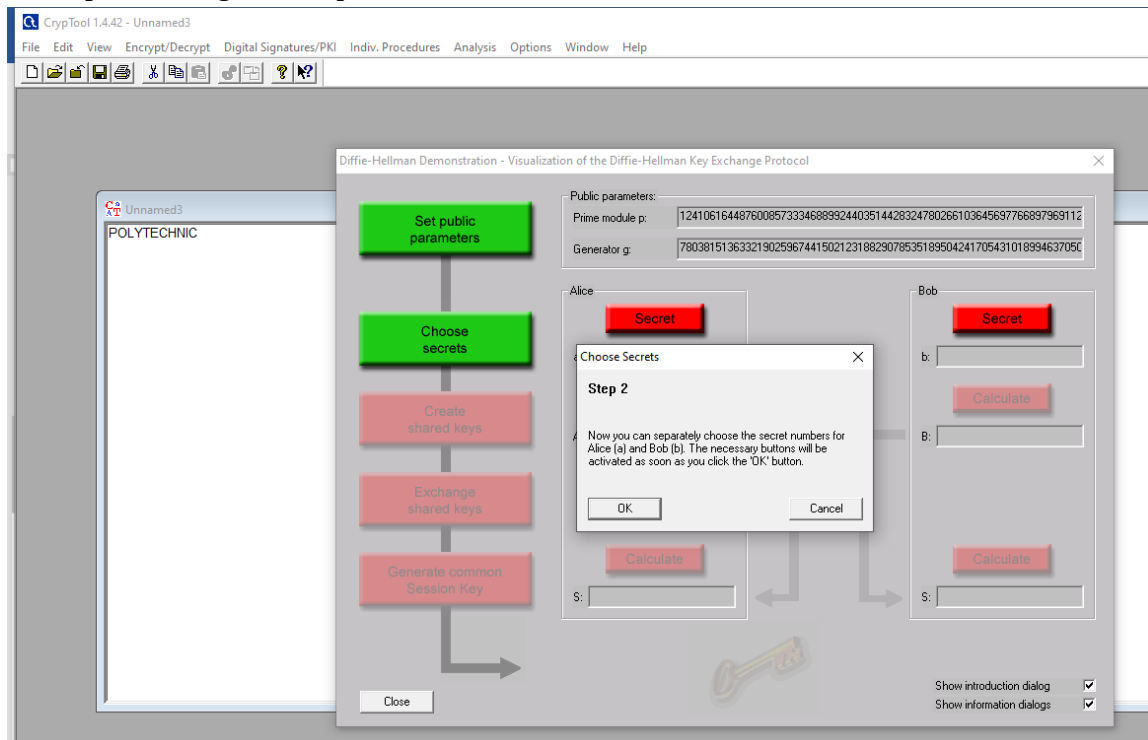
- Alice selects a private number a .

- Bob selects a private number b .
- Both values remain hidden.



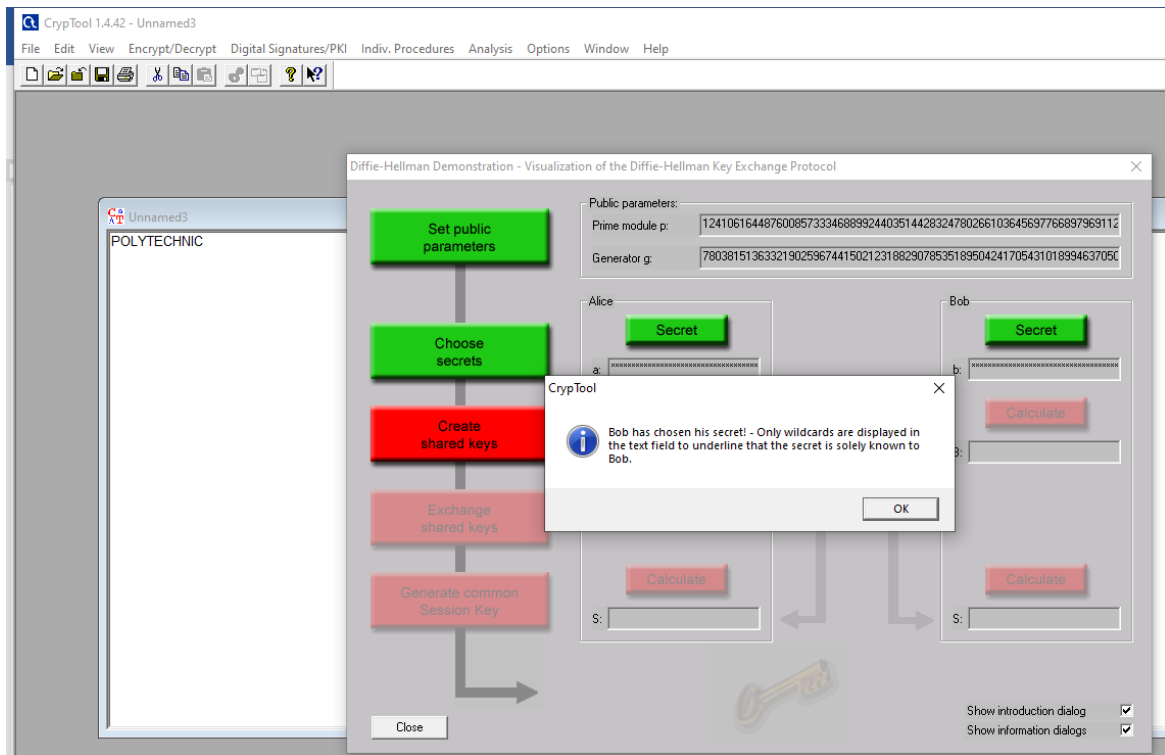
Step 5: Create shared keys (public values):

- Alice computes $A = g^a \text{ mod } p$.
- Bob computes $B = g^b \text{ mod } p$.



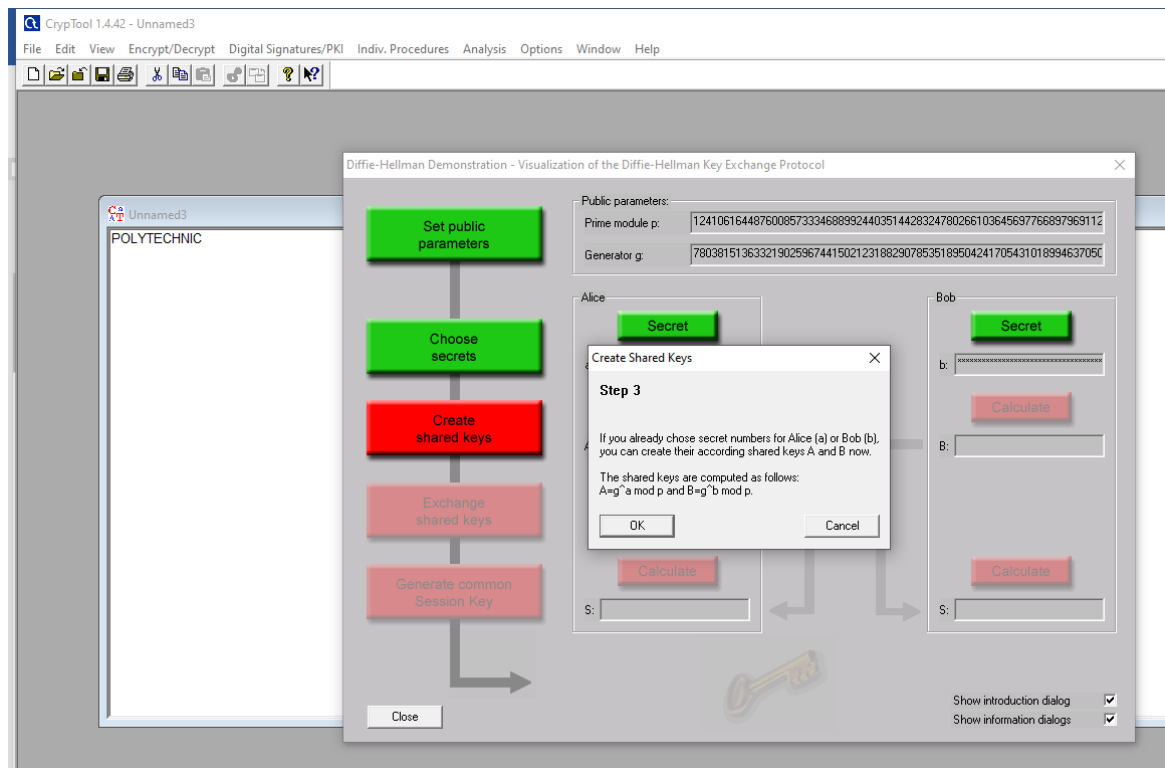
Step 6: Exchange keys:

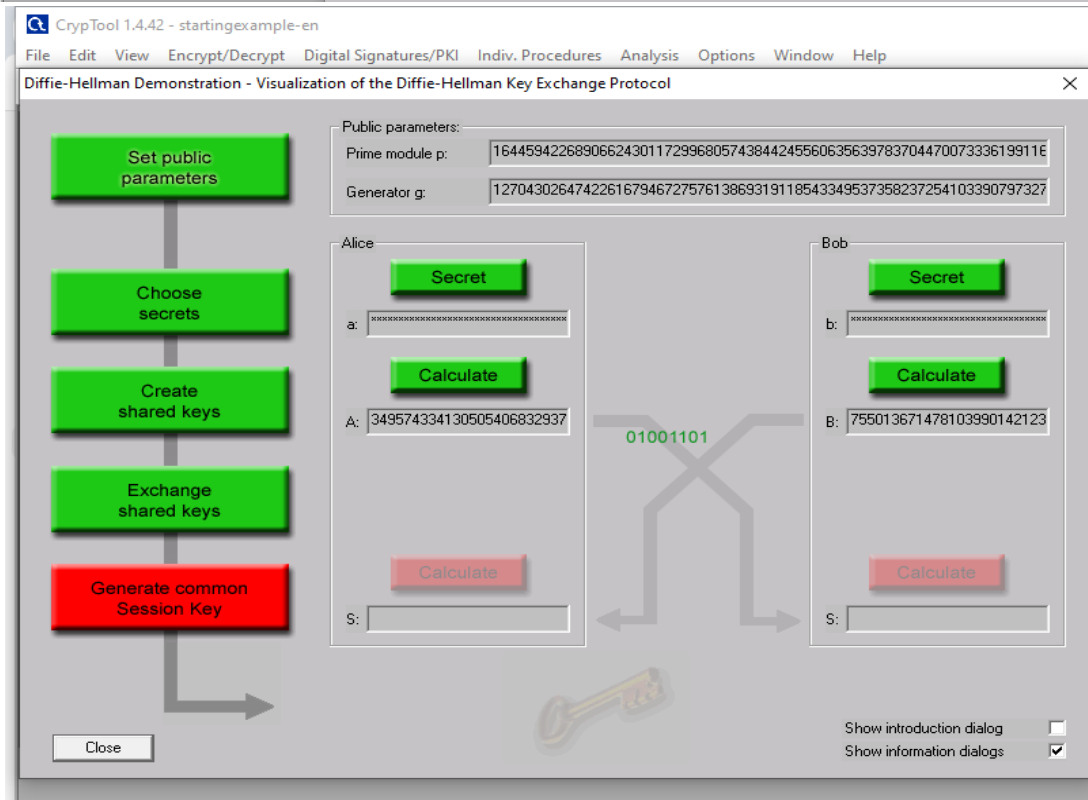
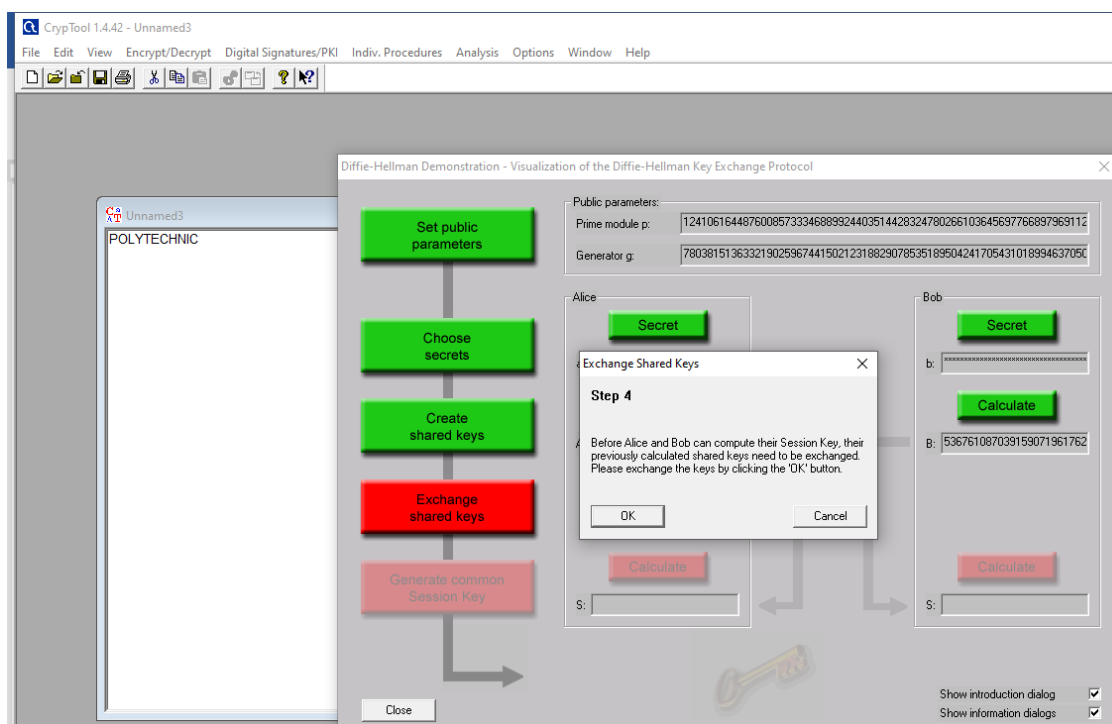
- Alice sends A to Bob.
- Bob sends B to Alice.



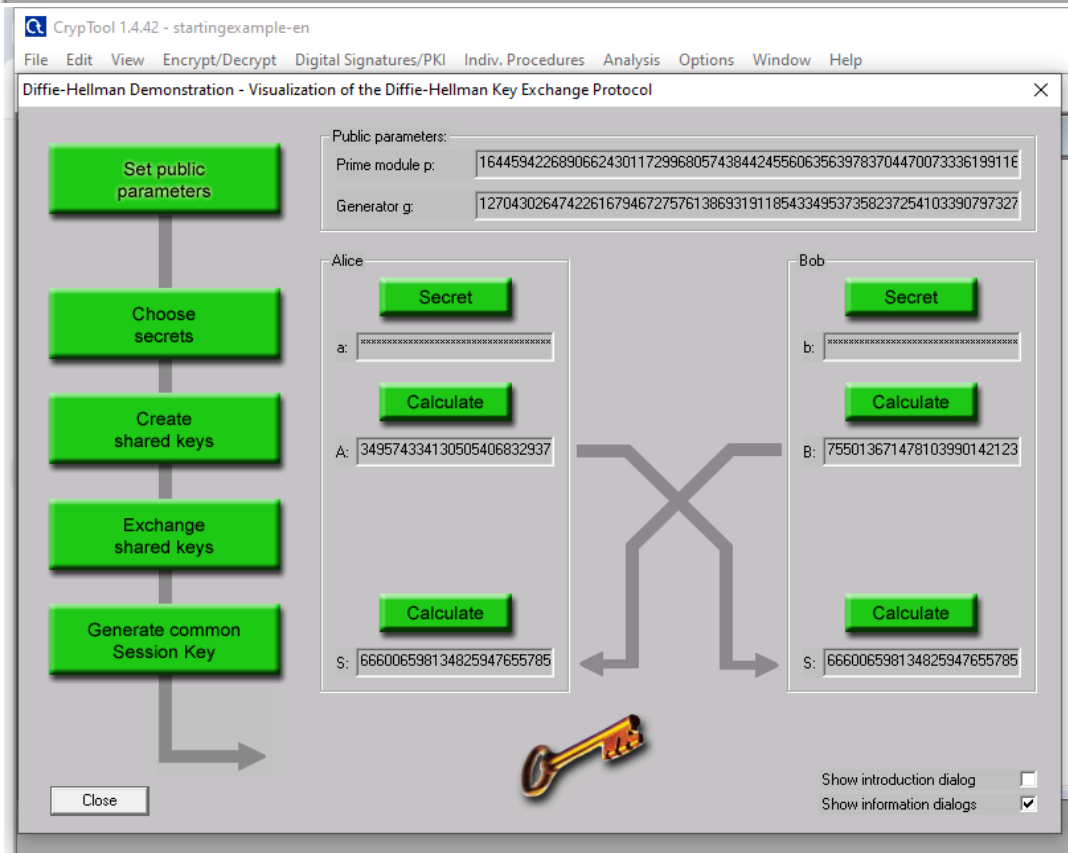
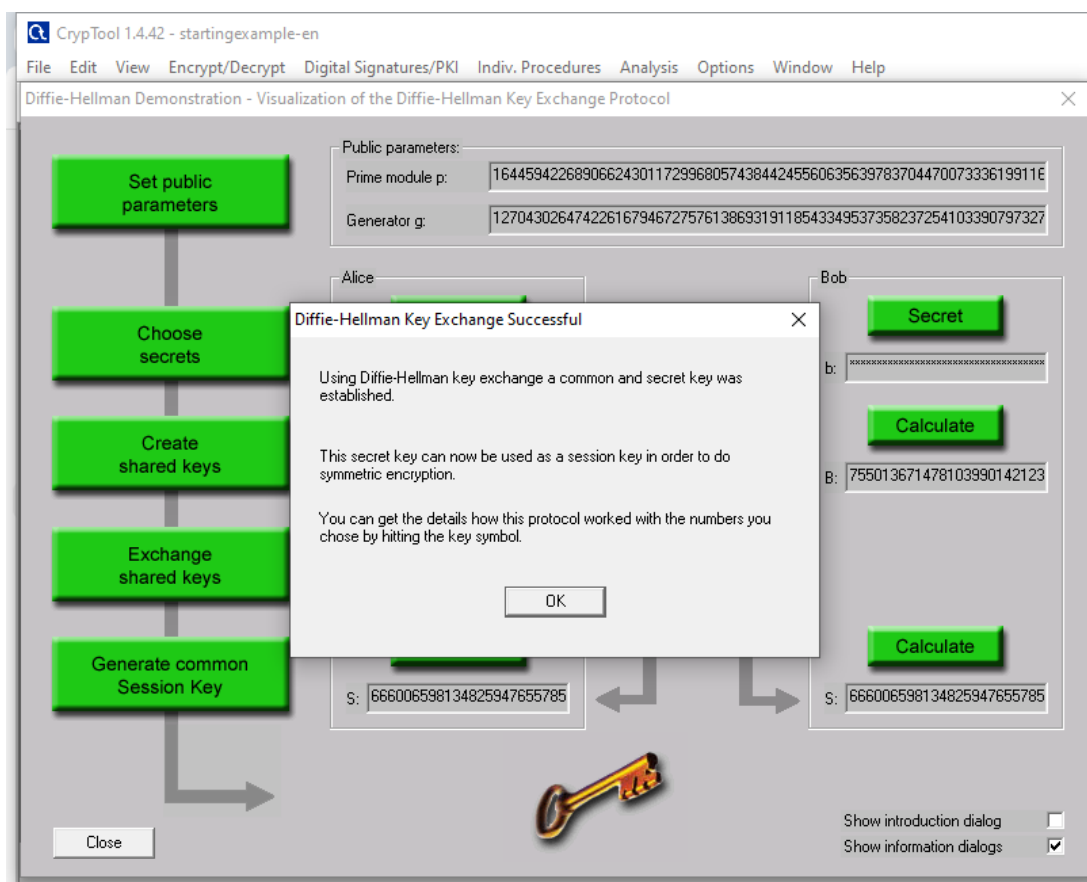
Step 7: Generate session key:

- Alice computes $S = B^a \text{ mod } p$.
- Bob computes $S = A^b \text{ mod } p$.
- Both values are identical.





Step 8: Verify: CrypTool displays the same session key for Alice and Bob → key exchange successful.



A common secret session key S shared by both Alice and Bob.

[illegible]

VI. Exercise

1. Modify the C program to accept keys in binary format.
2. Demonstrate encryption of a paragraph of text using Vernam cipher..
3. Compare Vernam cipher with modern stream ciphers (RC4, Salsa20).

VII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. CrypTool Official Website: <https://www.cryptool.org>
4. NPTEL Course: Introduction to Cryptography and Network Security.
5. Virtual Labs, IIIT Hyderabad, n.d. Cryptography Experiments. Available at: <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

VIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct implementation of Vernam cipher in C	20 %
2	Successful use of CrypTool for encryption/decryption	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Correctness of ciphertext and plaintext recovery	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 11: Use Steganography to encode and decode the message using any Open source tool (Example – OpenStego)

I. Practical Significance

This practical introduces steganography, a technique used to conceal information within another file such as an image, audio, or video so that the existence of the hidden data remains undetectable. Students will learn to embed (encode) and extract (decode) a secret message using OpenStego, an open-source tool for digital data hiding. The aim is to understand the concept of data confidentiality and covert communication.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO4 – Use tools and techniques to prevent cyber attacks.

IV. Laboratory Learning Outcome (LLO)

LLO 11.1: Implement steganography.

V. Relevant Affective domain related Outcomes(s)

1. Responsibility: Apply tools properly to protect information confidentiality.
2. Ethical Awareness: Understand ethical use of steganography in secure systems.
3. Analytical Skills: Analyze and verify the integrity of extracted hidden messages.

VI. Theoretical Background

Steganography is the art and science of **embedding secret messages in a cover message** in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message. It conceals the fact that communication is taking place, providing an additional layer of security beyond traditional cryptography.

As the image depicts, both the **cover file (X)** and **secret message (M)** are fed into a **Steganographic Encoder** as inputs.

The **Steganographic Encoder function**, represented as $f(X, M, K)$, embeds the secret message into the cover file using a key **K**. The resulting **Stego Object** looks very similar to the original cover file, with **no visible changes**, thereby concealing the hidden data. This completes the **encoding** process. To retrieve the hidden message, the **Stego Object** is fed into a **Steganographic Decoder**, which extracts the original secret message **M** using the same key **K**.

In simpler terms, steganography hides the **existence of information**, whereas **cryptography** hides the **meaning of information**.

Common Steganographic Media:

- **Image files** (e.g., .bmp, .png)
- **Audio files** (e.g., .wav, .mp3)
- **Video files** (e.g., .mp4, .avi)

OpenStego is an open-source tool that provides two major functionalities:

1. **Data Hiding (Message Embedding)**
2. **Watermarking (Authentication)**

Applications:

- Secure and covert communication
- Digital watermarking for copyright protection
- Data confidentiality in sensitive communications

VII. Resources required

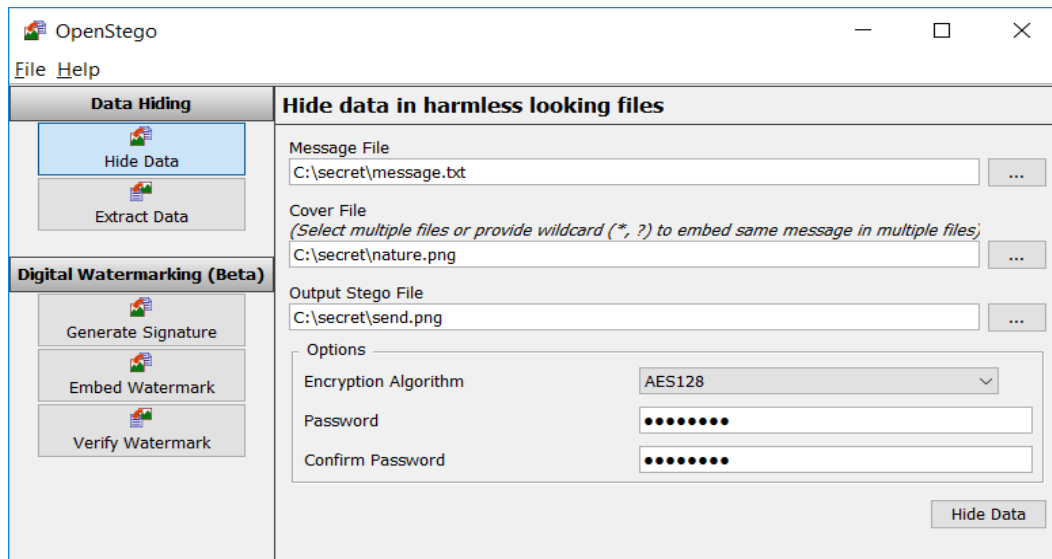
Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB)	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	CrypTool 1.4.42		

VIII. Procedure**Step-by-Step Procedure****Step 1: Download and Install OpenStego**

- OpenStego is available for multiple operating systems such as **Windows, Linux, and macOS**.
- Visit the official website: <https://www.openstego.com>
- Download the version compatible with your operating system.
- Follow the installation instructions to install OpenStego on your computer.

Step 2: Launch OpenStego

- After successful installation, **launch** the OpenStego application.
- The main interface window will appear.



Step 3: Select the File to Embed Data

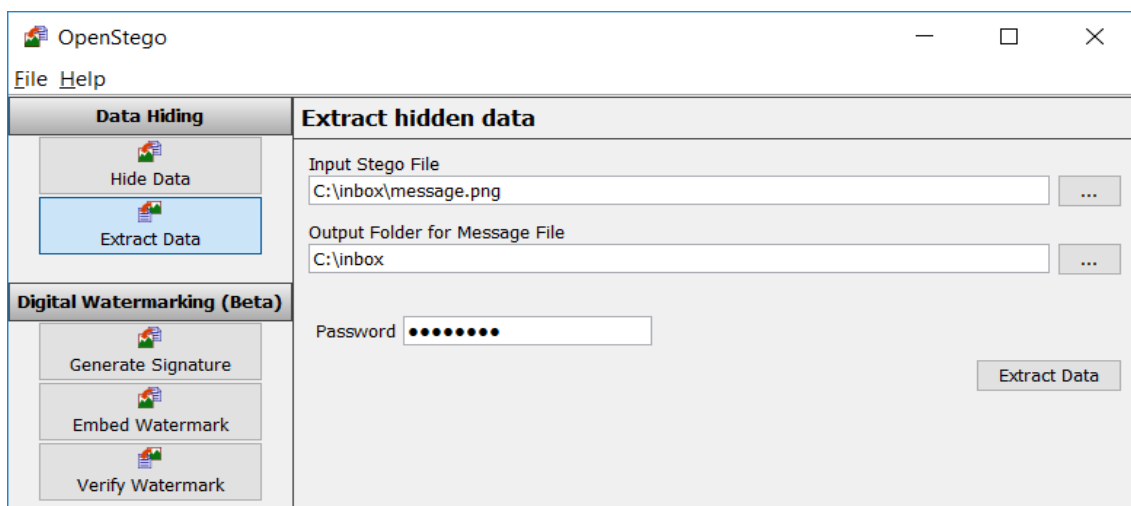
- Click on the “**Embed**” option in the OpenStego interface.
- Under the “Cover File” section, **browse and select** the file in which you want to embed (hide) your data.
- This file is known as the **carrier** or **cover object**.

Step 4: Select the Data to Embed

- Under the “Message File” section, **browse and select** the file containing the **data to be hidden**.
- OpenStego supports embedding various file types such as **text files, images, and other data files**.

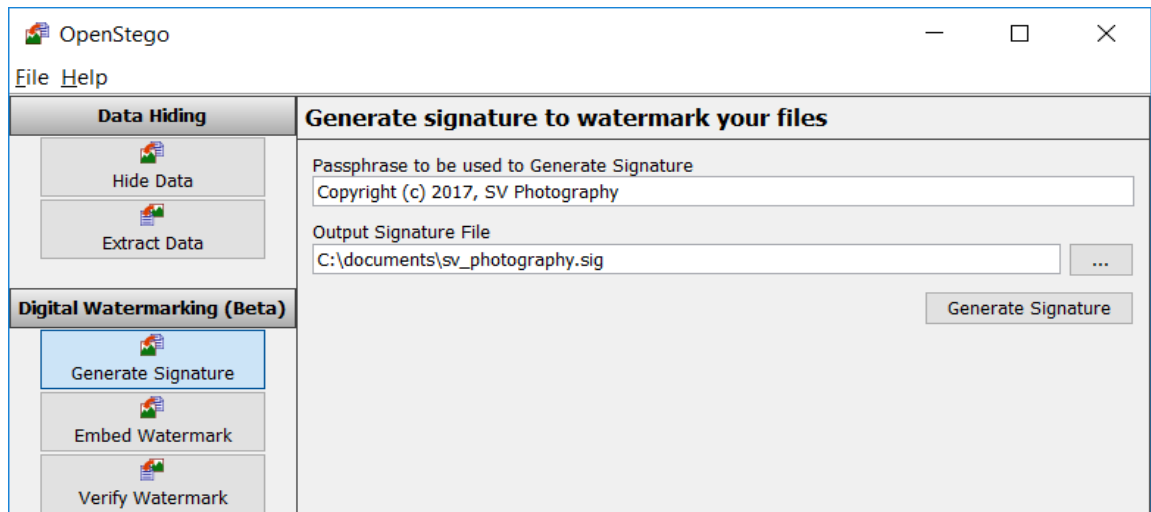
Step 5: Configure Encryption Settings

- OpenStego provides **encryption and password protection options** to secure hidden data.
- Enter a **password/key** for encryption or configure other settings as required.
- You may also specify the **output file name** and location where the stego file will be saved.



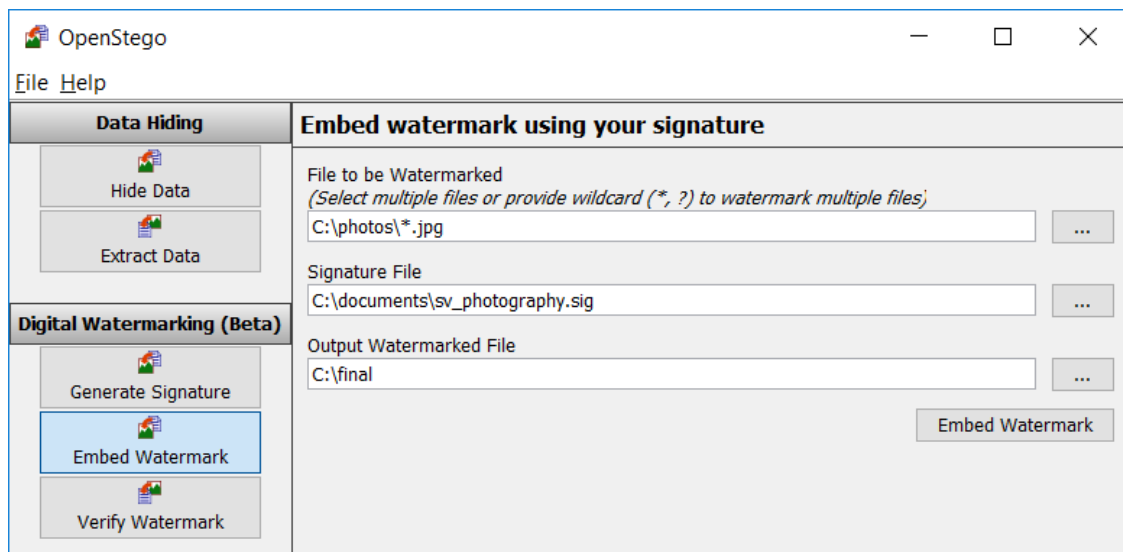
Step 6: Embed the Data

- After configuration, click the “**Embed**” button.
- OpenStego will process the files and generate a **new stego file** that contains the hidden data.
- A confirmation message will appear once the embedding process is completed successfully.



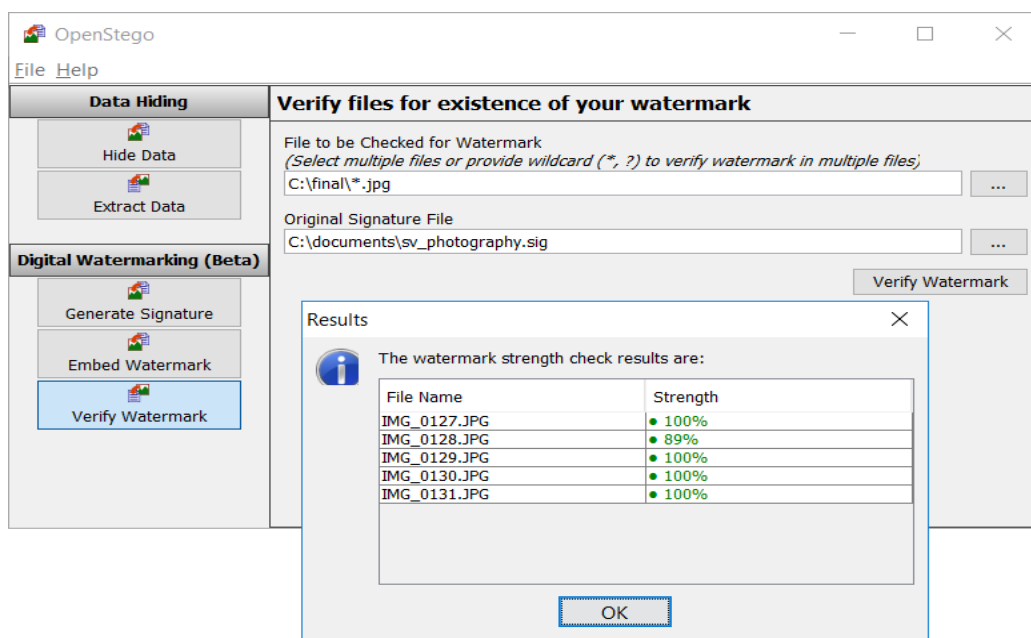
Step 7: Extract the Hidden Data

- To retrieve the hidden data, go back to the main screen and click the “**Extract**” button.
- In the extraction window:
 - Select the **stego file** (the one that contains the hidden data).
 - Specify the **output location** where extracted data will be saved.
- Click “**Extract**” to begin the extraction process.



Step 8: View Extracted Data

- Once extraction is complete, OpenStego will save the **recovered hidden file** in the specified folder.
- Open the extracted file to verify that the data has been retrieved successfully.



IX. Precautions

1. Use high-quality image files (e.g., .png) for better embedding capacity.
2. Avoid compressing stego files after encoding (may damage hidden data).
3. Keep original and stego files secure and clearly labeled.
4. Do not share passwords or hidden data publicly.

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Describe steganography. How does it differ from cryptography?
2. Mention three common file formats used in steganography.
3. Explain the two main functions of OpenStego.
4. What precautions should be taken when using steganography tools?
5. Perform encoding and decoding using OpenStego and record your observations.

Space for Answer

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

[illegible]

.....

.....

.....

.....

.....

.....

.....

XI. Exercise

1. Encode and decode a message using OpenStego with different file formats (e.g., .bmp, .png).
2. Compare stego file sizes before and after embedding.
3. Research and report on the use of steganography in digital watermarking.

XII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. NPTEL Course: Introduction to Cryptography and Network Security.
4. OpenStego Official Website: <https://www.openstego.com>
5. Virtual Labs (IIIT Hyderabad): <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct encoding (hiding) of message using OpenStego	20 %
2	Accurate decoding and retrieval of hidden message	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Correctness and clarity of hidden message	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 12: Create and verify Digital Signature using any Open-source tool (Example – CrypTool).

I. Practical Significance

This practical introduces the concept of digital signatures, a cryptographic technique used to ensure the authenticity, integrity, and non-repudiation of digital messages or documents. Students will learn how digital signatures work using public and private key pairs and practice creating and verifying digital signatures using open-source tools such as CrypTool.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO4 – Use tools and techniques to prevent cyber attacks.

IV. Laboratory Learning Outcome (LLO)

LLO 12.1: Generate digital signature.

V. Relevant Affective domain related Outcomes(s)

1. Responsibility: Correct use of cryptographic tools to ensure message authenticity.
2. Ethical Awareness: Understanding how digital signatures help prevent data forgery and impersonation.
3. Analytical Skills: Interpreting and validating digital signature verification outputs.

VI. Theoretical Background

A Digital Signature is a mathematical technique used to validate the authenticity, integrity, and non-repudiation of a message, software, or digital document. It provides a secure and verifiable way of confirming that a message originates from a verified sender and has not been altered during transmission. Digital signatures are a crucial component of modern cryptographic communication systems and are widely used in secure email, software distribution, electronic transactions, and e-governance.

A digital signature uses public key cryptography (asymmetric encryption), which involves a private key and a public key:

The private key is known only to the sender and is used to create the signature.

The public key is shared with the receiver and is used to verify the signature.

A hash function is applied to the message to create a message digest, which is a fixed-length unique representation of the data. This digest is then encrypted using the sender's private key to form the digital signature.

At the receiver's end, the signature is decrypted using the sender's public key, and the resulting hash is compared with a newly computed hash of the received message. If both match, the message is verified as authentic and unaltered.

Structure of a Digital Signature Scheme

A digital signature scheme typically consists of **three core algorithms**:

1. Key Generation Algorithm:

- Selects a **private key** uniformly at random from a set of possible keys.
- Generates a corresponding **public key**.
- Ensures both keys are mathematically linked but computationally infeasible to derive one from the other.

2. Signing Algorithm:

- Generates a **one-way hash** of the message using a hashing function (e.g., SHA-256).
- Encrypts this hash using the sender's **private key** to produce a **digital signature**.
- This signature, along with the message, is transmitted to the receiver.

3. Verification Algorithm:

- The receiver decrypts the digital signature using the sender's **public key** to retrieve the original hash.
- The receiver computes a new hash from the received message using the same hash function.
- If both hash values are identical, the message is **authentic** and **untampered**.

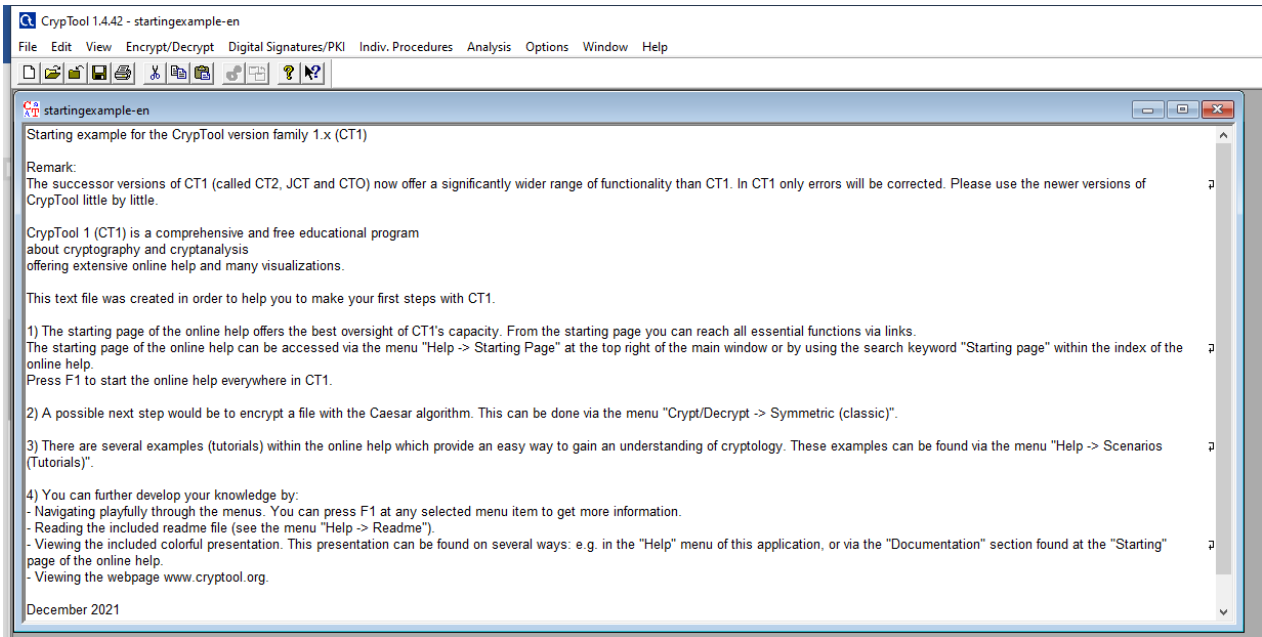
VII. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	CrypTool 1.4.42		

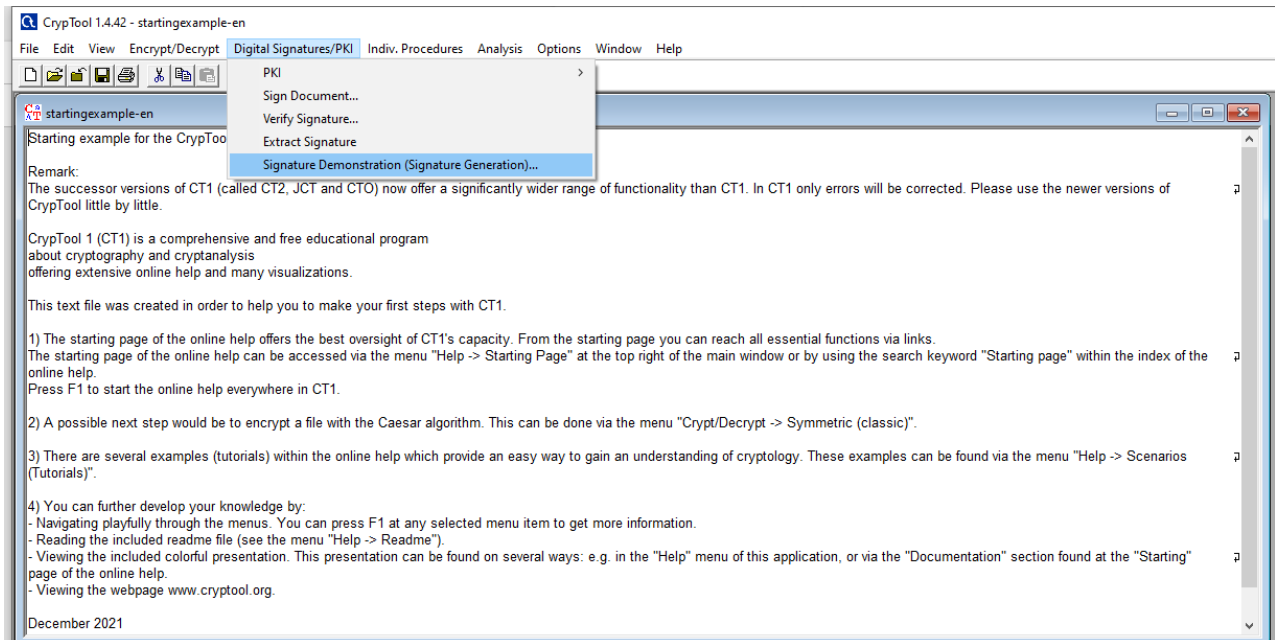
VIII. Procedure

Step-by-Step Procedure

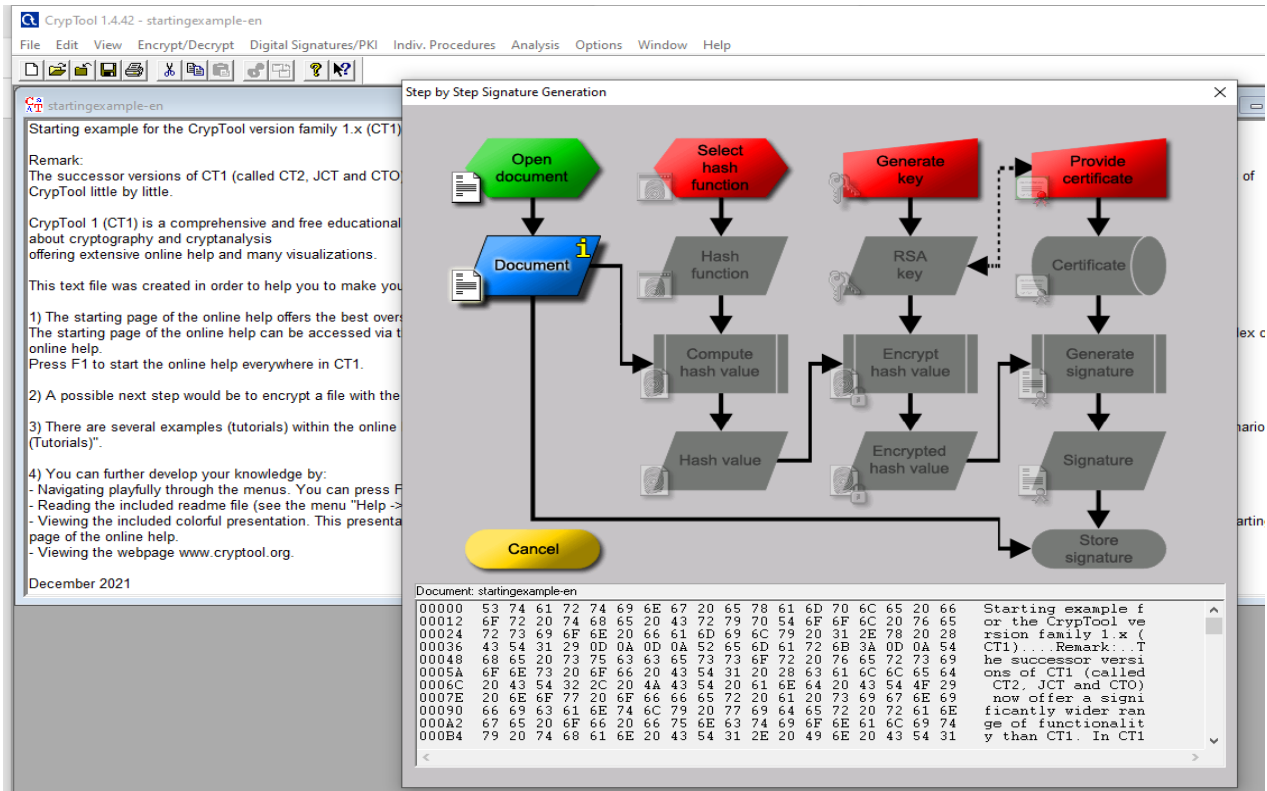
Step 1. Create a new file or open an existing file.



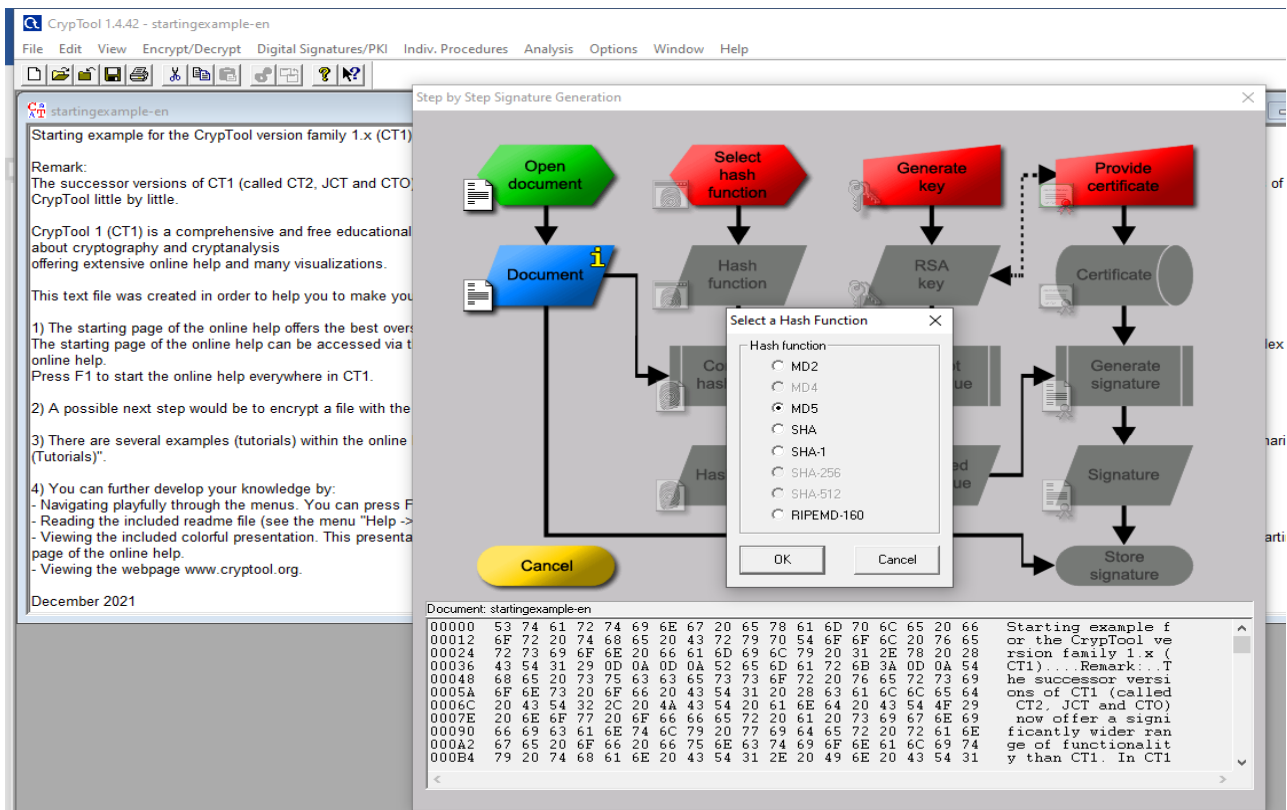
Step 2. From the menu, select Digital Signature/PKI → Signature Demonstration.
The flow chart of the Digital Signature process will be displayed.



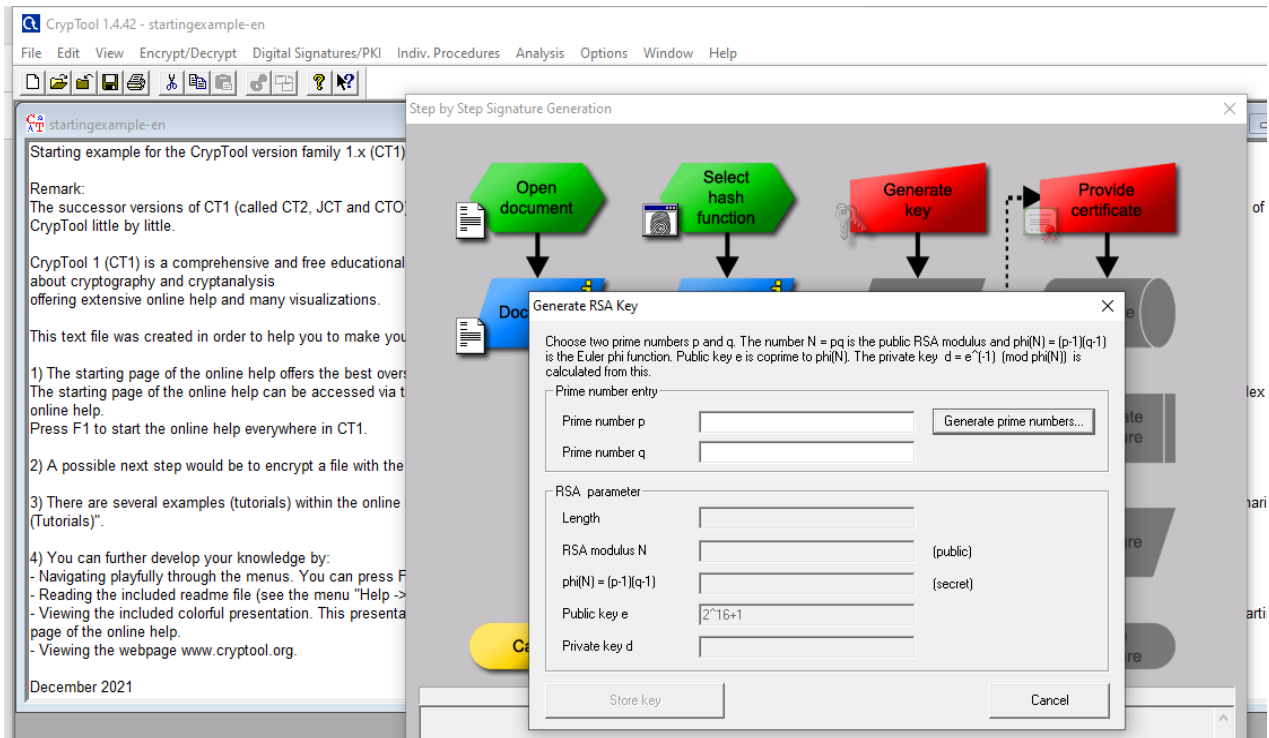
Step 3. Click on Select Hash Function and choose any hash function (e.g., MD5) as per your choice.
Then click OK to confirm the selection.



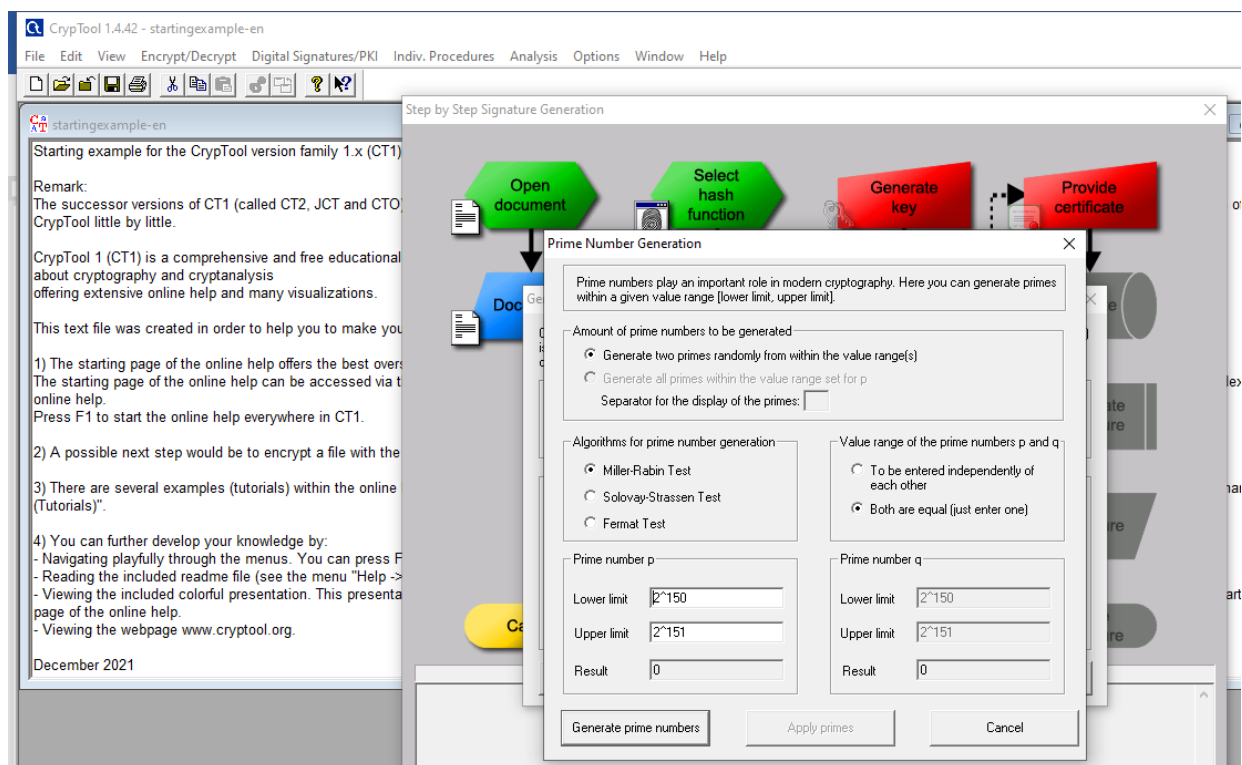
Step 4. Next, click on Generate Key. A Generate RSA Key window will appear. Click on Generate Prime.



Step 5. Select the required options as shown in the demonstration interface.

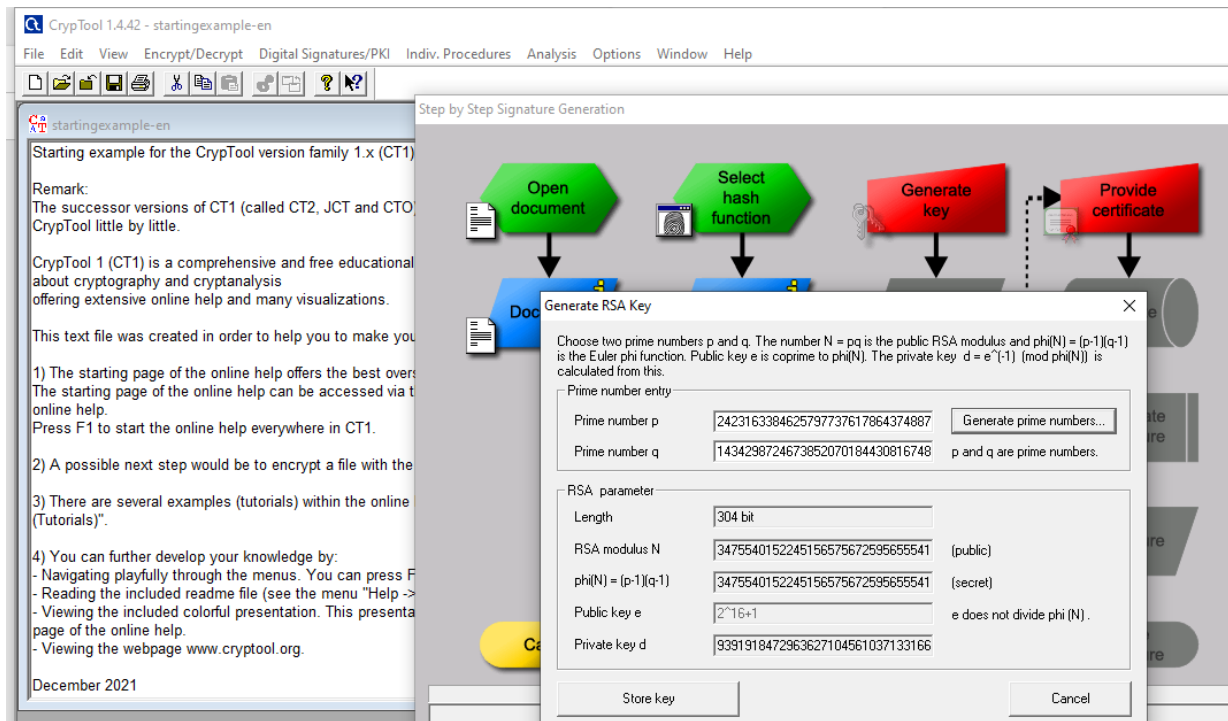


Step 6. Click on Generate Prime Numbers and then click on the Apply Primes button. After this, click on Store Key to save the generated key pair.



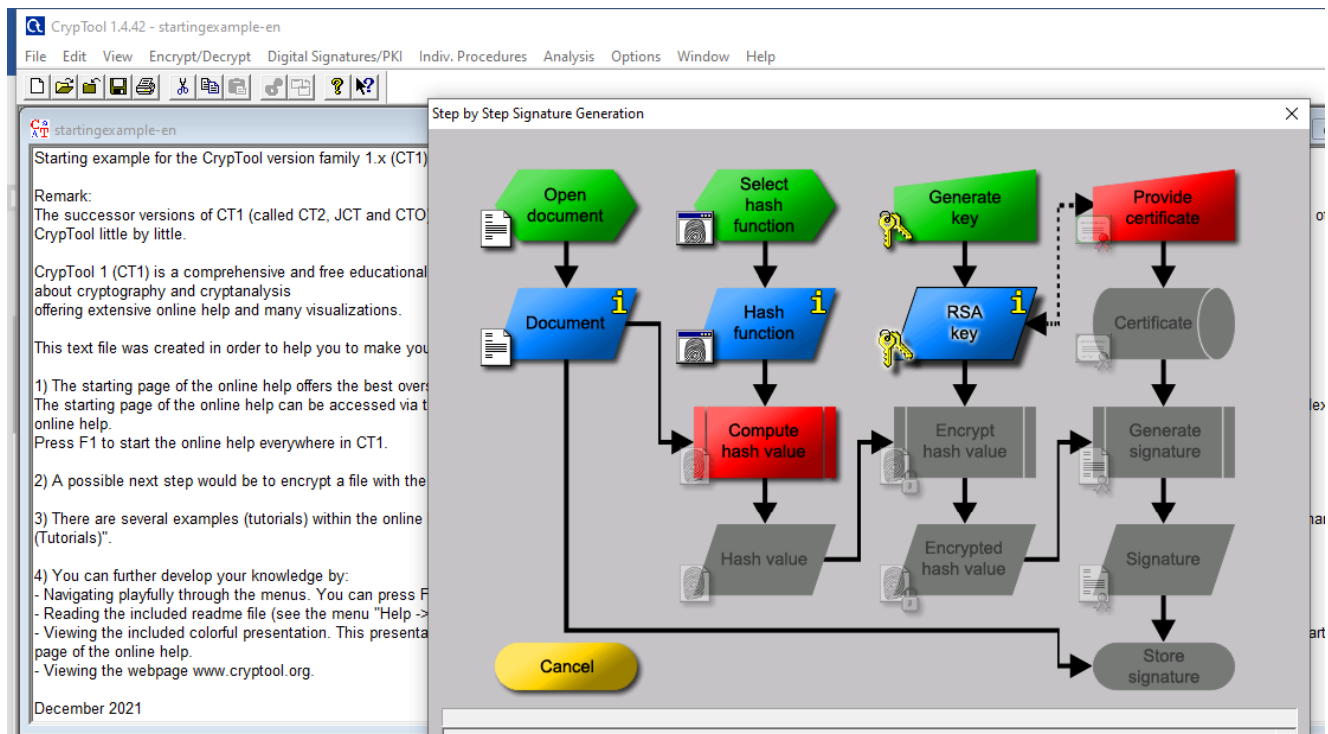
Step 7. In the flow chart, click on Provide Certificate.

The Create Certificate and PSE (Personal Security Environment) window will appear.



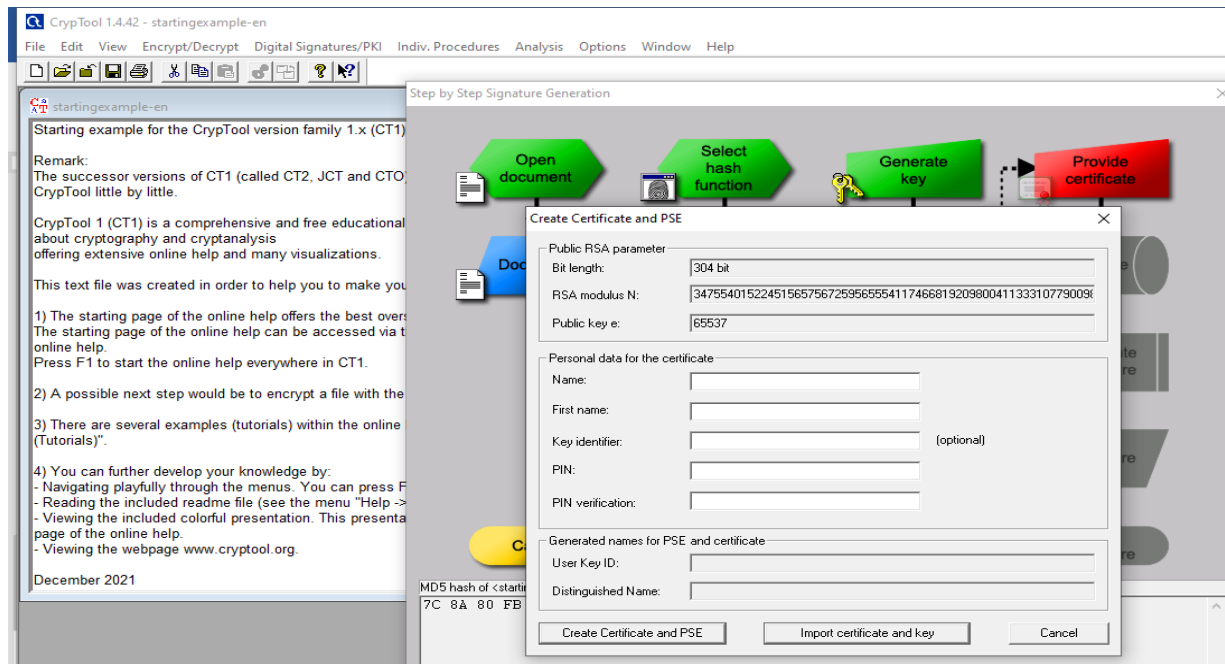
Step 8. Enter the required personal data (name, organization, email, etc.) for the certificate.

Click on the Create Certificate and PSE button to generate your certificate.



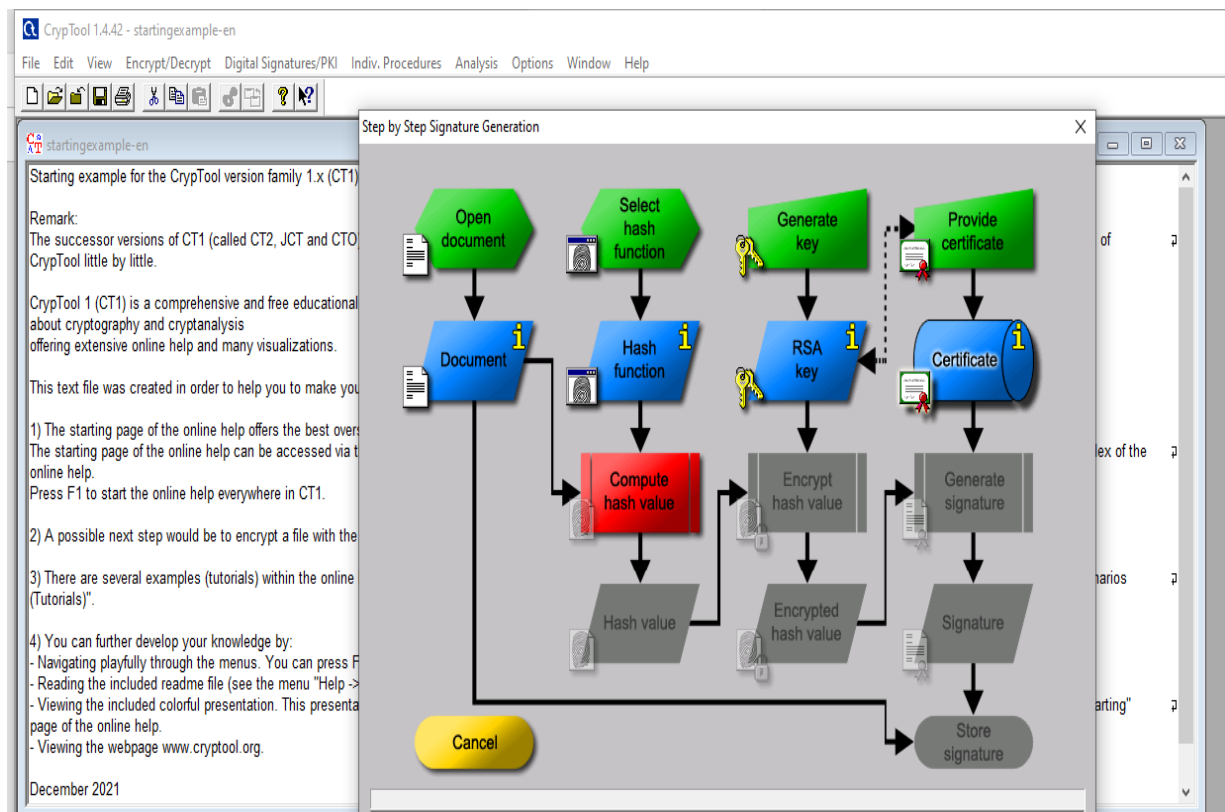
Step 9. On the flow chart, click on Hash Value.

This will compute the hash of your selected data using the chosen hash function.



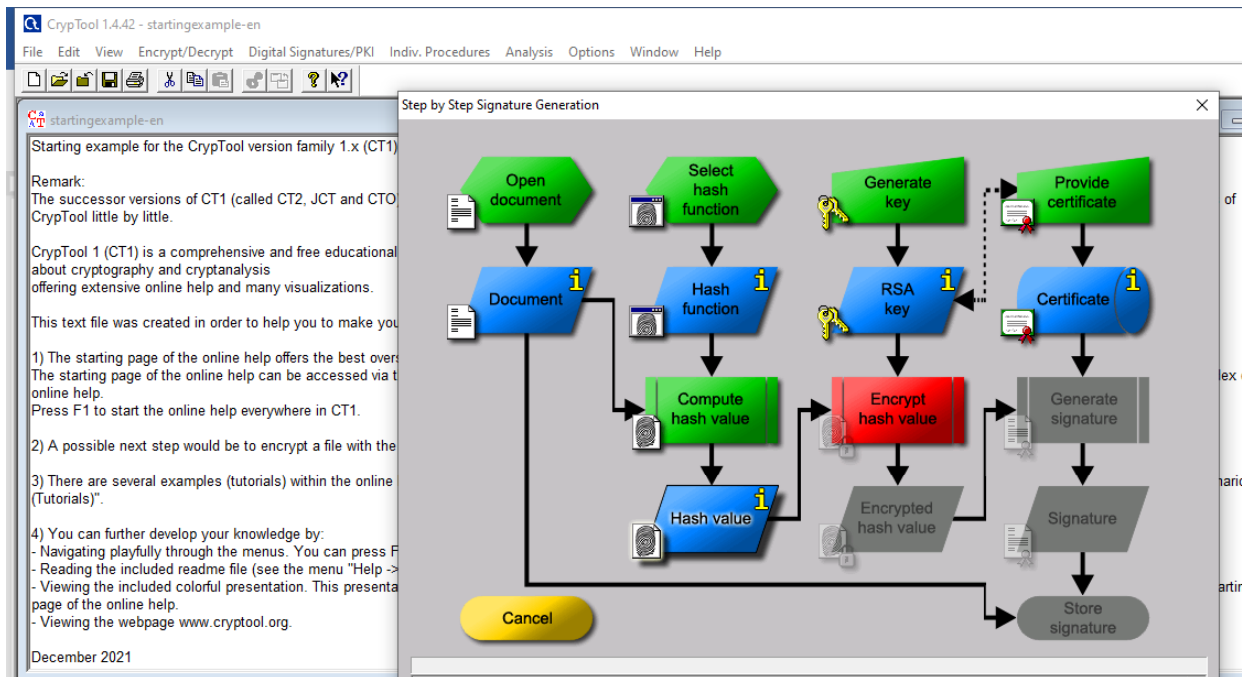
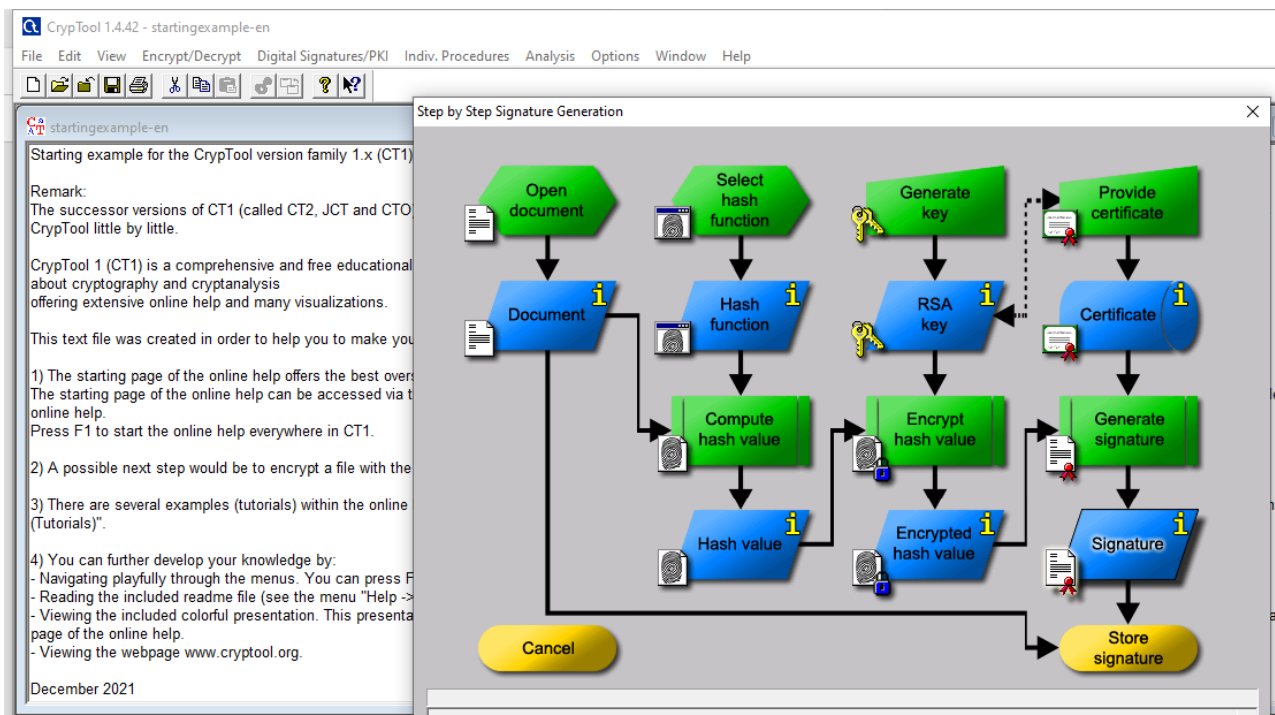
Step 10. Next, click on Encrypt Hash Value.

This step encrypts the computed hash using your private key to generate the digital signature.



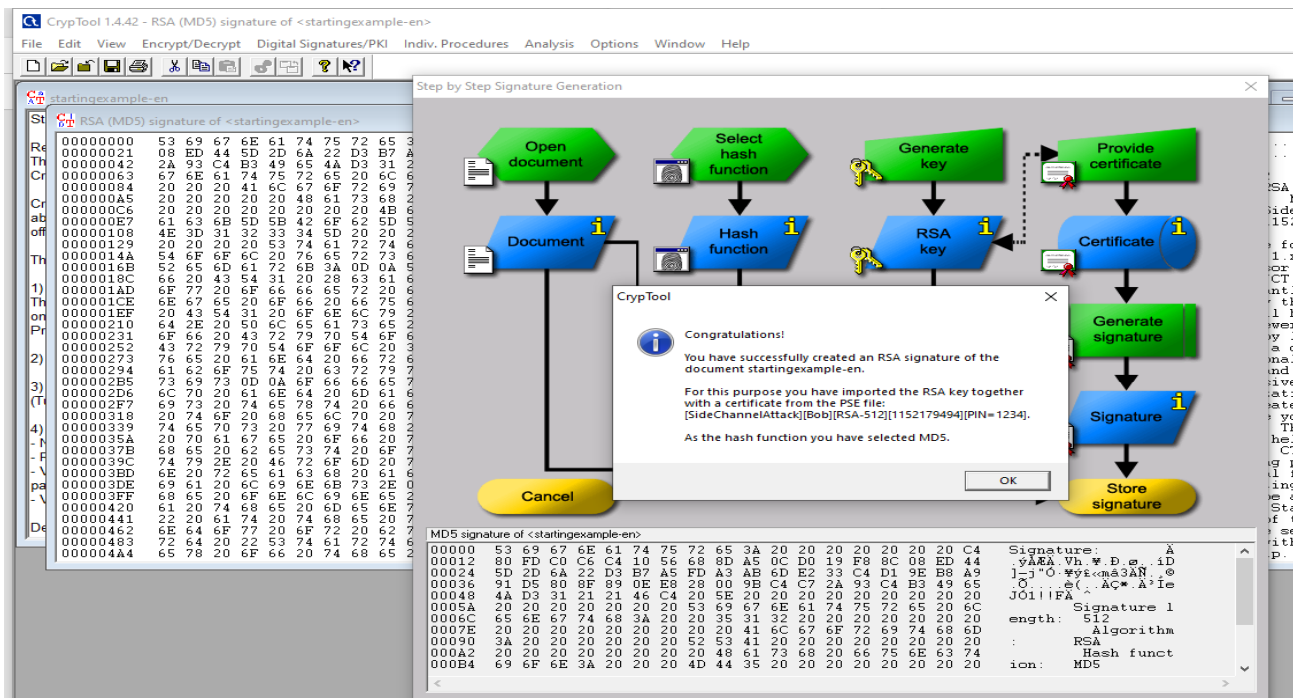
Step 11. Click on Generate Signature.

The tool will create the Digital Signature using the encryption algorithm.

**Step 12.** After the signature is generated, click on Store Signature to save it.**Step 13.** A message box will appear stating:

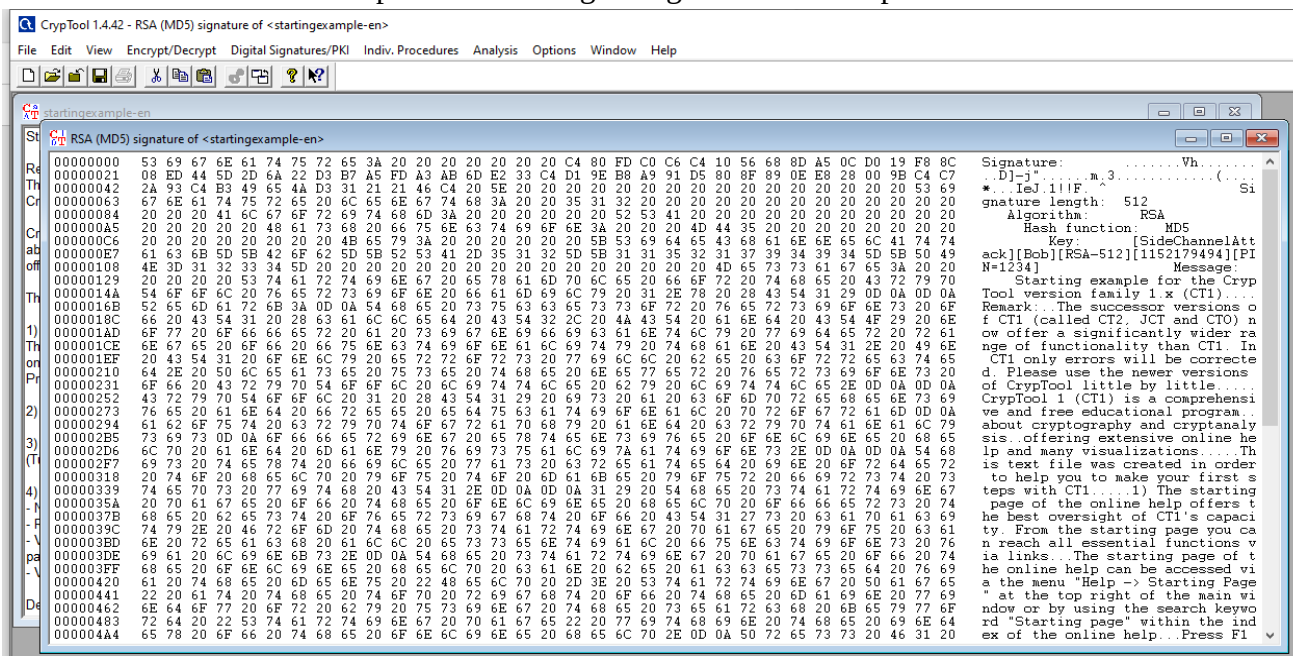
“Congratulations! You have successfully created an RSA Signature.”

Click OK to confirm.



Step 14. Finally, the system will display a window showing the document created with a unique digital signature.

This indicates successful completion of the digital signature creation process.



IX. Precautions

1. Keep your private key confidential.
2. Use strong key lengths (minimum 2048 bits for RSA).
3. Verify both the message and the signature file for integrity.
4. Do not modify the signed message before verification.

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Describe three main objectives of a digital signature?
2. Difference between a digital signature and an electronic signature.
3. What happens if the signed message is altered after signing?
4. Generate and verify a digital signature using RSA and SHA-256 in CrypTool.
5. Write a C program to simulate the generation of a digital signature.

Space for Answer

[illegible]

[illegible]

XI. Exercise

1. Demonstrate digital signature creation and verification using both RSA and DSA in CrypTool.
2. Observe how message alteration leads to signature verification failure.
3. Research and summarize how digital certificates are used to distribute public keys.

XII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. NPTEL Course: Introduction to Cryptography and Network Security.
4. CrypTool Official Website: <https://www.cryptool.org>
5. Virtual Labs, IIIT Hyderabad: <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct generation of digital signature using CrypTool	20 %
2	Verification of digital signature using the corresponding public key	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Accuracy of signature verification results	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 13: Create and verify Digital Certificate using any Open-source tool (Example – CrypTool).

I. Practical Significance

This practical introduces digital certificates, which are electronic credentials issued by a trusted authority (CA – Certificate Authority) to verify the identity of entities involved in digital communication. Students will learn the process of generating, signing, and verifying digital certificates using an open-source cryptographic tool such as CrypTool, gaining a clear understanding of how certificates ensure trust, authentication, and secure communication over the Internet.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO4 – Use tools and techniques to prevent cyber attacks.

IV. Laboratory Learning Outcome (LLO)

LLO 13.1: Generate digital Certificate.

V. Relevant Affective domain related Outcomes(s)

1. Responsibility: Apply tools properly for secure certificate creation and verification.
2. Ethical Awareness: Understand the significance of certificate-based authentication in secure communications.
3. Analytical Skills: Analyze the certificate details to interpret issuer, validity period, and public key information.

VI. Theoretical Background

A **digital certificate** is an electronic document that binds a **public key** with an entity's identity (such as a person, organization, or website).

It is issued and digitally signed by a trusted **Certificate Authority (CA)** as part of the **Public Key Infrastructure (PKI)** system.

A valid digital certificate contains:

- **Subject name** (owner's identity)
- **Public key**
- **Issuer name (CA)**
- **Validity period**
- **Digital signature of CA**

Digital certificates are used in:

- **SSL/TLS** protocols for secure web communication.
- **Email encryption and signing.**

- **Software signing** to prove authenticity of programs.

Types of Certificates:

- **Self-signed certificate** (created and verified locally)
- **CA-signed certificate** (issued by trusted third-party authority)

VII. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows XP / 7 onwards / LINUX V 5.0 or latest		
3	Tools / Software Required	CrypTool 1.4.42		

VIII. Procedure

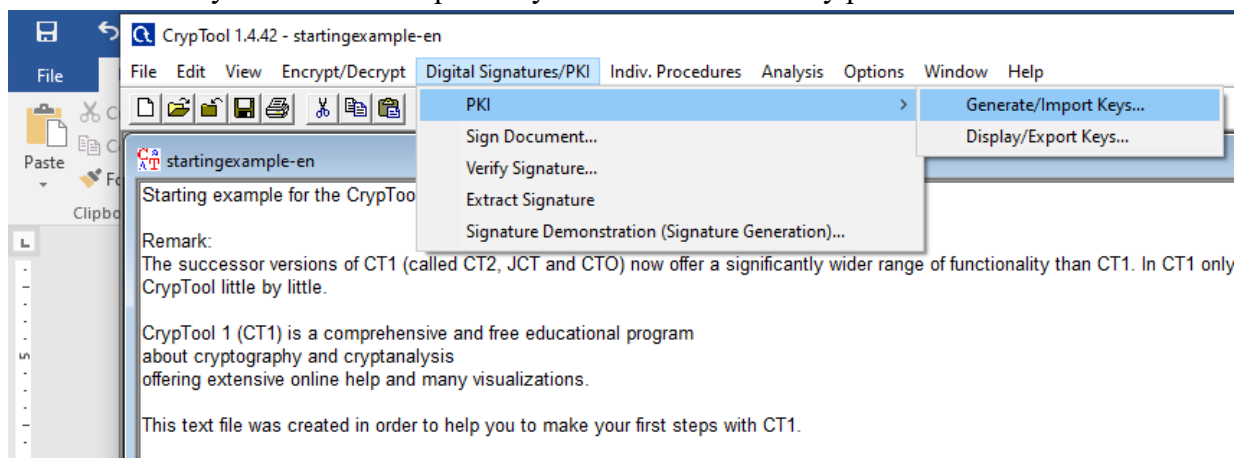
Step-by-Step Procedure

Step 1. Perform all the steps as outlined in Practical No. 12 to generate a Digital Signature.

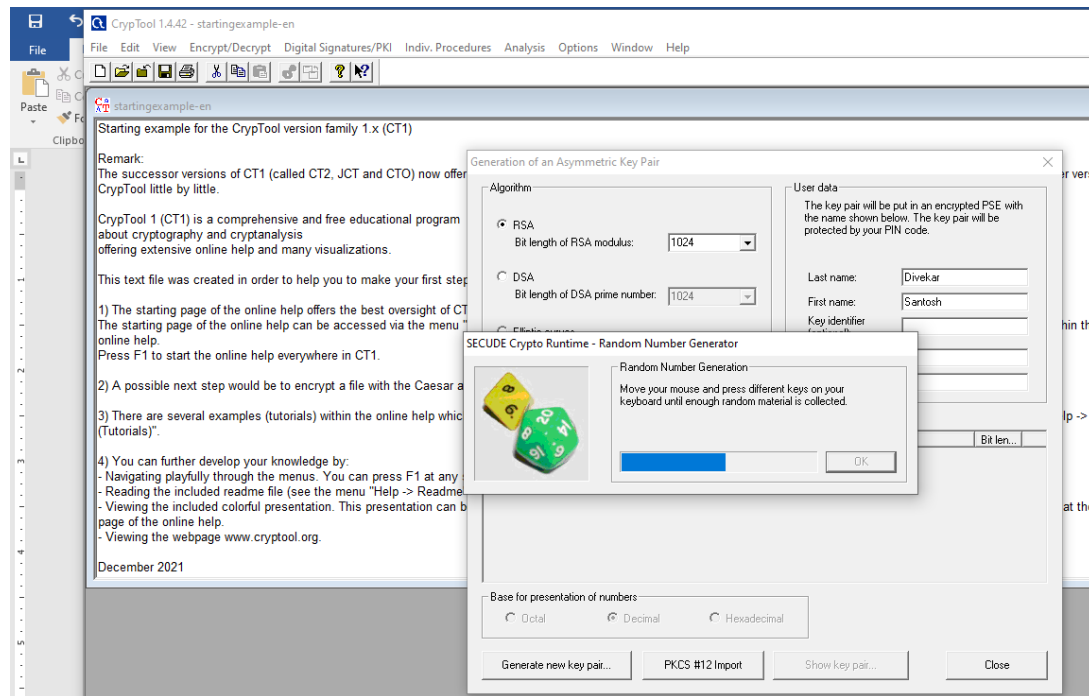
A. Creating Keys and Certificate

1. Open CrypTool 1.4.42.
2. Go to:

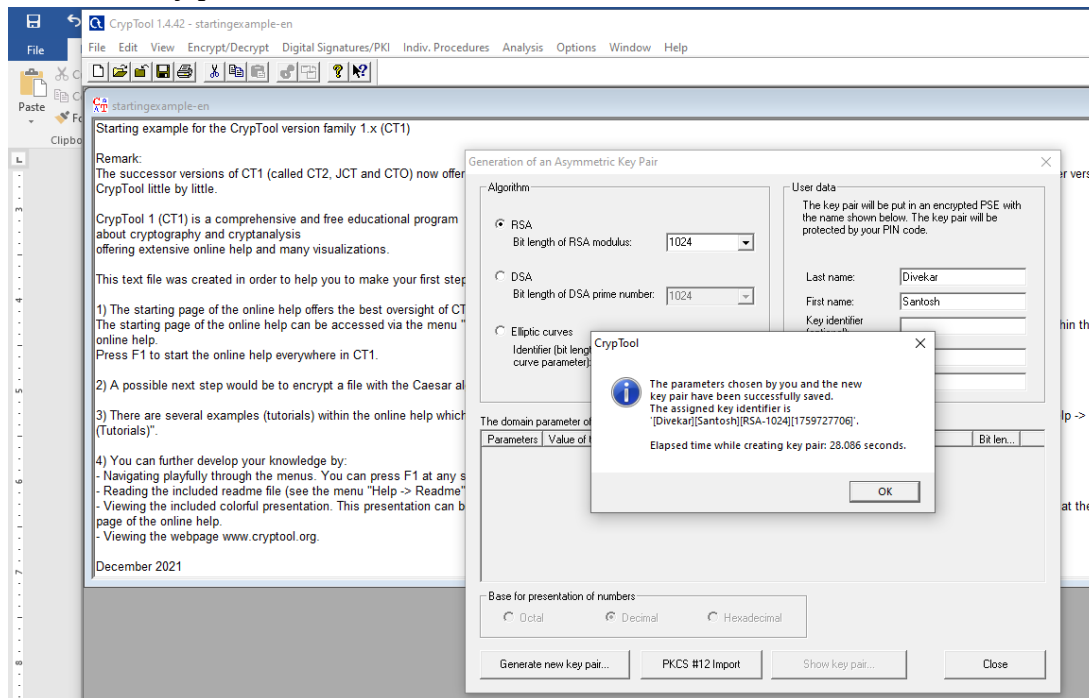
Menu → Public Key → Generate/Import Keys → Generate new key pair



3. Choose algorithm (e.g., RSA).
4. Enter:
 - Owner's name
 - Email address
 - Key length (e.g., 1024 bits)



5. Click Generate key pair.

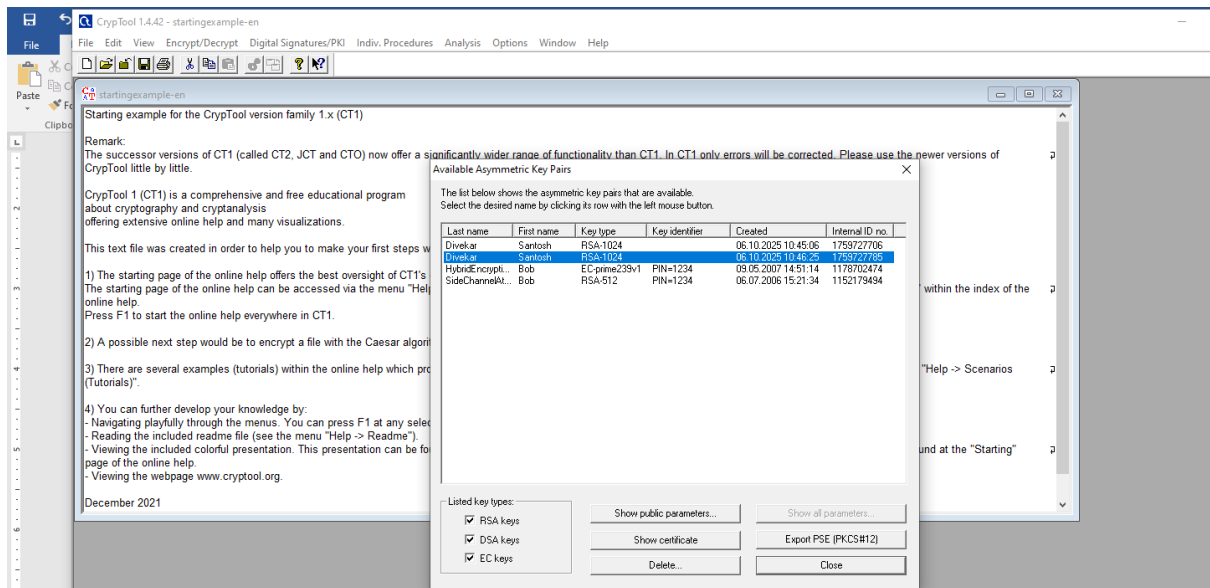


6. Now go to:

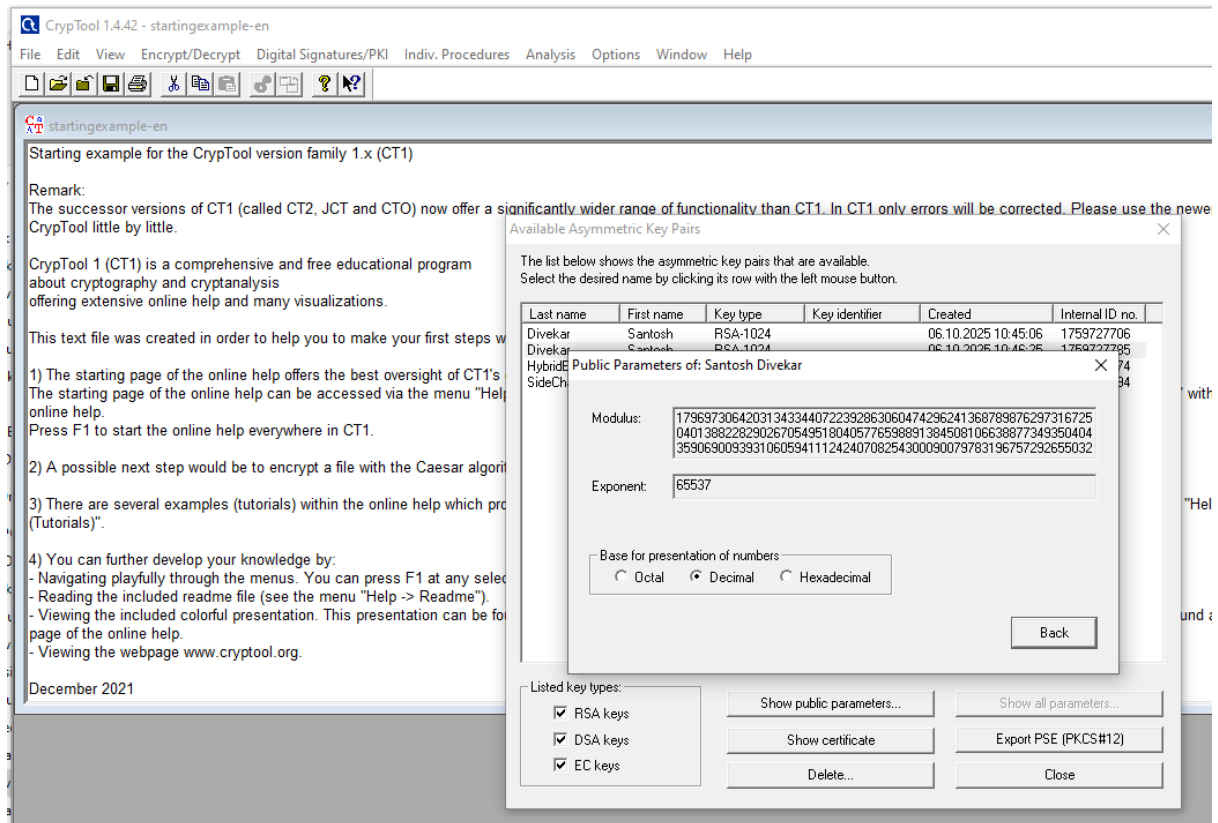
Menu → Public Key → Certificates → Create X.509 Certificate

7. Fill in the certificate details:

- Owner name
- Organization
- Validity period

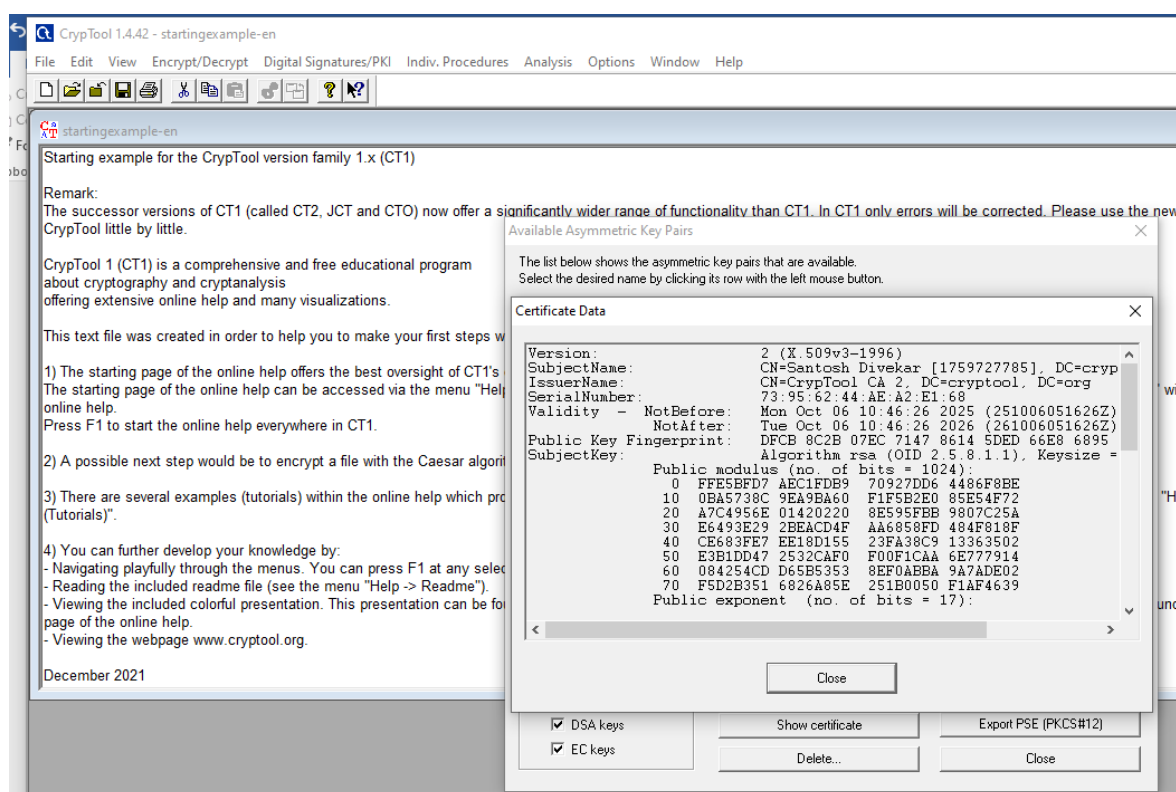


8. Select the private key you just generated.



9. Click Sign Certificate.

CrypTool will use your private key (or a CA's private key) to sign the certificate.



10. Save the generated certificate (e.g., user_cert.cer).

IX. Precautions

1. Ensure the private key is securely stored and not shared.
2. Use valid certificate parameters (e.g., key size ≥ 2048 bits).
3. Check the validity period and issuer name before using certificates.
4. Do not modify certificate contents manually.

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Describe digital certificate. How does it differ from a digital signature?
2. Explain the role of a Certificate Authority (CA) in PKI.
3. How does a certificate ensure authenticity and trust in communication?
4. Create a self-signed certificate using CrypTool and verify its validity.
5. Identify key fields in a digital certificate (Issuer, Subject, Validity, Public Key, Signature).

Space for Answer

.....

.....

.....

.....

.....

This image shows a full page of white paper with horizontal dotted lines. The lines are evenly spaced and run across the width of the page, providing a guide for handwriting practice. There are no margins, text, or other markings on the page.

.....

.....

.....

.....

XI. Exercise

1. Create and verify both a self-signed and CA-signed certificate using CrypTool.
2. Analyze certificate details and validity information.
3. Research how SSL/TLS uses certificates for website authentication.

XII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. NPTEL Course: Introduction to Cryptography and Network Security.
4. CrypTool Official Website: <https://www.cryptool.org>
5. Virtual Labs, IIIT Hyderabad: <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html> .

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct creation of digital certificate using CrypTool	20 %
2	Verification and validation of certificate authenticity	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Accuracy of certificate details (validity, issuer, subject)	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 14: Configure Firewall Settings on any Operating System.

I. Practical Significance

This practical introduces the concept and configuration of a firewall, an essential security mechanism used to control incoming and outgoing network traffic. Students will learn how to configure firewall rules, ports, and permissions on an operating system to protect against unauthorized access and cyberattacks. The goal is to understand how firewalls safeguard systems by applying predefined security policies.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO4 – Use tools and techniques to prevent cyber attacks.

IV. Laboratory Learning Outcome (LLO)

LLO 14.1: Configure firewall.

V. Relevant Affective domain related Outcomes(s)

1. Responsibility: Properly manage system security configurations.
2. Ethical Awareness: Recognize the importance of safe and authorized network configurations.
3. Analytical Skills: Analyze the effect of firewall rules on network connectivity.

VI. Theoretical Background

A **firewall** is a network security system that monitors and controls incoming and outgoing network traffic based on predetermined security rules.

It acts as a barrier between a **trusted internal network** and an **untrusted external network (such as the Internet)**.

Types of Firewalls:

- **Packet Filtering Firewall:** Filters packets based on source/destination IP address, ports, and protocol.
- **Stateful Firewall:** Tracks the state of active connections and allows only legitimate packets.
- **Application Layer Firewall:** Filters traffic based on specific applications (e.g., HTTP, FTP).
- **Next-Generation Firewall (NGFW):** Integrates deep packet inspection, intrusion prevention, and application control.

Basic Firewall Operations:

1. Allow or block specific network traffic.
2. Control access to ports and protocols.
3. Prevent unauthorized inbound or outbound connections.
4. Log and monitor suspicious network activities.

Firewalls can be configured using **graphical interfaces** (Windows Firewall) or **command-line tools** (e.g., ufw in Linux).

VII. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB)	As per batch Size	--
2	Operating System	Windows 10 / 11 / Linux (Ubuntu/CentOS)		
3	Tools / Software Required	Built-in Firewall Tools (Windows Defender Firewall / UFW)		

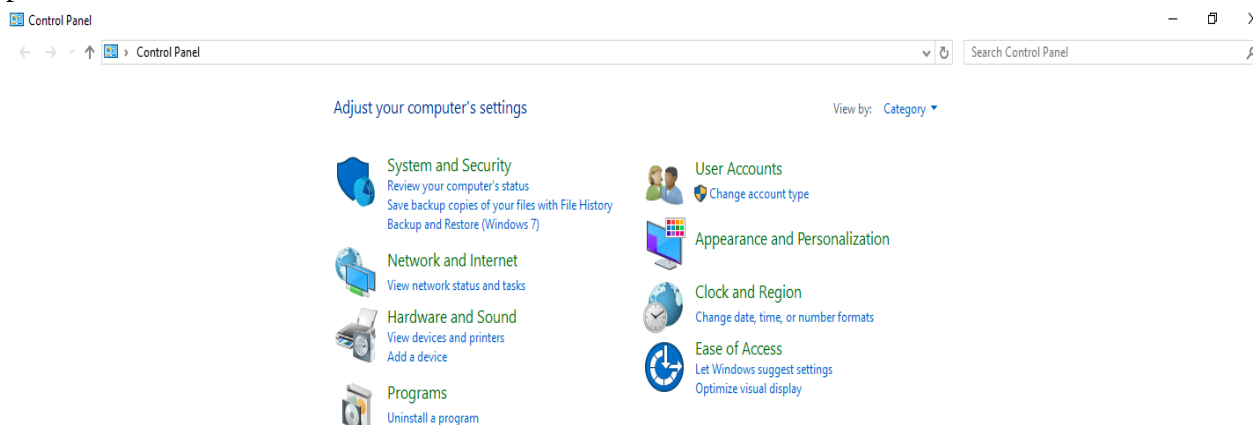
VIII. Procedure

Step-by-Step Procedure

A. Install Firewall on Windows Operating System

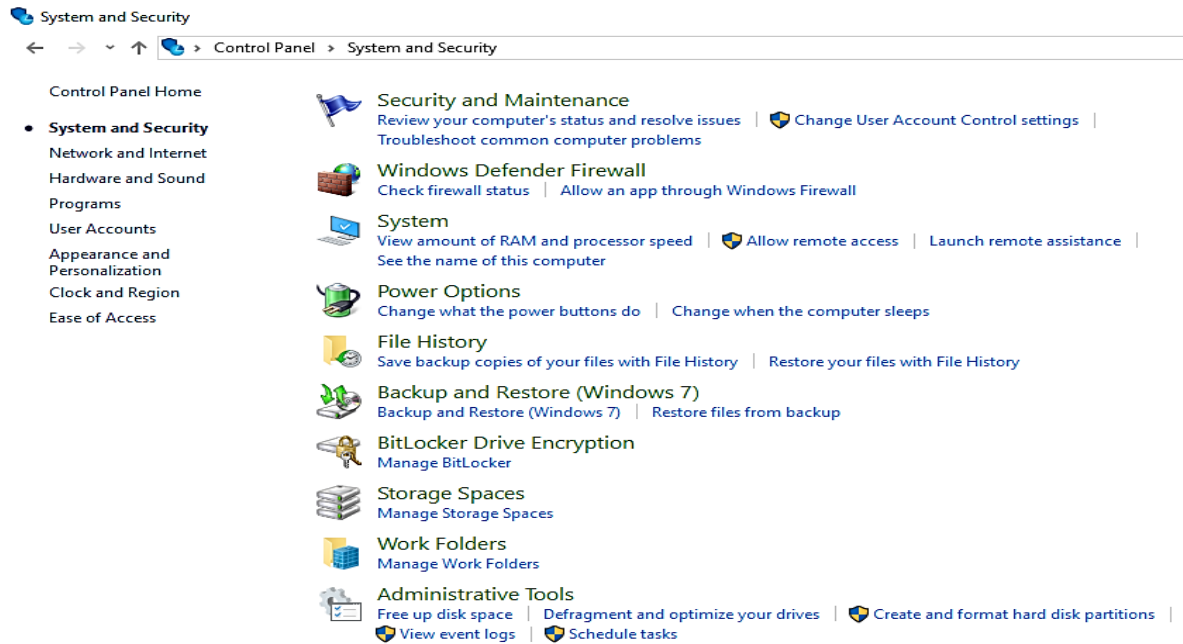
Step 1:

Open the Start Menu and click on Control Panel.



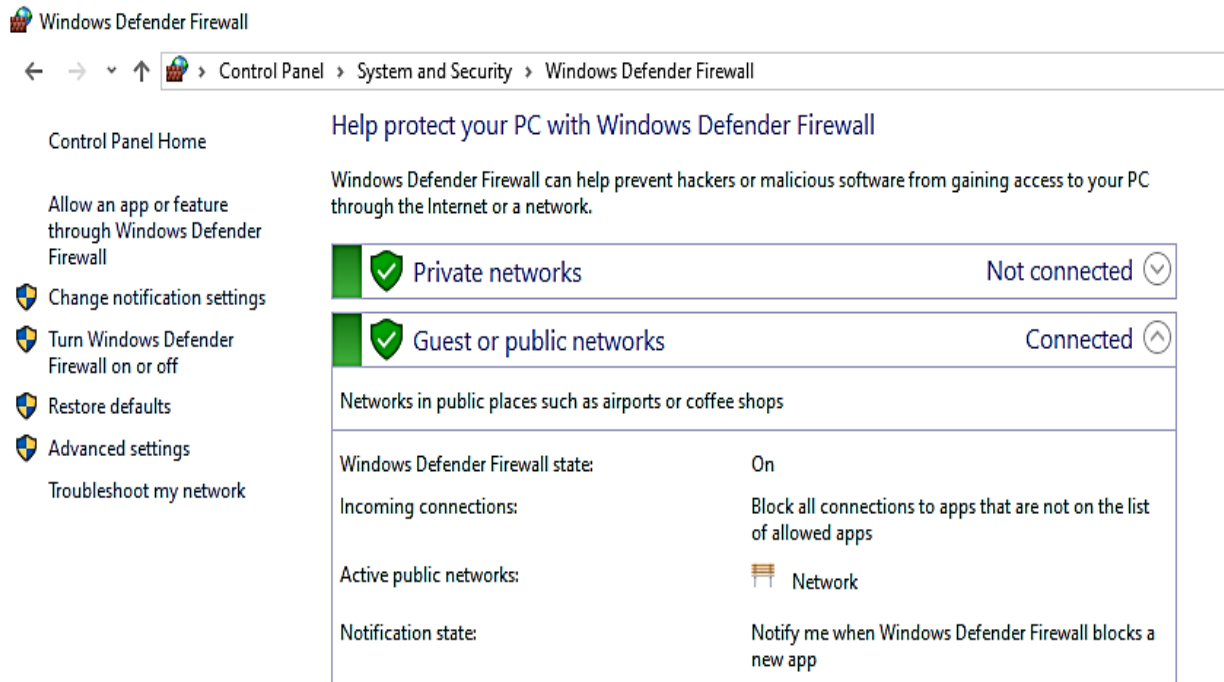
Step 2:

In the Control Panel window, select System and Security.

**Step 3:**

Click on “Windows Defender Firewall” option.

Then, click on “Turn Windows Defender Firewall on or off” from the left panel.



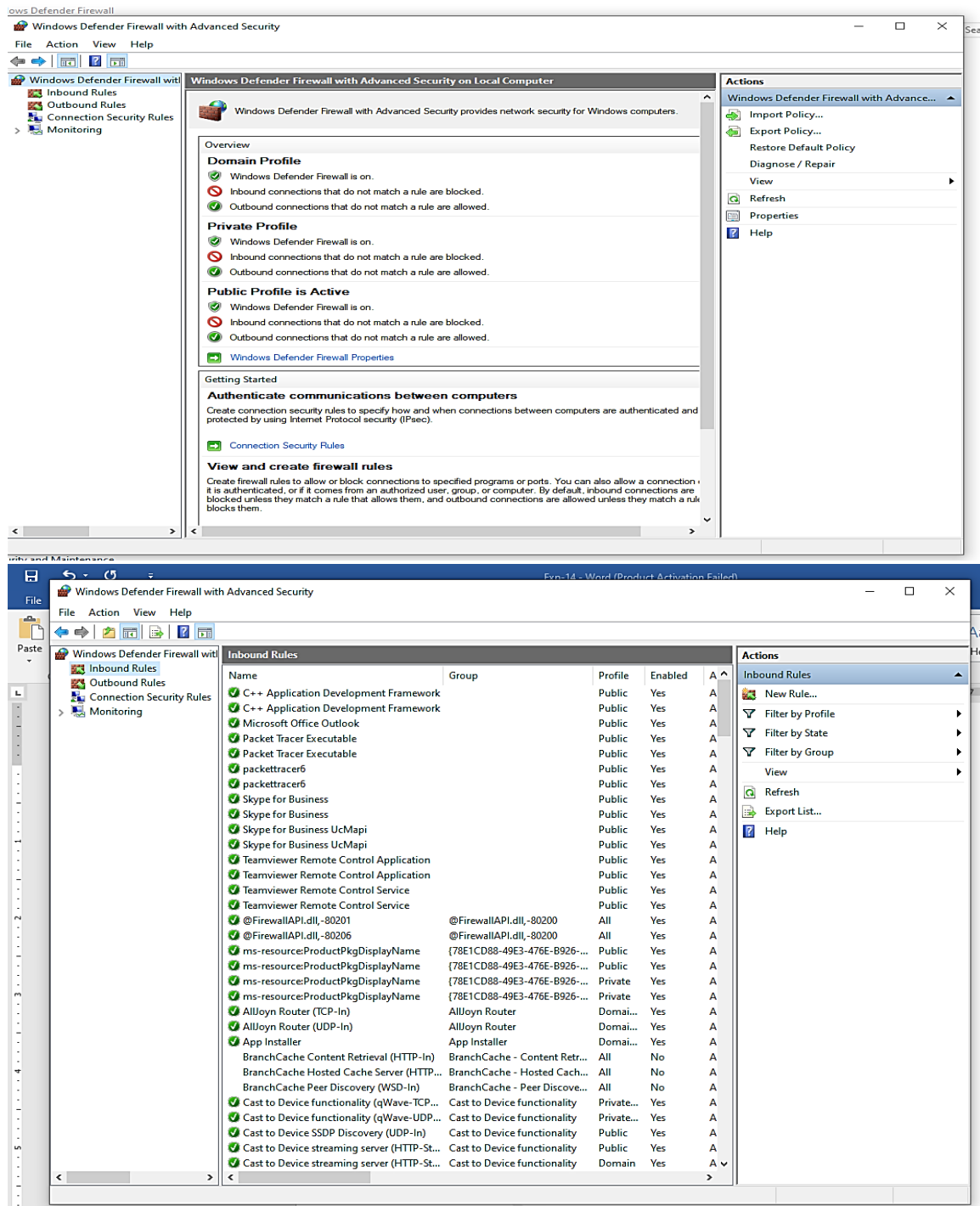
Step 4:

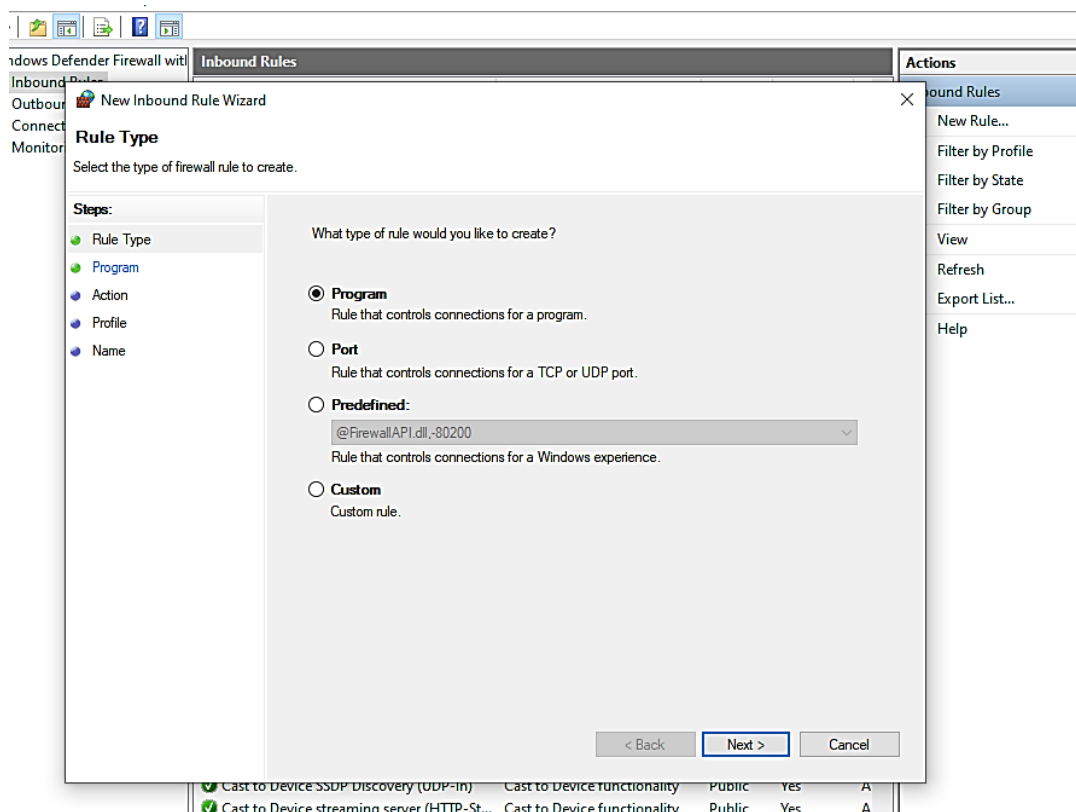
Choose Turn on Windows Defender Firewall under both Private and Public network settings. Click on OK to apply changes.

Step 5:

Modify settings as per requirement (for example, allow or block incoming connections, enable notifications, etc.).

Finally, click OK to save the configuration.





IX. Precautions

1. Always test rules before applying them in production systems.
2. Ensure essential services (like SSH or RDP) are not permanently blocked.
3. Keep a backup of existing firewall configurations.
4. Avoid disabling the firewall unless absolutely necessary.

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. What is the purpose of a firewall in a network?
2. Differentiate between hardware and software firewalls.
3. Write any two advantages of a stateful firewall.
4. List the steps to allow or block a specific port in Windows Firewall.

Space for Answer:

.....

.....

.....

.....

.....

[illegible]

.....

.....

.....

.....

.....

XI. Exercise

1. Configure a rule to block all incoming ICMP (ping) requests.
2. Create a rule to allow only port 443 (HTTPS) communication.
3. Compare the operation of Windows Firewall and Linux UFW using command outputs.

XII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. NPTEL Course: Introduction to Information Security.
4. Microsoft Documentation: Windows Defender Firewall with Advanced Security.
5. Ubuntu Documentation: UFW — Uncomplicated Firewall.

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct configuration of firewall rules on chosen OS	20 %
2	Testing and verifying allowed and blocked connections	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Correctness and effectiveness of firewall settings	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 15: Send a Test Mail Securely using any Open-source Tool (Example – Pretty Good Privacy with GnuPG).

I. Practical Significance

This practical introduces email security mechanisms using Pretty Good Privacy (PGP) or GNU Privacy Guard (GnuPG) to ensure confidentiality, integrity, and authenticity of email communication. Students will learn how to encrypt, decrypt, sign, and verify emails using public and private key pairs. The experiment helps in understanding how cryptographic protection safeguards emails from eavesdropping and tampering.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO5 – Apply security on Network and Database.

IV. Laboratory Learning Outcome (LLO)

LLO 15.1: Implement email security.

V. Relevant Affective domain related Outcomes(s)

1. **Responsibility:** Apply correct encryption practices while handling confidential information.
2. **Ethical Awareness:** Recognize the importance of secure and authentic digital communication.
3. **Analytical Skills:** Verify message authenticity and understand key-based security mechanisms.

VI. Theoretical Background

Email security is the practice of protecting email content and user accounts against unauthorized access, loss, or compromise.

PGP (Pretty Good Privacy) and **GnuPG (GNU Privacy Guard)** are encryption programs that use a combination of **symmetric** and **asymmetric cryptography** to secure email communication.

Working of PGP/GnuPG:

1. **Key Generation:** Each user generates a pair of keys — a **public key** (shared) and a **private key** (kept secret).
2. **Encryption:** The sender encrypts the message using the receiver's public key.
3. **Decryption:** The receiver decrypts the message using their private key.
4. **Digital Signature:** The sender signs the message with their private key to ensure authenticity.
5. **Verification:** The receiver verifies the signature using the sender's public key.

Email Security Features:

- **Confidentiality:** Message encryption ensures only the intended recipient can read it.
- **Integrity:** Digital signatures ensure the message has not been altered.
- **Authentication:** Verifies sender identity using key-based signing.

- **Non-repudiation:** Sender cannot deny sending the signed message.

Common Tools:

- GnuPG (command-line tool)
- Mozilla Thunderbird with Enigmail plugin
- PGP Desktop

VII. Resources required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB	As per batch Size	--
2	Operating System	Windows 10 / 11 / Linux (Ubuntu/CentOS)		
3	Tools / Software Required	GnuPG / PGP / Thunderbird with Enigmail		

VIII. Procedure

Step-by-Step Procedure

Step 1:

To install GPG on Windows, download and install the “**Kleopatra**” application from:

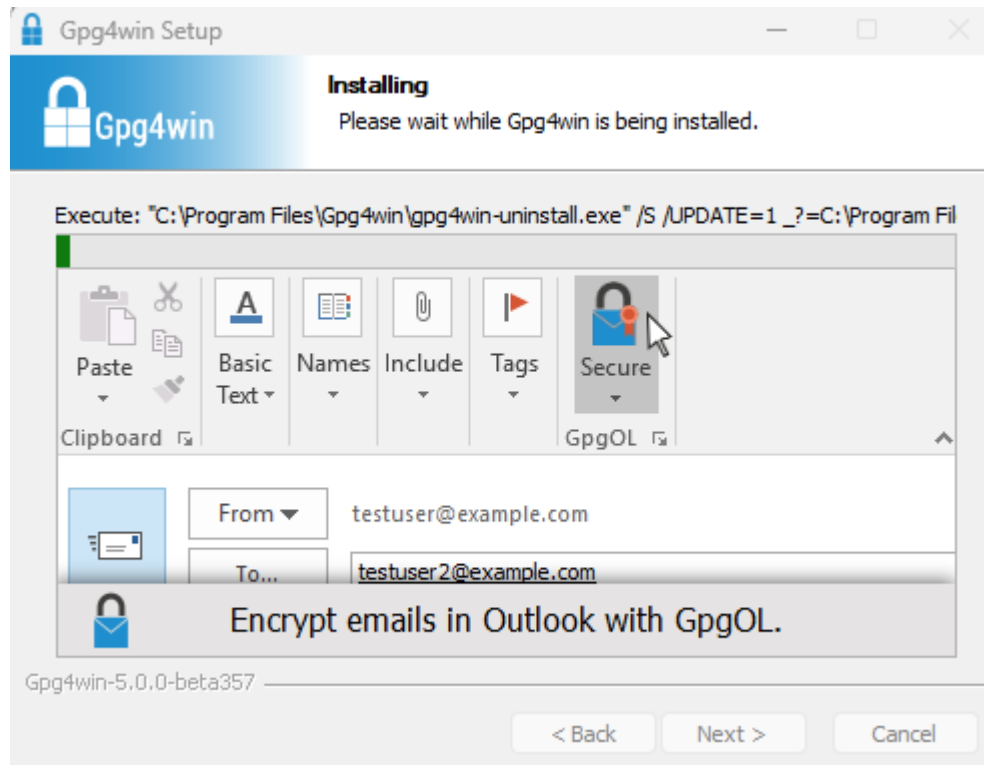
<https://www.gpg4win.org/download.html>

After installation, **start the Kleopatra application.**



Step 2:

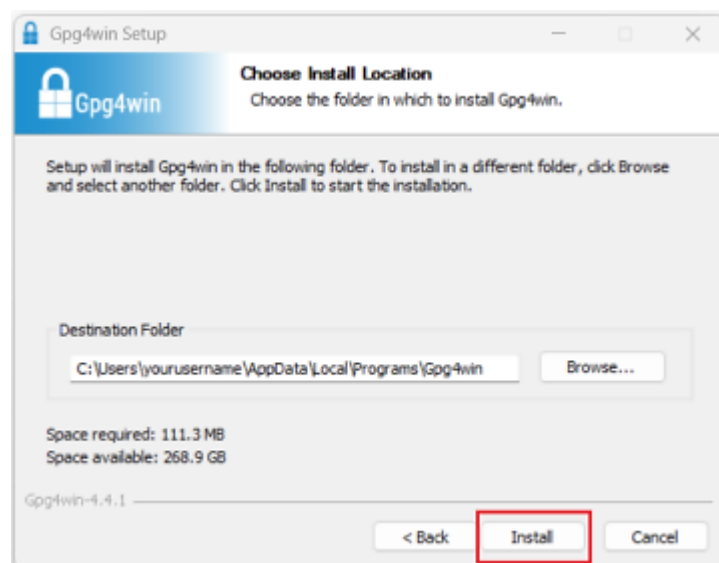
After launching the installer, click on **Next** to continue.

**Step 3:**

Specify the **destination folder path** for installation.

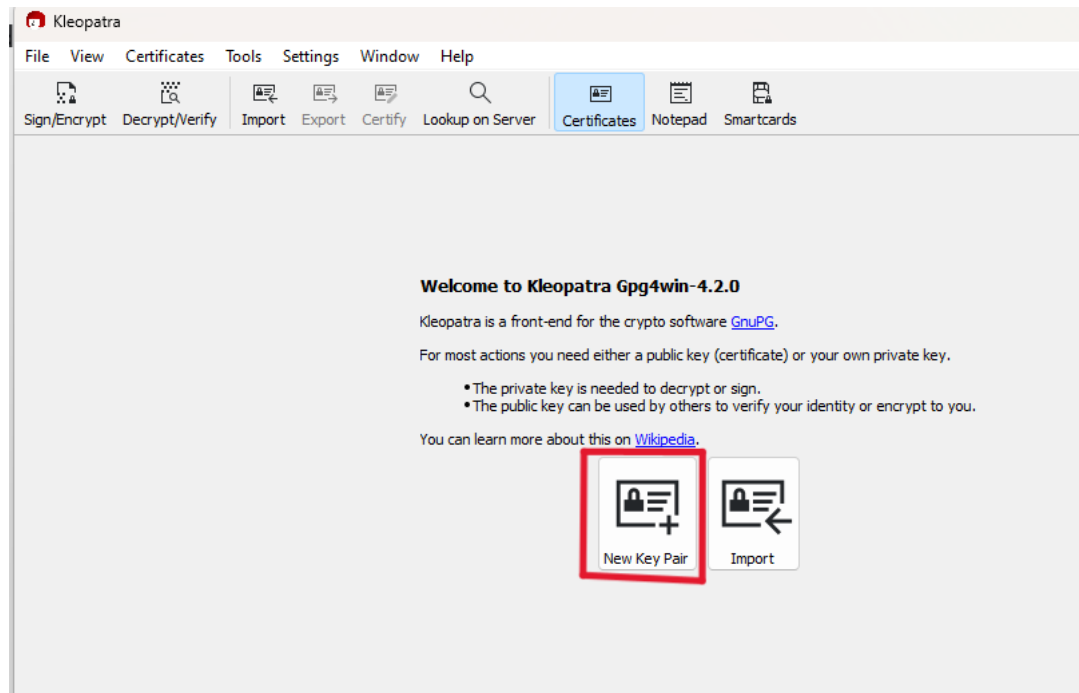
You can browse and select the desired path.

Click on **Install** after confirming the folder path.



Step 4:

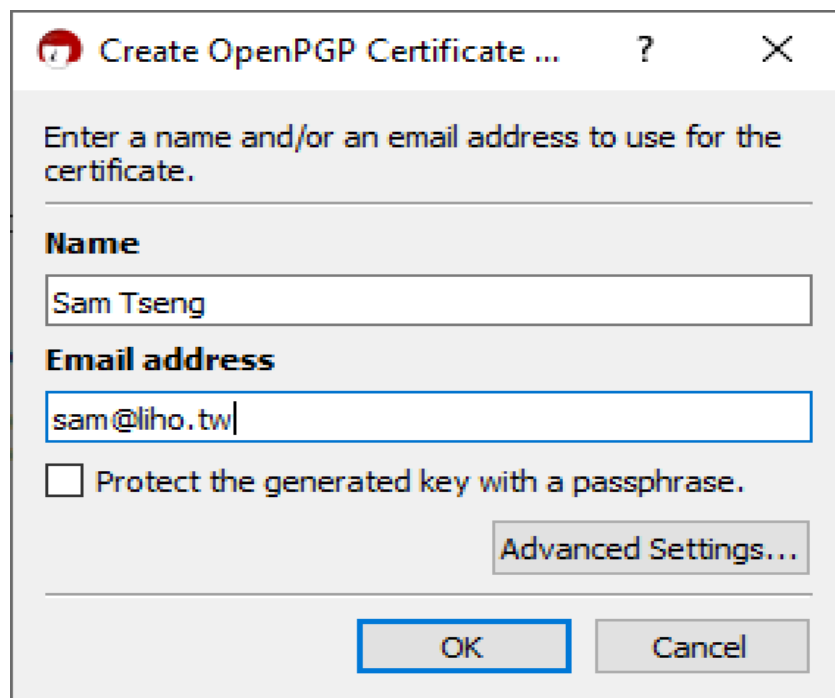
After successful installation, the **welcome page** of the Kleopatra application will appear.

**Step 5 (Key Generation):**

On the welcome screen, click on “**Certificates**” option.

Select “**Create OpenPGP Certificate.**”

Enter your **Name** and **Email Address**, then click on “**Advanced Settings...**” to open key configuration options.

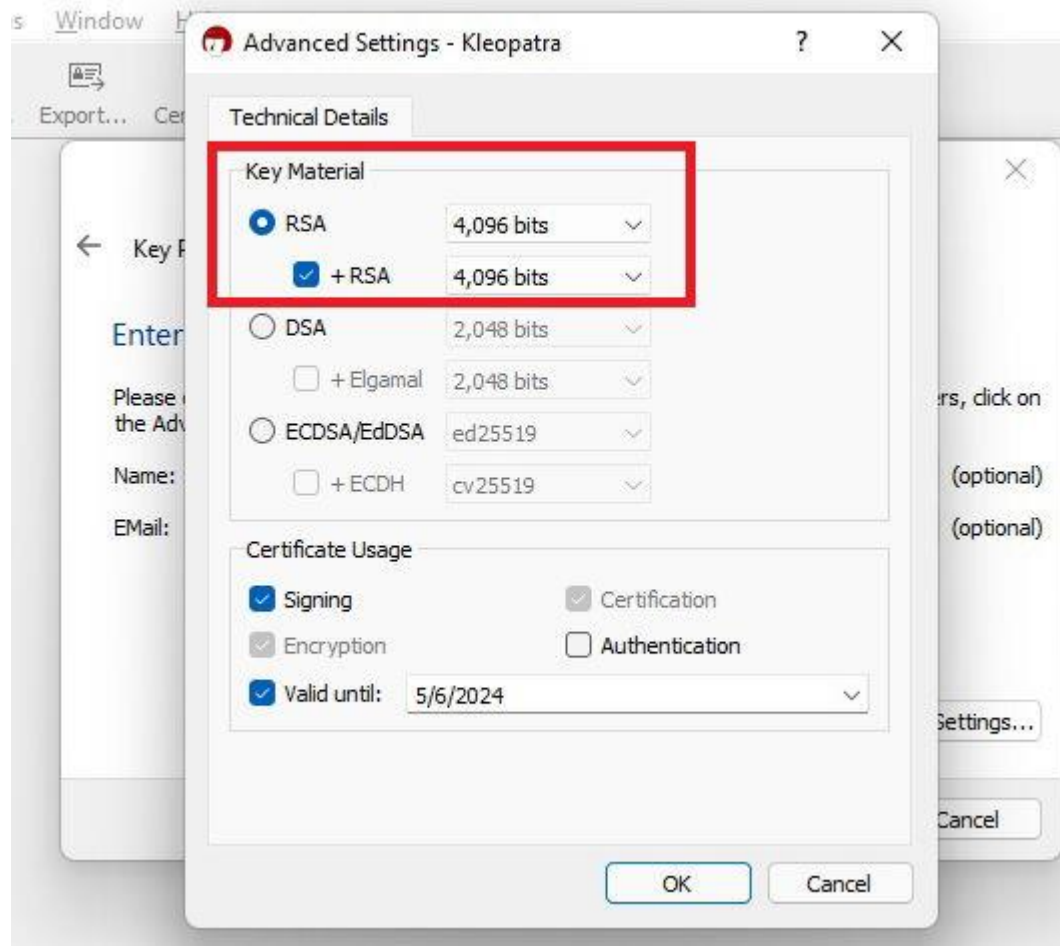


Step 6:

Provide additional key generation settings such as:

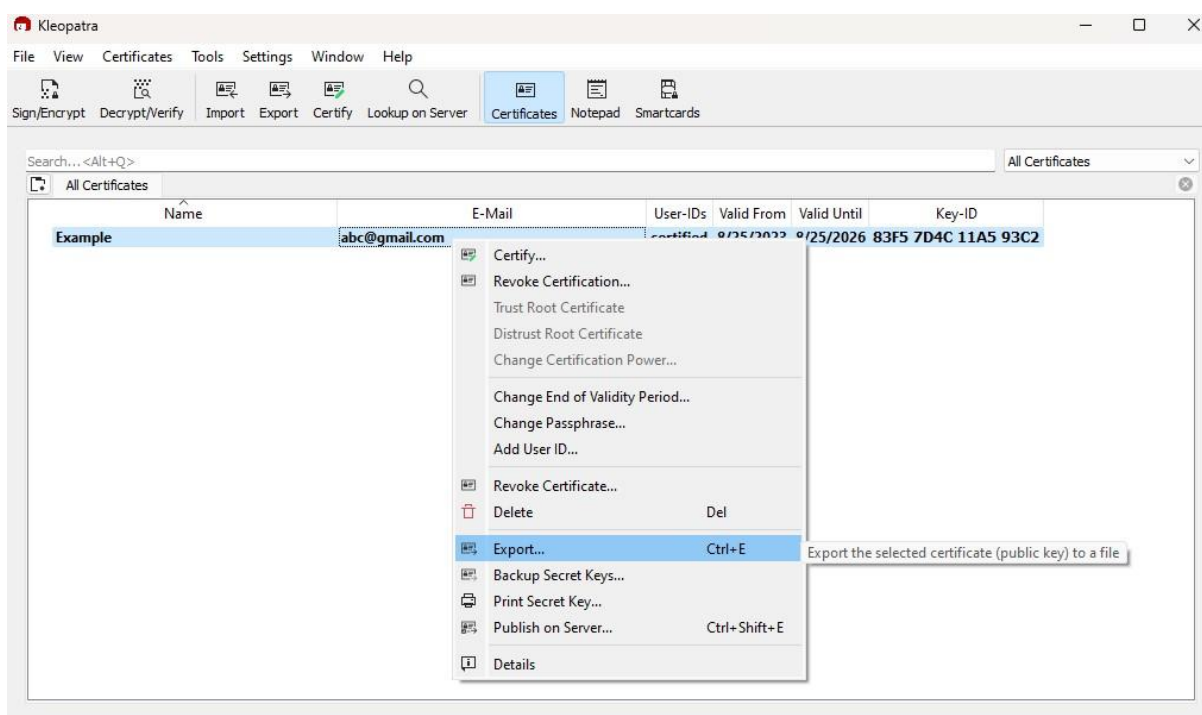
- Key type (RSA, ECC, etc.)
- Key size
- Expiration date

Click **OK** to proceed.

**Step 7 (Export Public Key):**

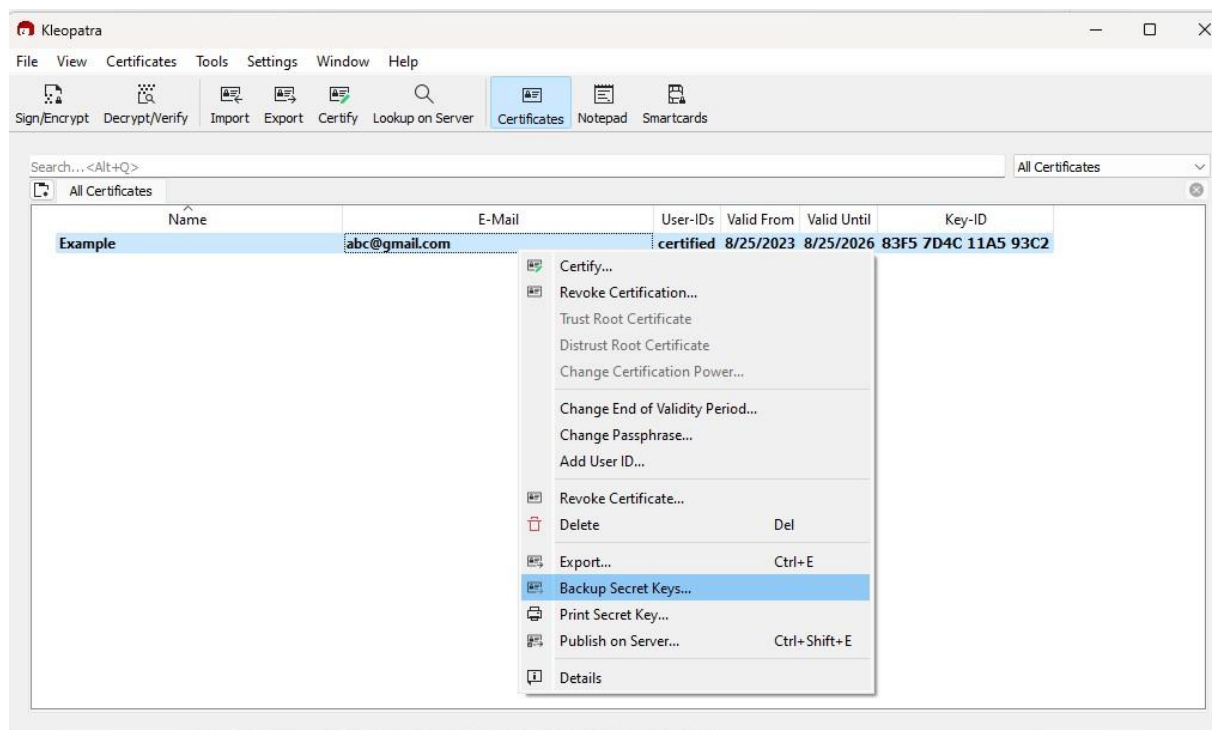
Once key generation is completed, a certificate is created containing **Name, Email, User ID, Validity, and Key ID**.

Select the created certificate → right-click on your name → choose “**Export...**” or press **Ctrl + E** to export the public key.



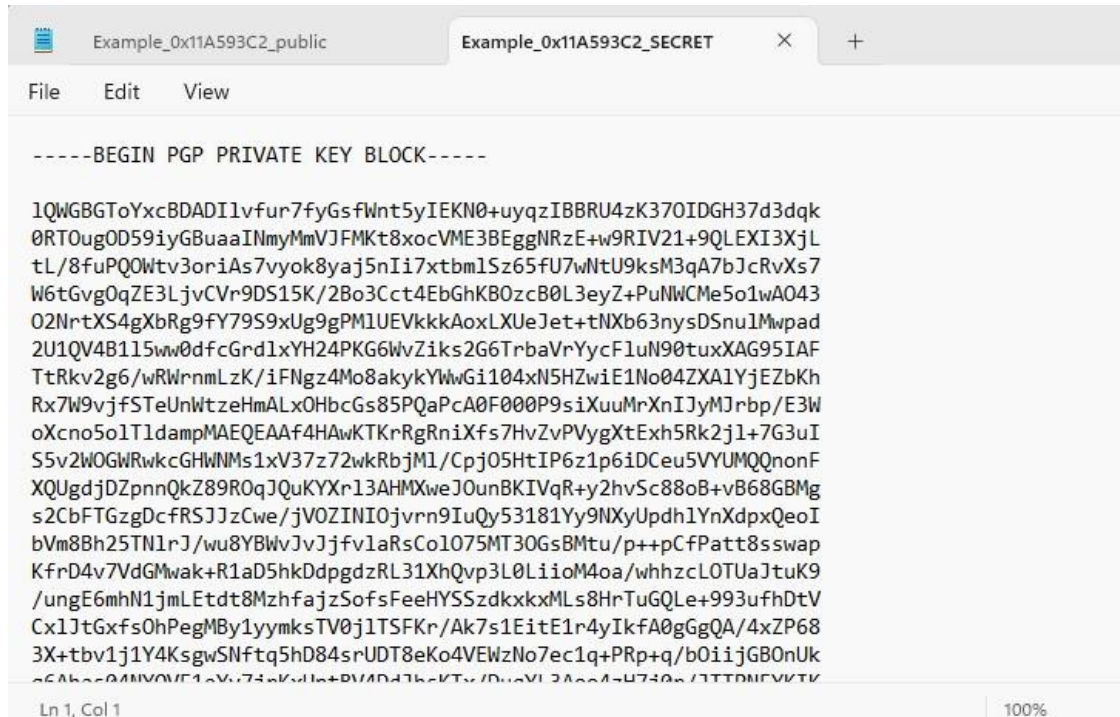
Step 8 (Backup Private Key):

Click on the **Email field** and select “**Backup Security Keys...**” from the dropdown list. This step backs up your private key safely.



Step 9:

Copy the **exported public key** and save it into a new .txt file.



The screenshot shows a text editor with two tabs: 'Example_0x11A593C2_public' and 'Example_0x11A593C2_SECRET'. The 'Example_0x11A593C2_SECRET' tab is active, displaying a PGP private key block. The text starts with '-----BEGIN PGP PRIVATE KEY BLOCK-----' and ends with a long base64-encoded string. The status bar at the bottom indicates 'Ln 1, Col 1' and '100%' zoom.

```
-----BEGIN PGP PRIVATE KEY BLOCK-----

1QWGBGT0YxcBDADI1vfur7fyGsFWnt5yIEKN0+uyqzIBBRU4zK370IDGH37d3dqk
0RT0ugOD59iyGBuaaINmyMmVJFMKt8xocVME3BEggNRzE+w9RIV21+9QLEXI3XjL
tL/8fuPQOWtv3oriAs7vyok8yaj5nIi7xtbm1Sz65fU7wNtU9ksM3qA7bJcRvXs7
W6tGvg0qZE3LjvCvR9DS15K/2Bo3Cct4EbGhKB0zcB0L3eyZ+PuNWCMe5o1wA043
02NrtXS4gXbRg9fY79S9xUg9gPM1UEVkkkAoxLXUeJet+tNXb63nysDSnu1Mwpad
2U1QV4B115w0dfcGrdlxYH24PKG6WvZiks2G6TrbaVrYycFluN90tuxXAG95IAF
TtRkv2g6/wRWrnmLzK/iFNgz4Mo8akykYwWGi104xN5HZwiE1No04ZXA1YjEZbKh
Rx7W9vjfSTeUnWtzeHmALxOHbcGs85PQaPcA0F000P9siXuuMrXnIJyMJrbp/E3W
oXcno5o1T1dampMAEQEAAf4HAWKTKrRgRniXfs7HvZvPVyGxtExh5Rk2j1+7G3uI
S5v2WOGWRwkcGHWNM51xV37z72wkRbjM1/Cpj05HtIP6z1p6iDCeu5VYUMQQnonF
XQUgdjDZpnnQkZ89R0qJQuKYXr13AHMXweJ0unBKIVqR+y2hvSc88oB+vB68GBMg
s2CbFTGzgDcFRSJJzCwe/jVOZINI0jvrn9IuQy53181Yy9NXyUpdh1YnXdpXQeoI
bVm8Bh25TNlrJ/wu8YBwVJvJjfv1aRsCo1075MT30GsBMtu/p++pCfPatt8sswap
KfrD4v7VdGMwak+R1aD5hkDdpgdzRL31XhQvp3L0LioM4oa/whhzcLOTUaJtuK9
/ungE6mhN1jmLEtdt8MzhfajzSofsFeeHYSSzdkxkxMLs8HrTuGQLe+993ufhDtV
Cx1JtGxfS0hPegMBY1yymksTV0j1TSFKr/Ak7s1EitE1r4yIkfA0gGgQA/4xZP68
3X+tbv1j1Y4KsgwSNftq5hD84srUDT8eKo4VEWzNo7ec1q+PRp+q/b0iijGB0nUk
c6Abae04MYQV51eYv74eKdletBVMDd3bKTy/Duay13Aoc4-H740e/1TTPMEVKTk
-----
```

Step 10:

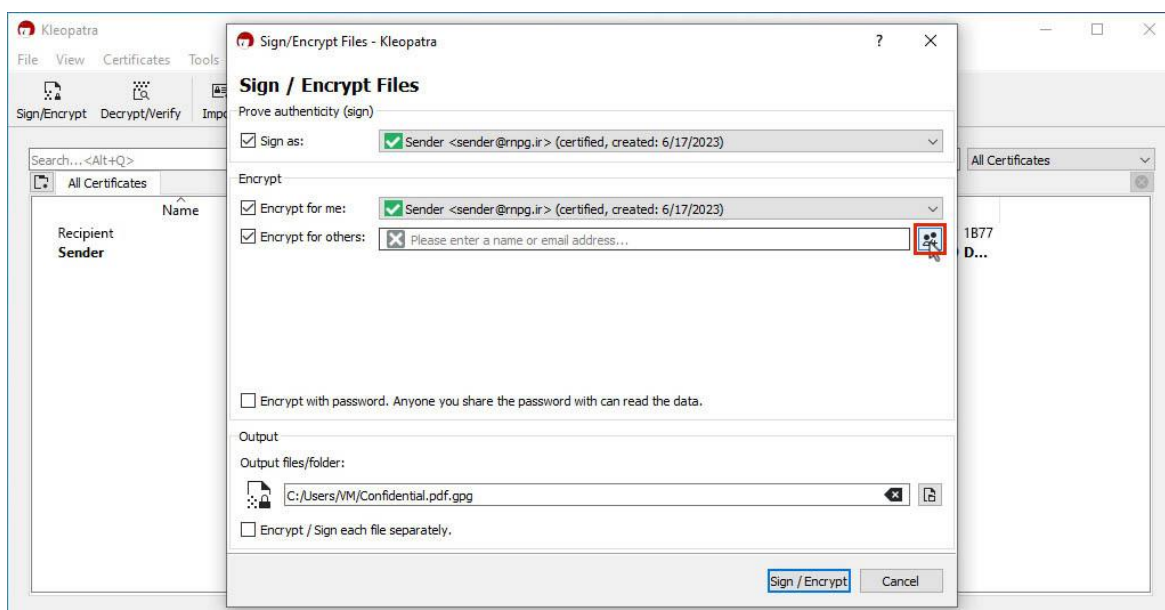
Create a **demo file** (e.g., message.txt) containing the text you wish to encrypt.



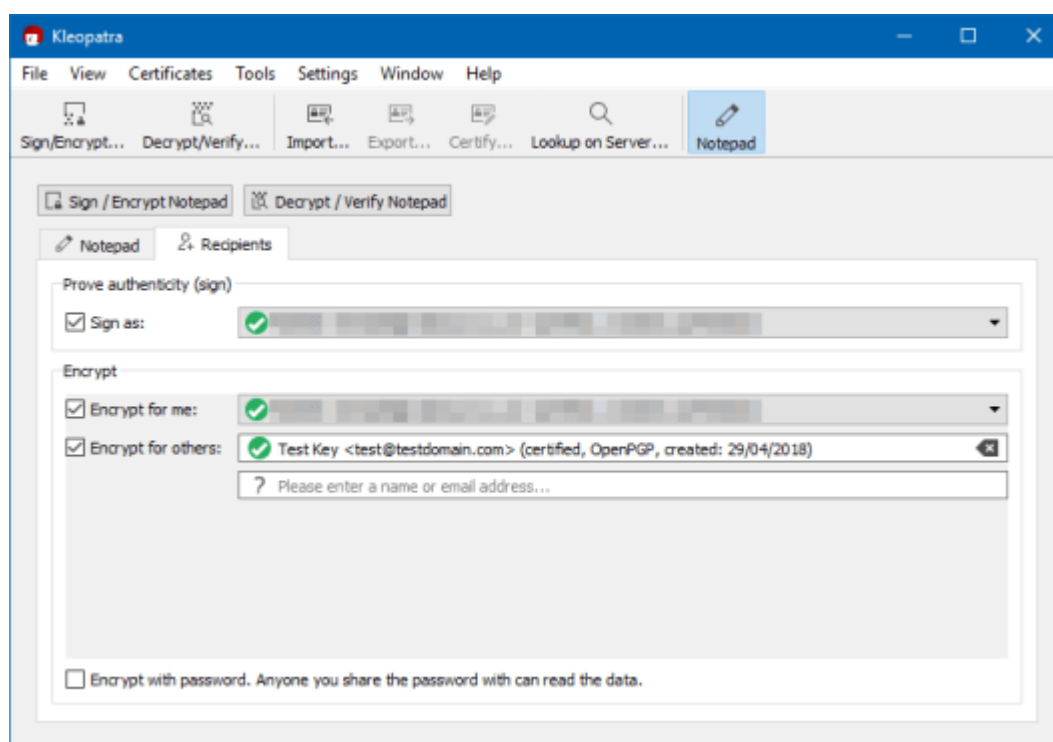
The screenshot shows a text editor with three tabs: 'public key.txt', 'public key2.txt', and 'hello this is a test message.'. The 'hello this is a test message.' tab is active, displaying the text 'hello this is a test message.' in a blue font. The status bar at the bottom indicates 'Ln 1, Col 1' and '100%' zoom.

Step 11:

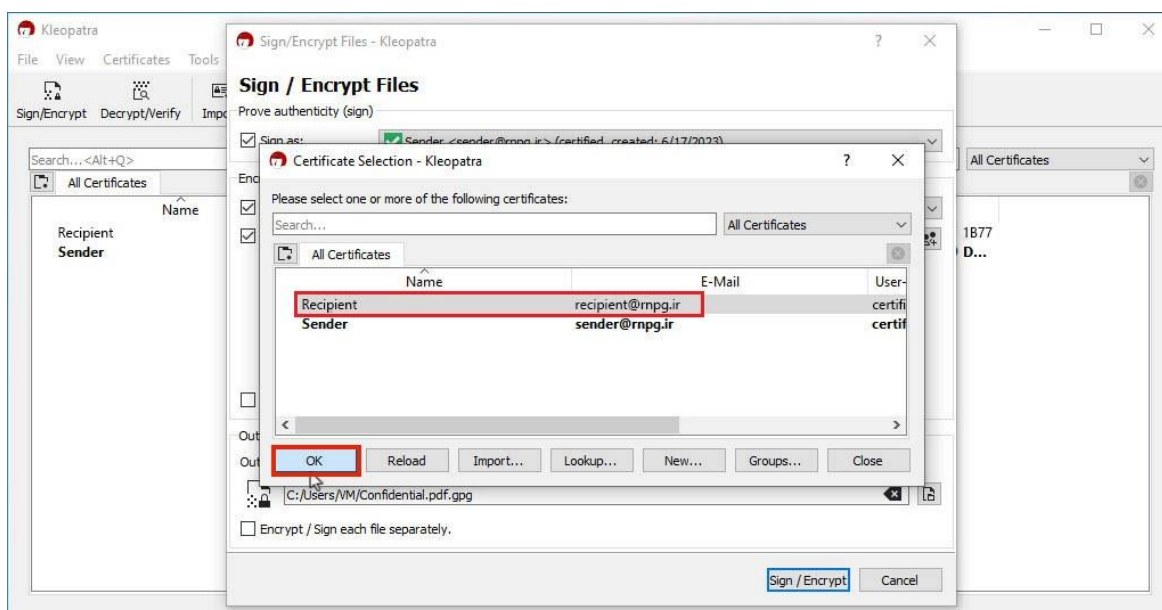
In Kleopatra, select your **public key** for encryption.

**Step 12:**

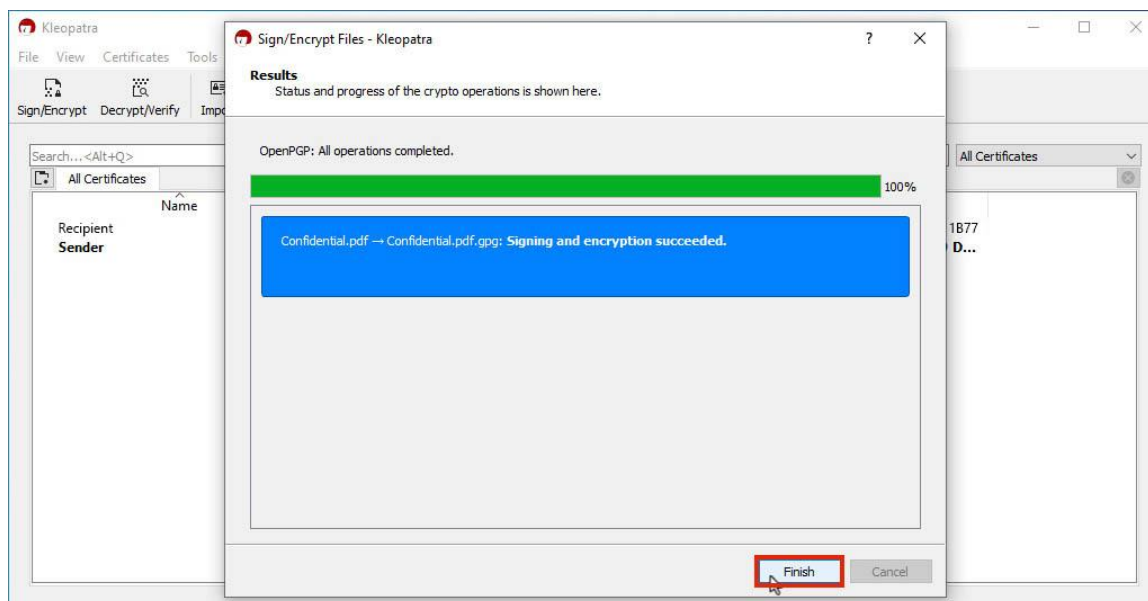
Optionally, you can also **sign** the file using your email ID for authenticity. Once all information is filled, click on “**Sign/Encrypt**” button.

**Step 13:**

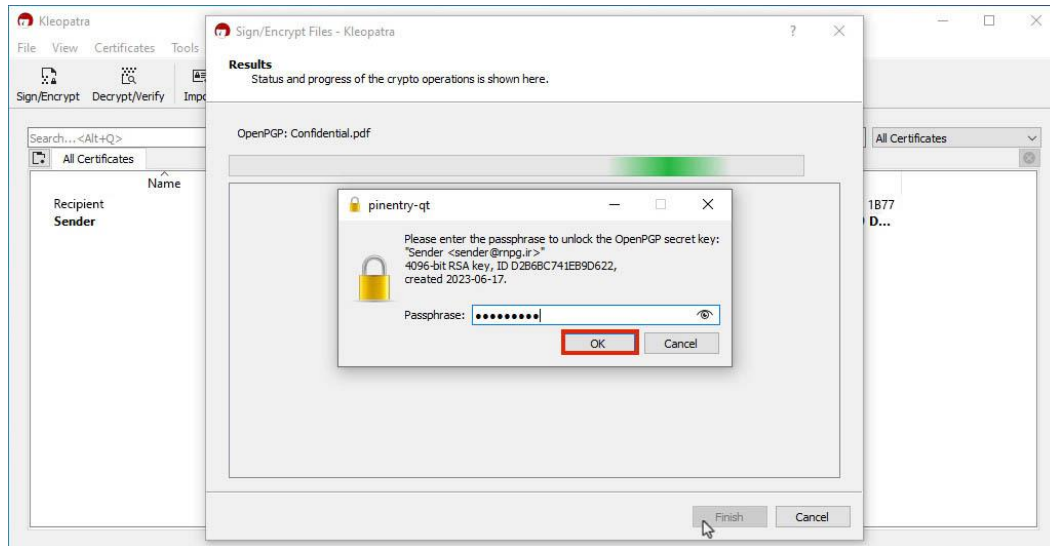
Select one or more **certificates** if required for encryption.

**Step 14:**

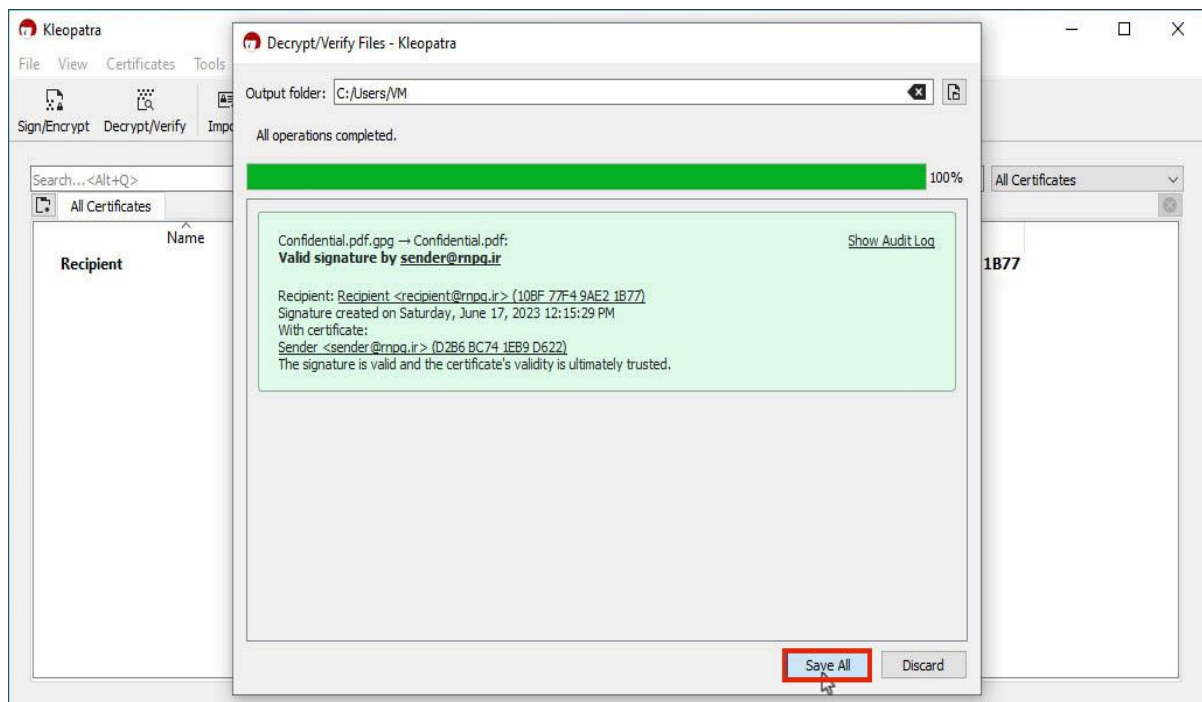
Wait until all encryption operations are completed successfully.

**Step 15:**

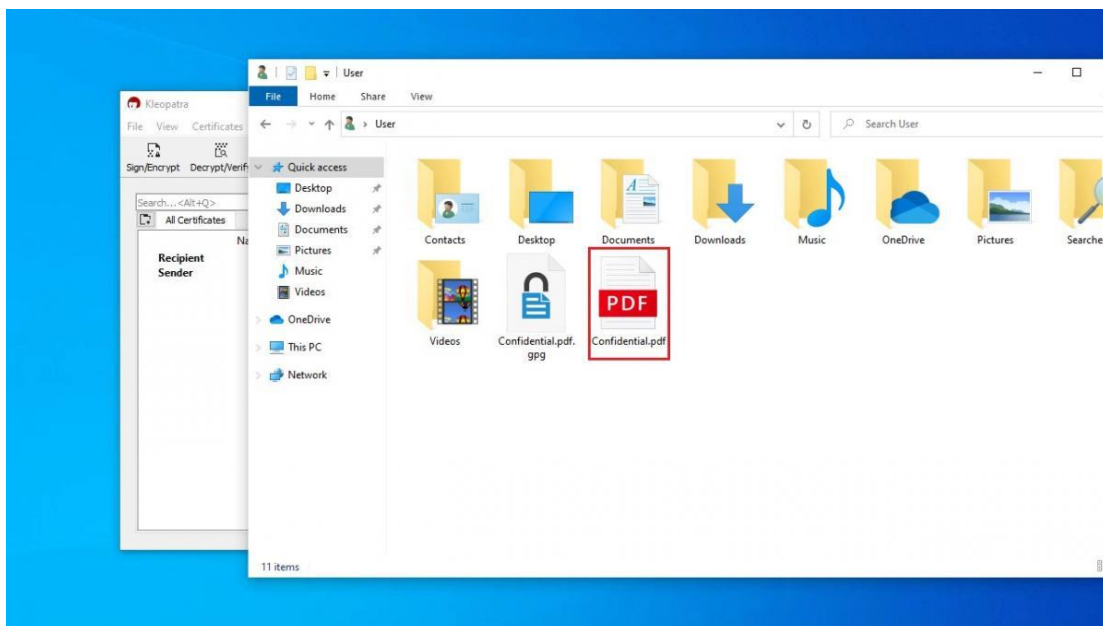
Enter your **key password (passphrase)** when prompted, to authorize the encryption or decryption process.

**Step 16 (Decryption):**

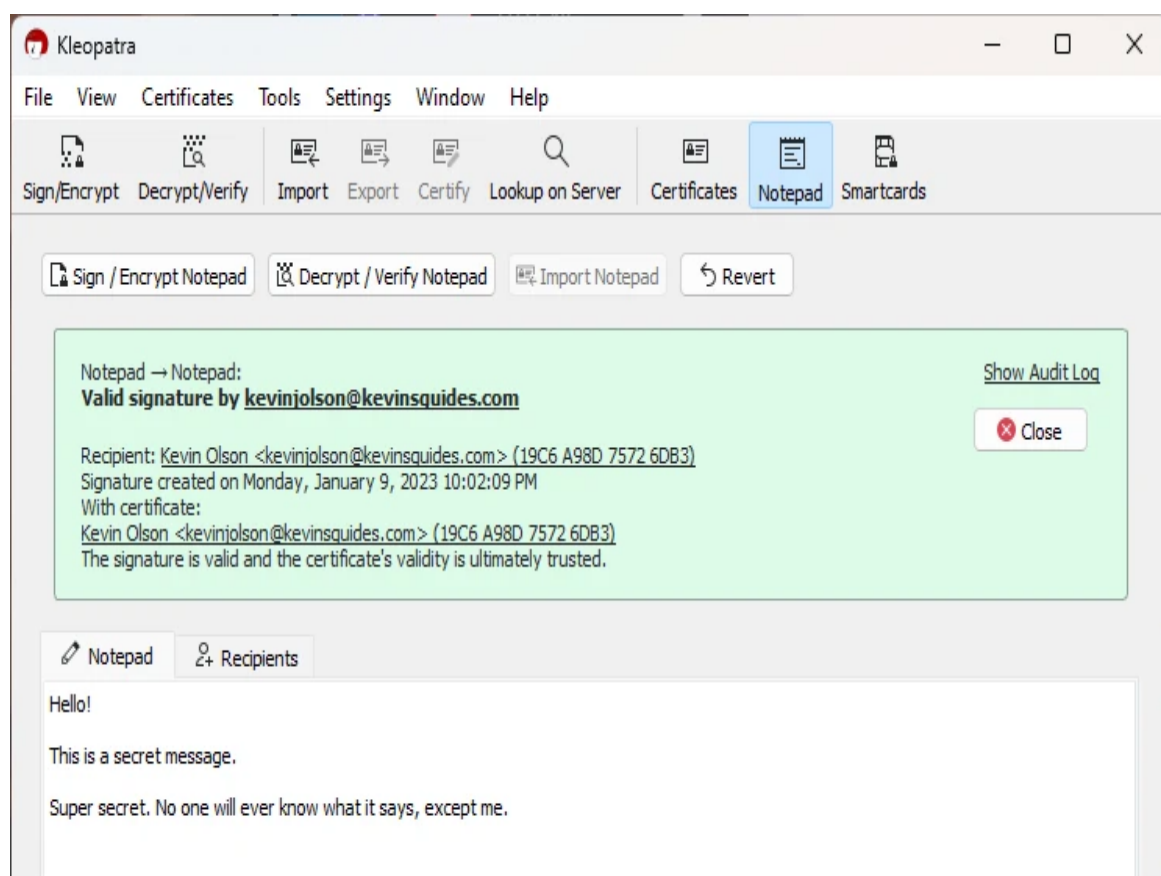
To decrypt, select **Decrypt/Verify** option from the toolbar.
Browse and select the **encrypted file** you want to decrypt.

**Step 17:**

Confirm the selection of the encrypted file for decryption.

**Step 18:**

After successful decryption, the **original message** will be displayed.



IX. Precautions

1. Keep your **private key confidential** and protected with a strong passphrase.
2. Always verify the authenticity of a contact's public key before using it.
3. Backup your keys in a secure location.
4. Do not use untrusted systems for encryption or signing activities.

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. What is the purpose of using PGP in email security?
2. Explain how public and private keys work in GnuPG.
3. What are the main security features provided by encrypted email communication?
4. Perform encryption and decryption of an email using GnuPG.
5. Explain the difference between signing and encrypting an email.

Space for Answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

This image shows a full page of white paper with horizontal dotted lines. The lines are evenly spaced and run across the width of the page, providing a guide for handwriting or typing. There are no margins, text, or other markings on the page.

.....

.....

.....

.....

.....

XI. Exercise

1. Encrypt and send a sample message using GnuPG or Thunderbird.
2. Verify a signed message using the sender's public key.
3. Research and report the difference between PGP and S/MIME protocols.

XII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. NPTEL Course: Introduction to Information Security.
4. GnuPG Official Website: <https://gnupg.org>
5. Thunderbird Enigmail Documentation: <https://www.enigmail.net>
6. Virtual Labs (IIIT Hyderabad): <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct generation and sharing of public/private keys	20 %
2	Successful encryption, signing, and decryption of email	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Accuracy of encrypted/decrypted message	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 16: Find the Origin of an Email using Email Tracker Pro

I. Practical Significance

This practical introduces email tracing and analysis techniques using Email Tracker Pro, a tool that helps identify the origin, sender's IP address, location, and route taken by an email. Students will learn how to analyze email headers to detect spam, phishing, or spoofed emails, and understand how tracing contributes to cyber forensics and incident response.

II. Industry / Employer Expected Outcome

The aim of this course is to help the students to attain the following Industry Identified Outcomes through various teaching learning experiences: Implement policies and guidelines to maintain data security and privacy during data transmission.

III. Course Level Learning Outcomes(s)

CO5 – Apply security on Network and Database.

IV. Laboratory Learning Outcome (LLO)

LLO 16.1: Use of email tracker pro.

V. Relevant Affective domain related Outcomes(s)

1. Responsibility: Use tracking tools ethically and only for legitimate analysis.
2. Ethical Awareness: Understand privacy implications and lawful use of tracking tools.
3. Analytical Skills: Interpret header data and identify key routing details.

VI. Theoretical Background

Email tracing is a process used to determine the path and origin of an email message. Every email sent across the Internet carries a header containing metadata — such as sender and recipient information, IP addresses, time stamps, and mail server details.

Email Tracker Pro is a Windows-based utility that automatically reads and analyzes these headers to provide insights like:

- **Sender's IP address**
- **Approximate geographical location**
- **Mail transfer route (Received path)**
- **Email client and server information**

Structure of an Email Header:

An email header typically includes lines such as:

```
Received: from mail.example.com (123.45.67.89)
by smtp.gmail.com with ESMTP id abc123;
Mon, 6 Oct 2025 10:12:34 +0530
```

Here, the **IP address (123.45.67.89)** can be used to trace the sender's network origin.

Uses of Email Tracing:

- Detecting **spam or phishing** sources.
- Verifying sender authenticity.
- Supporting **digital forensics investigations**.
- Tracking **cybercrime evidence**.

VII. Resources required

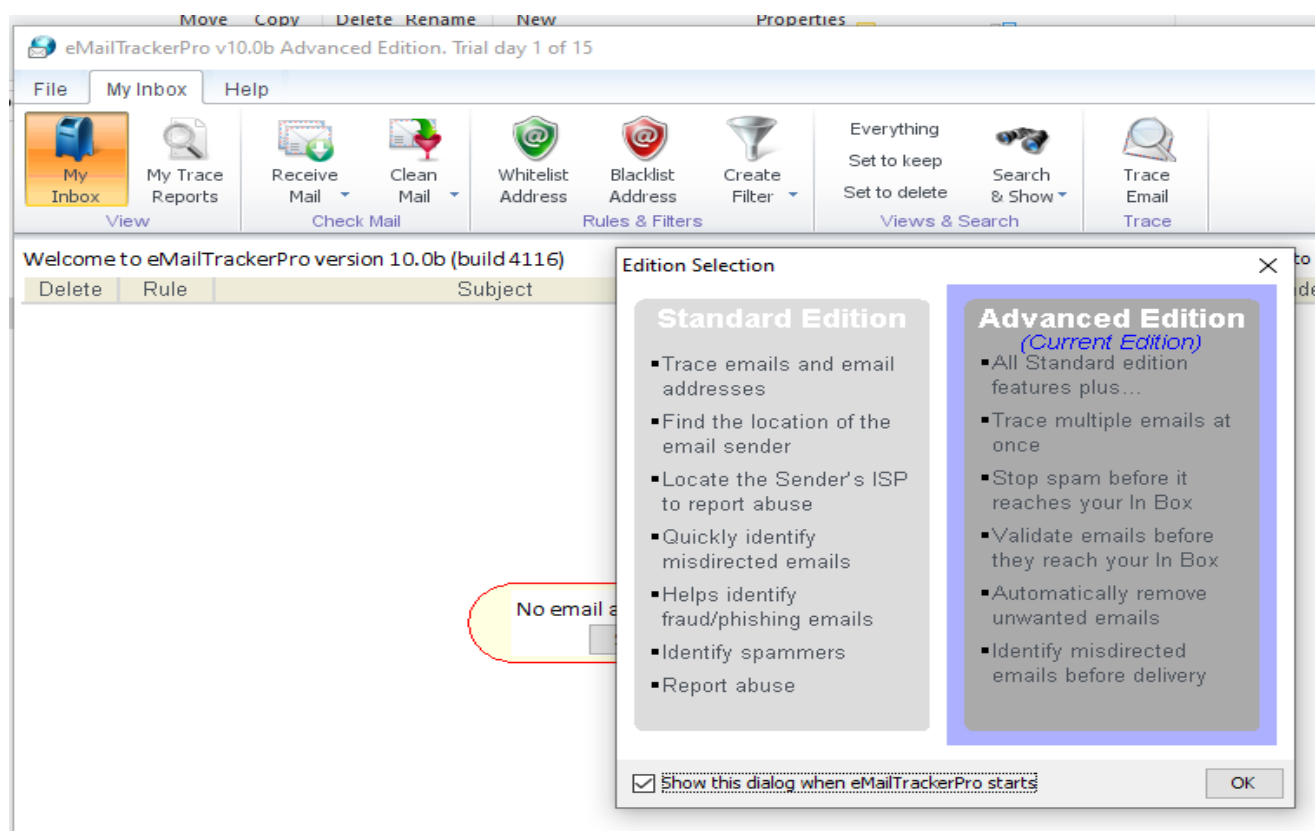
Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer (i3 – i5 RAM minimum 2 GB and onwards and HDD minimum 10 GB)	As per batch Size	--
2	Operating System	Windows 10 / 11 / Linux (Ubuntu/CentOS)		
3	Tools / Software Required	Email Tracker Pro (Free or Trial Version)		

VIII. Procedure

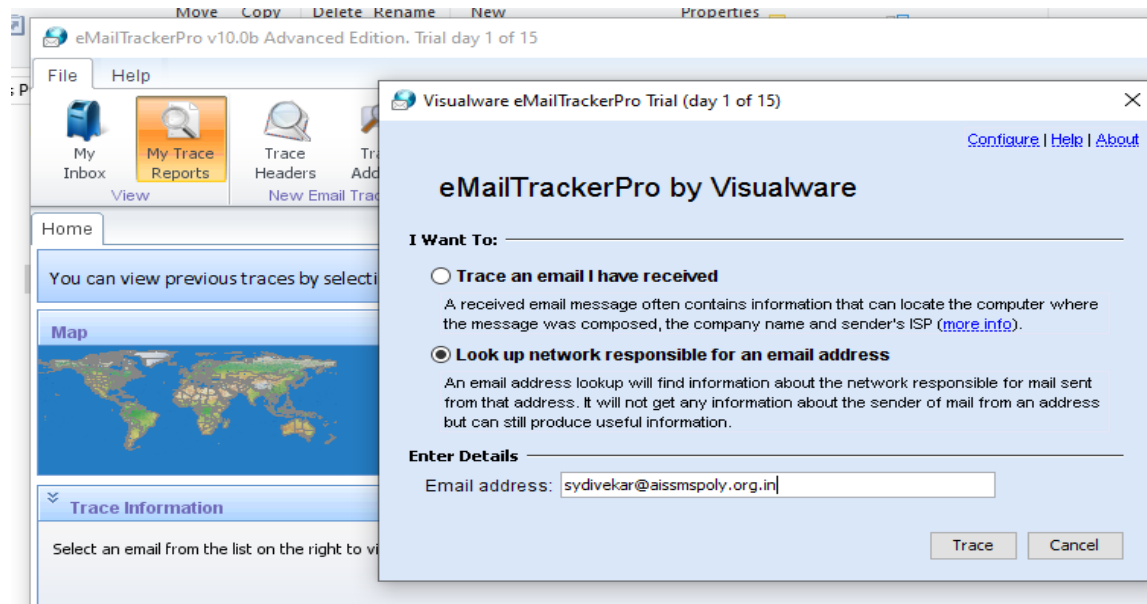
Step-by-Step Procedure

Step 1: Launch Email Tracker Pro on your system.

Double-click the application icon or open it from the Start Menu.



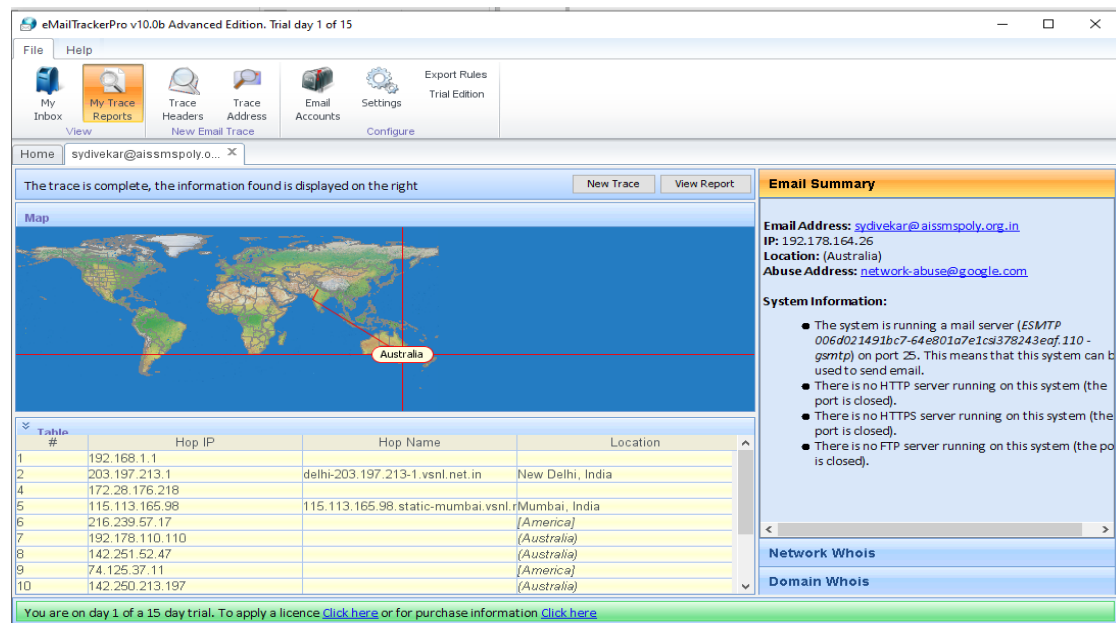
Step 2: In the main window of Email Tracker Pro, click on “Analyze Email Address.”



Step 3: Click on the “Trace Email” button to begin analysis.

The software will process the header and display information such as:

- Sender’s IP address
- Originating location
- Mail server path
- Delivery hops and timestamps



Step 4: Review the analysis results displayed on screen.

You can identify the **geographical source** of the email and detect possible **spoofing or spam** activity.

IX. Precautions

1. Do not trace personal or confidential emails without consent.
2. Use tracing only for **educational or forensic analysis** purposes.
3. Be aware that some emails (especially from large providers) use **relay servers**, making exact tracing difficult.
4. Avoid using third-party websites that store header data without permission.

X. Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. What is an email header and why is it important in email security?
2. How can the origin of an email be determined using Email Tracker Pro?
3. Explain the purpose of the “Received” field in an email header.
4. What are some indicators of a phishing or spoofed email?
5. Perform an email trace and identify the IP address and location of the sender.

Space for Answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

[illegible]

.....

.....

.....

.....

.....

XI. Exercise

1. Trace at least two sample emails (one genuine and one spam) using Email Tracker Pro.
2. Compare the routing paths and origins of both messages.
3. Research and report one real-world cybercrime case solved through email forensics.

XII. References / Suggestions for further reading

1. Stallings, W. Cryptography and Network Security: Principles and Practice.
2. Kahate, A. Cryptography and Network Security.
3. NPTEL Course: Introduction to Information Security.
4. Email Tracker Pro Official Website: <https://www.emailtrackerpro.com>
5. Gmail Help: View Original Message and Headers.
6. Virtual Labs (IIIT Hyderabad): <https://cse29-iiith.vlabs.ac.in/List%20of%20experiments.html>

XIII. Assessment schemes

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Correct extraction of email header information	20 %
2	Successful tracing and interpretation of sender's origin	30 %
3	Quality of output achieved(LLO mapped)	10 %
Product related (10 Marks)		40 %
4	Accuracy of trace report (origin, IP, location)	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	