Name: Aastha Kumar
Reg. No: 21BCE5067
Course Code: BCSE354E
Date of submission: 23/04/24

INFORMATION SECURITY AND MANAGEMENT LAB

CONSOLIDATED LAB EXPERIMENTS

EXPERIMENT 1

Aim:

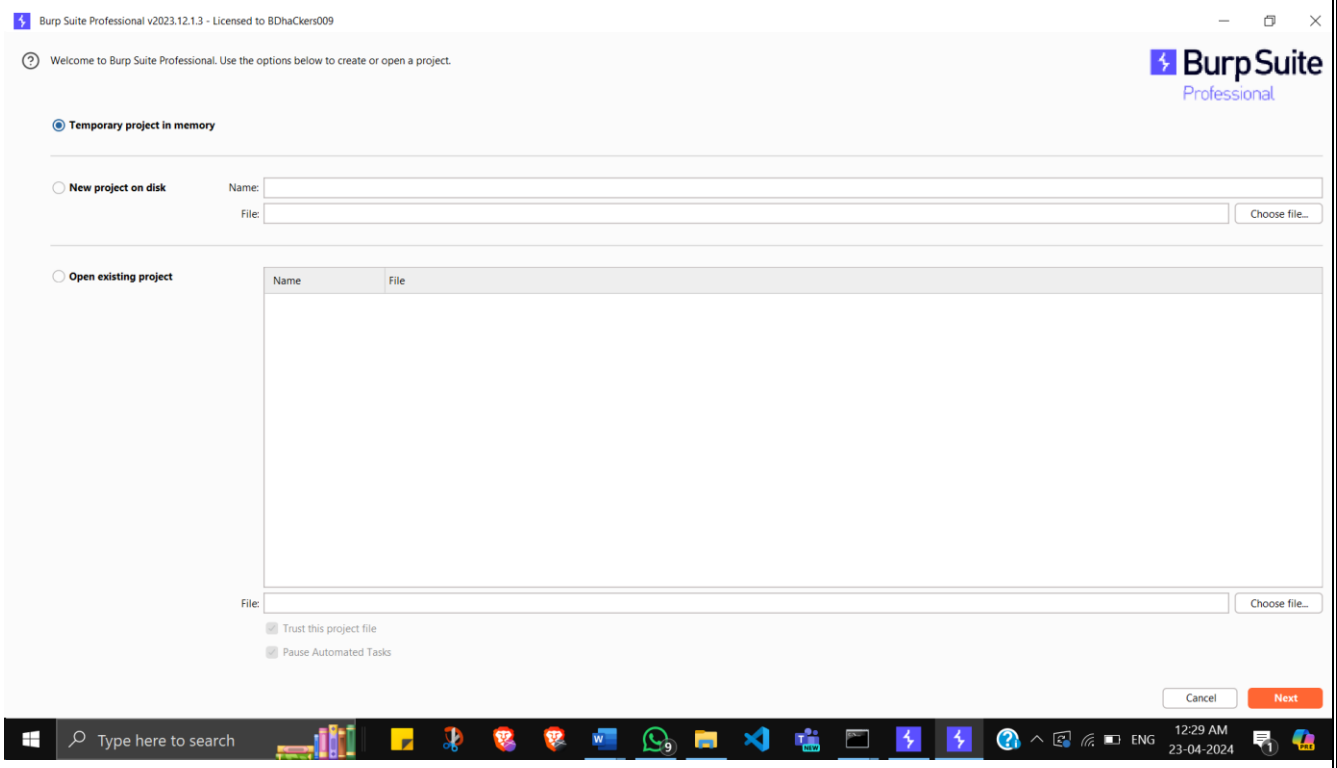
To study about the tool Burp Suite and Intercept the connection between the system and the internet.

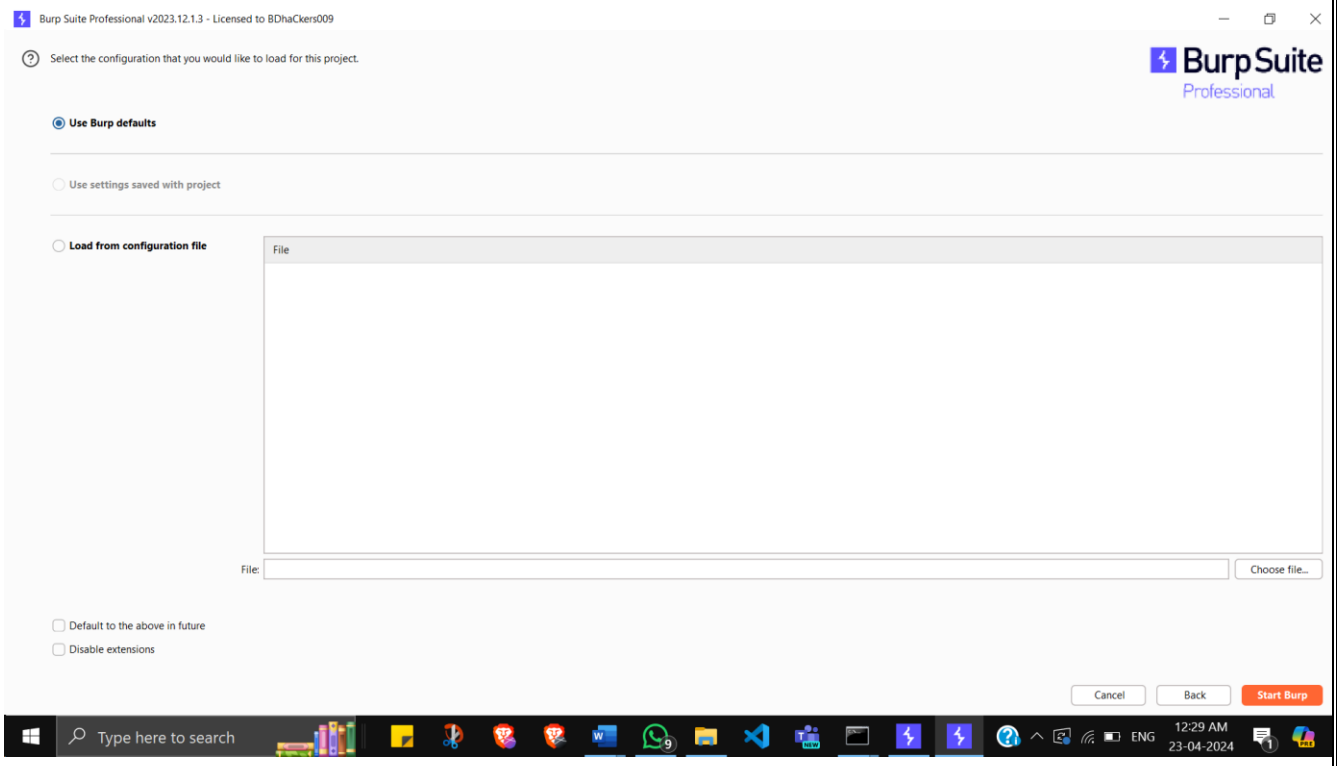
Apparatus Required:

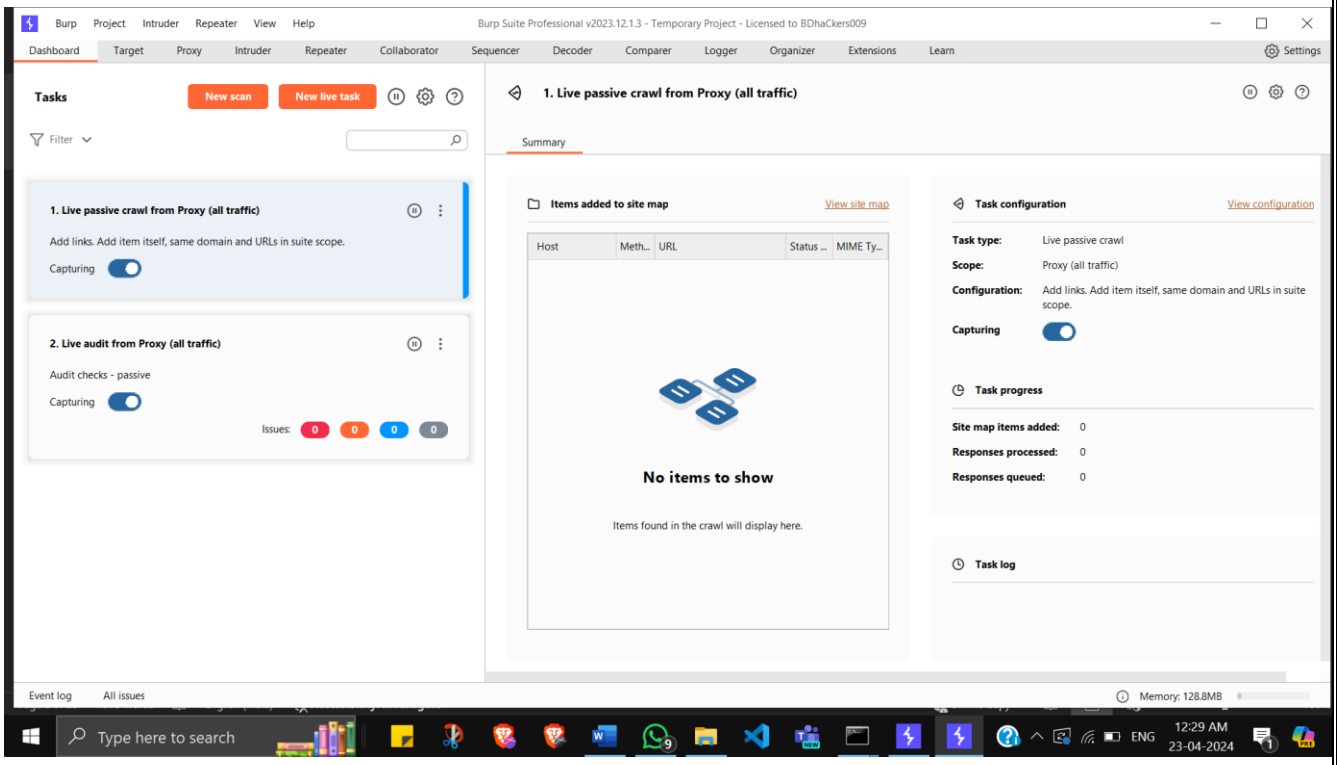
Burp Suite

Procedure:

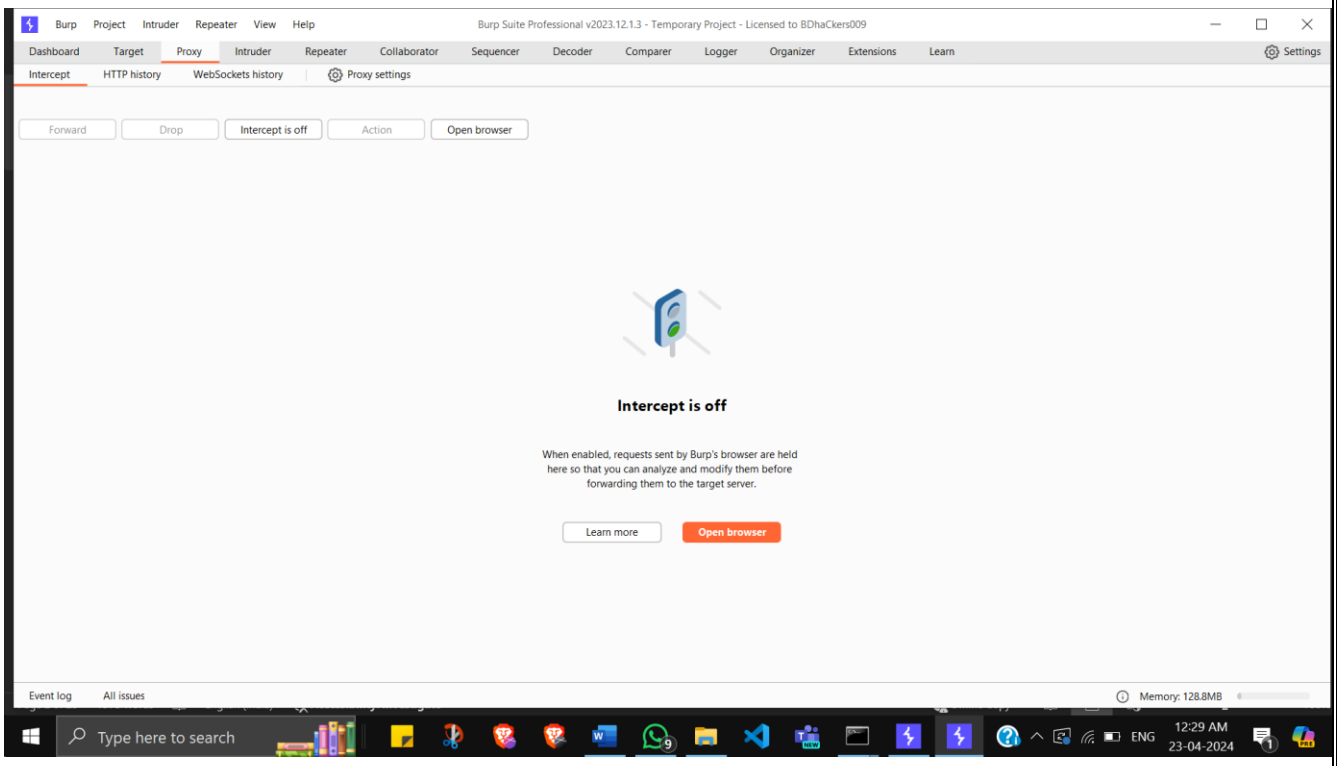
Step 1: Download Burp Suite and open it.



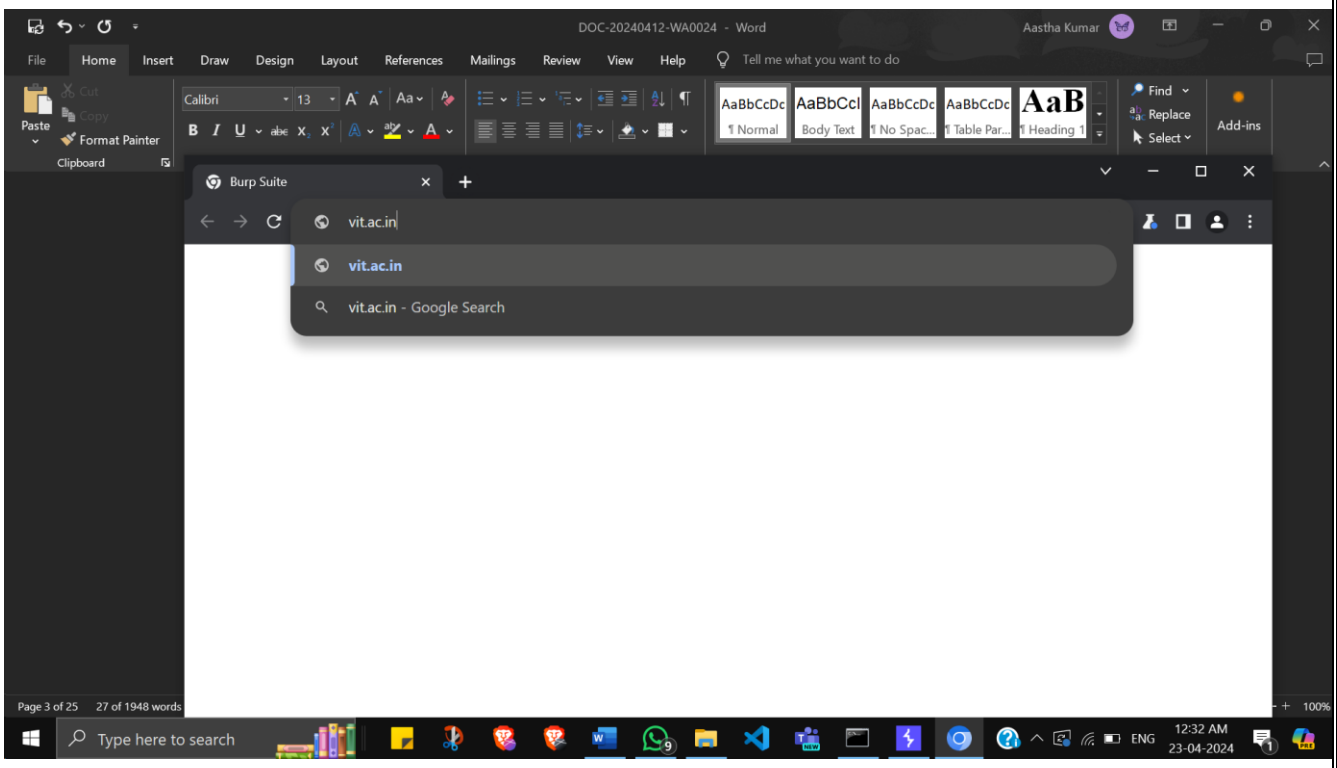


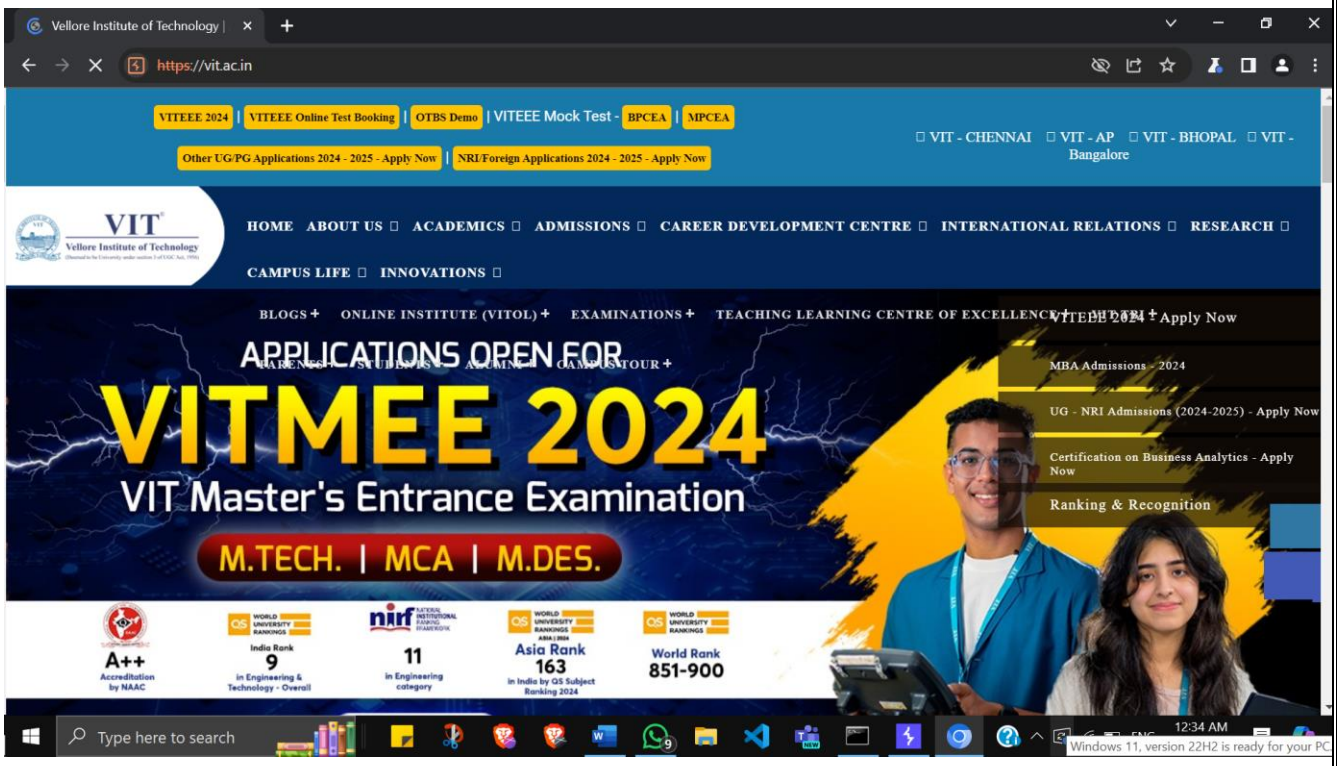


Step 2: Go to Proxy tab and click “Open Browser”

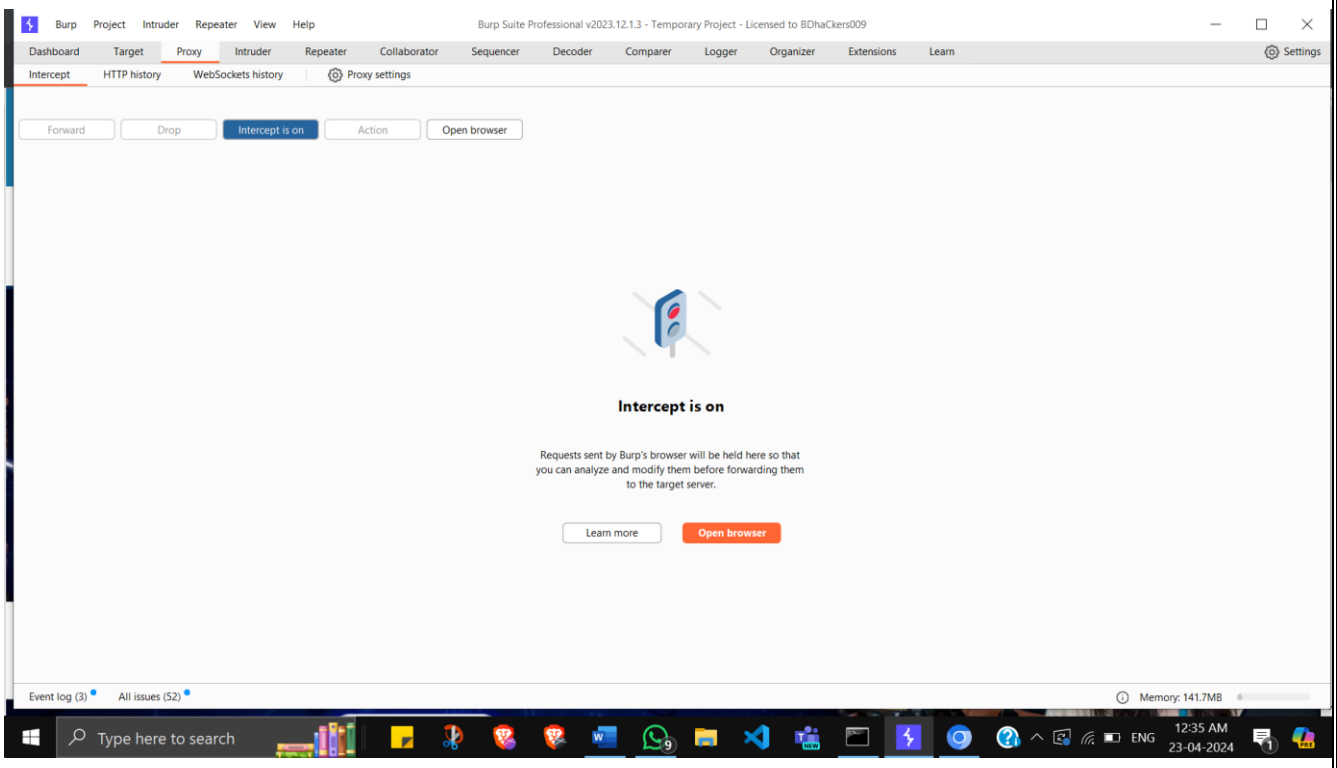


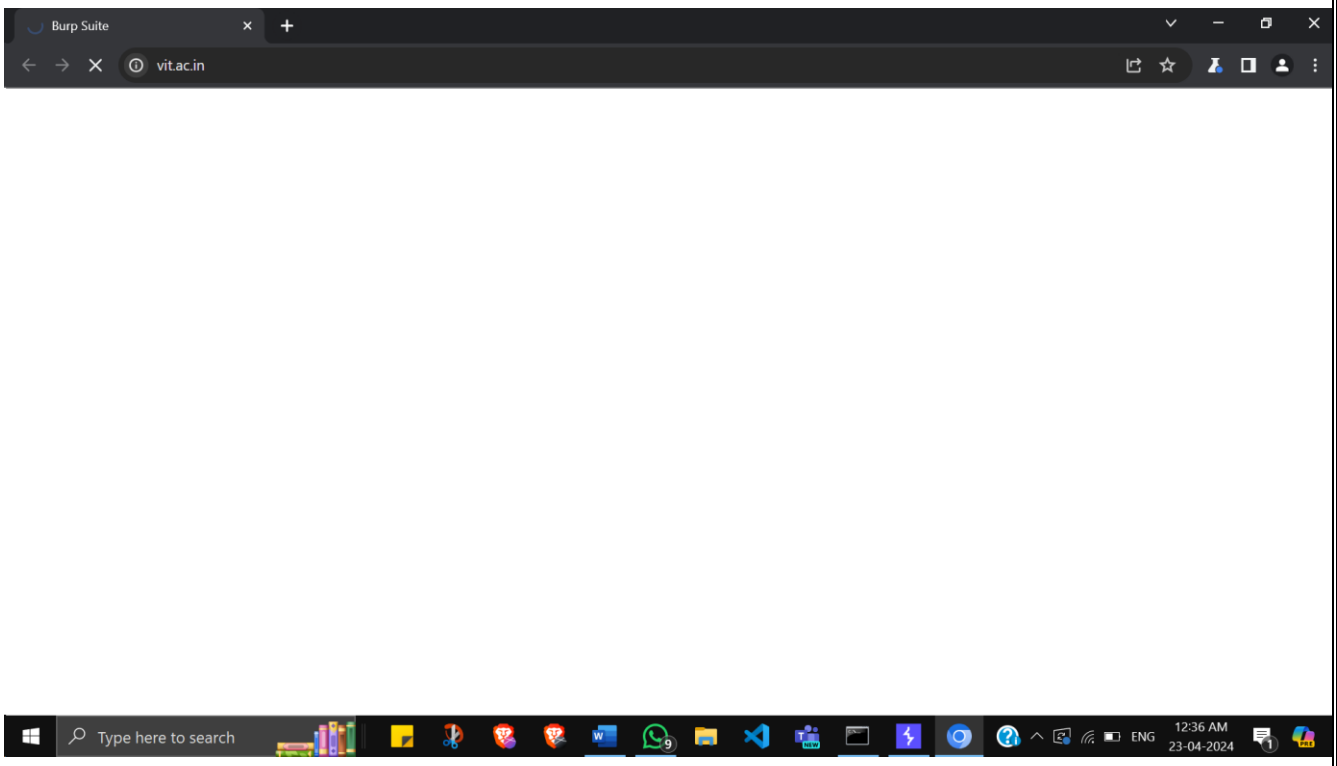
Step 3: Type in vit.ac.in in the search bar and wait for results. You will notice that the site loads without any timeout error because the intercept is off.





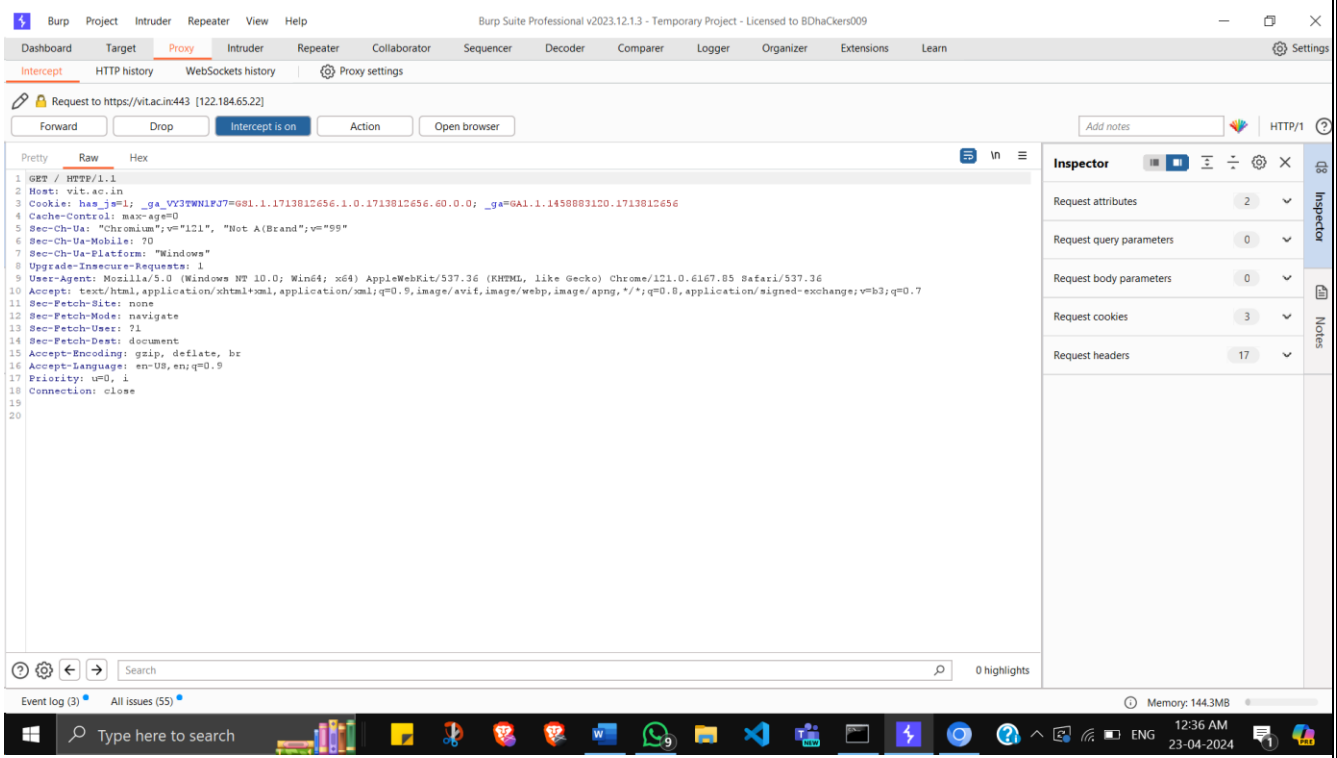
Step 4: Now go back to Proxy tab and switch on the intercept. Try loading the same site again. This time you will face timeout error.





It will send a request but will then be stuck waiting for a response

Step 5: In BurpSuite, we can view the request sent



If we click forward, the request will be sent to the server. The webpage will be visible again.

If we drop the request in Burp Suite, you will face timeout error.

Conclusion: Tools offered by BurpSuite

1. Proxy: BurpSuite contains an intercepting proxy that lets the user see and modify the contents of requests and responses while they are in transit.
2. Intruder: It is a fuzzer. This is used to run a set of values through an input point. The values are run and the output is observed for success/failure and content length.
3. Repeater: Repeater lets a user send requests repeatedly with manual modifications.
4. Sequencer: The sequencer is an entropy checker that checks for the randomness of tokens generated by the webserver. These tokens are generally used for authentication in sensitive operations: cookies and anti-CSRF tokens are examples of such tokens.
5. Decoder: Decoder lists the common encoding methods like URL, HTML, Base64, Hex, etc. This tool comes handy when looking for chunks of data in values of parameters or headers. Itis also used for payload construction for various vulnerability classes. It is used to uncover primary cases of IDOR and session hijacking.
6. Extender: BurpSuite supports external components to be integrated into the tools suite to enhance its capabilities.
7. Scanner: The scanner is not available in the community edition. It scans the website automatically for many common vulnerabilities and lists them with information on confidence over each finding and their complexity of exploitation. It is updated regularly to include new and less known vulnerabilities.

EXPERIMENT 2

Aim:

Using Burp to Brute Force a Login Page

Experiment Objective:

The objective of this experiment is to demonstrate how to use Burp Suite to perform a brute force attack on a login page. By simulating an attack on the authentication mechanism of a web application.

Introduction:

Authentication is a critical aspect of application security, and a vulnerable authentication system can lead to unauthorized access to sensitive data and functionalities. This experiment aims to highlight the significance of secure authentication practices and the potential consequences of authentication bypass vulnerabilities.

Setup:

Software and Tools:

1. Burp Suite
2. Web browser
3. Access to a vulnerable web application

Experiment Steps:

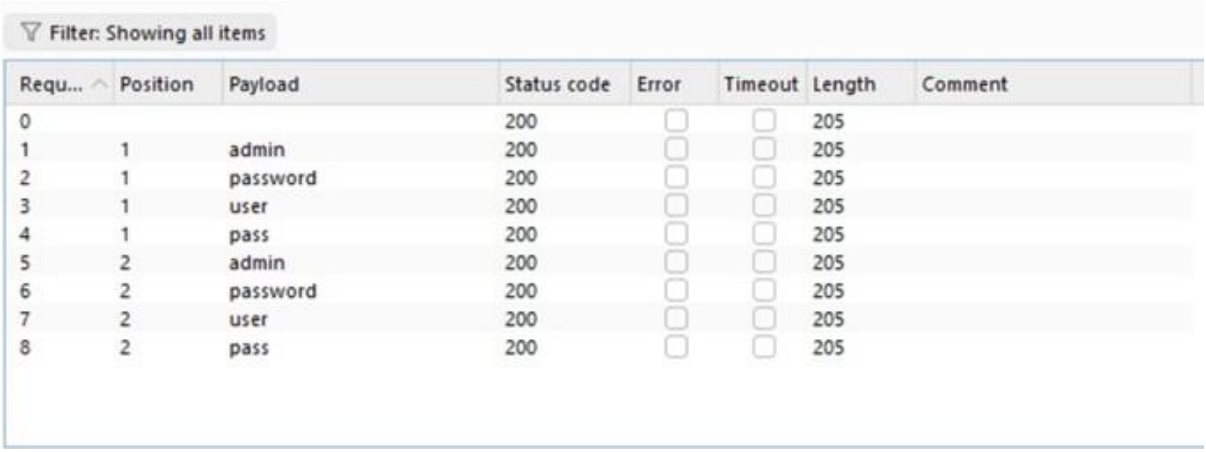
1. Disable interception in the Burp Proxy tab and navigate to the login page of the target web application in your browser.
2. Enter credentials into the login form and submit the request.
3. View the captured request in the Burp Proxy "HTTP history" tab and right-click to send it to the Intruder.
4. Set the attack type to "Cluster bomb/ Sniper/ Battering ram/ Pitchfork" and configure the payload sets in the "Payloads" tab.
5. In the Intruder "Positions" tab, designate the "username" and "password" parameters as payload positions for Pitchfork/ Cluster bomb otherwise either of password or username for Sniper/ Battering ram attack.
6. Start the attack and observe the results in the Intruder attack window.
7. Analyze the responses to identify successful login attempts.
8. Use the gathered credentials to authenticate on the web application's login page to confirm the success of the brute force attack.

Attacks:

1. Sniper:

Payload set	**1**
Payload type	**Simple list**
Payload options	**Admin** **password** **pass user**

Result:



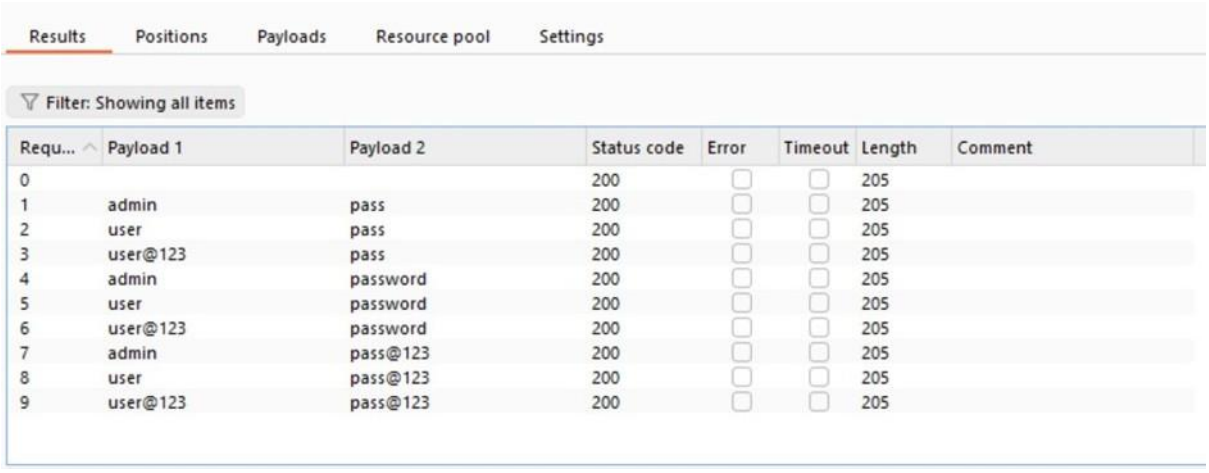
Requ...	Position	Payload	Status code	Error	Timeout	Length	Comment
0			200			205	
1	1	admin	200			205	
2	1	password	200			205	
3	1	user	200			205	
4	1	pass	200			205	
5	2	admin	200			205	
6	2	password	200			205	
7	2	user	200			205	
8	2	pass	200			205	

Since, all result lengths are equal, there is no different response and we have not foundthe valid password.

2. Pitchfork:

Payload set	1	2
Payload type	Simple list	Simple list
Payload options	admin user user@123	pass password pass@123

Result:



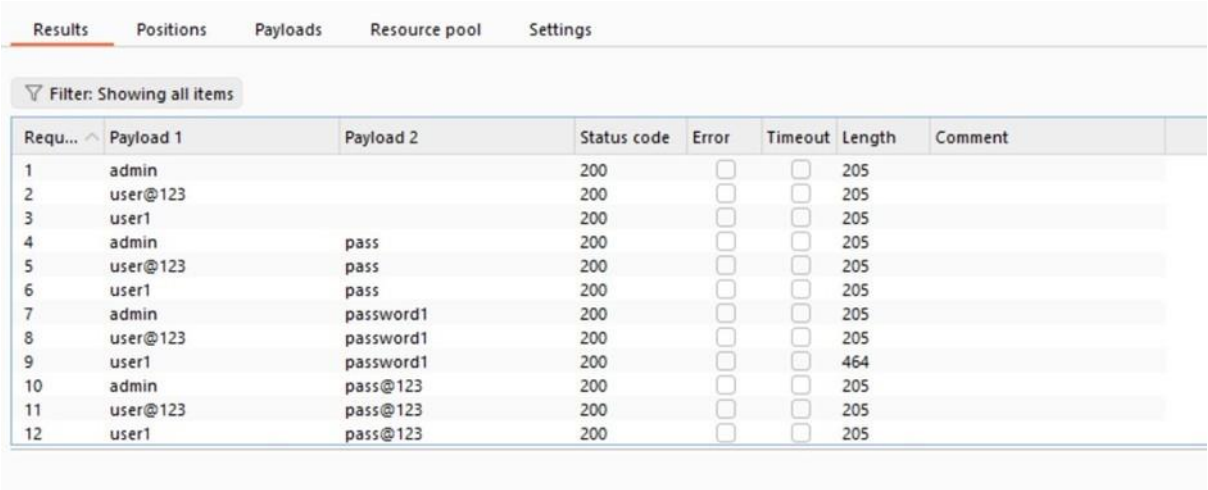
Requ...	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment
0			200			205	
1	admin	pass	200			205	
2	user	pass	200			205	
3	user@123	pass	200			205	
4	admin	password	200			205	
5	user	password	200			205	
6	user@123	password	200			205	
7	admin	pass@123	200			205	
8	user	pass@123	200			205	
9	user@123	pass@123	200			205	

Since, all result lengths are equal, there is no different response and we have not foundthe valid password.

3. Cluster Bomb:

Payload set	1	2
Payload type	Simple list	Simple list
Payload options	admin user1 user@123	pass password1 pass@123

Result:



Requ...	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment
1	admin		200			205	
2	user@123		200			205	
3	user1		200			205	
4	admin	pass	200			205	
5	user@123	pass	200			205	
6	user1	pass	200			205	
7	admin	password1	200			205	
8	user@123	password1	200			205	
9	user1	password1	200			464	
10	admin	pass@123	200			205	
11	user@123	pass@123	200			205	
12	user1	pass@123	200			205	

Since, the 9th entry is different with a result length of 464, we can deduce it is the correct username password combination.

Observations and Results:

Therefore, we've established that employing a Cluster Bomb attack and exploring varied combinations within list1 and list2 enables us to proficiently utilize Burp Suite for implementing the brute force technique, ultimately allowing us to decipher our website's password successfully.

EXPERIMENT 3

Aim:

SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

Experiment Objective:

The objective of this experiment is to demonstrate how to exploit a SQL injection vulnerability to bypass authentication on a vulnerable login page. By simulating a SQL injection attack, participants will understand the risks associated with improper input validation and the importance of secure coding practices in preventing such vulnerabilities.

Introduction:

SQL injection is a common vulnerability that occurs when an application fails to properly sanitize user-supplied input before using it in an SQL query. Attackers can exploit this vulnerability to manipulate the SQL queries executed by the application, potentially leading to unauthorized access, data leakage, and other security breaches. This experiment aims to illustrate the impact of SQL injection vulnerabilities on authentication mechanisms and the overall security of web applications.

Setup:

Software and Tools:

1. Burp Suite
2. Web browser
3. Access to a vulnerable web application

Instructions:

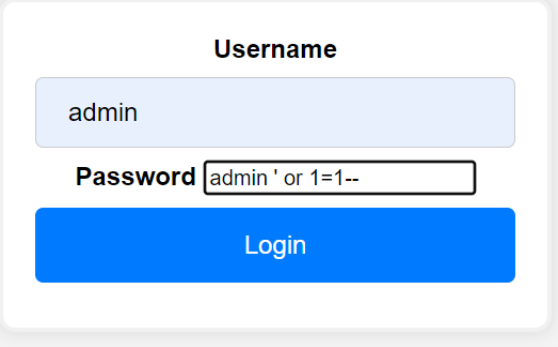
1. Install and configure Burp Suite on your system.
2. Download and set up a vulnerable web application.
3. Ensure that Burp Suite is correctly configured with your web browser for proxying HTTP requests.
4. Access the vulnerable login page of the web application in your browser.

Experiment Steps:

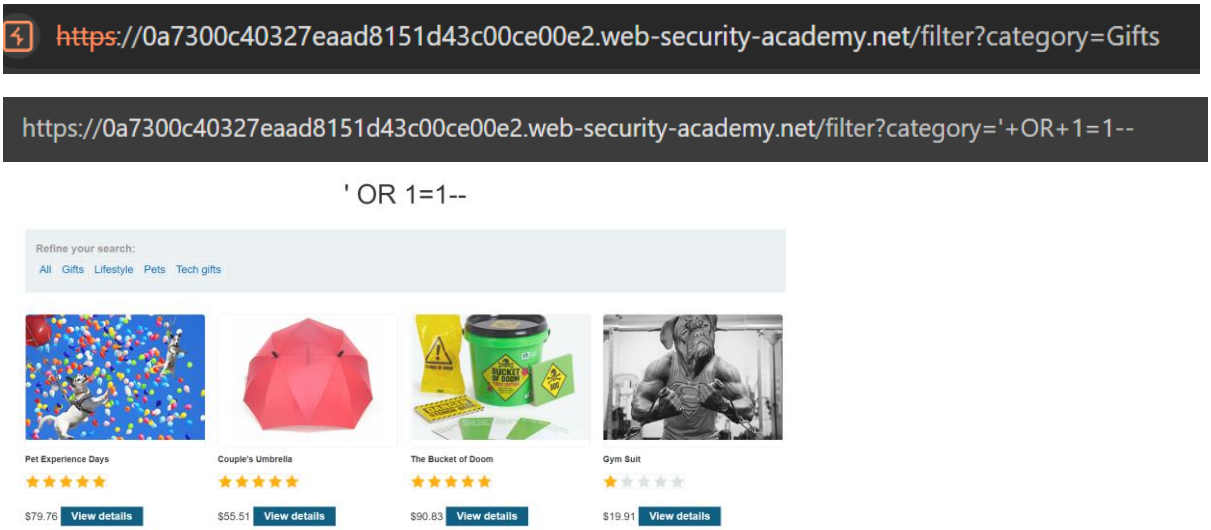
1. Disable interception in the Burp Proxy tab and navigate to the login page of the target web application in your browser.
2. Submit the login request with credentials and observe any error messages.
3. Modify the password parameter in the login page to inject a SQL code (e.g. admin ' or 1=1 --).
4. Verify if the SQL injection attack was successful by bypassing authentication.

Observations:

The SQL injection attack successfully bypassed the authentication mechanism, allowing access to the application as the administrator user.



SQL injection vulnerability in WHERE clause allowing retrieval of hidden data



Results:

1. The experiment underscores the critical importance of implementing robust input validation mechanisms to mitigate the risk of SQL injection attacks.
2. Developers should utilize secure coding practices, such as parameterized queries and input sanitization, to prevent SQL injection vulnerabilities in their applications.
3. Security professionals should regularly assess and test web applications for SQL injection vulnerabilities.

EXPERIMENT 4

Aim:

Cross site scripting

Experiment Objective:

To understand and demonstrate the impact of Cross-Site Scripting (XSS) vulnerabilities in web applications using Burp Suite.

Introduction :

XSS vulnerabilities pose a significant threat by allowing attackers to inject malicious scripts into websites, leading to potential data theft, session hijacking, and other harmful consequences. This experiment provides an opportunity to delve into the realm of web security, emphasizing the importance of identifying and mitigating XSS vulnerabilities.

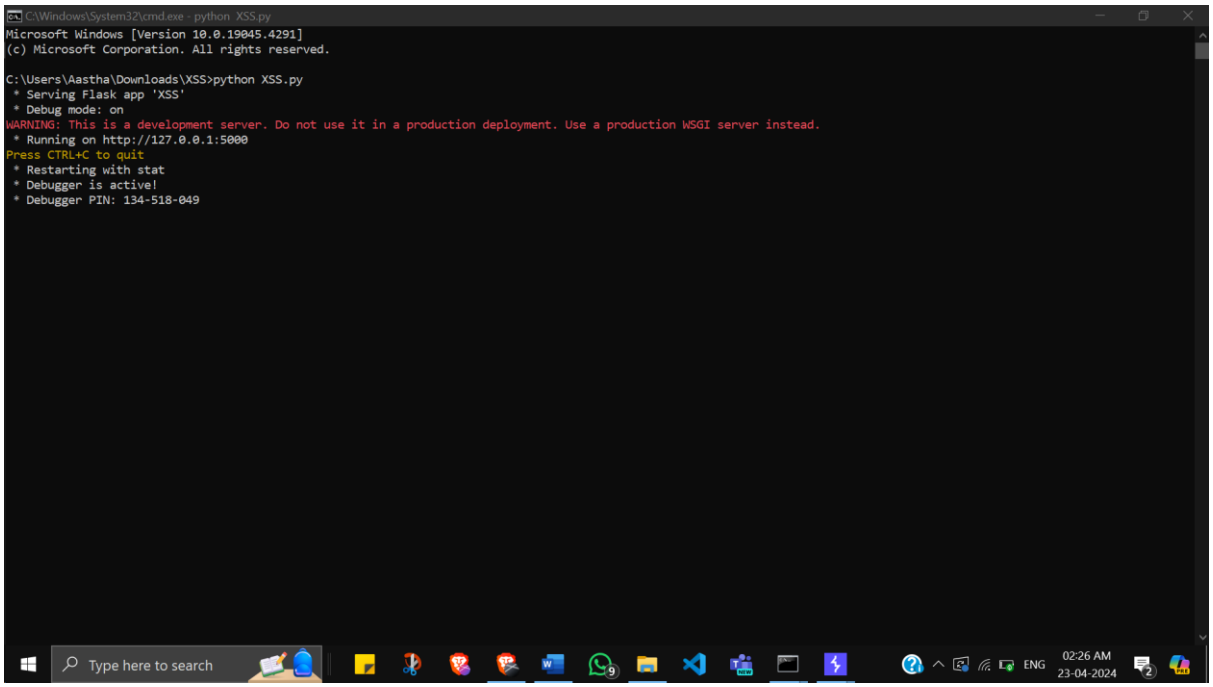
Setup:

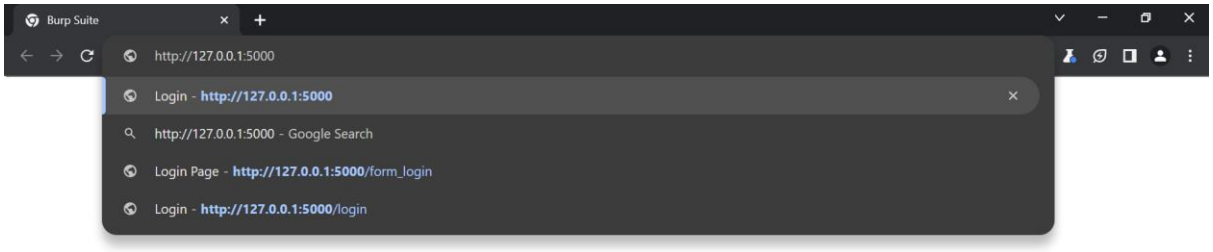
Our primary tool for this experiment is Burp Suite, a powerful web application testing toolkit widely used by cybersecurity professionals and ethical hackers. Burp Suite helps in identifying security vulnerabilities, understanding how they can be exploited, and recommending preventive measures.

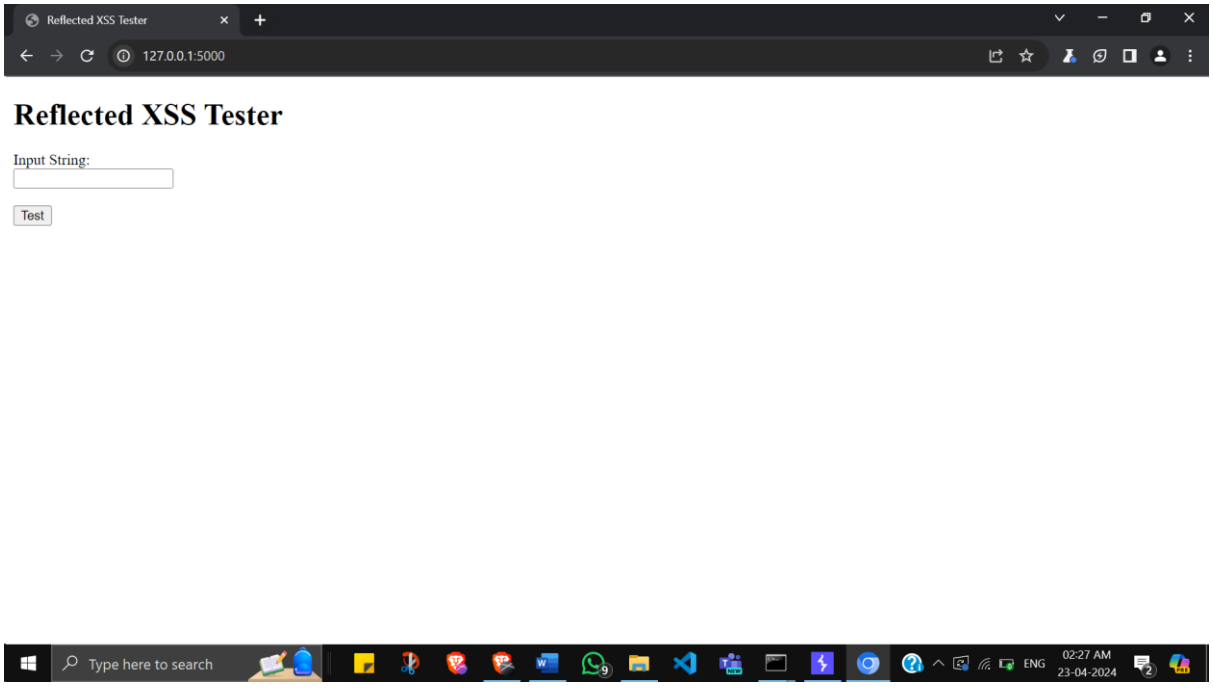
Flask app made by me

Experiment Steps:

Step 1: First, ensure that Burp is correctly configured with your browser.With intercept turned off in the Proxy "Intercept" tab, visit the web application you are testing in your browser.

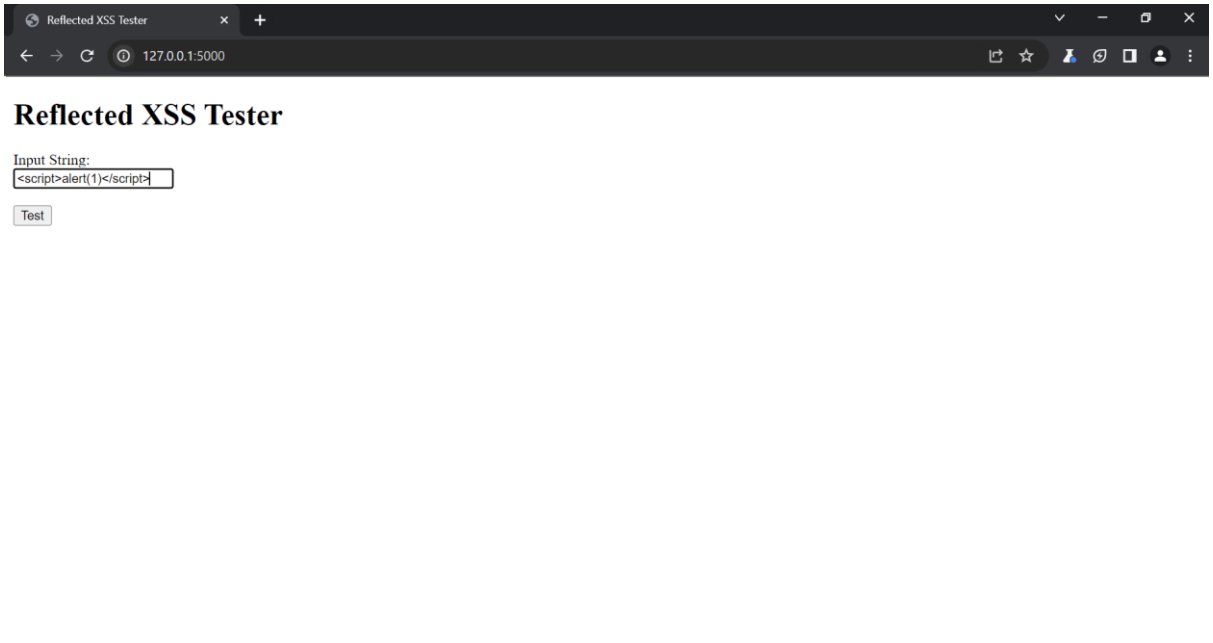


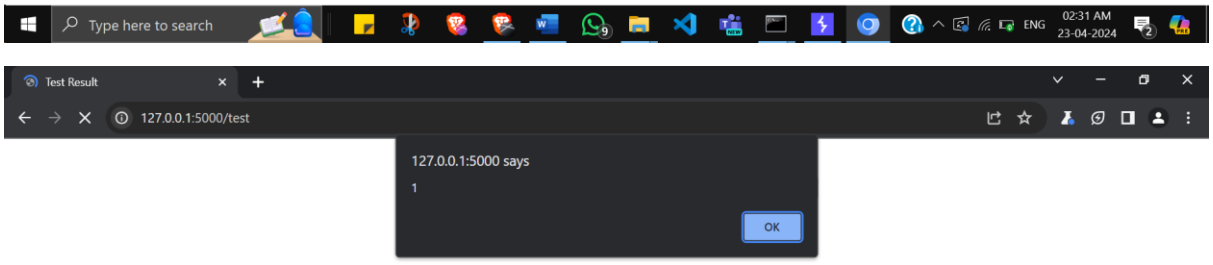




Step 2: Return to Burp. In the Proxy "Intercept" tab, ensure "Intercept is on". A simple payload such as <s> can often be used to check for issues. In this example we have used a command that attempts to perform a pop up in our browser.

Step 3: Type in "`<script>alert(1)</script>`" in the input box. A pop up message will immediately show implying scripting is being used.





Observation and Results:

1. The experiment underscores the critical importance of implementing robust input validation mechanisms to mitigate the risk of SQL injection attacks.
2. Developers should utilize secure coding practices, such as parameterized queries and input sanitization, to prevent SQL injection vulnerabilities in their applications.
3. Security professionals should regularly assess and test web applications for SQL injection vulnerabilities.

EXPERIMENT 5

Aim:

Cross Site request Forgery

Experiment Objective:

The main goal of the CSRF experiment using Burp Suite is to provide students with practical insights into the detection, exploitation, and prevention of Cross-Site Request Forgery vulnerabilities in web applications.

Introduction :

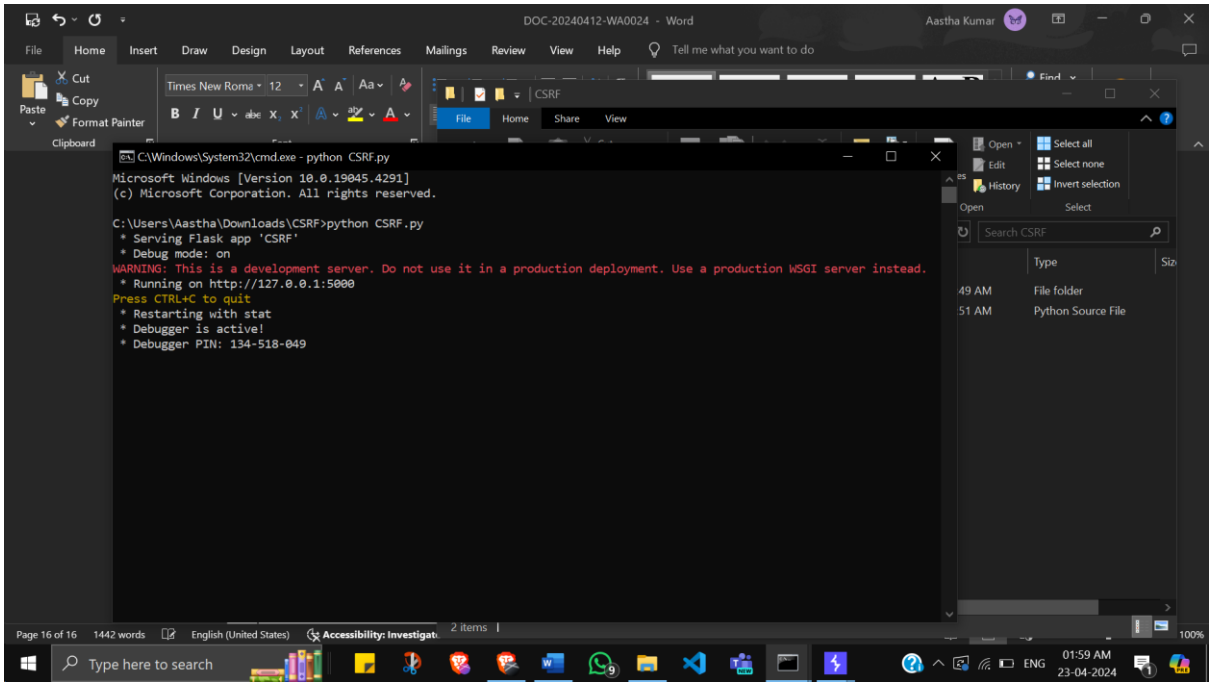
Cross-site request forgery (CSRF) is an attack which forces an end user to execute unwanted actions on a web application to which they are currently authenticated. CSRF vulnerabilities may arise when applications rely solely on HTTP cookies to identify the user that has issued a particular request. Because browsers automatically add cookies to requests regardless of the request's origin, it may be possible for an attacker to create a malicious web site that forges a cross-domain request to the vulnerable application.

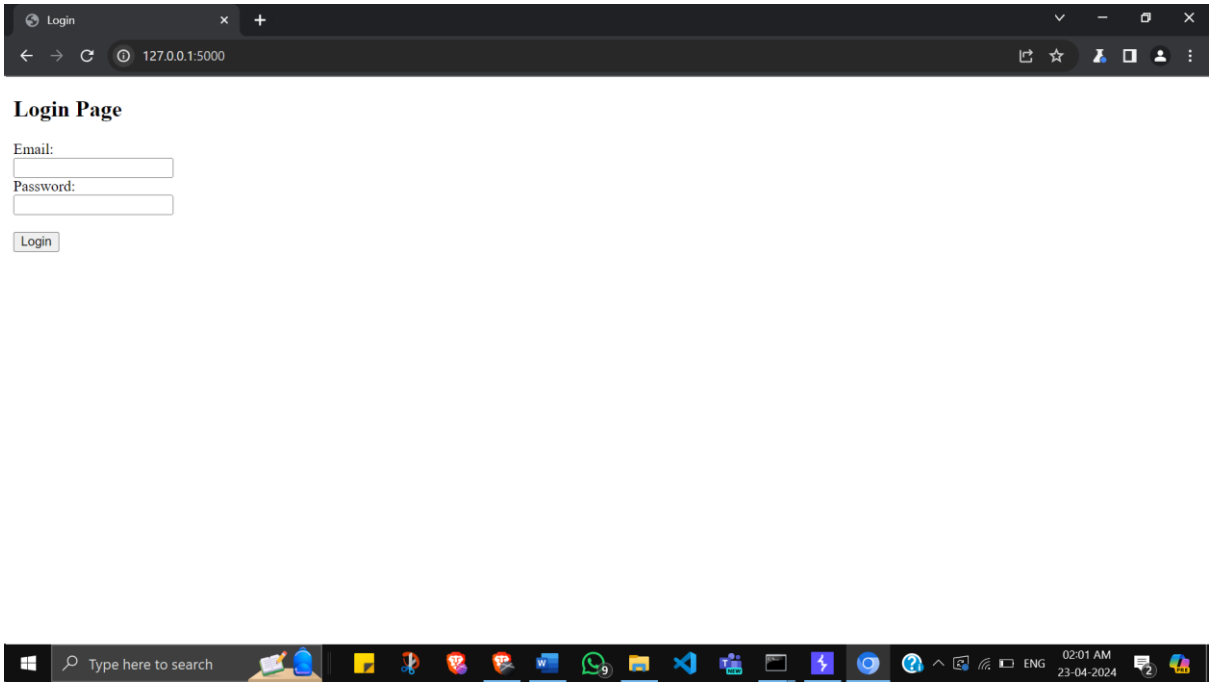
Setup:

Burp Suite
Flask App created by me (login page)

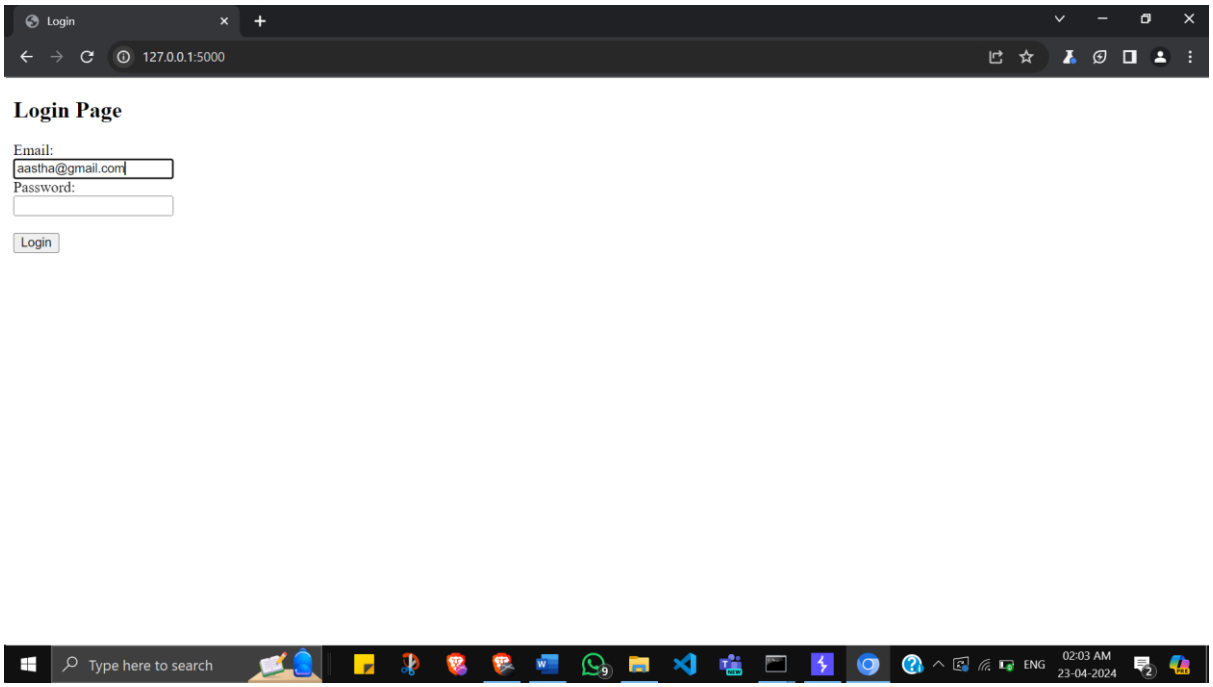
Experiment Steps:

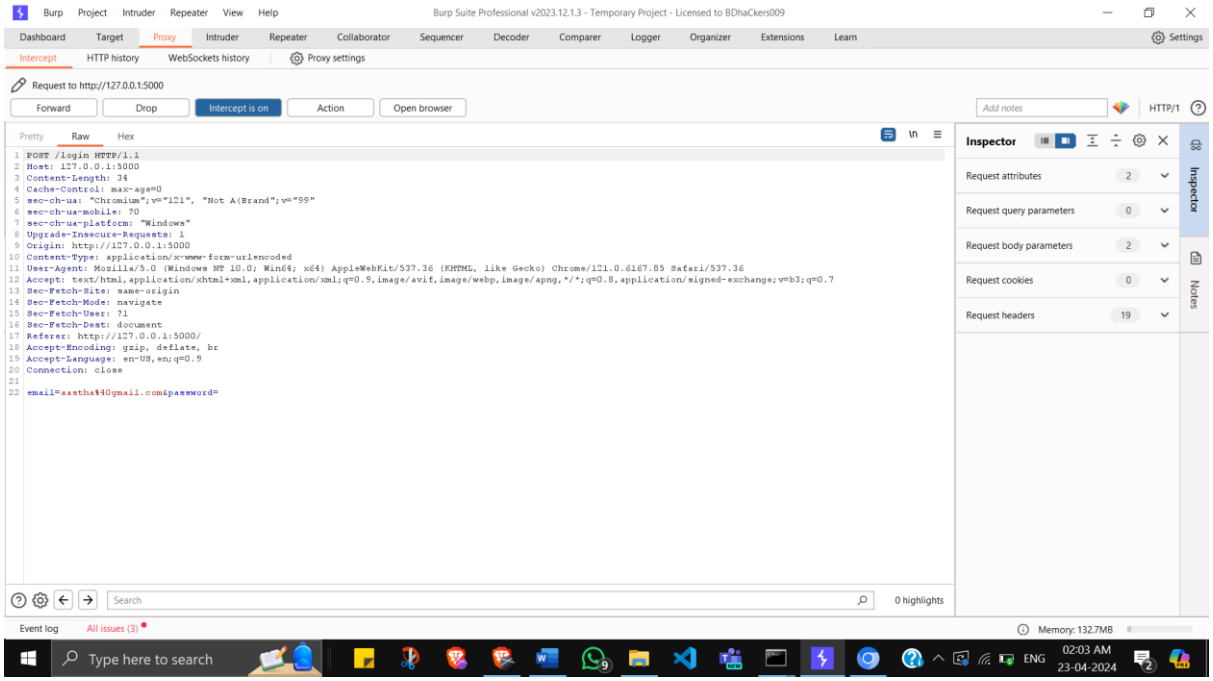
Step 1: To manually test for CSRF vulnerabilities, first, ensure that Burp is correctly configured with your browser.In the Burp Proxy "Intercept" tab, ensure "Intercept is off". Visit the web application you are testing in your browser.



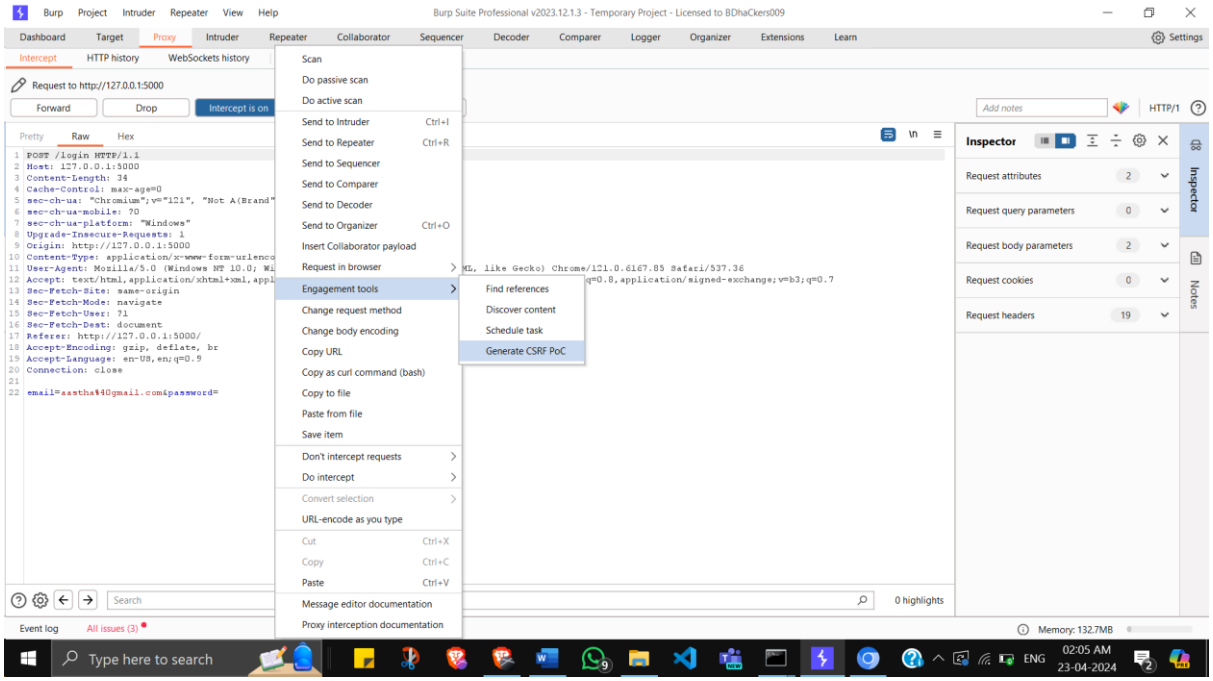


Step 2: Alter the value in the field/s you wish to change, in this case "Email". Make sure “Intercept is on”

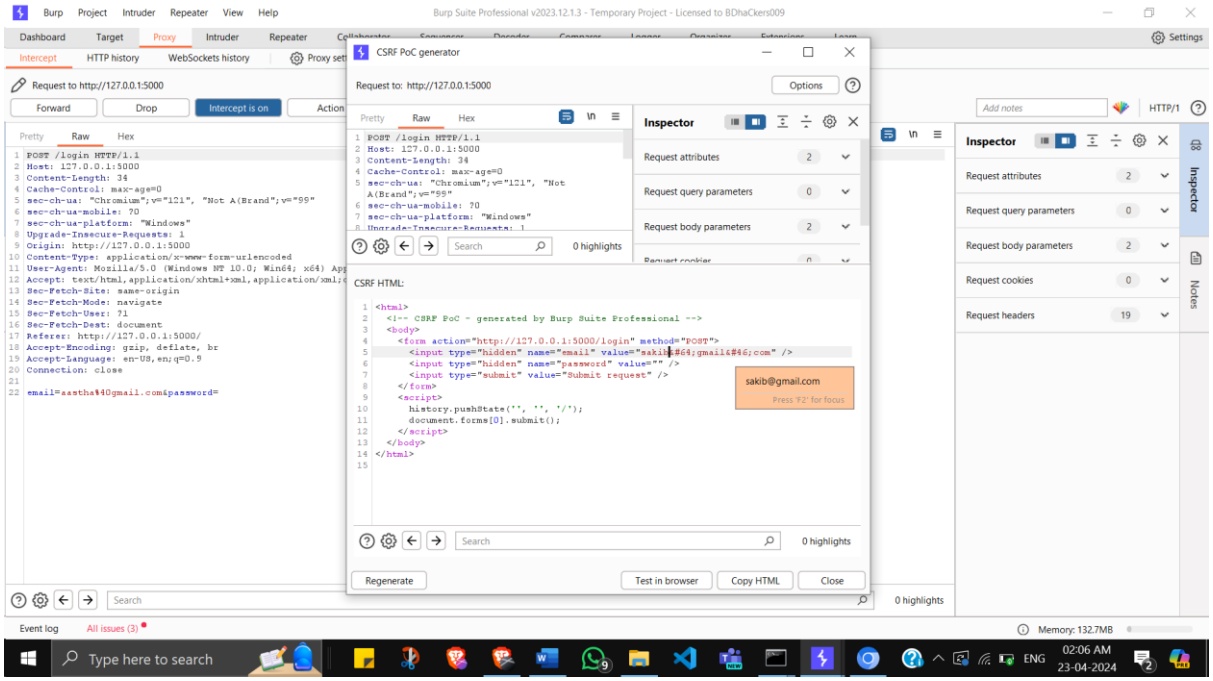




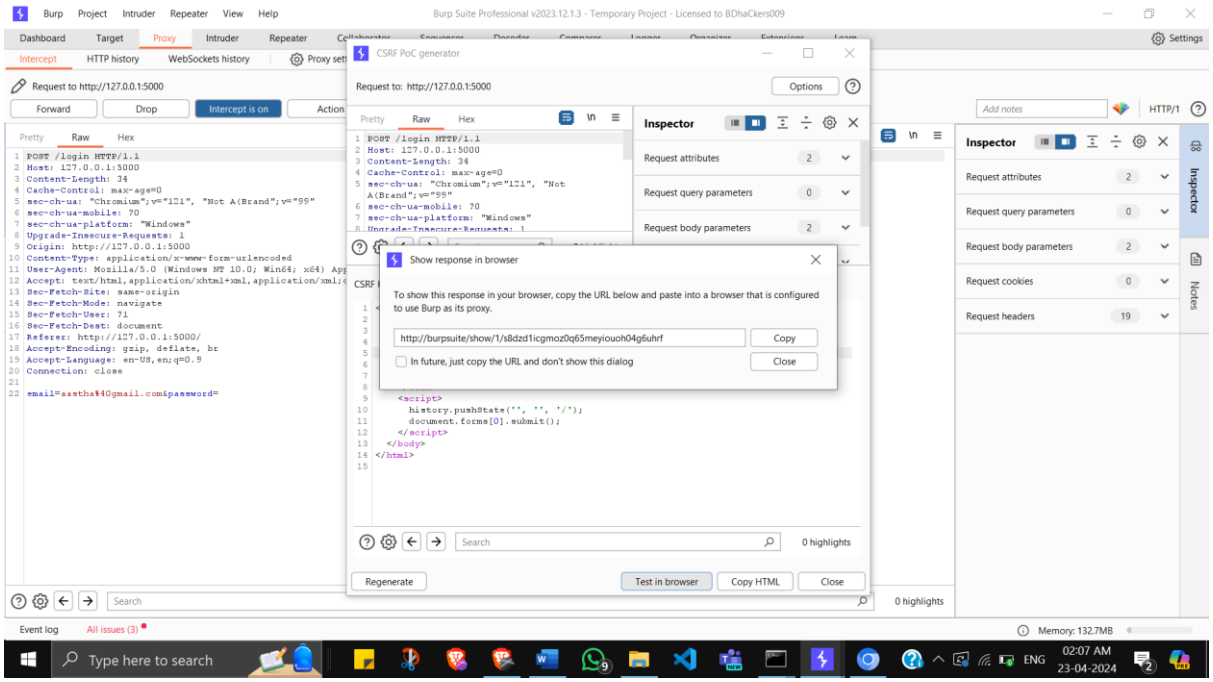
Step 3: Submit the request so that it is captured by Burp. In the "Proxy" tab, right click on the raw request to bring up the context menu. Go to the "Engagement tools" options and click "Generate CSRF PoC".



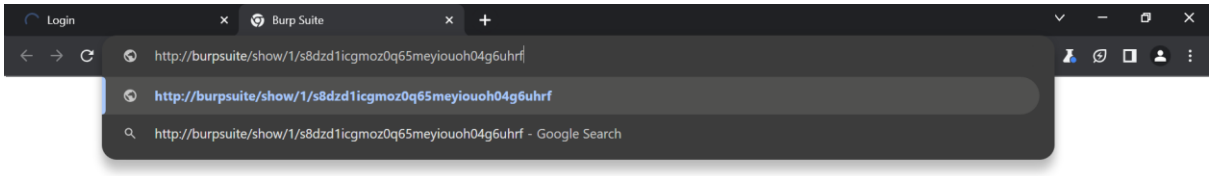
Step 4: In the "CSRF PoC generator" window you should alter the value of the user supplied input. In this example we will email from aastha@gmail.com to sakib@gmail.com



Step 5: Select “test in browser”. A dialog box will appear with a link. Copy it.

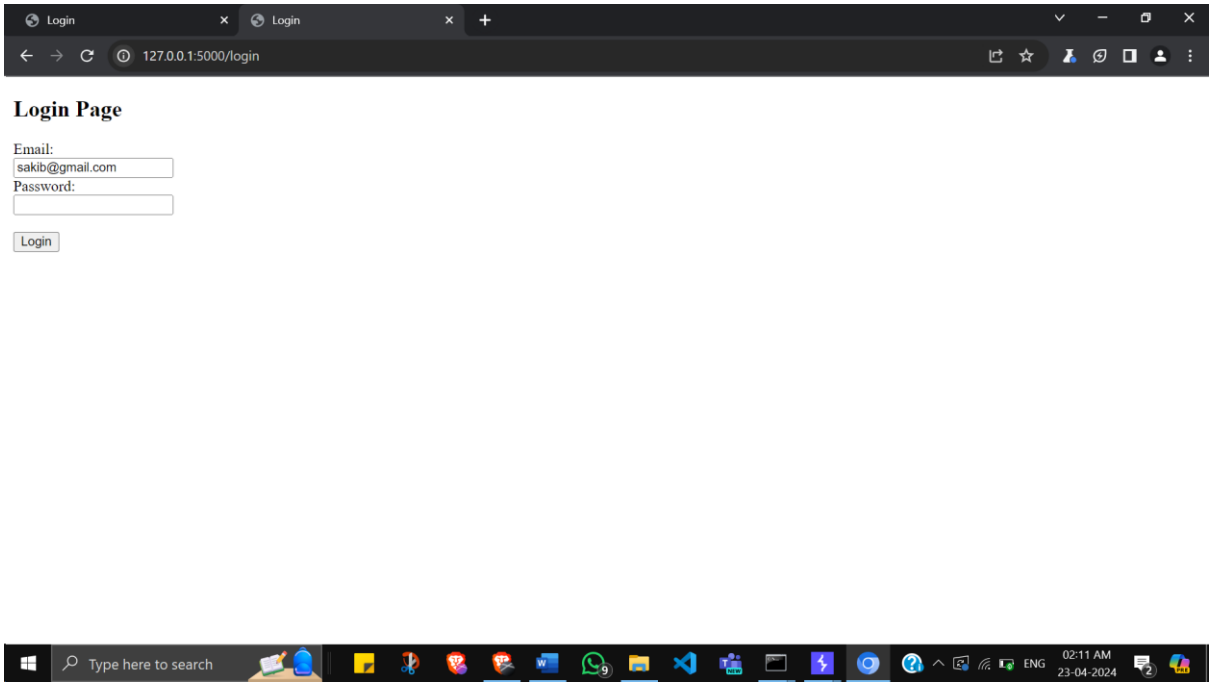


Step 6: In the Proxy "Intercept" tab, ensure "Intercept is off". Paste the URL in the same browser.



Step 7: Dependent on the CSRF PoC options you may need to submit the request or it may be submitted automatically. In this case we are submitting the request manually.

Step 8: If the attack has been successful and the account information has been successfully changed, this serves as an initial check to verify whether the attack is plausible.


Login
Login
127.0.0.1:5000/login
Login Page
Email:
sakib@gmail.com
Password:
Login
Type here to search
ENG
02:11 AM
23-04-2024

EXPERIMENT 6

Aim:

The aim of this experiment is to demonstrate the use of encryption and digital signatures for secure communication using Kleopatra software,

Experiment Objective:

The objective of this experiment is to:

1. Encrypt a message using Kleopatra with a chosen key and send it to a friend for decryption.

2. Encrypt a message and sign it using Kleopatra, then send it to a friend for verification of the signature.

Introduction :

In today's digital world, ensuring the confidentiality and integrity of communication is crucial. Encryption and digital signatures are essential techniques for achieving these goals. Encryption protects the content of a message from unauthorized access, while digital signatures verify the authenticity and integrity of the sender.

Kleopatra is an open-source encryption software that provides a user-friendly interface for managing encryption keys, encrypting and decrypting messages, and creating digital signatures. In this experiment, we will use Kleopatra to encrypt messages and demonstrate secure communication between two parties.

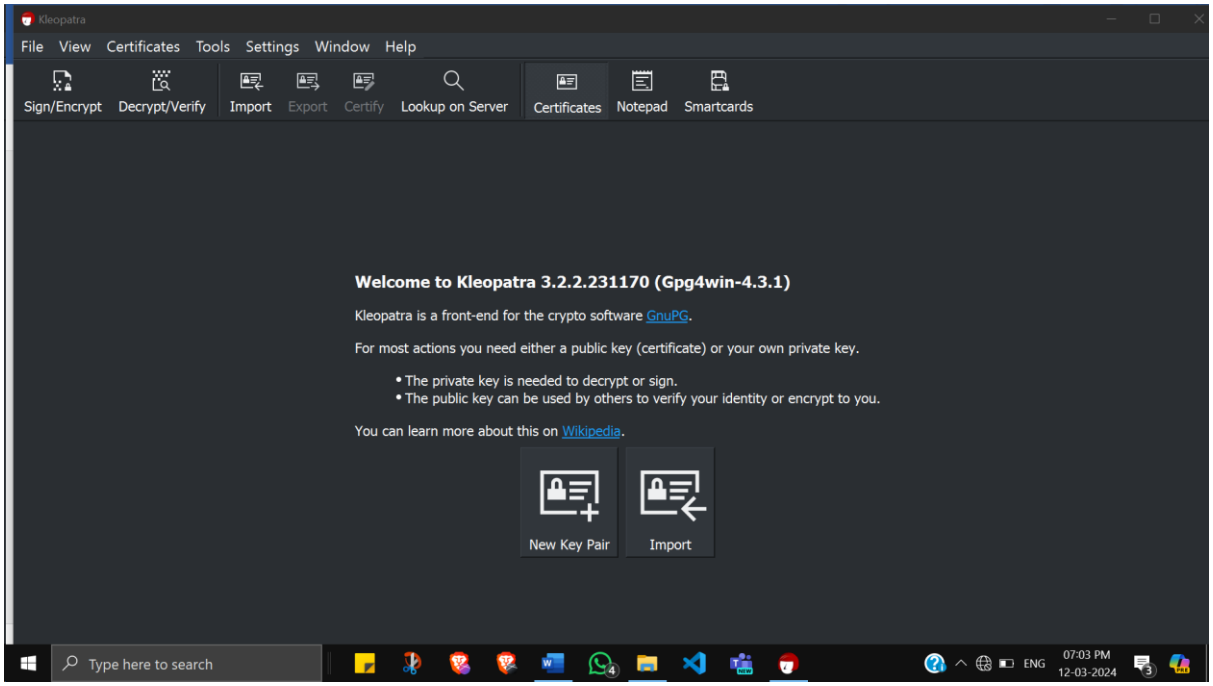
Setup:

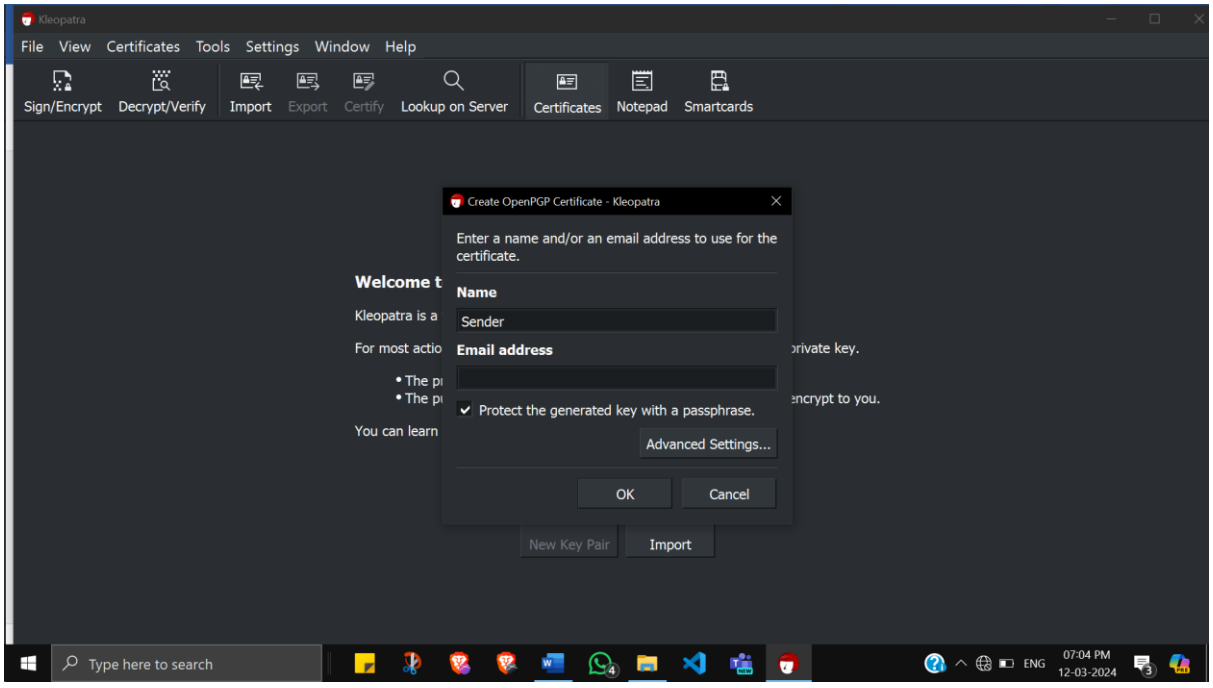
- Kleopatra: Kleopatra is an encryption software that is part of the GNU Privacy Guard (GnuPG) suite. It is available for Windows and Linux operating systems and can be downloaded from the Gpg4win website for Windows users or installed through package managers on Linux systems.
    - Download Kleopatra for Windows: Gpg4win website
    - Install Kleopatra following the installation instructions provided on the website.
- Email Client: You will need an email client to send encrypted messages to your friend. Popular email clients such as Microsoft Outlook, Mozilla Thunderbird, or web-based clients like Gmail can be used.
- Public Key: Obtain the public key of your friend with whom you want to communicate securely. You can import the public key into Kleopatra to encrypt messages for them.
    - Your friend should provide you with their public key, which you can import into Kleopatra.
    - In Kleopatra, go to "File" > "Import Certificates" and select the public key file provided by your friend.

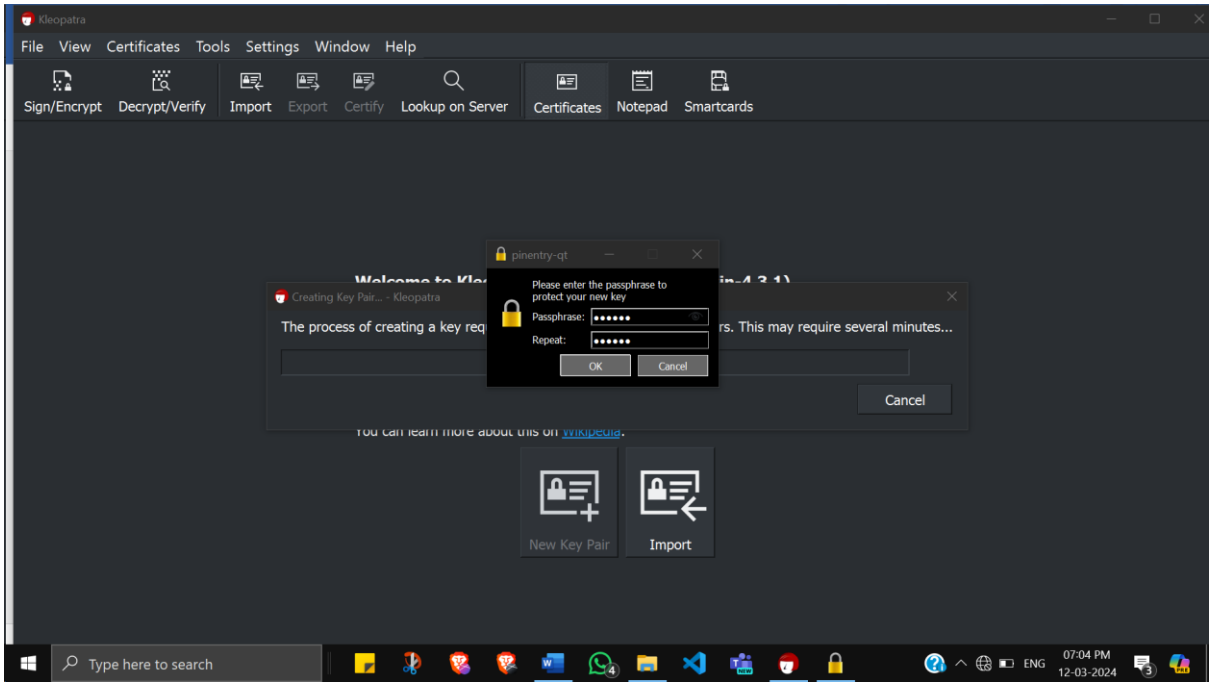
Experiment Steps:

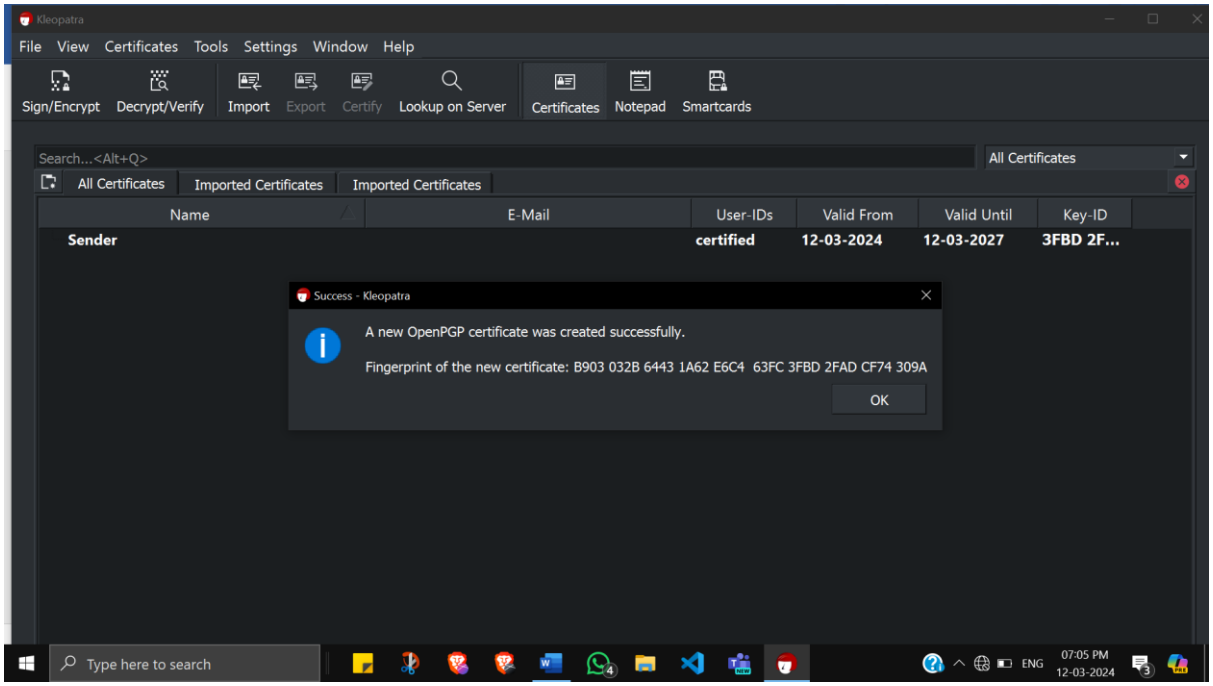
Task 1: Encrypt a message and send it to a friend using any key of your choice. They should be able to decrypt it and retrieve original message.

1. Create a new key pair for the sender's side. Enter a Name and click OK. Enter a passphrase and click OK.

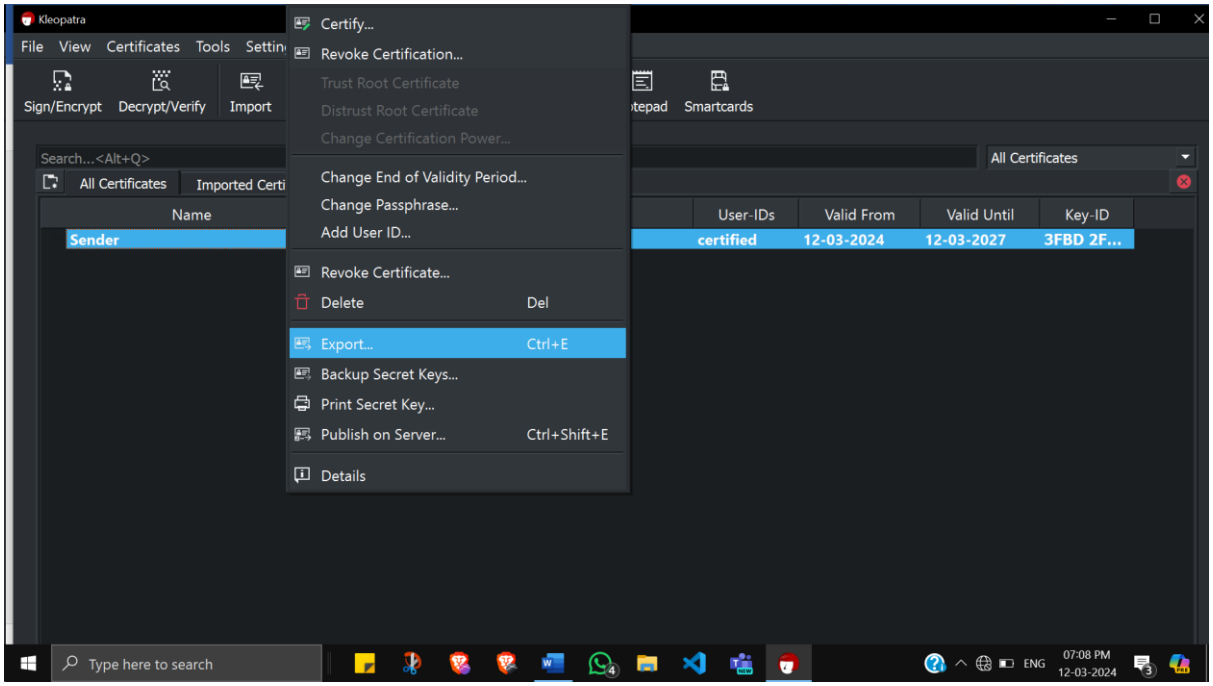


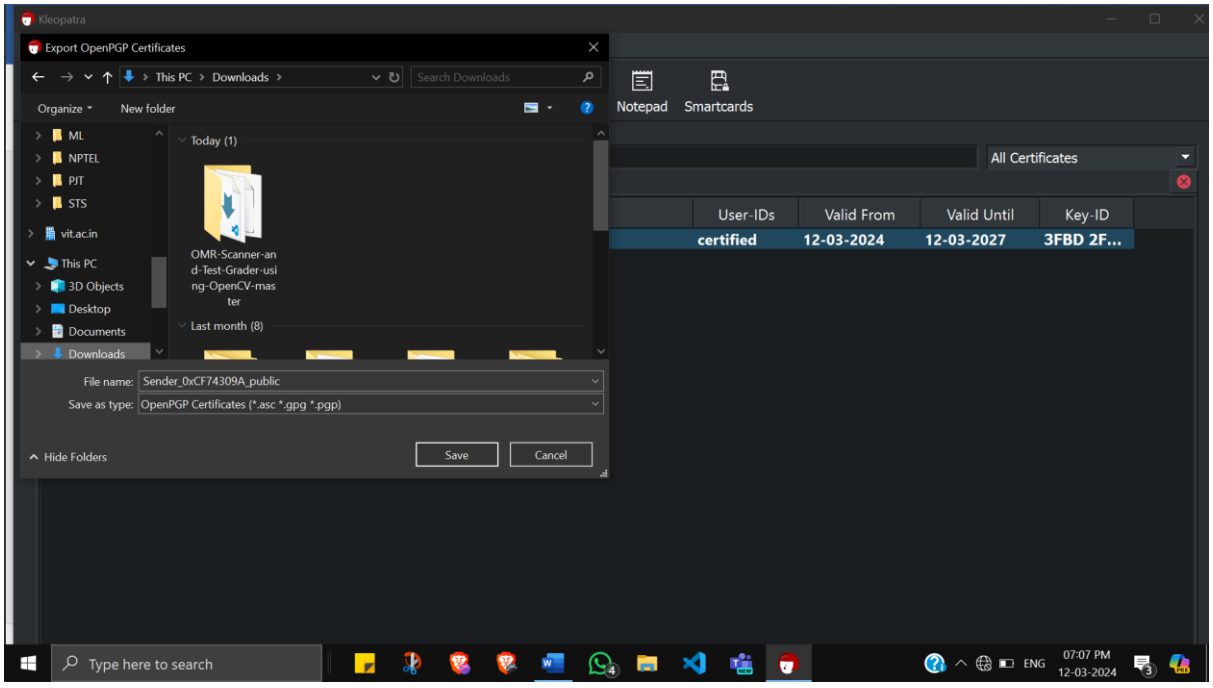




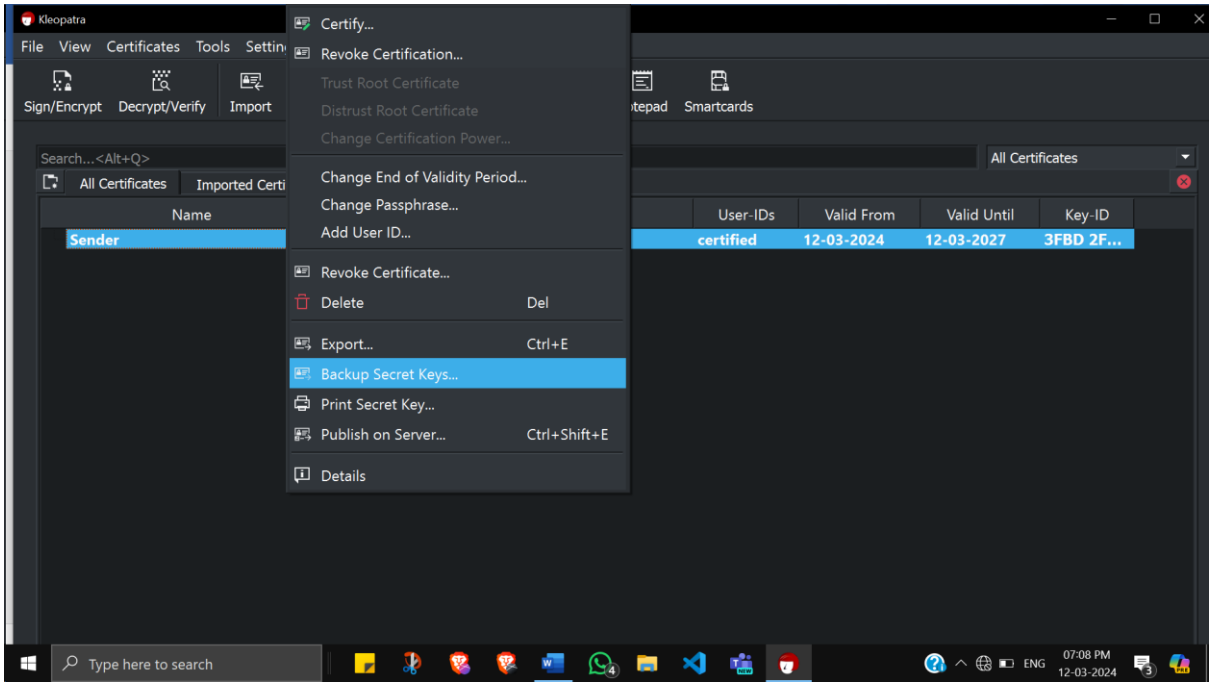


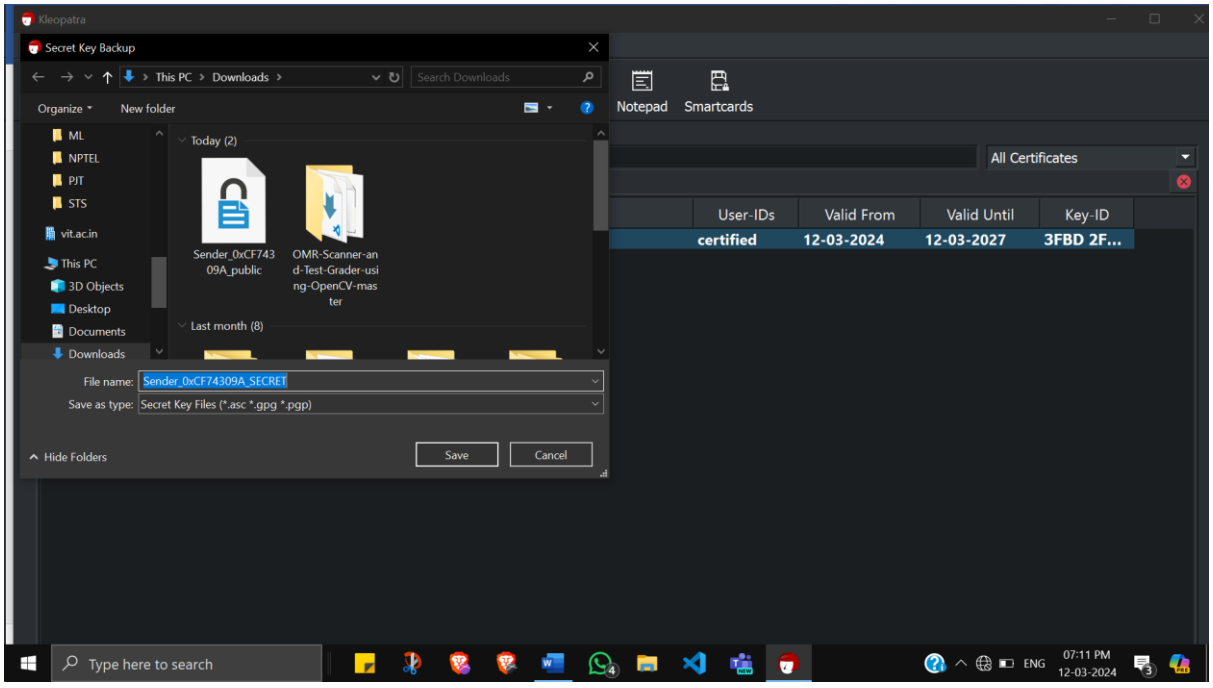
2. Right click on the freshly created certificate and select “Export”. Save the public key.

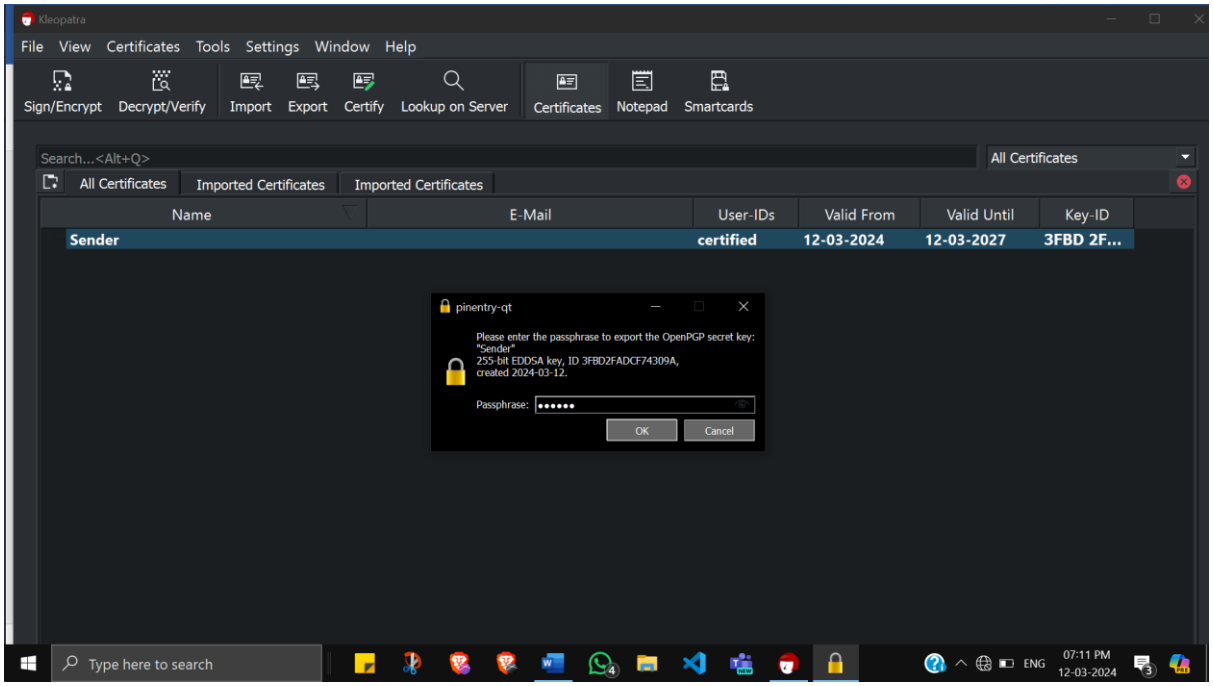




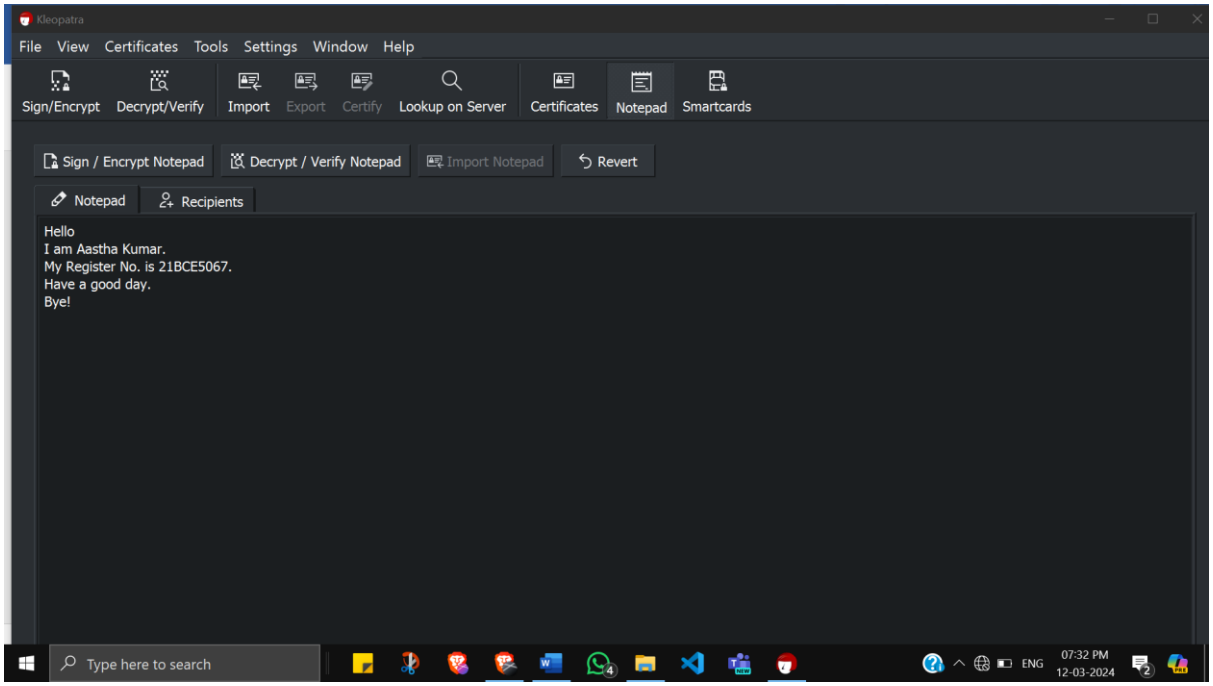
3. Right click on the freshly created certificate and select “Backup Secret Keys”. Save the private key and enter the passphrase.


Kleopatra
File View Certificates Tools Settin
Sign/Encrypt
Decrypt/Verify
Import
Notepad
Smartcards
Search...<Alt+Q>
All Certificates
Imported Certi
Name
Sender
All Certificates
User-IDs
Valid From
Valid Until
Key-ID
certified
12-03-2024
12-03-2027
3FBD 2F...
Certify...
Revoke Certification...
Trust Root Certificate
Distrust Root Certificate
Change Certification Power...
Change End of Validity Period...
Change Passphrase...
Add User ID...
Revoke Certificate...
Delete Del
Export... Ctrl+E
Backup Secret Keys...
Print Secret Key...
Publish on Server... Ctrl+Shift+E
Details
Type here to search
ENG
07:08 PM
12-03-2024

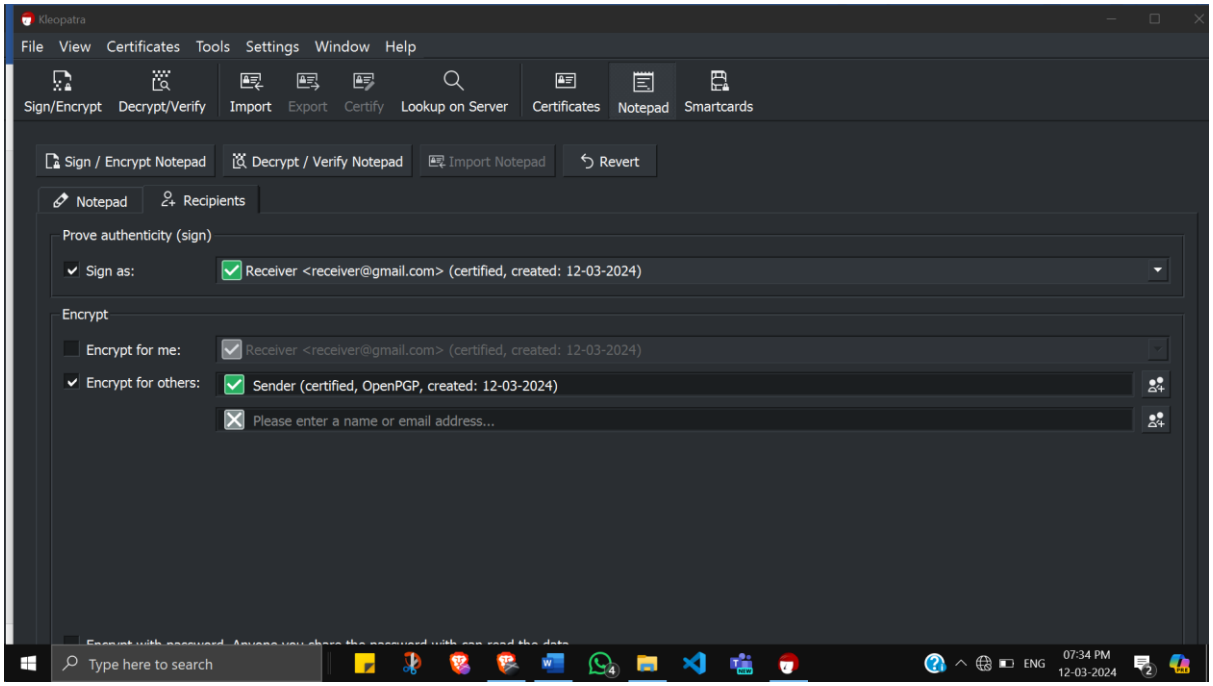

Kleopatra
Secret Key Backup
This PC > Downloads >
Search Downloads
Organize
New folder
ML
NPTEL
PJT
STS
vit.ac.in
This PC
3D Objects
Desktop
Documents
Downloads
Today (2)
Sender_0xCF74309A_public
OMR-Scanner-and-Test-Grader-using-OpenCV-master
Last month (8)
File name: Sender_0xCF74309A_SECRET
Save as type: Secret Key Files (*.asc *.gpg *.pgp)
Hide Folders
Save
Cancel
Notepad
Smartcards
All Certificates
User-IDs
Valid From
Valid Until
Key-ID
certified
12-03-2024
12-03-2027
3FBD 2F...
Type here to search
ENG
07:11 PM
12-03-2024



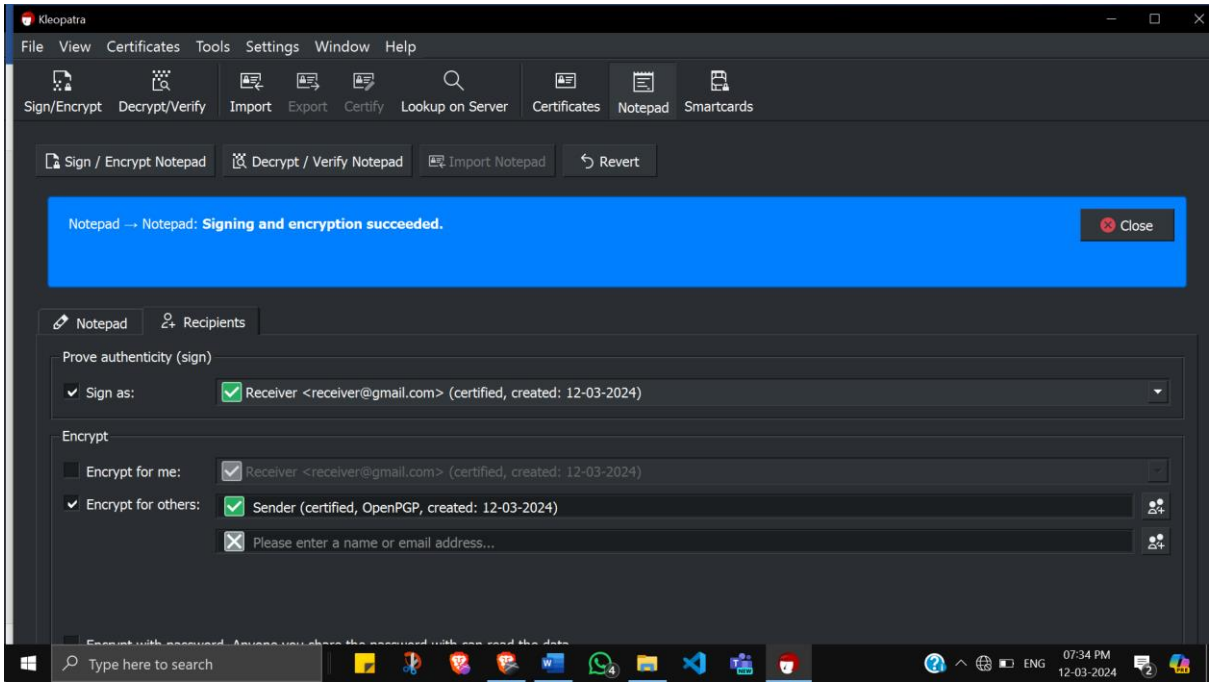
4. Go to Notepad and type in a message.



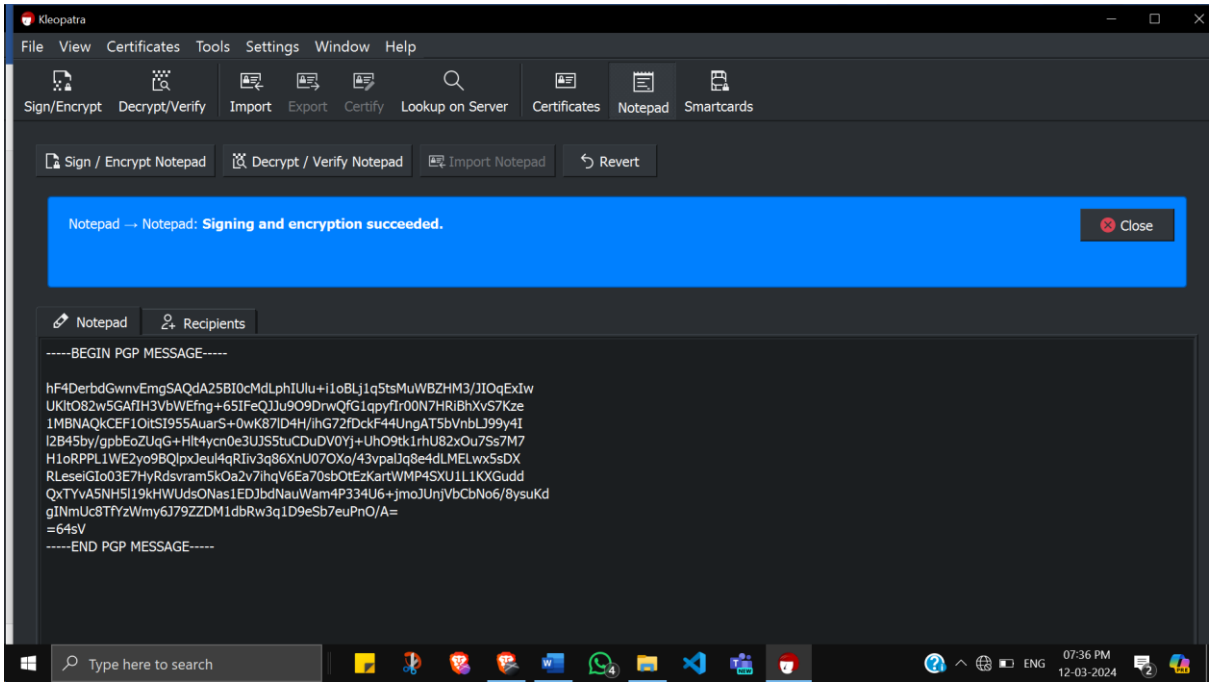
5. Under Recipients tab, select Sender beside “Encrypt for other” option and “Receiver” beside “Sign as” option.



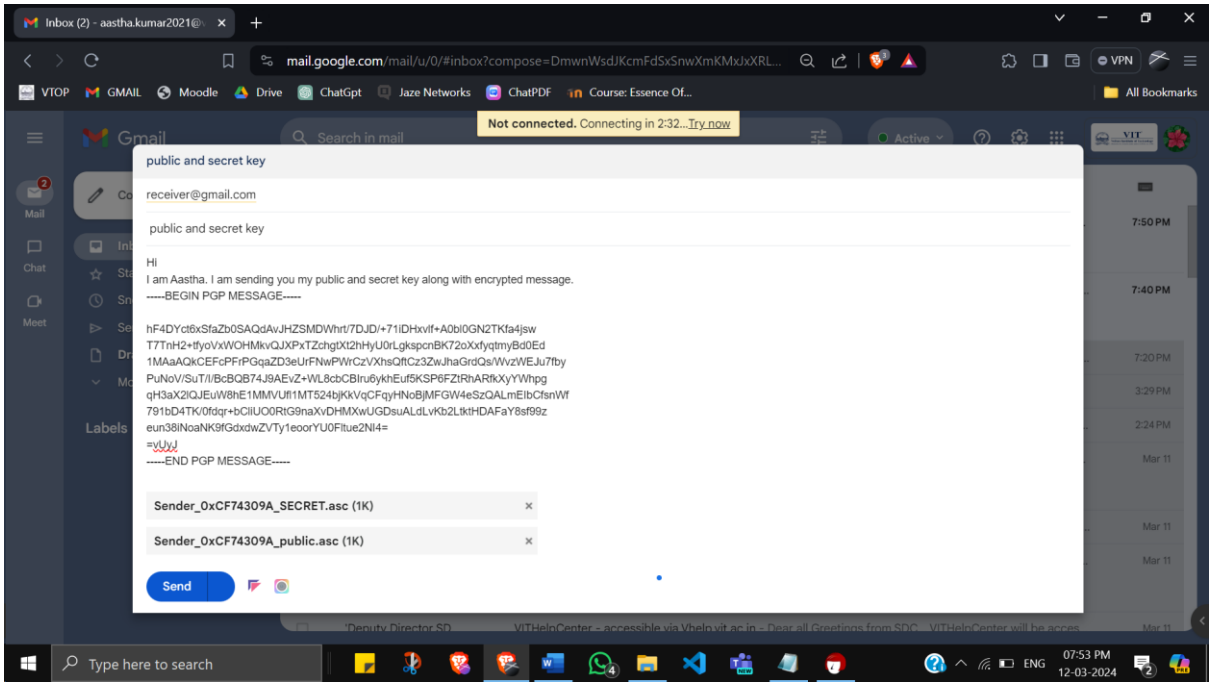
6. Encrypt the message by clicking on “Sign/Encrypt Notepad”



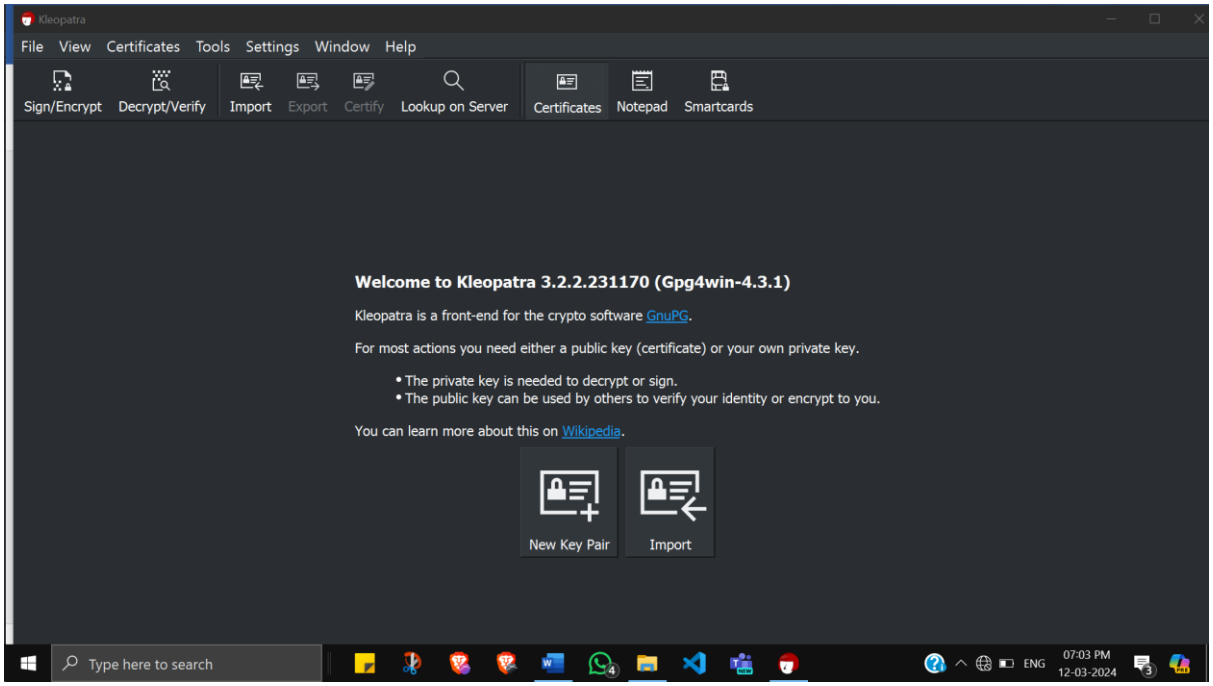
7. Go back to Notepad and now you can view the encrypted message.

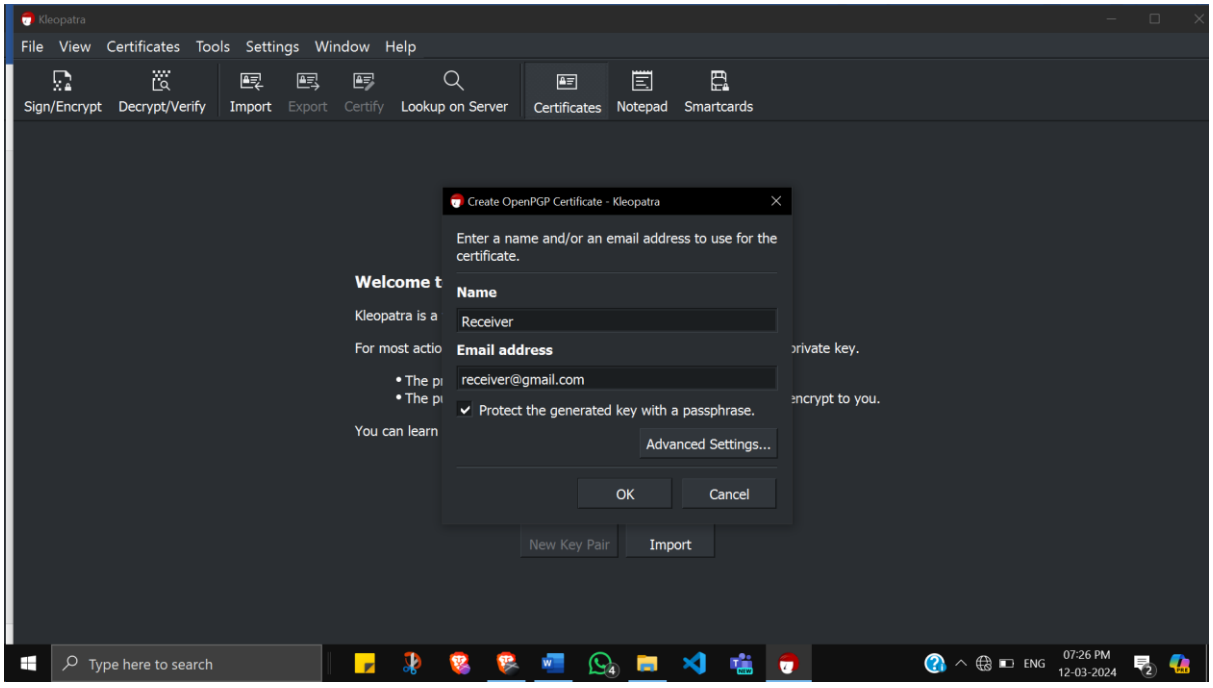


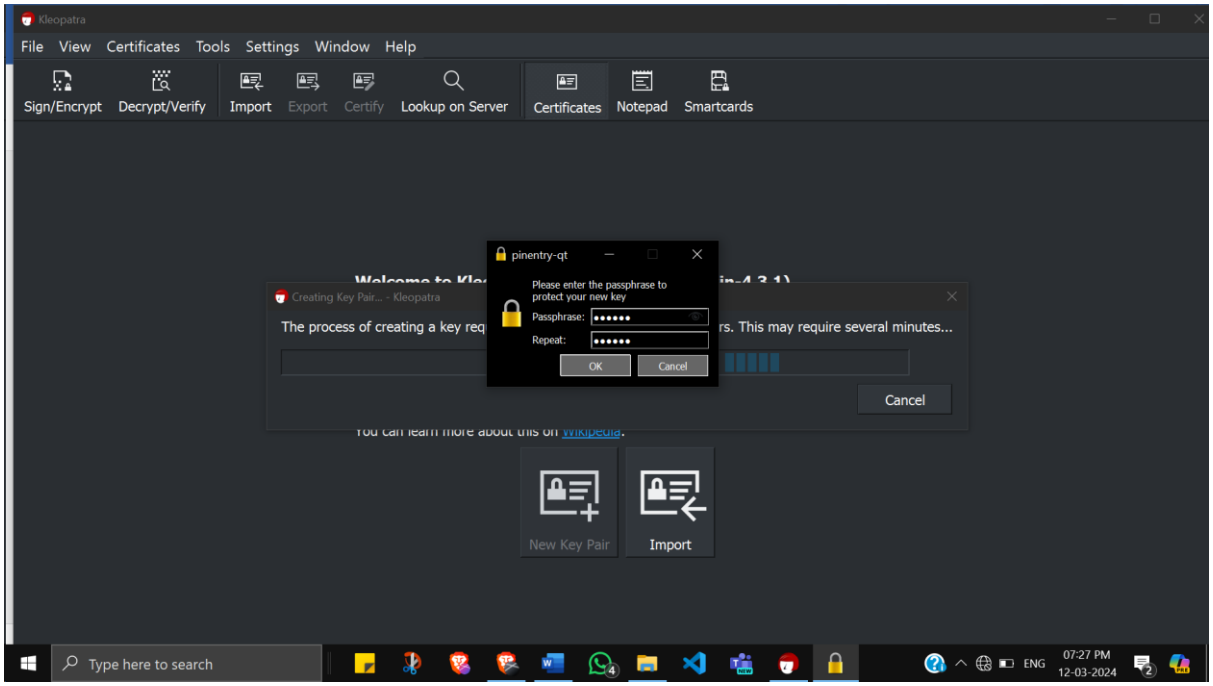
8. Mail the public key and private key to the recipient along with the encrypted message.

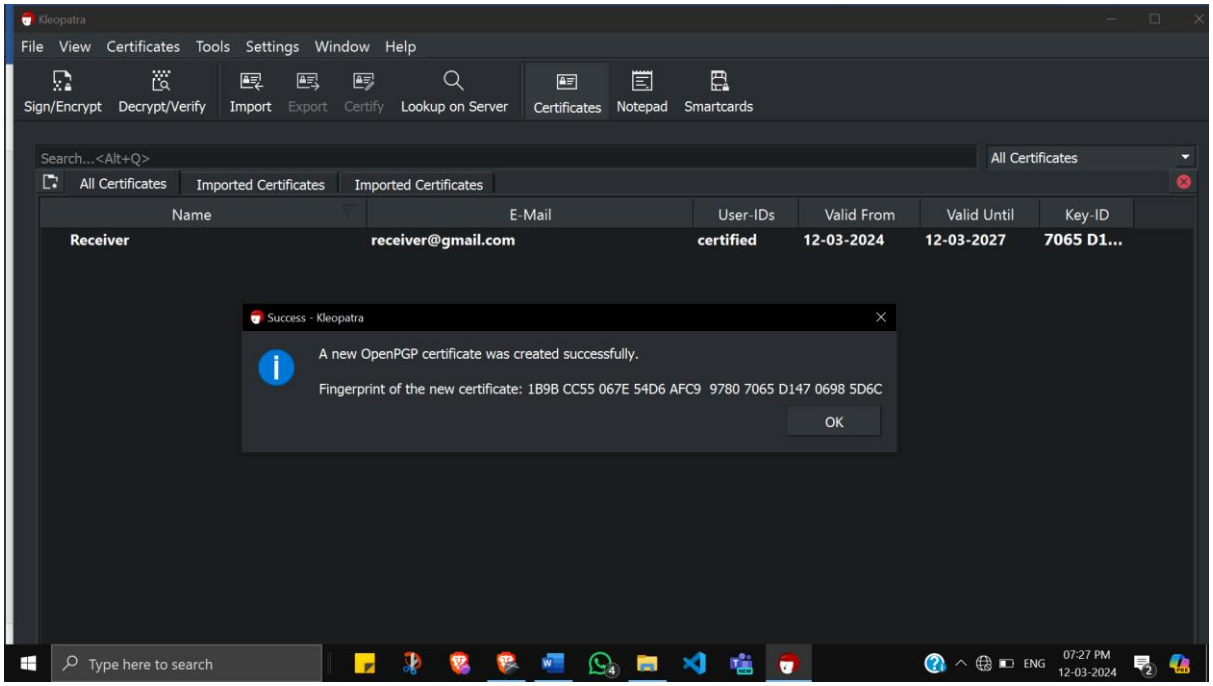


9. Receiver will download the public and private key into their PC.
10. Create a new key pair for the receiver's side. Enter a Name and click OK. Enter a passphrase and click OK.

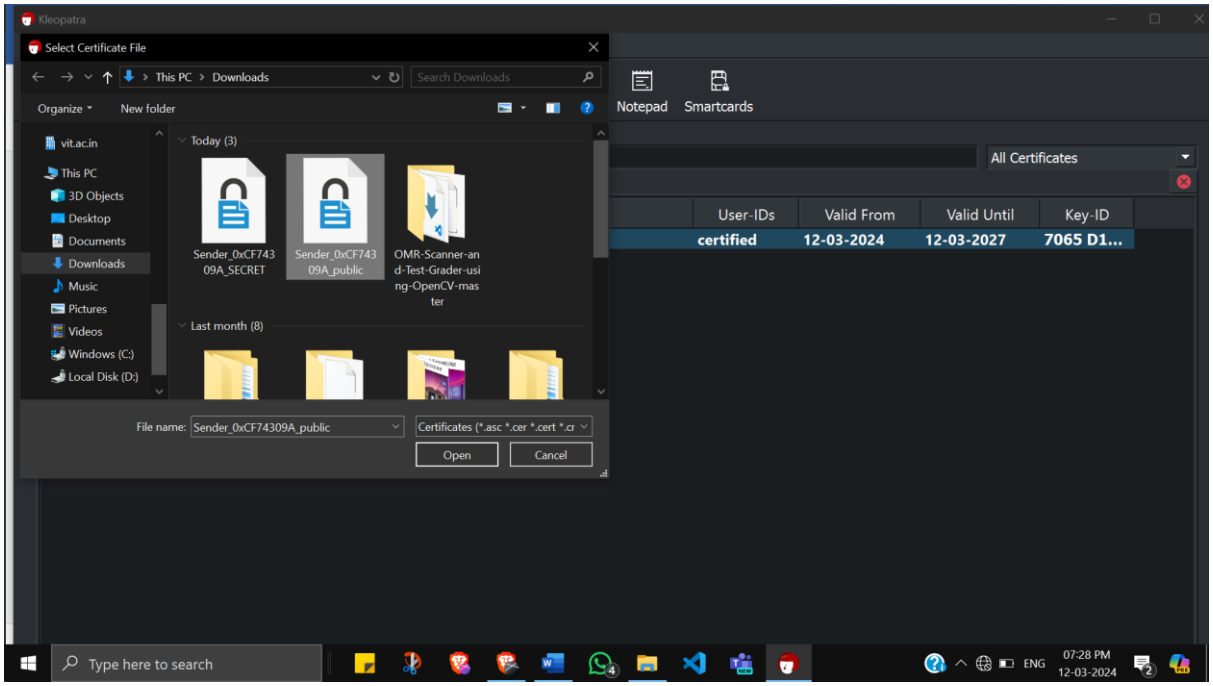

Kleopatra
File View Certificates Tools Settings Window Help
Sign/Encrypt
Decrypt/Verify
Import
Export
Certify
Lookup on Server
Certificates
Notepad
Smartcards
Welcome to Kleopatra 3.2.2.231170 (Gpg4win-4.3.1)
Kleopatra is a front-end for the crypto software GnuPG.
For most actions you need either a public key (certificate) or your own private key.
• The private key is needed to decrypt or sign.
• The public key can be used by others to verify your identity or encrypt to you.
You can learn more about this on Wikipedia.
New Key Pair
Import
Type here to search
ENG
07:03 PM
12-03-2024

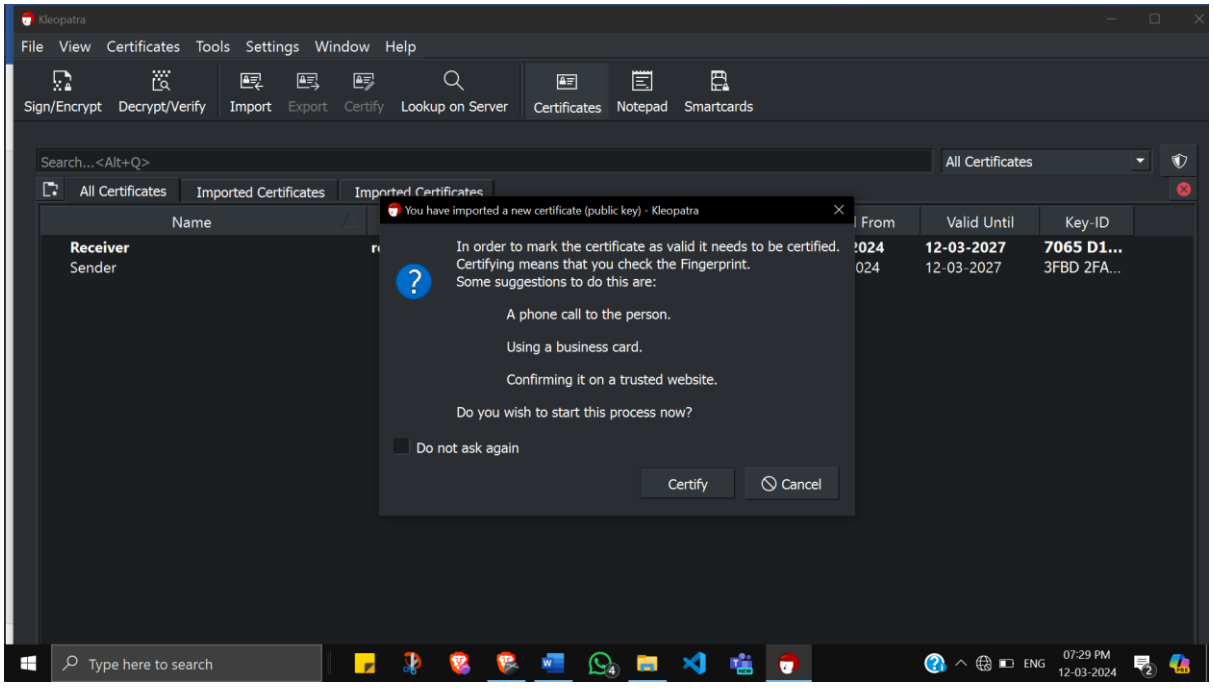

Kleopatra
File View Certificates Tools Settings Window Help
Sign/Encrypt
Decrypt/Verify
Import
Export
Certify
Lookup on Server
Certificates
Notepad
Smartcards
Create OpenPGP Certificate - Kleopatra
Enter a name and/or an email address to use for the certificate.
Name
Receiver
Email address
receiver@gmail.com
Protect the generated key with a passphrase.
Advanced Settings...
OK
Cancel
New Key Pair
Import
Type here to search
ENG
07:26 PM
12-03-2024

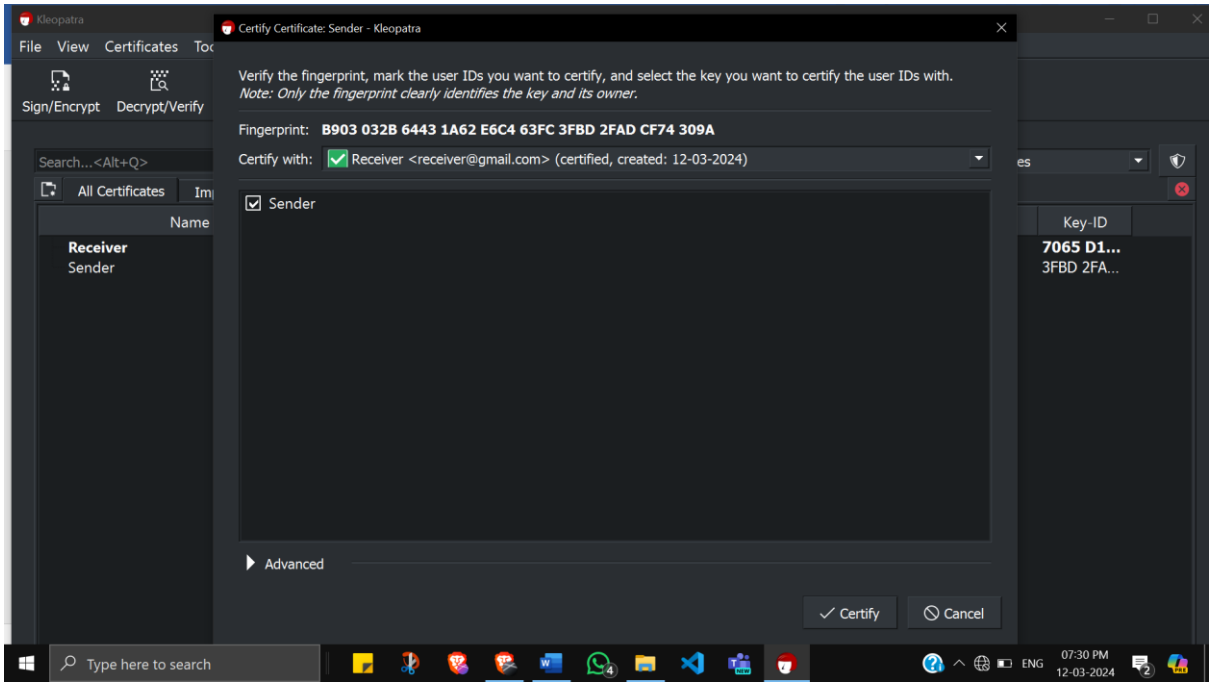




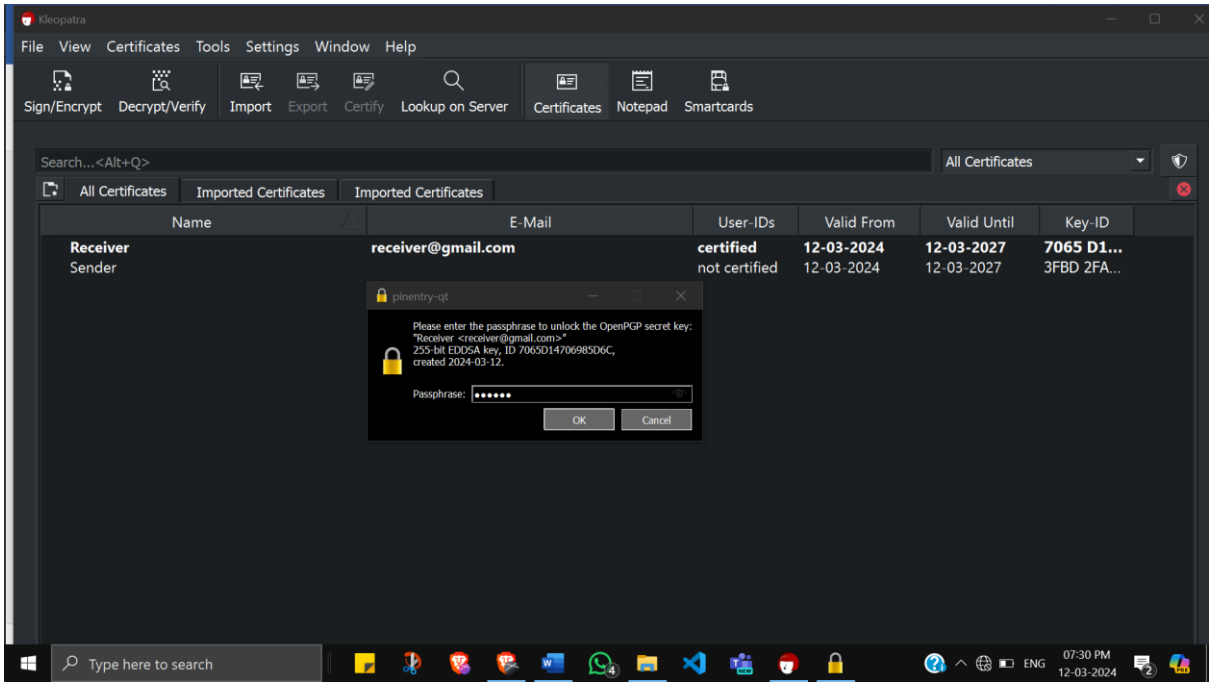
11. Click on “Import” and select public key. Click Open. Select “Certify” on the next dialogue box.

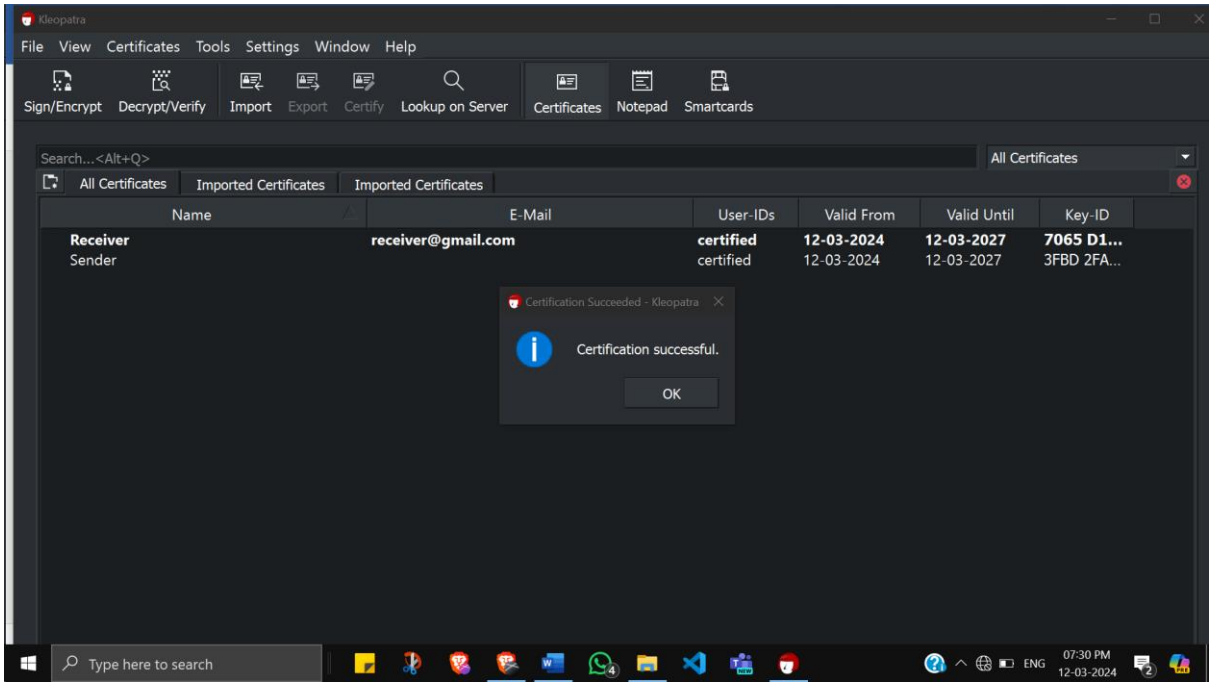

Kleopatra
Select Certificate File
This PC › Downloads
Search Downloads
Organize
New folder
vit.ac.in
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows (C:)
Local Disk (D:)
Today (3)
Sender_0xCF74309A_SECRET
Sender_0xCF74309A_public
OMR-Scanner-and-Test-Grader-using-OpenCV-master
Last month (8)
File name: Sender_0xCF74309A_public
Certificates (*.asc *.cer *.cert *.cr
Open
Cancel
Notepad
Smartcards
All Certificates
User-IDs
Valid From
Valid Until
Key-ID
certified
12-03-2024
12-03-2027
7065 D1...
Type here to search
ENG
07:28 PM
12-03-2024


Kleopatra
File View Certificates Tools Settings Window Help
Sign/Encrypt
Decrypt/Verify
Import
Export
Certify
Lookup on Server
Certificates
Notepad
Smartcards
Search...<Alt+Q>
All Certificates
All Certificates
Imported Certificates
Imported Certificates
Name
Receiver
Sender
Valid Until
Key-ID
12-03-2027
7065 D1...
12-03-2027
3FBD 2FA...
You have imported a new certificate (public key) - Kleopatra
In order to mark the certificate as valid it needs to be certified.
Certifying means that you check the Fingerprint.
Some suggestions to do this are:
A phone call to the person.
Using a business card.
Confirming it on a trusted website.
Do you wish to start this process now?
Do not ask again
Certify
Cancel
Type here to search
ENG
07:29 PM
12-03-2024

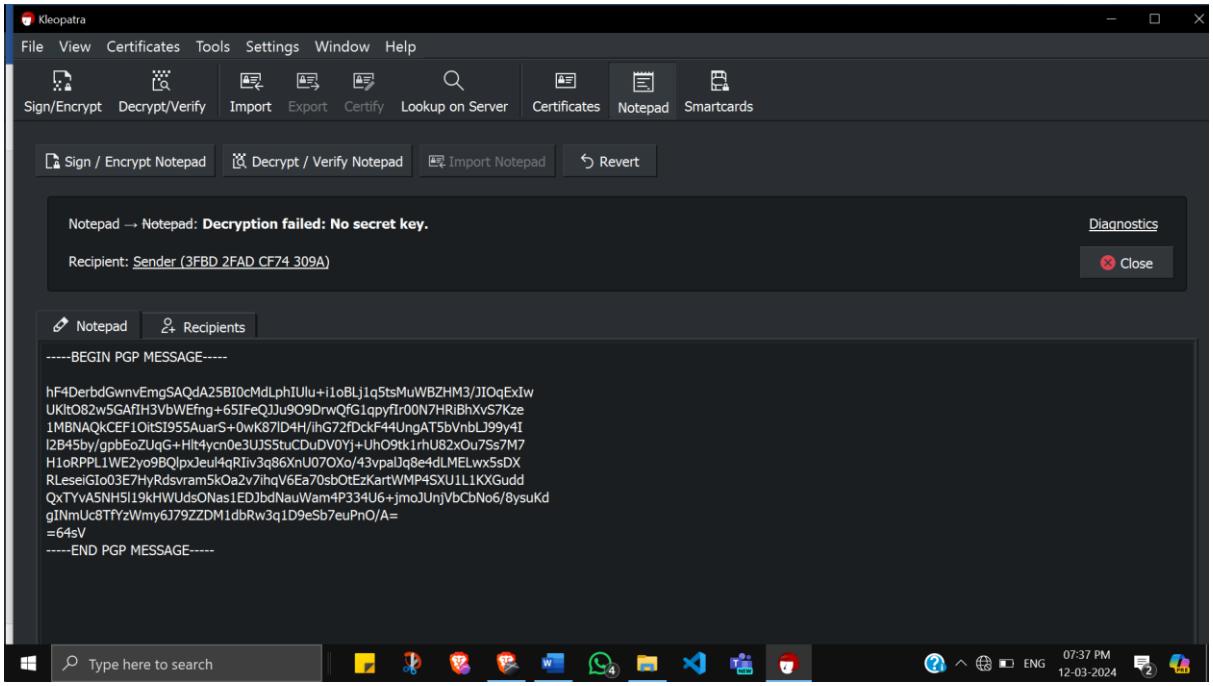


12. Enter the passphrase we had created earlier.

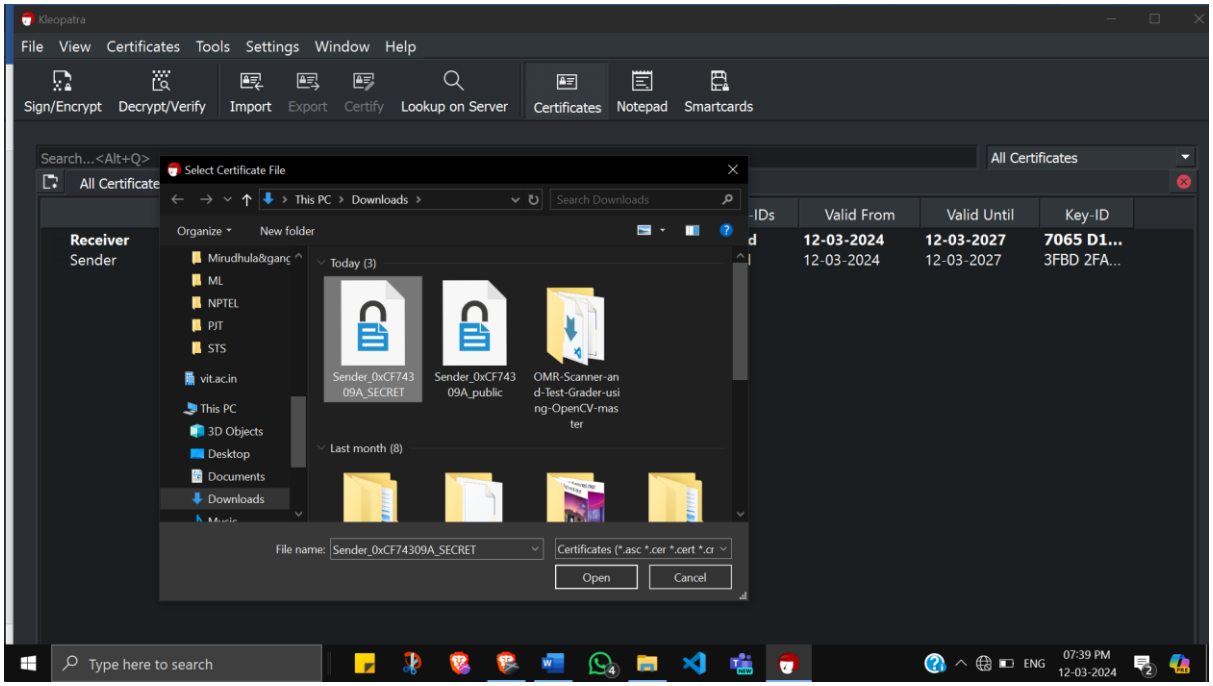


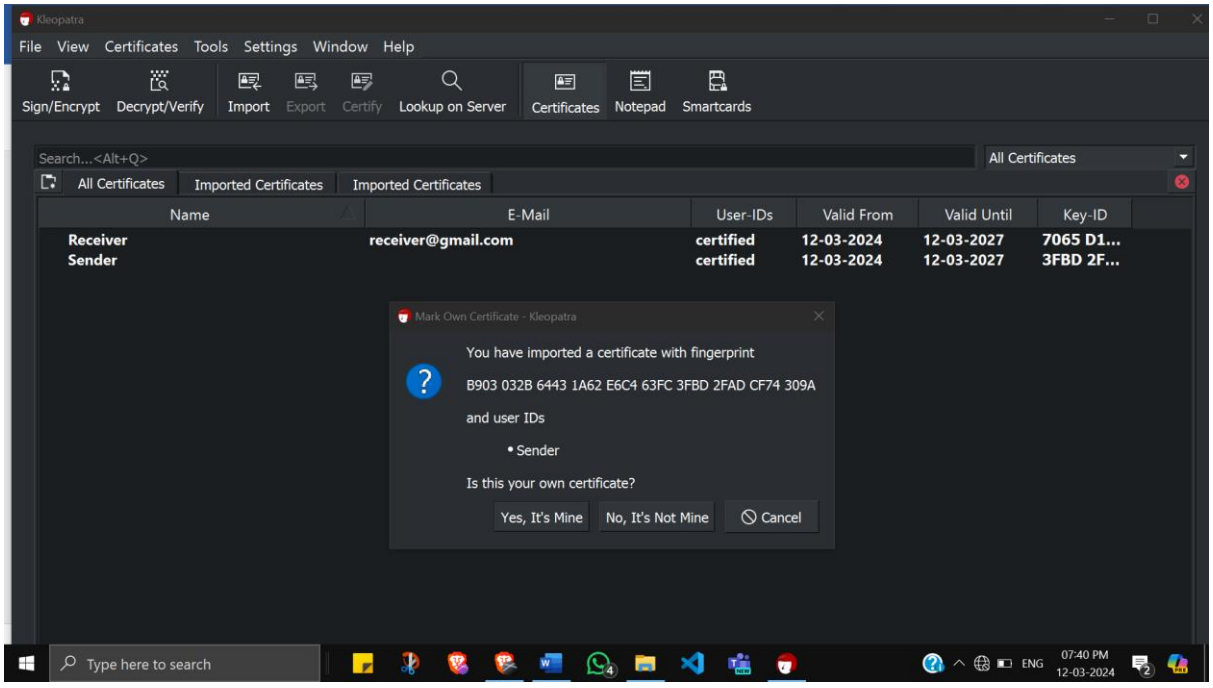


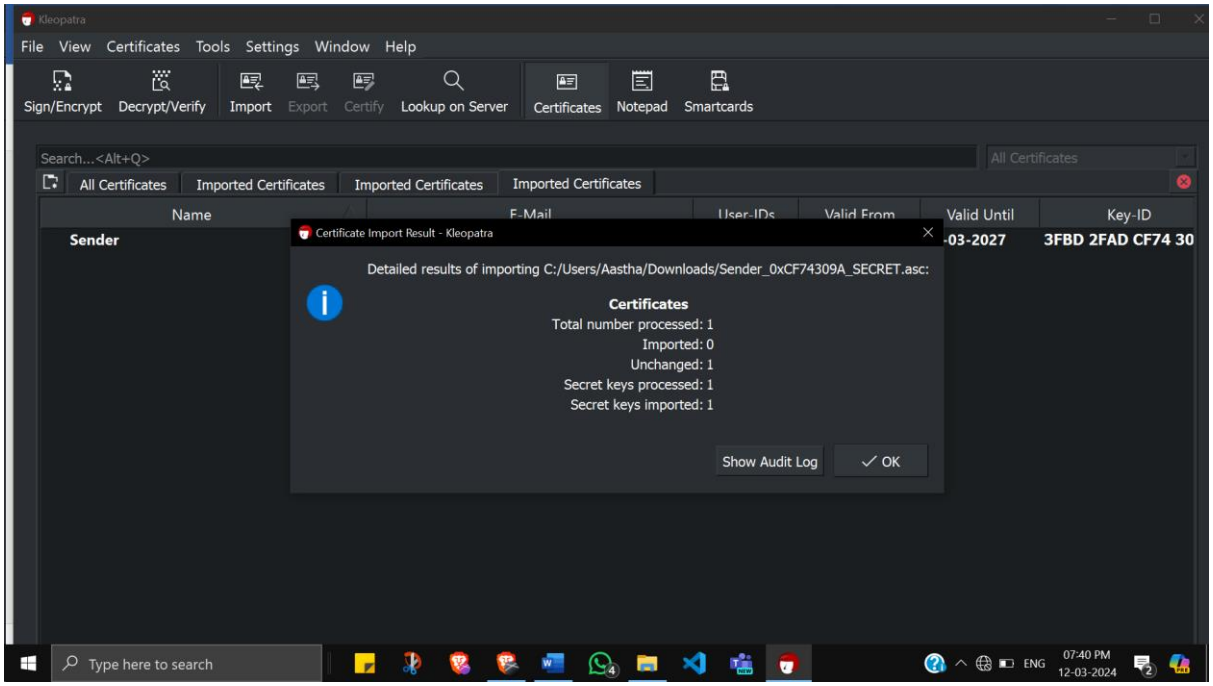
13. Now click on “Decrypt/Verify Notepad”. You will get an error.



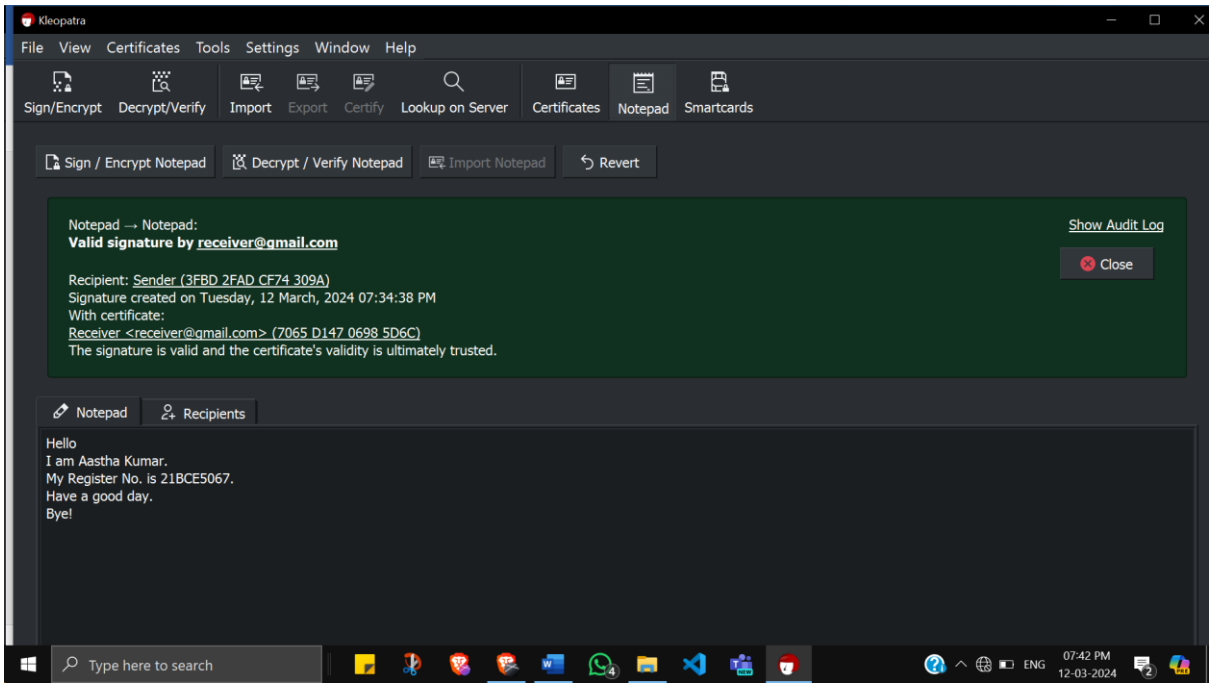
14. To tackle this error, go to Certificates and import the secret key.


Kleopatra
File View Certificates Tools Settings Window Help
Sign/Encrypt Decrypt/Verify Import Export Certify Lookup on Server Certificates Notepad Smartcards
Search...<Alt+Q>
All Certificates
Select Certificate File
This PC › Downloads ›
Search Downloads
Organize New folder
Mirudhula&gang
ML
NPTEL
PJT
STS
vit.ac.in
This PC
3D Objects
Desktop
Documents
Downloads
Today (3)
Sender_0xCF74309A_SECRET
Sender_0xCF74309A_public
OMR-Scanner-and-Test-Grader-using-OpenCV-master
Last month (8)
File name: Sender_0xCF74309A_SECRET
Certificates (*.asc *.cer *.cert *.cr
Open
Cancel
Receiver
Sender
Valid From
Valid Until
Key-ID
12-03-2024
12-03-2027
7065 D1...
12-03-2024
12-03-2027
3FBD 2FA...
Type here to search
ENG
07:39 PM
12-03-2024


Kleopatra
File View Certificates Tools Settings Window Help
Sign/Encrypt Decrypt/Verify Import Export Certify Lookup on Server Certificates Notepad Smartcards
Search...<Alt+Q>
All Certificates
All Certificates
Imported Certificates
Imported Certificates
Name
E-Mail
User-IDs
Valid From
Valid Until
Key-ID
Receiver
receiver@gmail.com
certified
12-03-2024
12-03-2027
7065 D1...
Sender
certified
12-03-2024
12-03-2027
3FBD 2F...
Mark Own Certificate - Kleopatra
You have imported a certificate with fingerprint
B903 032B 6443 1A62 E6C4 63FC 3FBD 2FAD CF74 309A
and user IDs
• Sender
Is this your own certificate?
Yes, It's Mine
No, It's Not Mine
Cancel
Type here to search
ENG
07:40 PM
12-03-2024

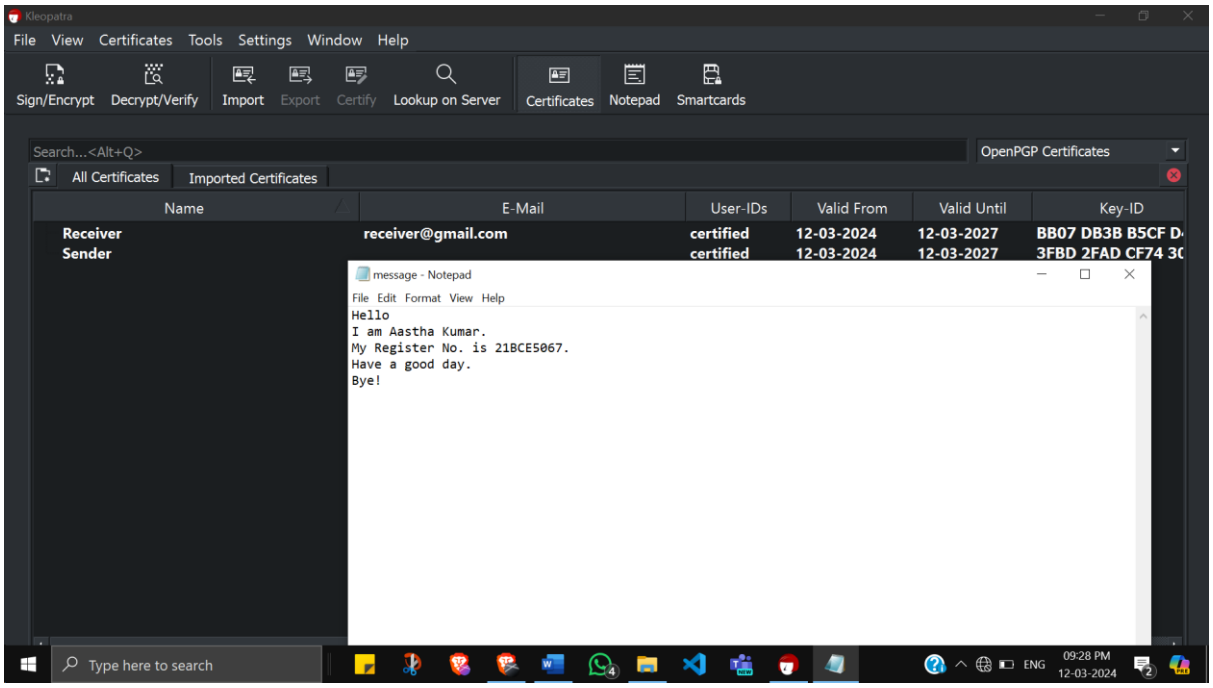


15. Now go to Notepad and click on “Decrypt/Verify Notepad”. The error is gone. The decrypted message can be viewed in the Notepad.

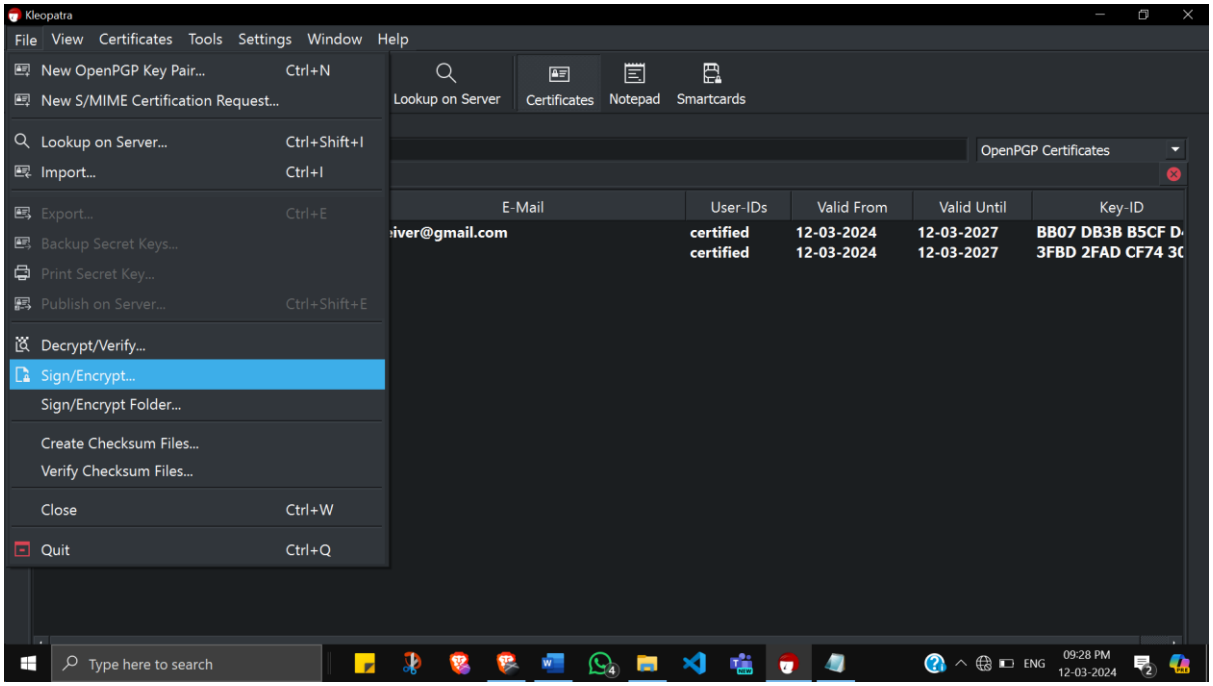


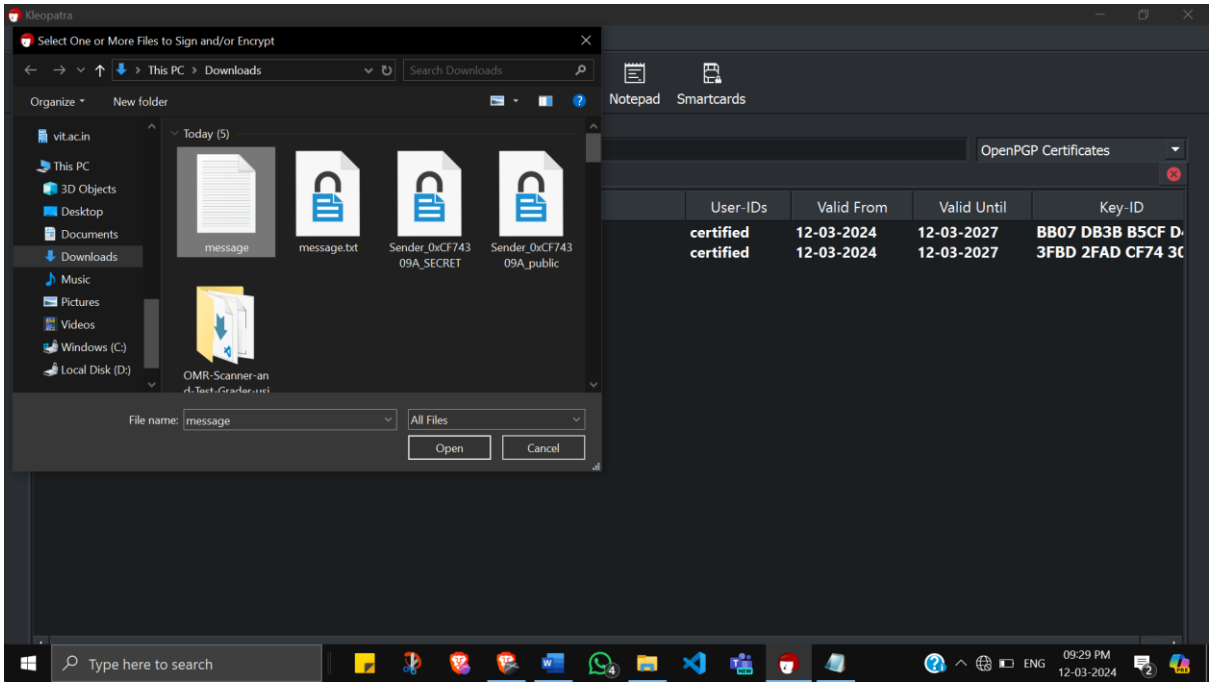
Task 2: Encrypt a message and sign it. Send it to your friend and verify the signature.

1. Open a text editor like Notepad and paste your message there. Save it as message.txt.

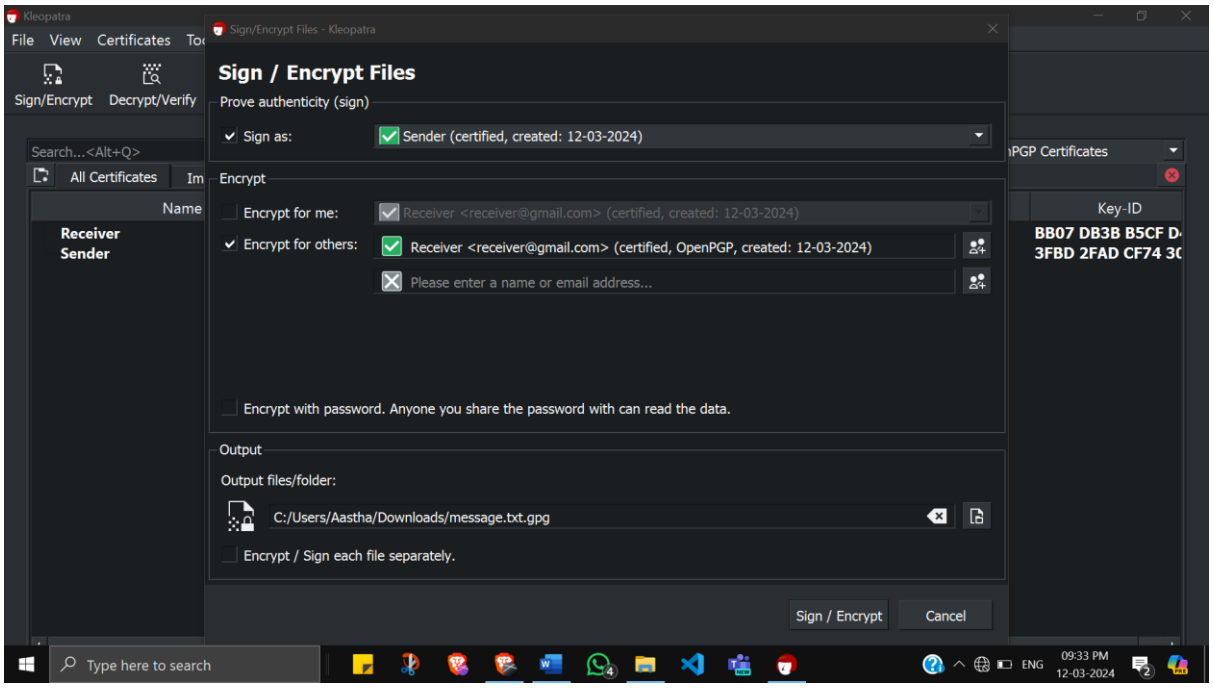


2. Go to File -> Sign/Encrypt. A pop up window will open from which you select message.txt.

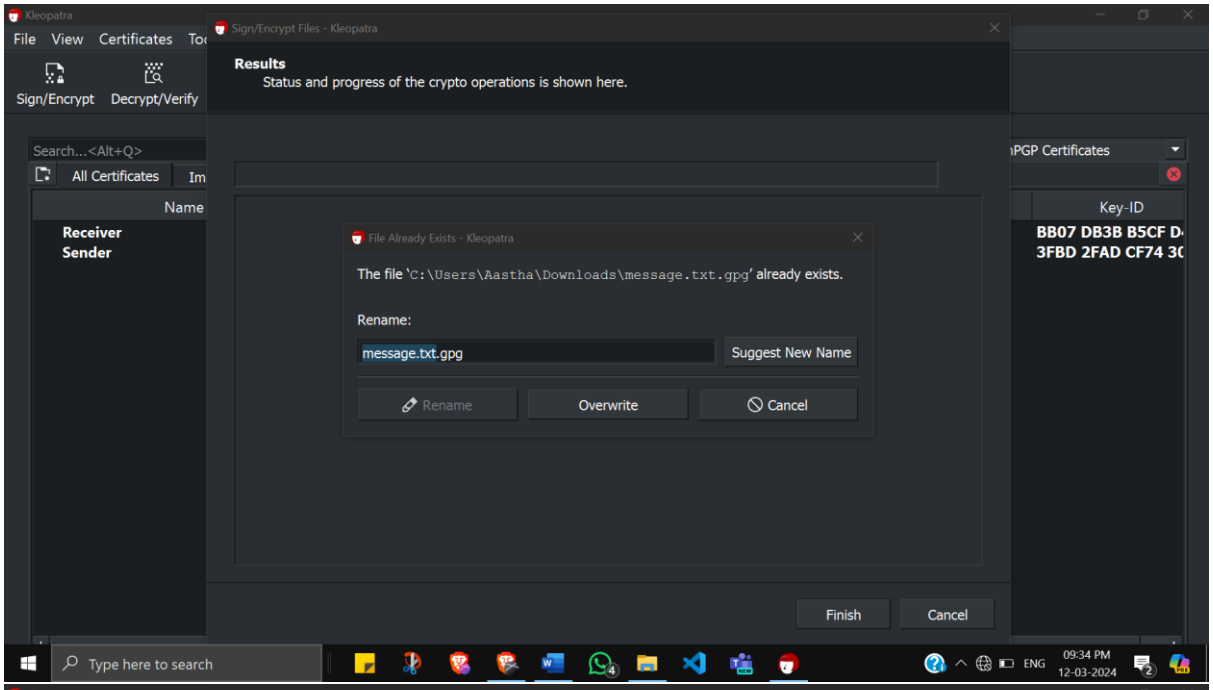


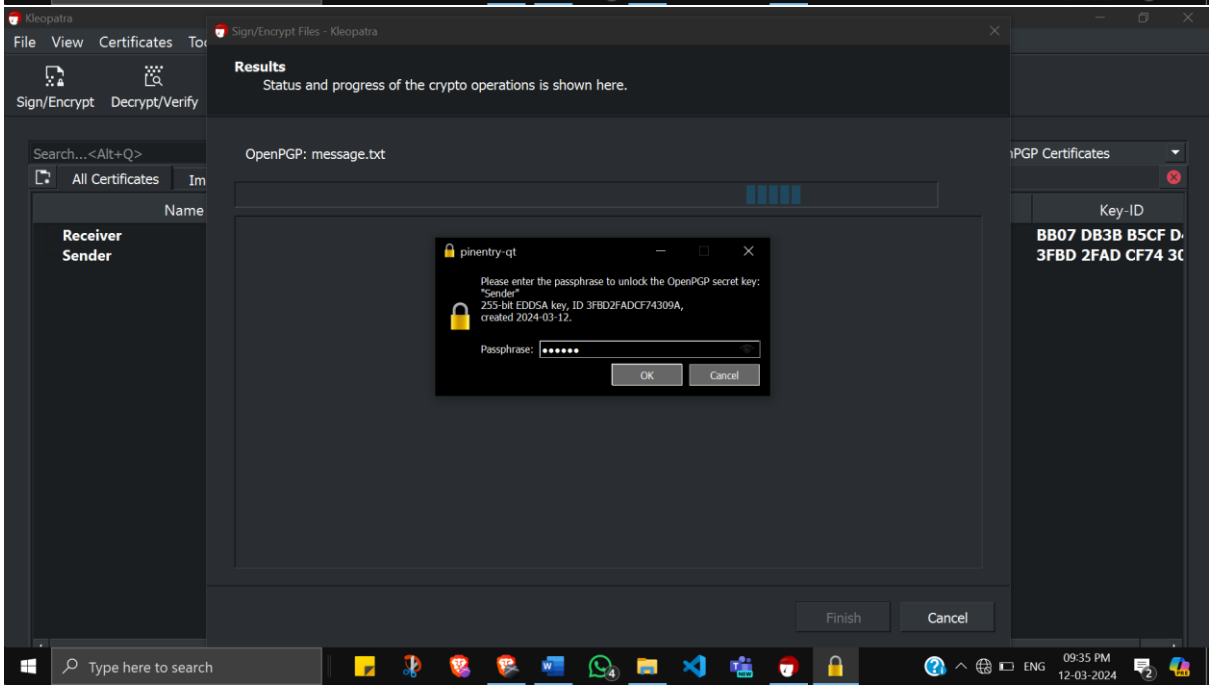


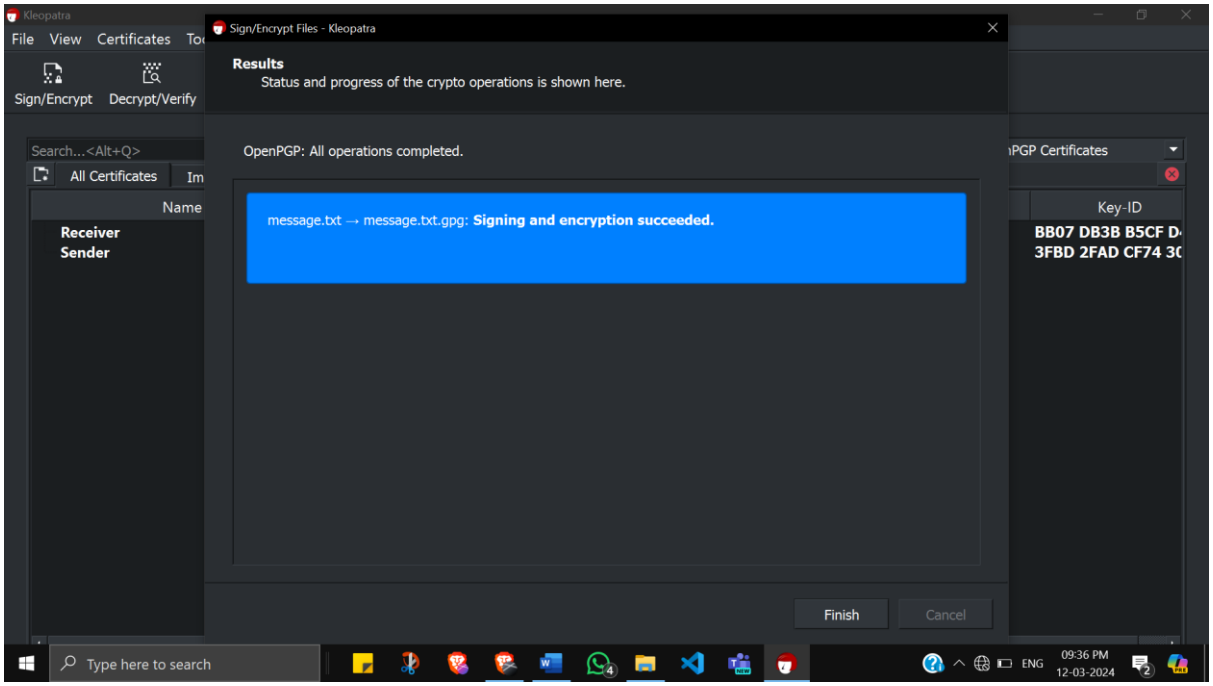
3. Select “Sender” beside “Sign as:” and “Receiver” beside “Encrypt for others:”. Click Sign/Encrypt.



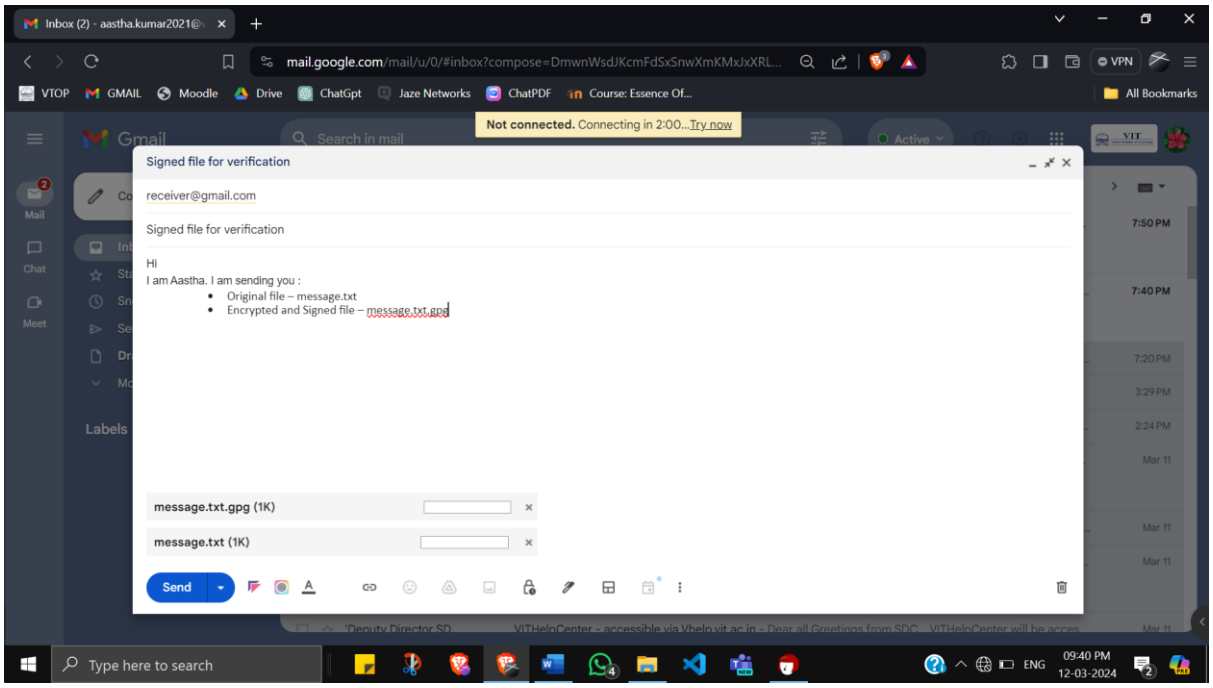
4. The new file will be encrypted, signed and saves as message.txt.gpg. Enter the Passphrase and click on “Finish”.


Kleopatra
File View Certificates
Sign/Encrypt
Decrypt/Verify
Sign/Encrypt Files - Kleopatra
Results
Status and progress of the crypto operations is shown here.
Search...<Alt+Q>
All Certificates
Name
Receiver
Sender
PGP Certificates
Key-ID
BB07 DB3B B5CF D
3FBD 2FAD CF74 3
File Already Exists - Kleopatra
The file 'C:\Users\Aastha\Downloads\message.txt.gpg' already exists.
Rename:
message.txt.gpg
Suggest New Name
Rename
Overwrite
Cancel
Finish
Cancel
Type here to search
ENG
09:34 PM
12-03-2024

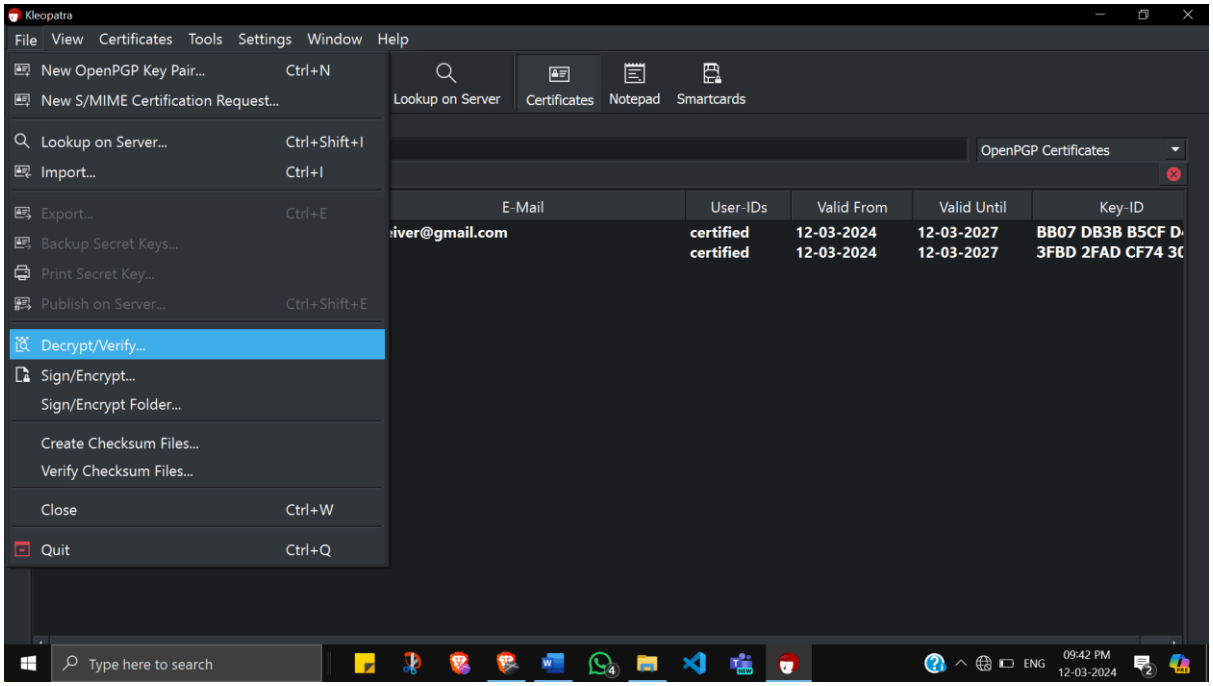

Kleopatra
File View Certificates
Sign/Encrypt
Decrypt/Verify
Sign/Encrypt Files - Kleopatra
Results
Status and progress of the crypto operations is shown here.
OpenPGP: message.txt
Search...<Alt+Q>
All Certificates
Name
Receiver
Sender
PGP Certificates
Key-ID
BB07 DB3B B5CF D
3FBD 2FAD CF74 3
pinentry-qt
Please enter the passphrase to unlock the OpenPGP secret key:
"Sender"
255-bit EDDSA key, ID 3FBD2FADCF74309A,
created 2024-03-12.
Passphrase:
OK
Cancel
Finish
Cancel
Type here to search
ENG
09:35 PM
12-03-2024



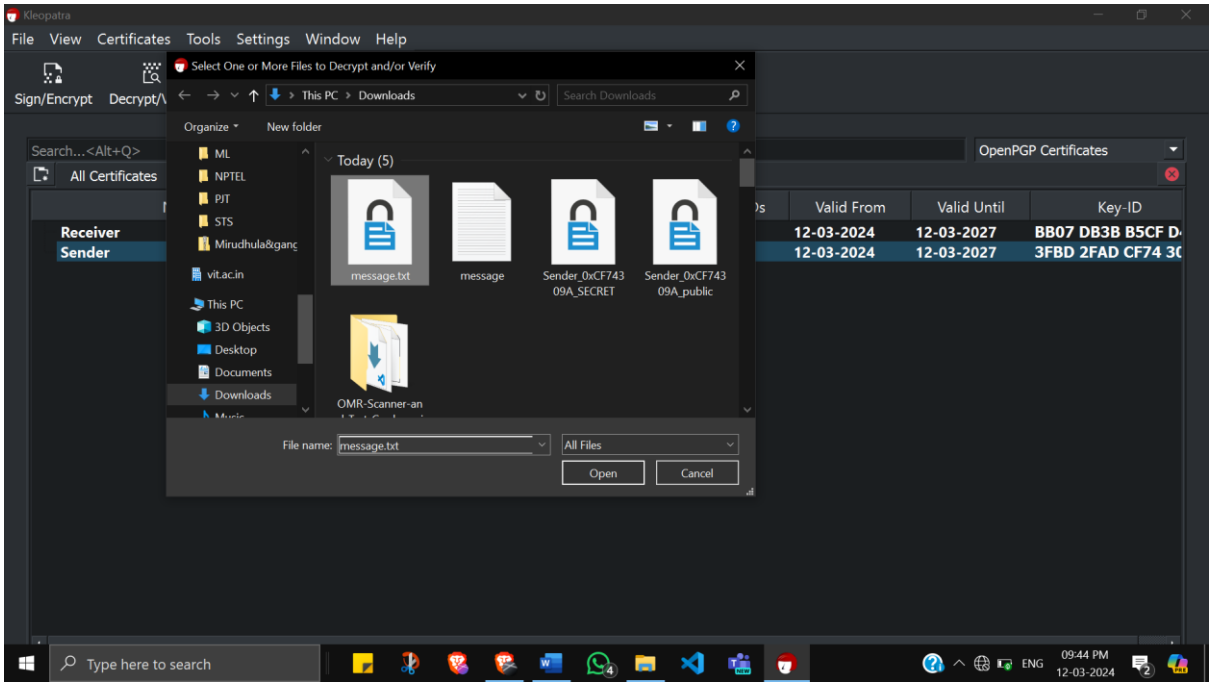
5. Now compose an email to the receiver and attach both files:-
    - Original file – message.txt
    - Encrypted and Signed file – message.txt.gpg

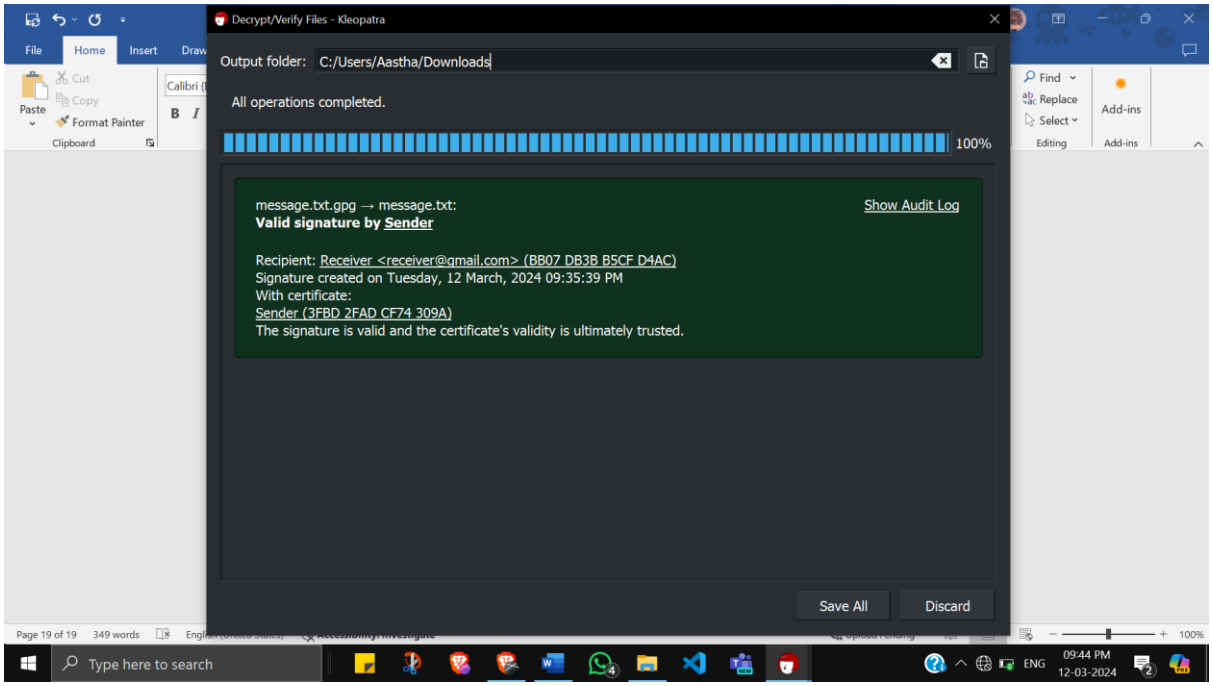


6. Now the recipient needs to verify the signature, for that they need to go to File -> Decrypt/Verify.



7. A pop up window will open from which you select message.txt.gpg. and click on “Open”





Observation:

- Encryption with Kleopatra: Encrypting a message using Kleopatra is straightforward. You can select the recipient's public key, enter the message, and encrypt it with just a few clicks. Kleopatra provides options for selecting encryption algorithms and adding additional security features such as digital signatures.
- Decryption with Kleopatra: Decrypting an encrypted message received from a sender is also simple using Kleopatra. You can import your private key and decrypt the message with ease. Kleopatra ensures secure decryption by verifying the authenticity of the sender's digital signature.
- Digital Signatures: Adding digital signatures to messages provides assurance of the sender's identity and the integrity of the message content. Verifying digital signatures using Kleopatra ensures that the message has not been tampered with during transmission.

Results:

1. Encryption and Decryption:

The encrypted message sent using Kleopatra was successfully decrypted by the recipient using their private key. This demonstrates the effectiveness of encryption in protecting the confidentiality of communication.

2. Digital Signatures:

The message signed with a digital signature using Kleopatra was successfully verified by the recipient using the sender's public key. The verification process ensures that the message originated from the claimed sender and has not been altered in transit.

EXPERIMENT 7

Aim:

Demonstrate the implementation of network intrusion detection using Snort

Experiment Objective:

1. Simulate a network attack from one machine to another within a controlled environment.

2. Configure Snort on the target machine to detect specific network activities.

3. Create a custom Snort rule to trigger an alert when outbound TCP traffic to a specific domain is detected.

4. Verify the effectiveness of the custom Snort rule by browsing to a designated website from the machine where Snort is running and observing the alert generated by Snort.

Introduction :

Network intrusion detection systems (NIDS) are crucial components of network security infrastructure, responsible for identifying and mitigating potential threats within a network. Snort is a widely used open-source NIDS capable of real-time traffic analysis and packet logging. In this experiment, we aim to demonstrate the practical application of Snort in detecting suspicious network activity.

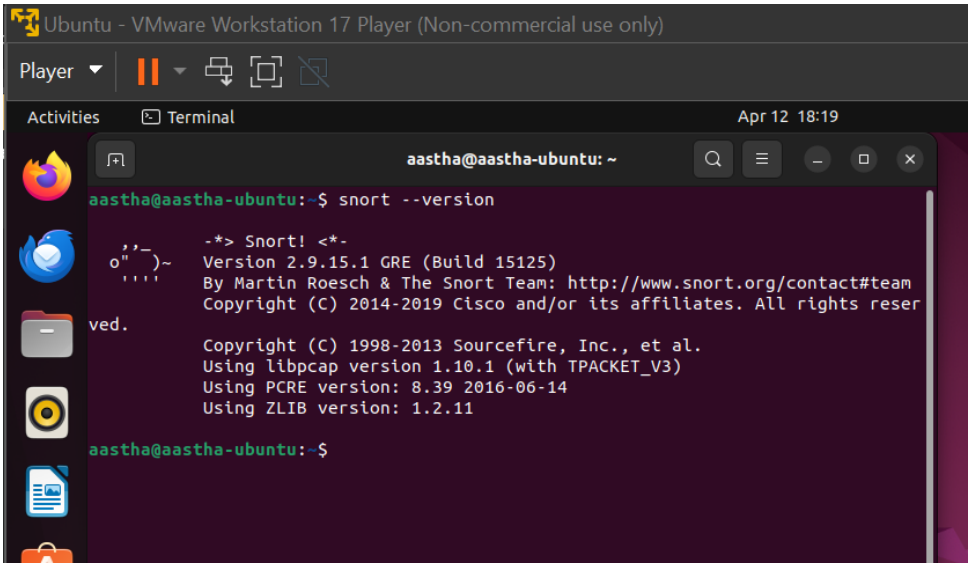
Setup:

- Operating Systems: Ubuntu and Kali Linux
- Snort: Install Snort on both the Ubuntu and Kali Linux machines. Snort can be installed using package managers or compiled from source. Ensure Snort is properly configured to monitor network traffic.
- Target Machine: Set up a target machine to simulate the attack. This machine will be the subject of the simulated attack and will run the services or applications to be exploited.
- Network Configuration: Ensure that the Ubuntu machine running Snort and the target machine are connected to the same network. This allows Snort to monitor network traffic between the two machines effectively.
- Custom Snort Rule: Create a custom Snort rule to detect outbound TCP traffic to the designated domain (craigslist.org). This rule should generate an alert when such traffic is detected.
- Testing: Perform testing by browsing to vit.ac.in from the machine where Snort is running. Monitor Snort alerts to verify that the custom rule correctly detects outbound TCP traffic to craigslist.org.

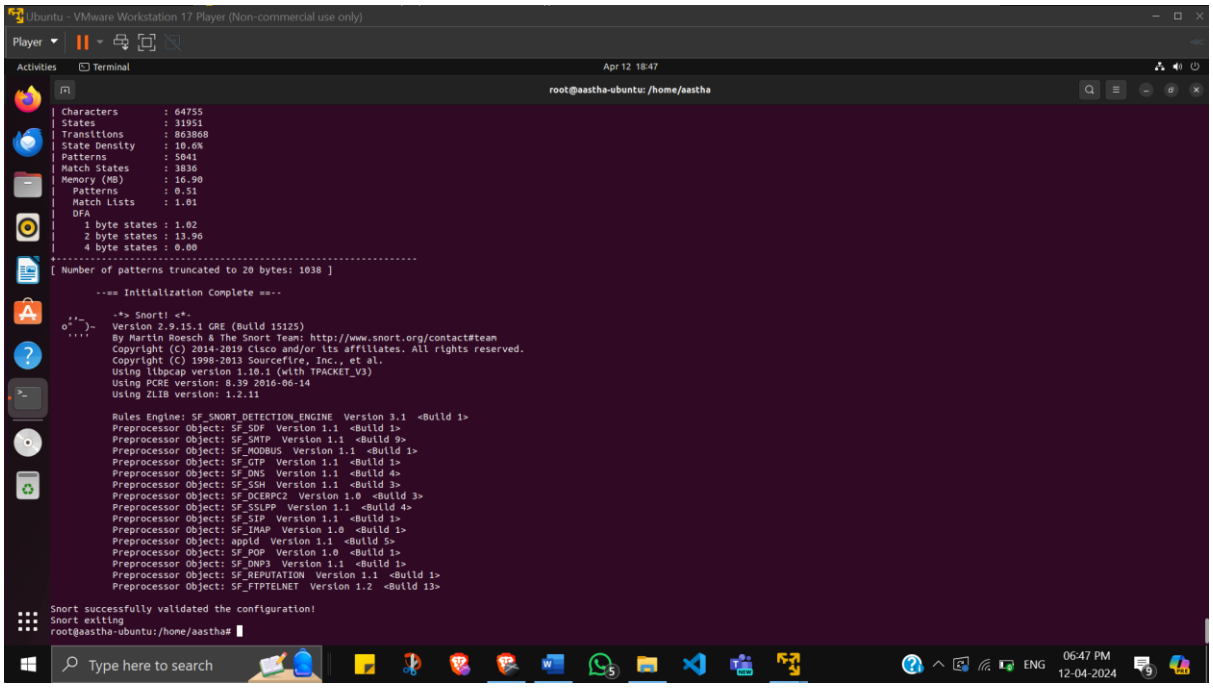
Experiment Steps:

Task 1: simulate attack on another machine.

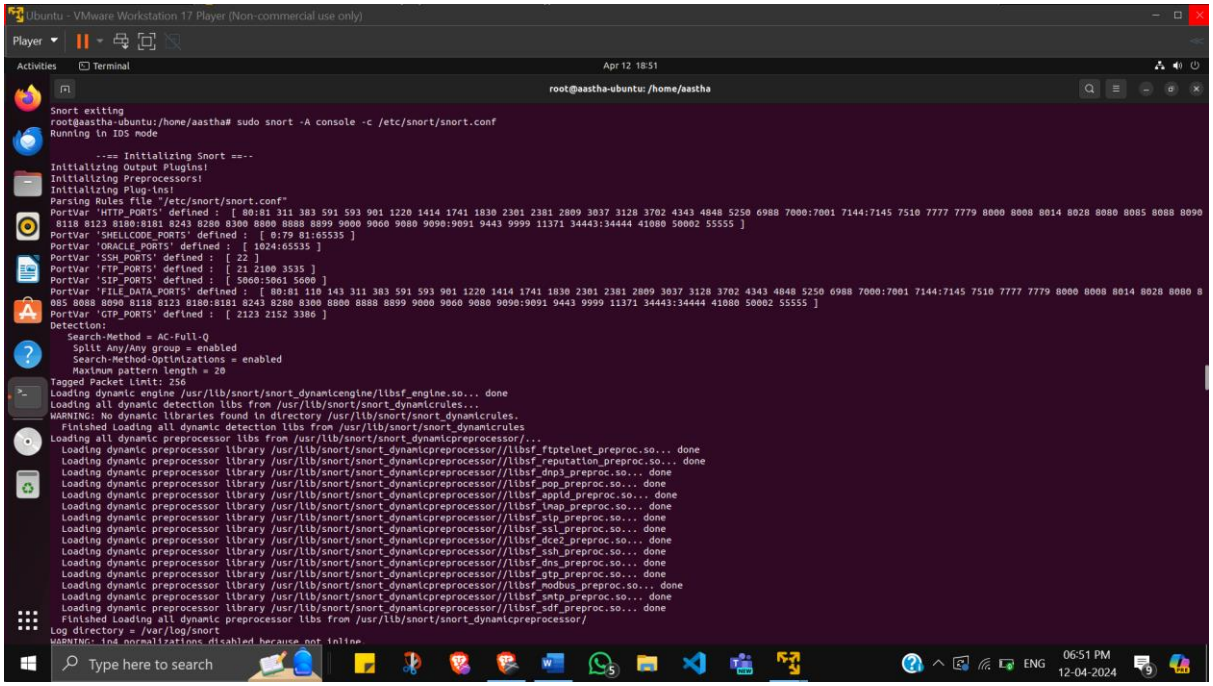
1. Ensure Snort is already downloaded and configured on the Ubuntu machine. It can be checked typing **snort –version**

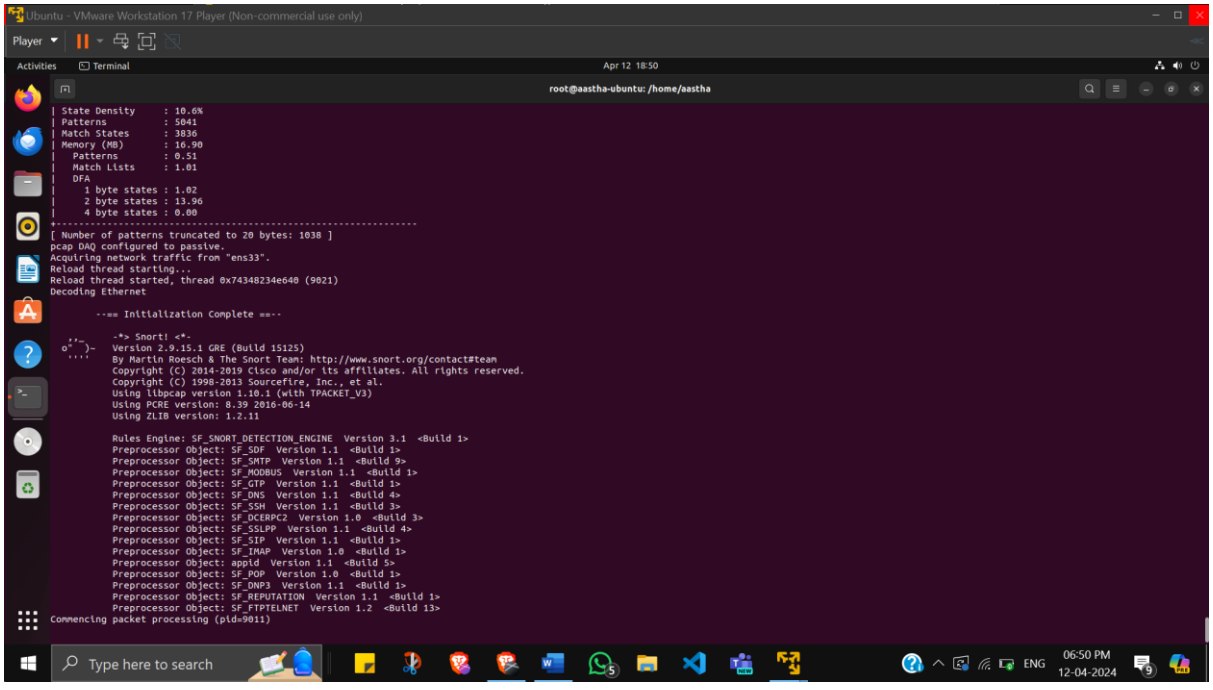


2. Switch the root mode by typing in **sudo su** followed by your password. Then type in the following command to test if snort has properly been configured **sudo snort -T -c /etc/snort/snort.conf**

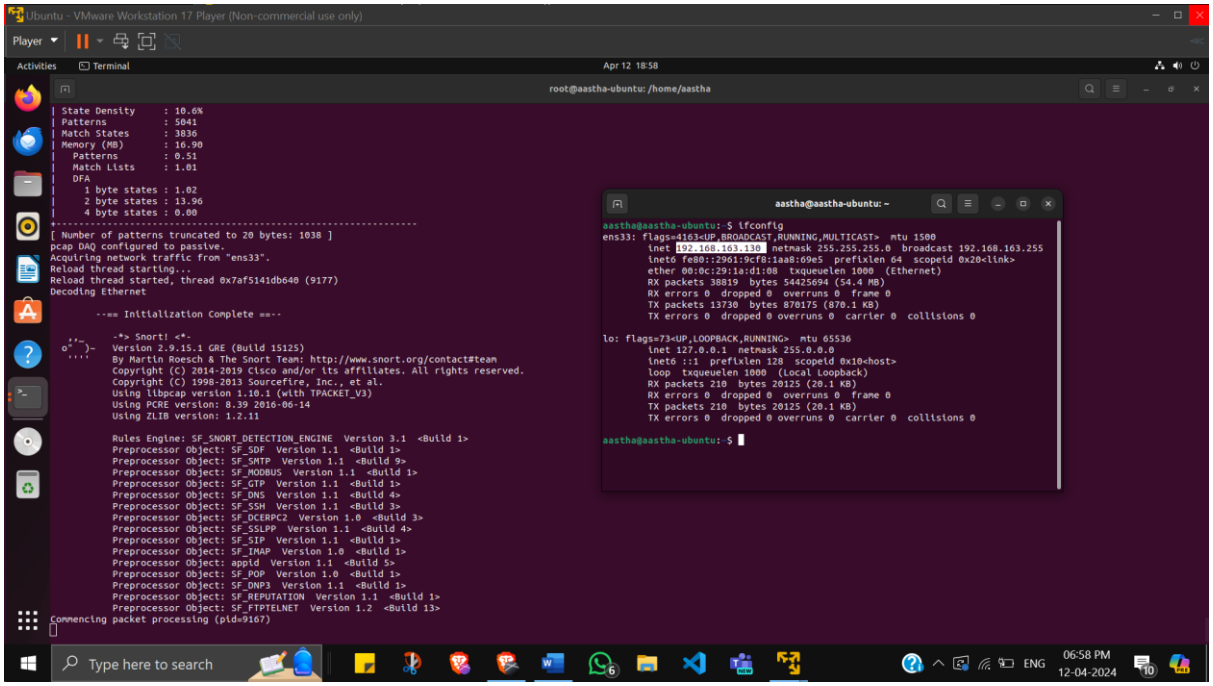


3. To start attack using Snort, type the following **sudo snort -A console -c /etc/snort/snort.conf**

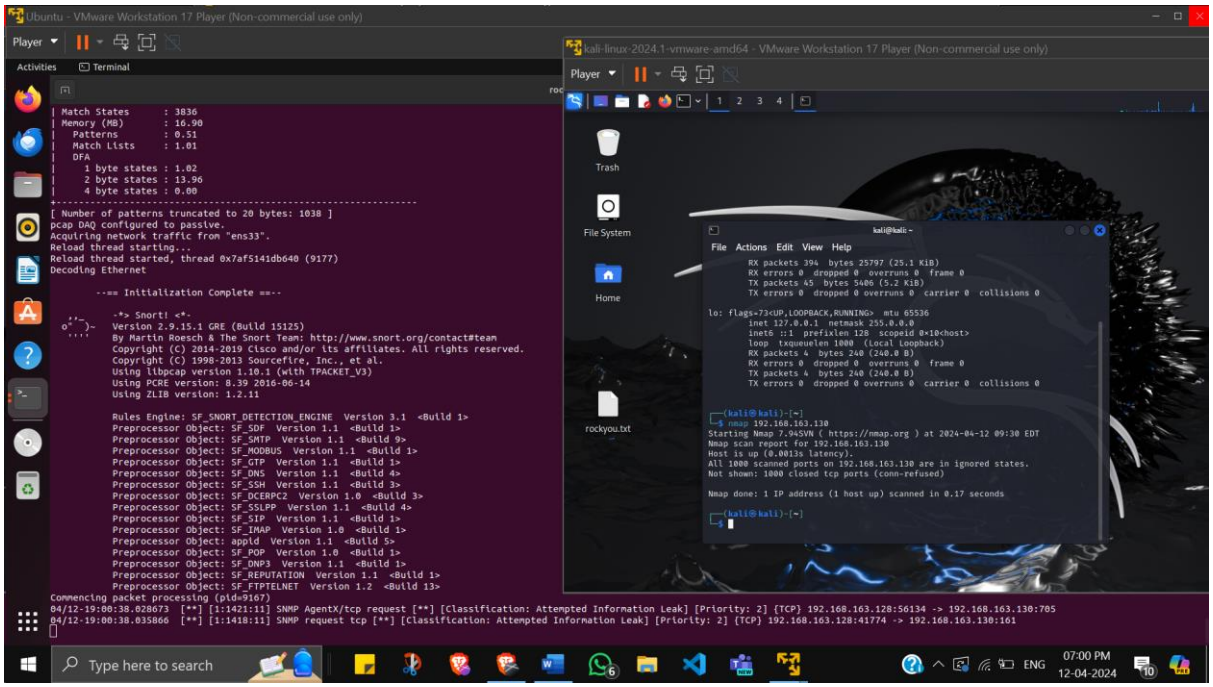




4. Snort is now listening to packets. Let us assume the Kali machine is trying to Now we need to find the ip address of the Ubunutu machine using **ifconfig** command



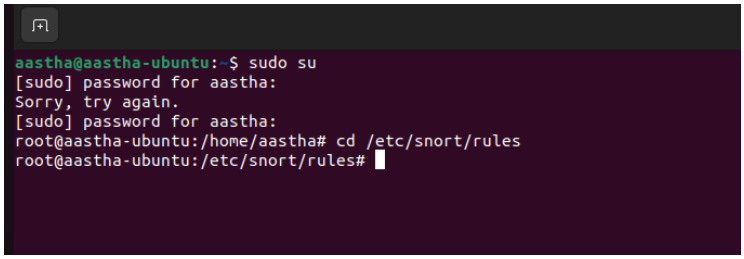
5. Now the Kali machine will try to gain information on open ports on our Ubuntu machine using **nmap <ip address of Ubuntu> i.e nmap 192.168.163.130**



Snort picks up “attempted information leak” packets from Kali machine.

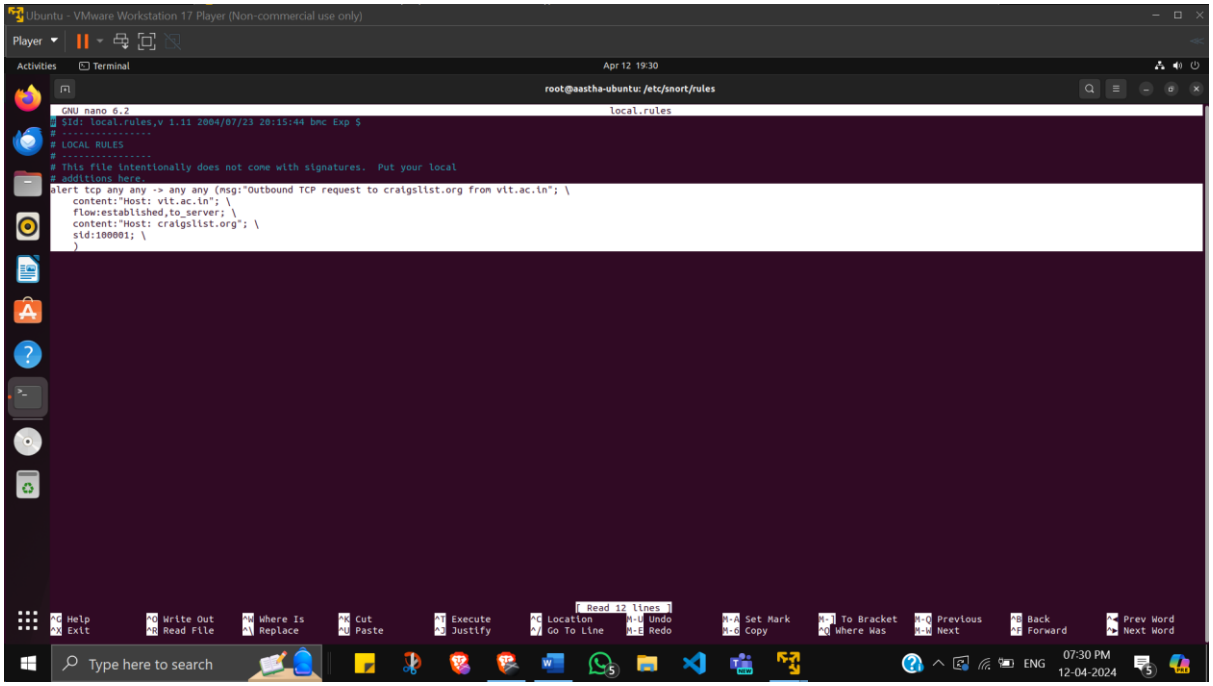
Task 2: create a rule in snort such that when you browse to vit.ac.in from the machine snort is running on, it should look for any outbound tcp request to craigslist.org and alert on it.

1. You need to create a rule to perform the above operation and then add it to the local rules directory under Snort rules. For this you need to navigate to the Snort Rules directory first by typing **cd /etc/snort/rules**



Then open the local rules file in edit mode by typing **nano local.rules.** Type in the following rule:

alert tcp any any -> any any (msg:"Outbound TCP request to craigslist.org from vit.ac.in"; \content:"Host: vit.ac.in"; \flow:established,to_server; \content:"Host: craigslist.org"; \sid:100001; \)



2. Save the updated local.rules file by hitting ctrl+X, then Y, then Enter. The rule gets created.

Observations:

1. Simulated Attack: The simulated attack on the target machine was successful, as evidenced by the generated network traffic.
2. Snort Alerts: Snort successfully detected the outbound TCP traffic to craigslist.org, triggered by the custom rule created for this purpose.
3. Real-time Monitoring: Snort provided real-time monitoring of network traffic, allowing for immediate detection and response to suspicious activities.
4. Rule Configuration: The custom Snort rule effectively identified the specific network activity targeted in the experiment, demonstrating the flexibility and customization capabilities of Snort.

Results:

1. Detection of Outbound Traffic: Snort generated alerts for the outbound TCP traffic to craigslist.org as per the custom rule configuration.
2. Alerts Generated: The Snort alerts provided detailed information about the detected activity, including the source IP address, destination IP address, timestamp, and rule triggered.
3. Response to Threats: The ability of Snort to detect and alert on suspicious network activities enables network administrators to respond promptly to potential threats, enhancing overall network security.
4. Effectiveness of Custom Rule: The custom Snort rule accurately identified the specified outbound TCP traffic pattern, demonstrating its effectiveness in detecting specific network behaviors.
5. Validation of Security Measures: The successful detection of the simulated attack and the generation of alerts validate the effectiveness of Snort as a network intrusion detection system and highlight the importance of implementing robust security measures to safeguard network assets.