Module 20: Cryptography

Scenario

With the increasing adoption of the Internet for business and personal communication, securing sensitive information such as credit-card and personal identification numbers (PINs), bank account numbers, and private messages is becoming increasingly important, and yet, more difficult to achieve. Today's information-based organizations extensively use the Internet for e-commerce, market research, customer support, and a variety of other activities. Thus, data security is critical to online businesses and privacy of communication.

Cryptography and cryptographic ("crypto") systems help in securing data from interception and compromise during online transmissions. Cryptography enables one to secure transactions, communications, and other processes performed in the electronic world, and is additionally used to protect confidential data such as email messages, chat sessions, web transactions, personal data, corporate data, e-commerce applications, etc.

As an ethical hacker or penetration tester, you should suggest to your client proper encryption techniques to protect data, both in storage and during transmission. The labs in this module demonstrate the use of encryption to protect information systems in organizations.

Objective

The objective of the lab is to use encryption to conceal data and perform other tasks that include, but is not limited to:

- Generate hashes and checksum files
- Calculate the encrypted value of the selected file
- Use encrypting/decrypting techniques
- Perform file and data encryption
- Create self-signed certificates
- Perform disk encryption

Overview of Cryptography

"Cryptography" comes from the Greek words kryptos, meaning "concealed, hidden, veiled, secret, or mysterious," and graphia, "writing"; thus, cryptography is "the art of secret writing."

Cryptography is the practice of concealing information by converting plain text (readable format) into cipher text (unreadable format) using a key or encryption scheme: it is the process of the conversion of data into a scrambled code that is sent across a private or public network.

There are two types of cryptography, determined by the number of keys employed for encryption and decryption:

- **Symmetric Encryption**: Symmetric encryption (secret-key, shared-key, and private-key) uses the same key for encryption as it does for decryption
- **Asymmetric Encryption**: Asymmetric encryption (public-key) uses different encryption keys for encryption and decryption; these keys are known as public and private keys

Lab Tasks

Ethical hackers or pen testers use numerous tools and techniques to perform cryptography to protect confidential data. Recommended labs that will assist you in learning various cryptography techniques include:

1. Encrypt the information using various cryptography tools
    - Perform multi-layer hashing using CyberChef
    - Perform file and text message encryption using CryptoForge
2. Create a self-signed certificate
    - Create and use self-signed certificates
3. Perform disk encryption
    - Perform disk encryption using VeraCrypt
4. Perform cryptography using AI
    - Perform cryptographic techniques using ShellGPT

Lab 1: Encrypt the Information using Various Cryptography Tools

Lab Scenario

As a professional ethical hacker and penetration tester, you should use various cryptography techniques or tools to protect confidential data against unauthorized access. Cryptography protects confidential data such as email messages, chat sessions, web transactions, personal data, corporate data, e-commerce applications, and many other kinds of communication. Encrypted messages can at times be decrypted by cryptanalysis (code breaking), although modern encryption techniques are virtually unbreakable.

The labs in this exercise demonstrate how you can use various cryptography tools to encrypt important information in the system.

Lab Objectives

- Perform multi-layer hashing using CyberChef
- Perform file and text message encryption using CryptoForge

Overview of Cryptography Tools

System administrators use cryptography tools to encrypt system data within their network to prevent attackers from modifying the data or misusing it in other ways. Cryptography tools can also be used to calculate or decrypt hash functions available in MD4, MD5, SHA-1, SHA-256, etc.

Cryptography tools are used to convert the information present in plain text (readable format) into cipher text (unreadable format) using a key or encryption scheme. The converted data are in the form of a scrambled code that is encrypted and sent across a private or public network.

Task 1: Perform Multi-layer Hashing using CyberChef

CyberChef enables a wide array of "cyber" tasks directly in browser. It offers a wide range of operations and transformations, from basic text manipulation to complex cryptographic functions which include various hashing techniques such as MD5, SHA-1, SHA-256, SHA-512, etc., and encoding techniques such as text to hexadecimal, binary, Base64, or URL encoding.

A multi-layer hash typically refers to a hierarchical or nested structure of hash functions applied successively to data. Instead of just applying a single hash function to a piece of data, multiple hash functions are employed in layers or stages, with the output of one hash function serving as the input to the next one.

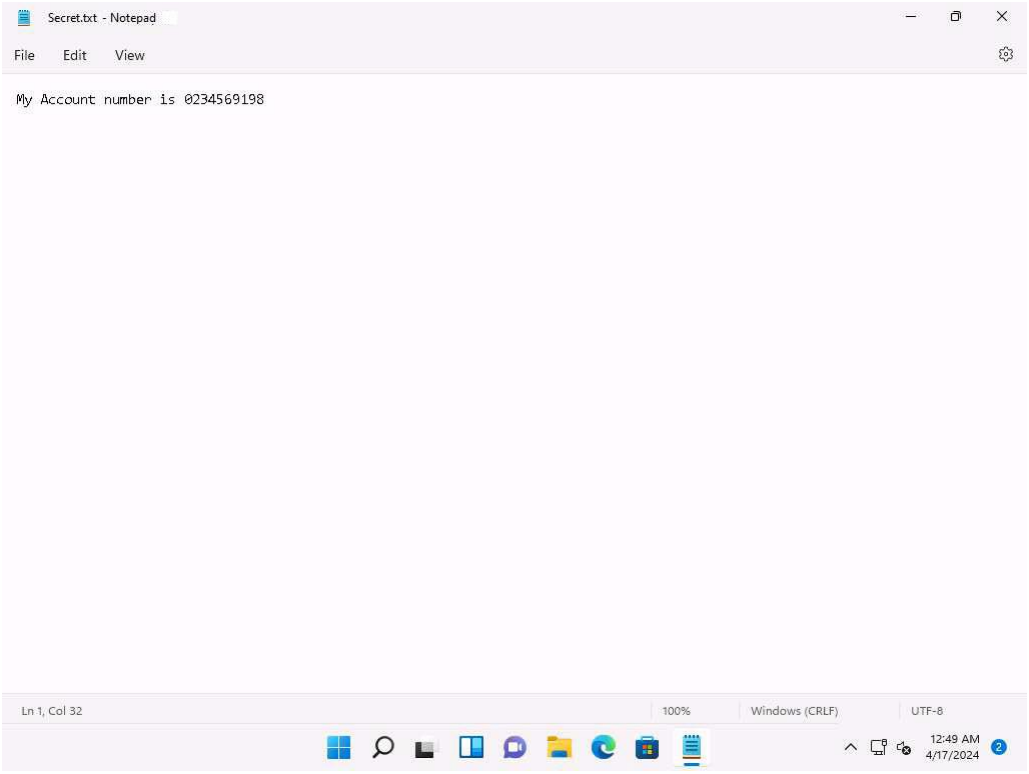
Here, we will perform multi-layer hashing using CyberChef.

1. Click Windows 11 to switch to the **Windows 11** machine.
   Click Ctrl+Alt+Delete to activate it and login with **Admin**/**Pa$$w0rd**.

   Alternatively, you can also click **Pa$$w0rd** under **Windows 11** machine thumbnail in the **Resources** pane.

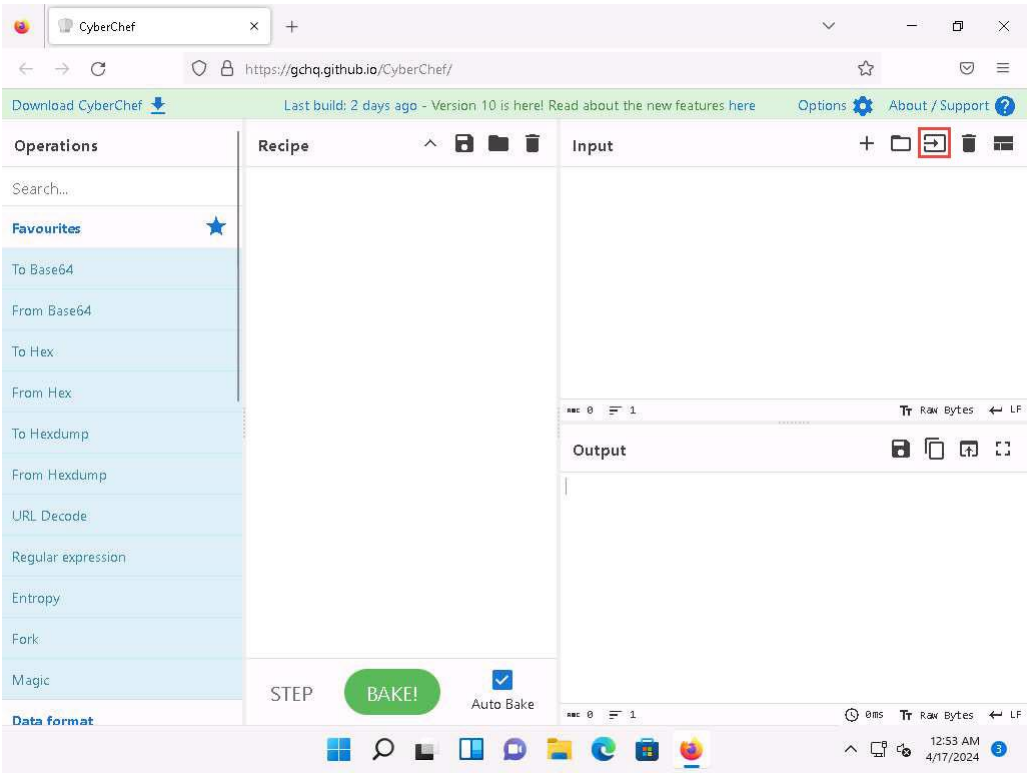
   Networks screen appears, click **Yes** to allow your PC to be discoverable by other PCs and devices on the network.

2. Navigate to **Desktop**, right-click on the **Desktop** window, and navigate to **New --> Text Document** to create a new text file. A newly created text file appears; rename it to **Secret.txt** and open it. Write some text in it (here, **My Account number is 0234569198**) and press **Ctrl+S** to save the file. Close the text file.



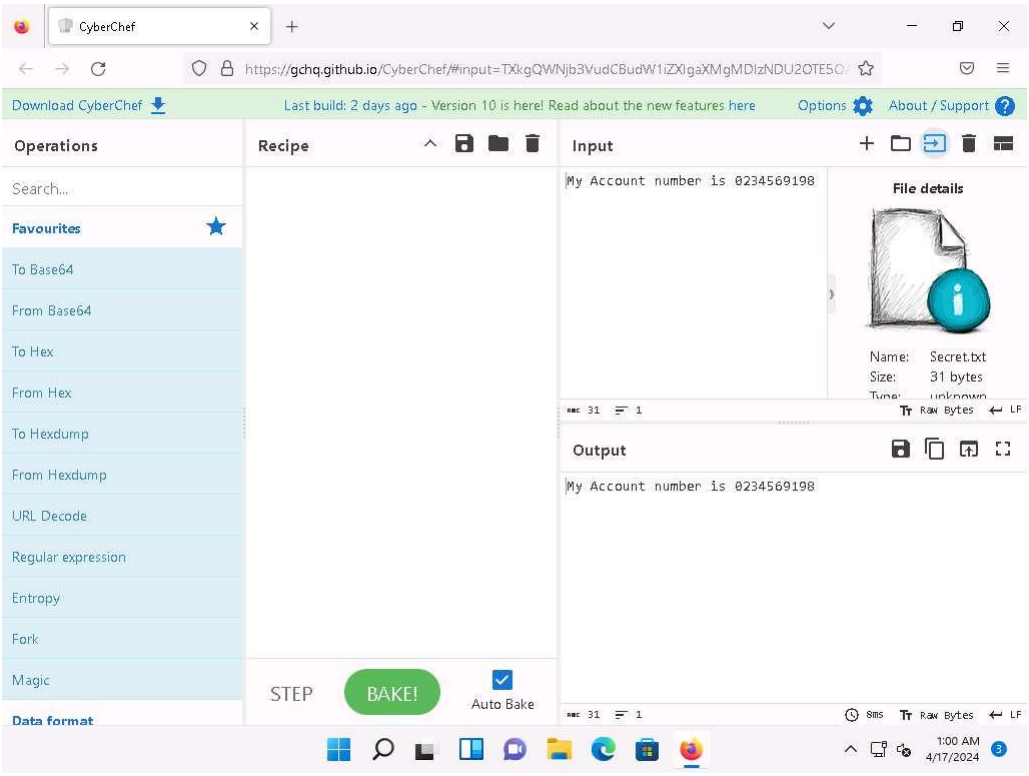
3. Launch any web browser, and go to **https://gchq.github.io/CyberChef/** (here, we are using **Mozilla Firefox**).

4. CyberChef website appears, click on **Open file as input** button present at the top of the **Input** section.



5. **File Upload** window appears, select **Secret.txt** file from **Desktop** and click **Open**.

6. The contents of the **Secret.txt** file will be displayed in the **Input** window.

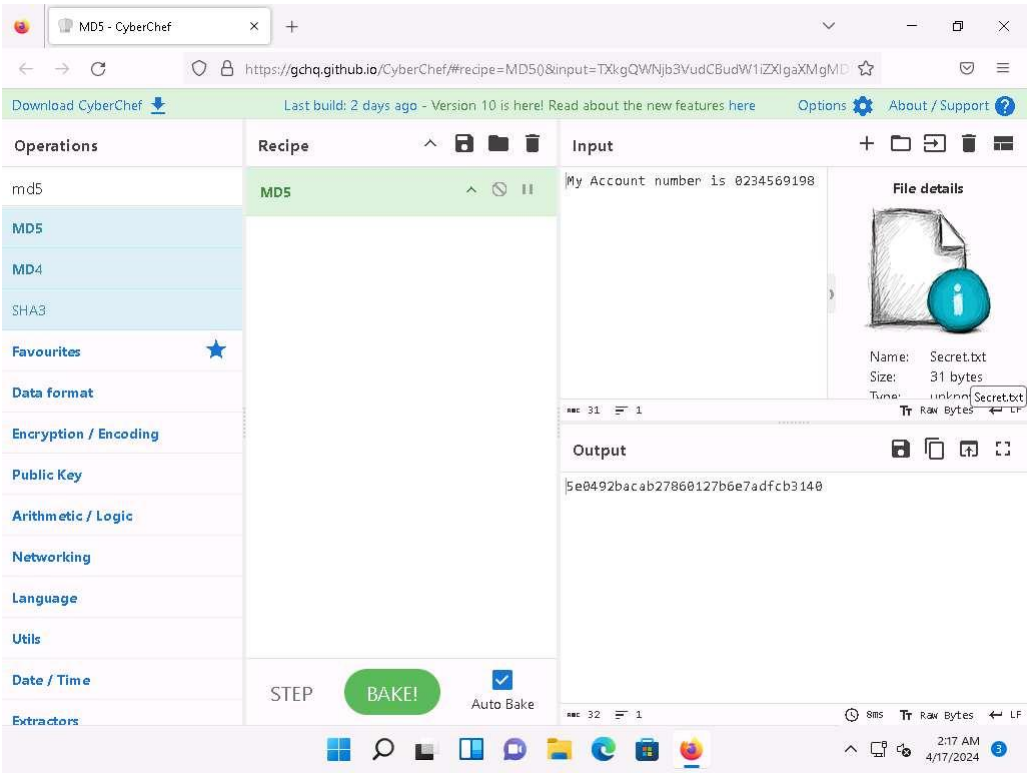


7. Now, we will calculate MD5 hash of the **Secret.txt** file, to do so, in the search field present under **Operations** section, type md5 and drag **MD5** from the results in the **Operations** section in to the **Recipe** section.

   Alternatively, you can expand **Hashing** node in the **Operations** section and select **MD5** algorithm.

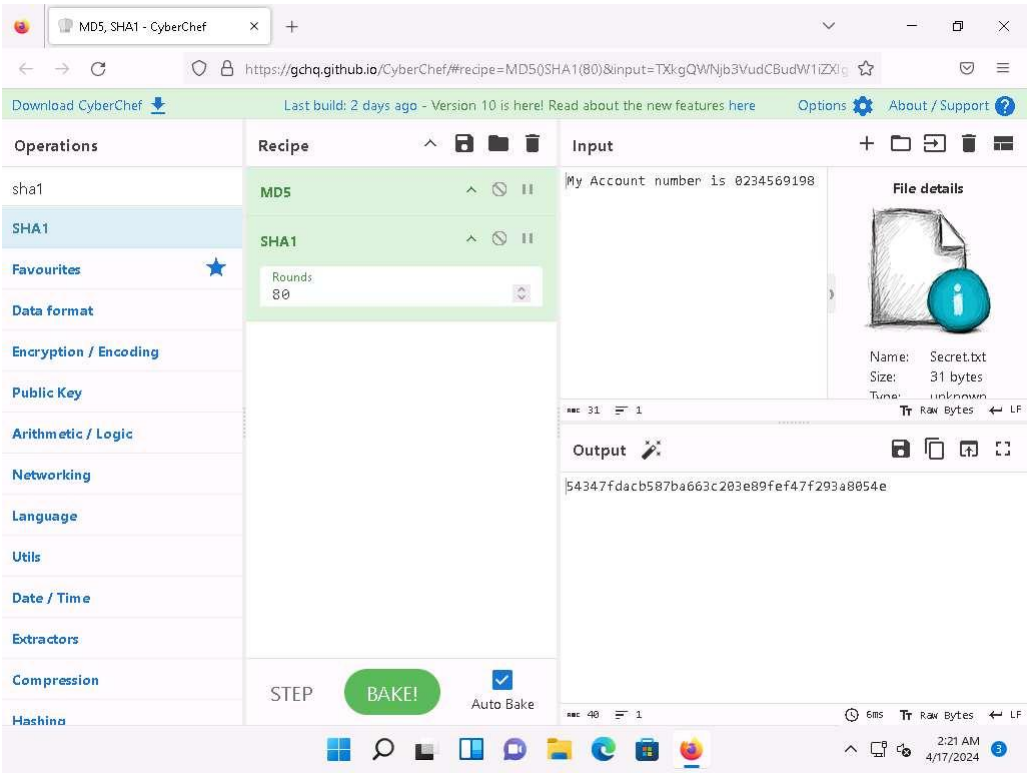
8. The tool calculates the **MD5** hash of the given input file and displays the output in the **Output** section.

   Ensure that **Auto Bake** option is checked. If "Auto Bake" is enabled, CyberChef will promptly "bake" and generate the output as soon as either the input or the recipe is modified.



9. Now, we will compute **SHA1** hash of the output. To do so, search for **sha1** and drag **SHA1** from the **Operations** section to **Recipe** section ensure the number of rounds is **80**.

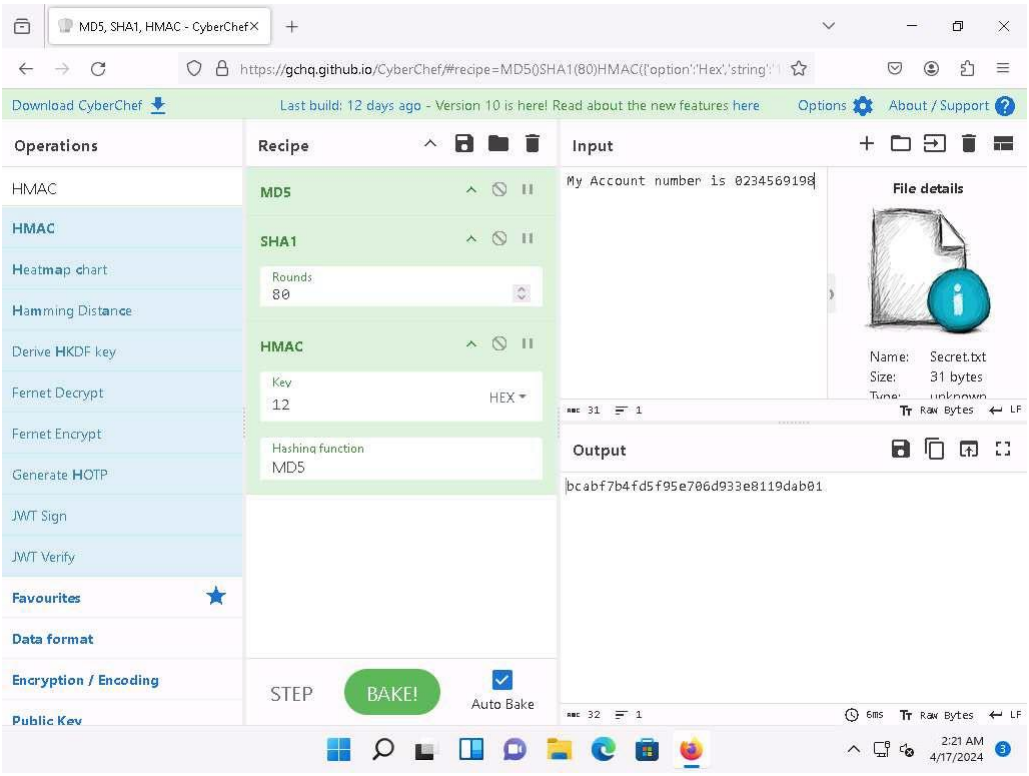
10. The **Output** section displays the computed **SHA1** hash of the **MD5** value.



11. Now, we will add **HMAC** as another layer of hash, search for hmac and drag **HMAC** from **Operations** section to **Recipe** section.

12. In **HMAC** Recipe, enter the key as **12** and select the **Hashing function** as **MD5** from the drop-down.

13. The **Output** section displays the **HMAC** hash of the **SHA1** hash.



14. In addition, we can set break point to the hash operation by clicking on the **Set breakpoint** button present in the Recipe.

    Breakpoints on an operation in the recipe will pause execution before running it.

15. We can also disable a hash operation by clicking the **Disable Operation** button present in the Recipie.

16. This concludes the demonstration of performing multi-layer hashing using CyberChef.

17. Close all open windows and document all the acquired information.

Question 20.1.1.1

In Windows 11 machine create a Secret.txt file with My Account number is 0234569198 and use CyberChef (https://gchq.github.io/CyberChef/) and perform multi-layer hashing. Which button in the recipe will pause execution before running it.

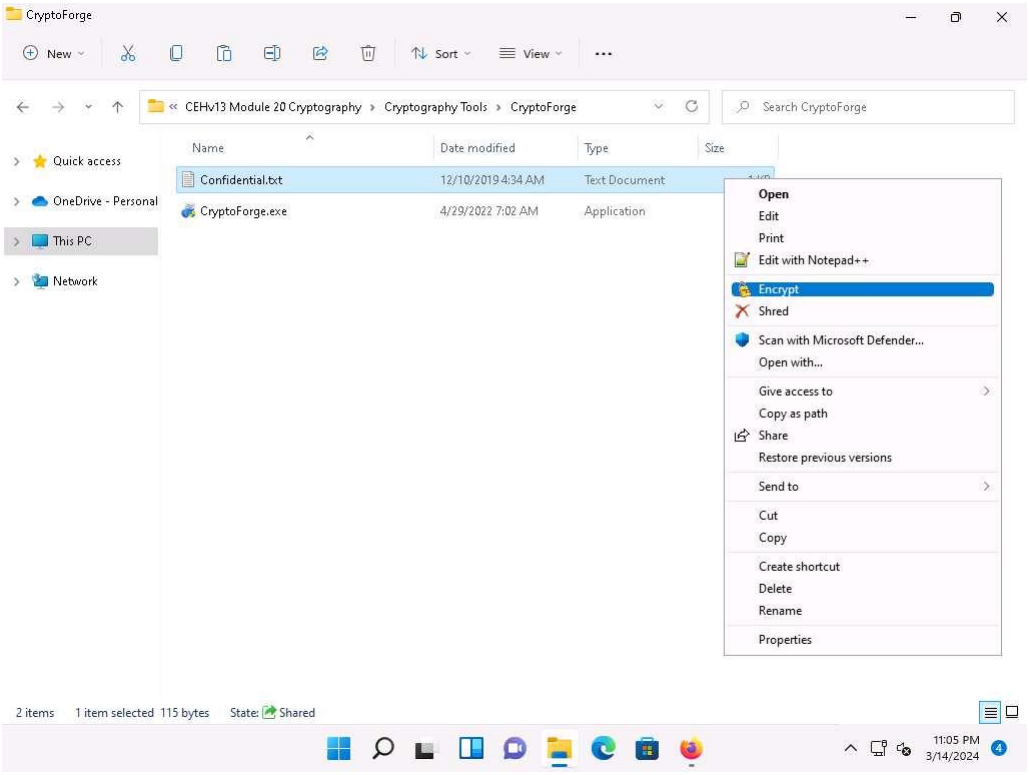
Task 2: Perform File and Text Message Encryption using CryptoForge

CryptoForge is a file encryption software for personal and professional data security. It allows you to protect the privacy of sensitive files, folders, or email messages by encrypting them with strong encryption algorithms. Once the information has been encrypted, it can be stored on insecure media or transmitted on an insecure network—such as the Internet—and remain private. Later, the information can be decrypted into its original form.

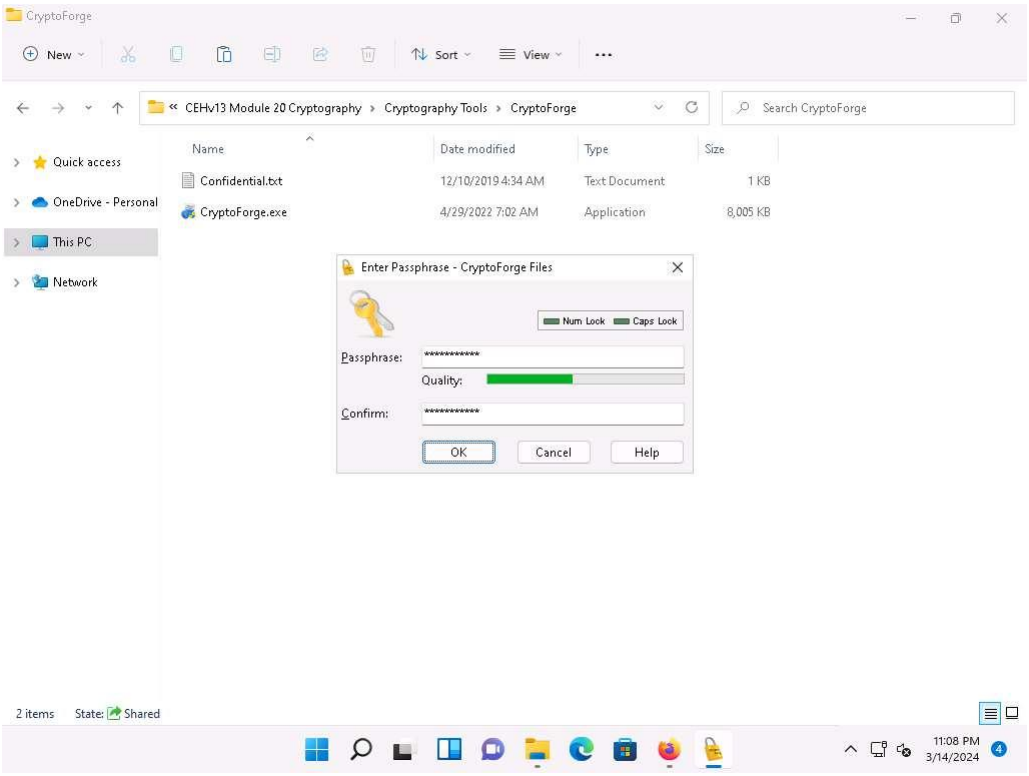
Here, we will use the CryptoForge tool to encrypt a file and text message.

1. Click on Windows 11 to switch to the **Windows 11** machine. Navigate to **E:\CEH-Tools\CEHv13 Module 20 Cryptography\Cryptography Tools\CryptoForge**. Right-click the **Confidential.txt** file and click **Show more options** and select **Encrypt** from the context menu.

   In this task, we are encrypting the **Confidential.txt** file, although you can encrypt any file of your choice.

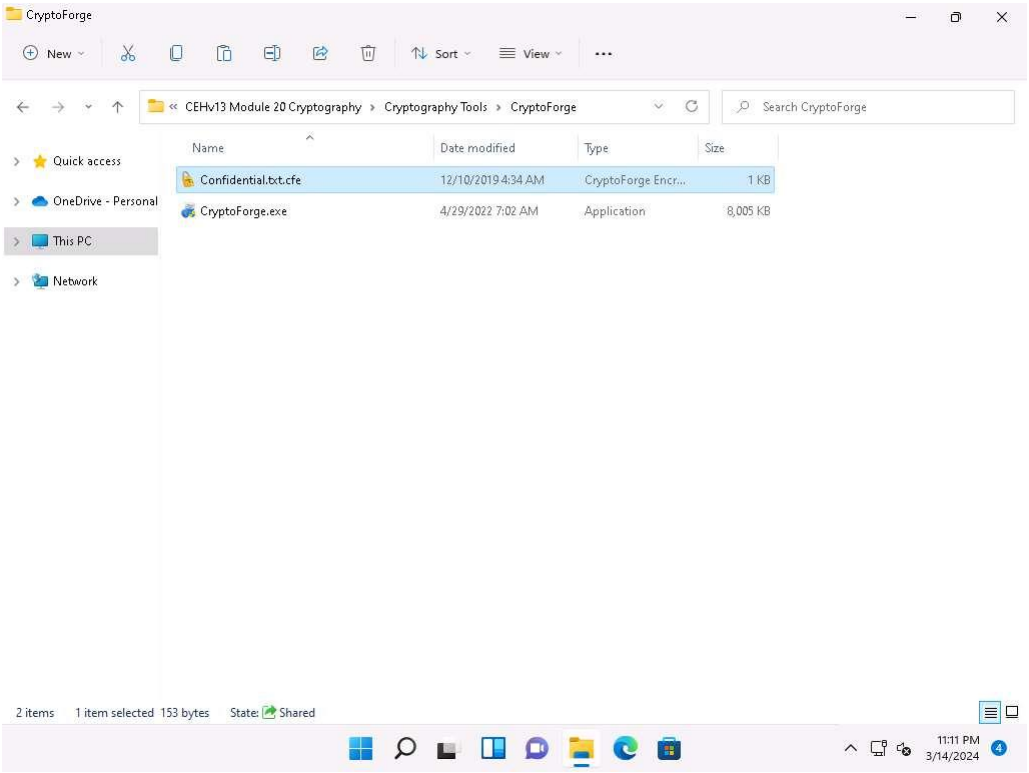


2. The **Enter Passphrase - CryptoForge Files** dialog-box appears; type a password in the **Passphrase** field, retype it in the **Confirm** field, and click **OK**. The password used in this lab is **qwerty@1234**.



3. Now, the file will be encrypted in the same location, and the old file will be deleted automatically, as shown in the screenshot.

   No one can access this file unless the user provides the password for the encrypted file. You will have to share the password with the user through message, email, or any other means.

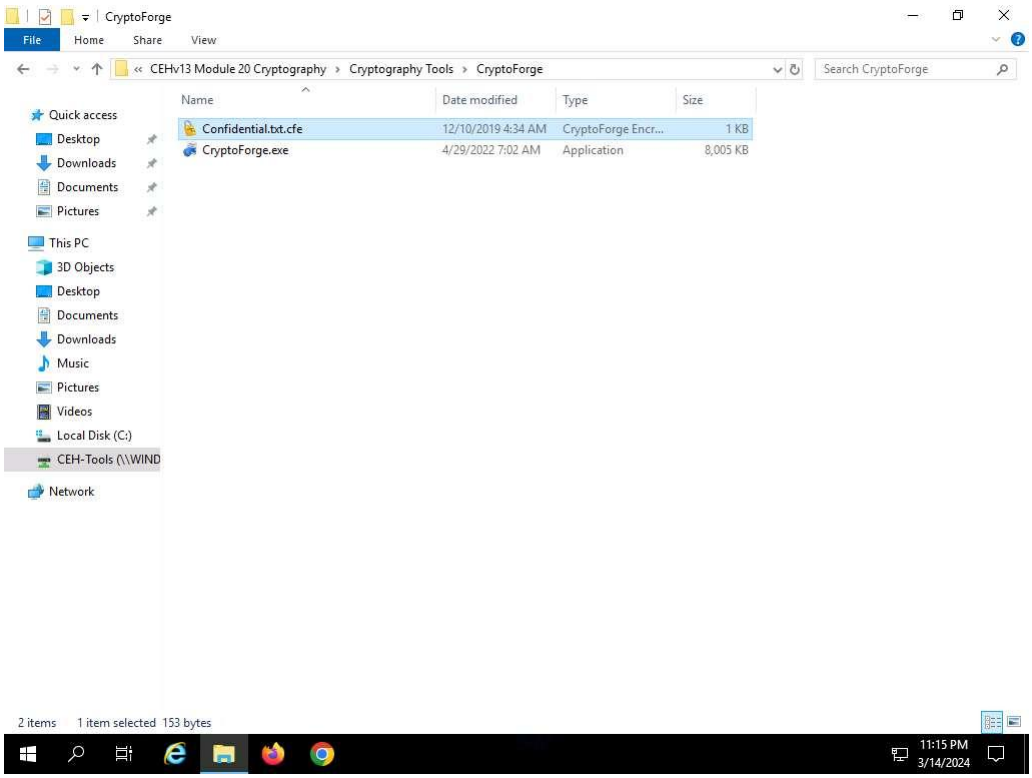


4. Let us assume that you shared this file through a shared network drive.

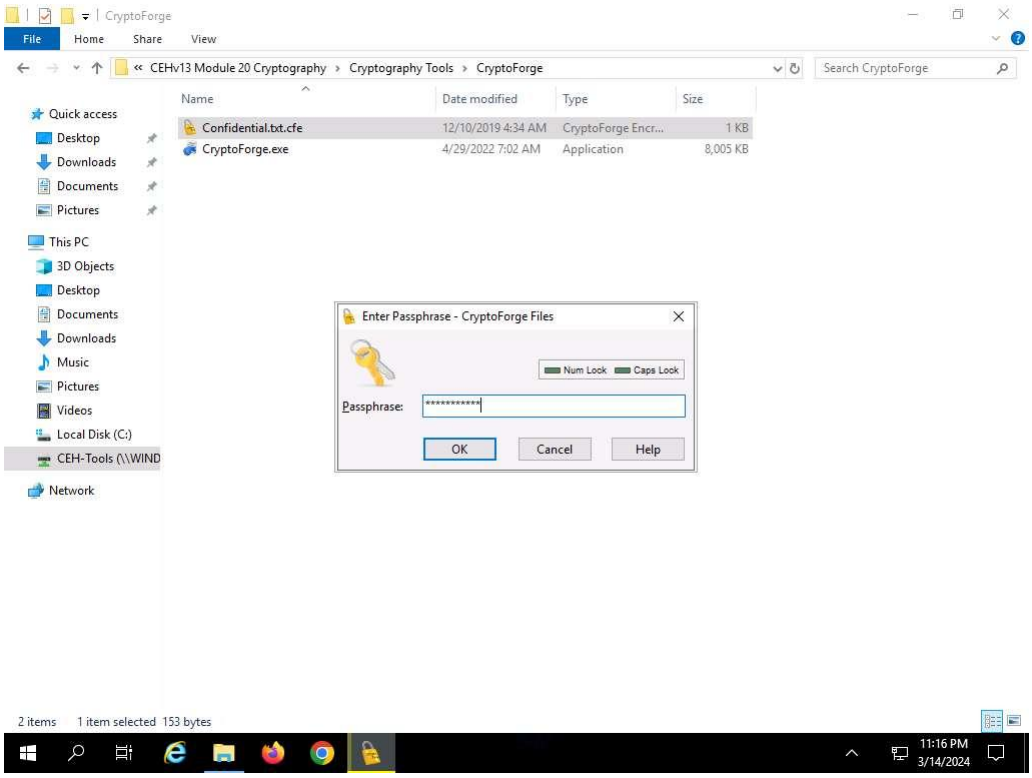
5. Now, click on Windows Server 2019 to switch to the **Windows Server 2019**, click Ctrl+Alt+Delete to activate the machine and login with **Administrator**/**Pa$$w0rd**.

6. Navigate to **Z:\CEHv13 Module 20 Cryptography\Cryptography Tools\CryptoForge**. You will observe the encrypted file in this location.

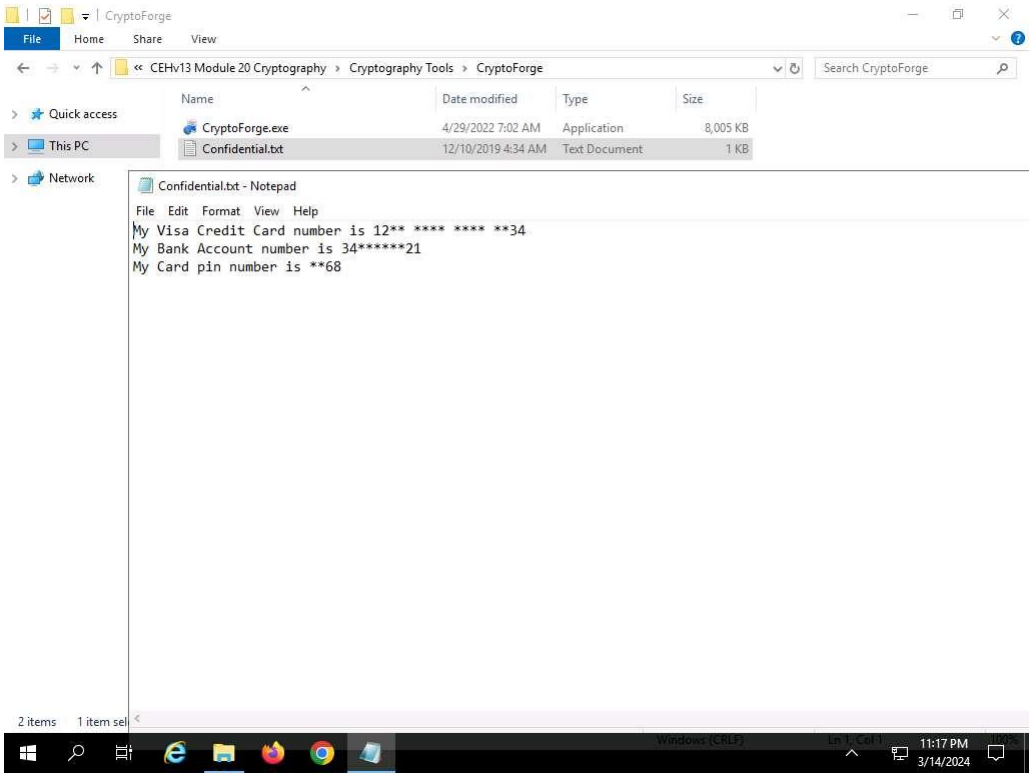
7. Double-click the encrypted file to decrypt it and view its contents.



8. The **Enter Passphrase - CryptoForge Files** dialog-box appears; enter the password that you have provided in **Step#2** to encrypt the file and click **OK**.



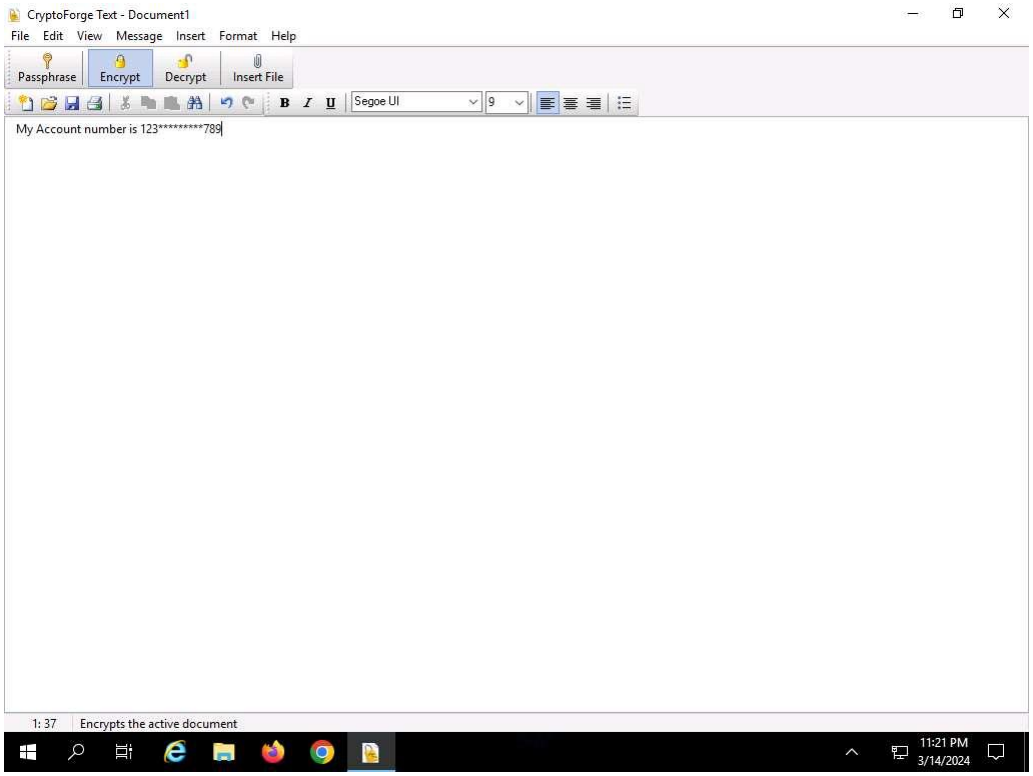
9. Upon entering the password, the file will be successfully decrypted. You may now double-click the text file to view its contents.



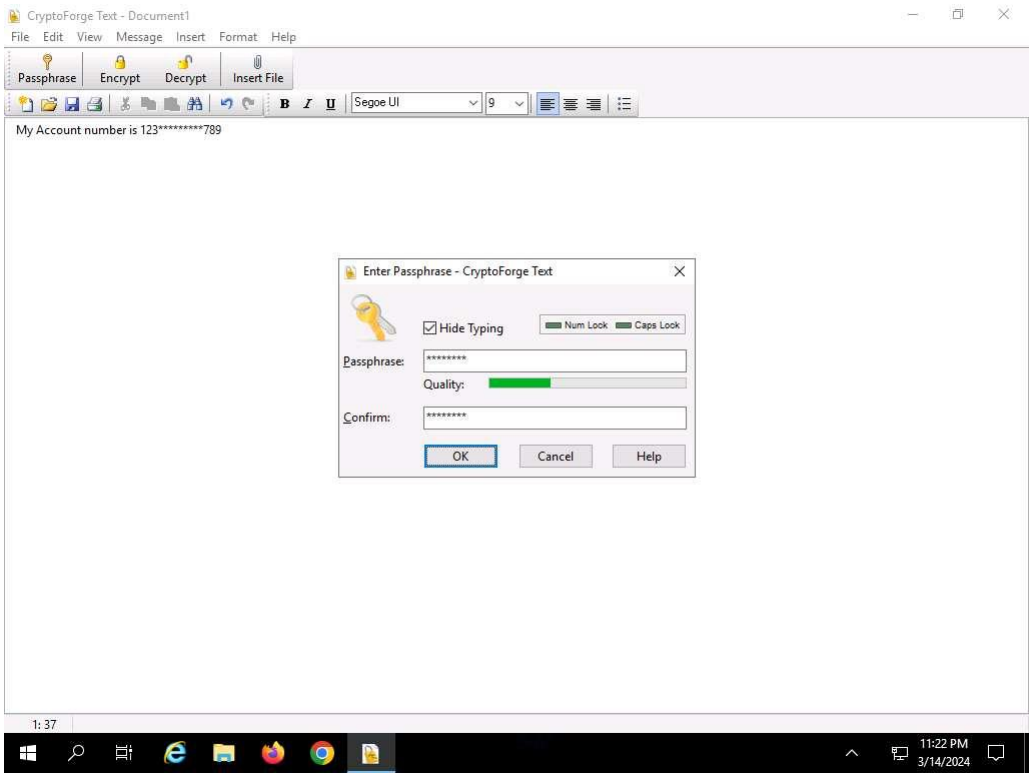
10. So far, you have seen how to encrypt a file and share it with the intended user. Now, we shall share an encrypted message with a user.

11. In the **Windows Server 2019** machine, click the **Type here to search** icon in the **Desktop**, type **crypto** in the search field and click **CryptoForge Text** from the apps to launch the application.

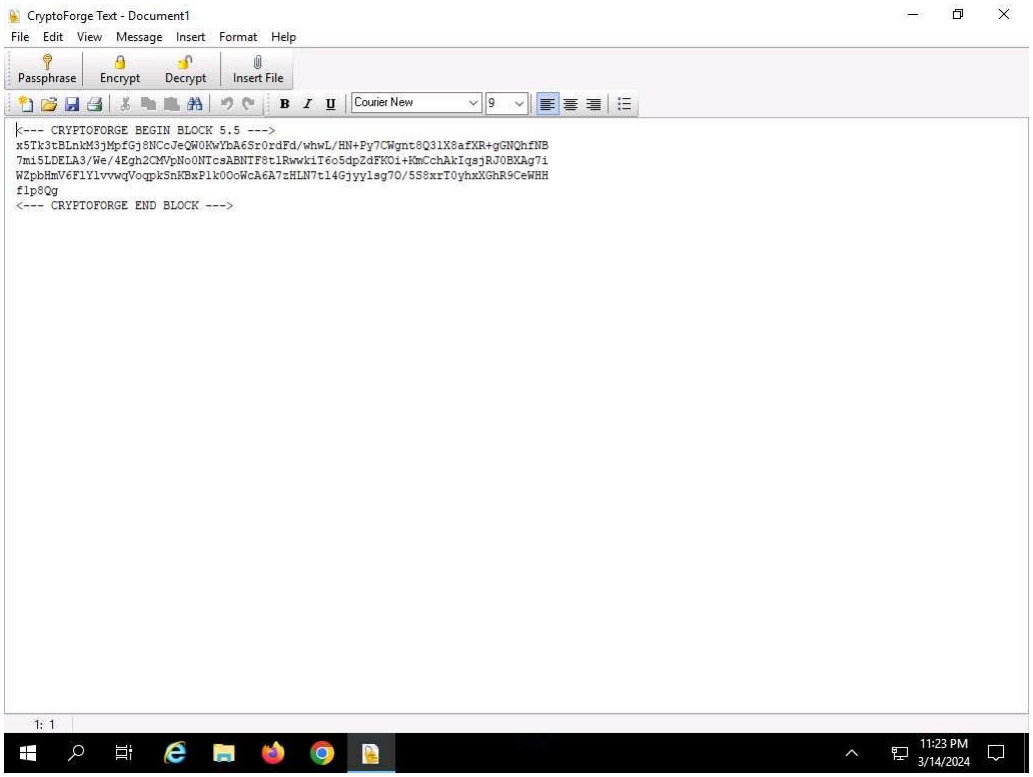
12. The **CryptoForge Text** window appears; type a message and click **Encrypt** from the toolbar.



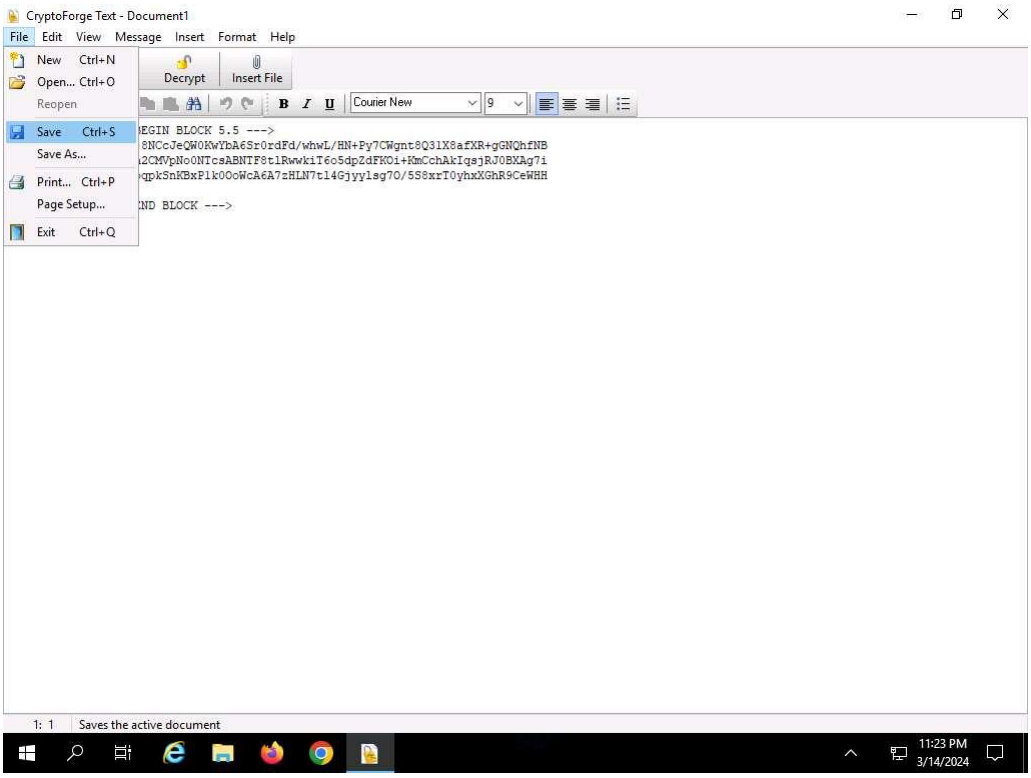
13. The **Enter Passphrase - CryptoForge Text** dialog-box appears; type a password in the **Passphrase** field, retype it in the **Confirm** field, and click **OK**. The password used in this lab is **test@123**.



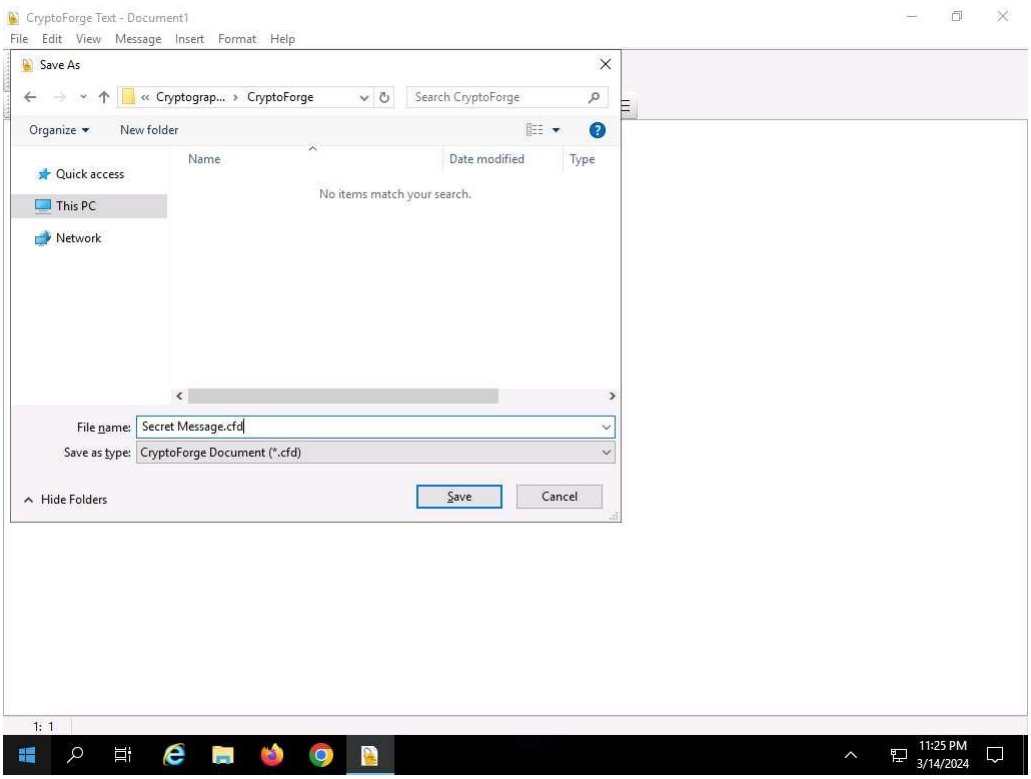
14. The message that you have typed will be encrypted, as shown in the screenshot.



15. Now, you need to save the file. Click **File** in the menu bar and click **Save**.



16. The **Save As** window appears; navigate to **Z:\CEHv13 Module 20 Cryptography\Cryptography Tools\CryptoForge**, specify the file name as **Secret Message.cfd**, and click **Save**.

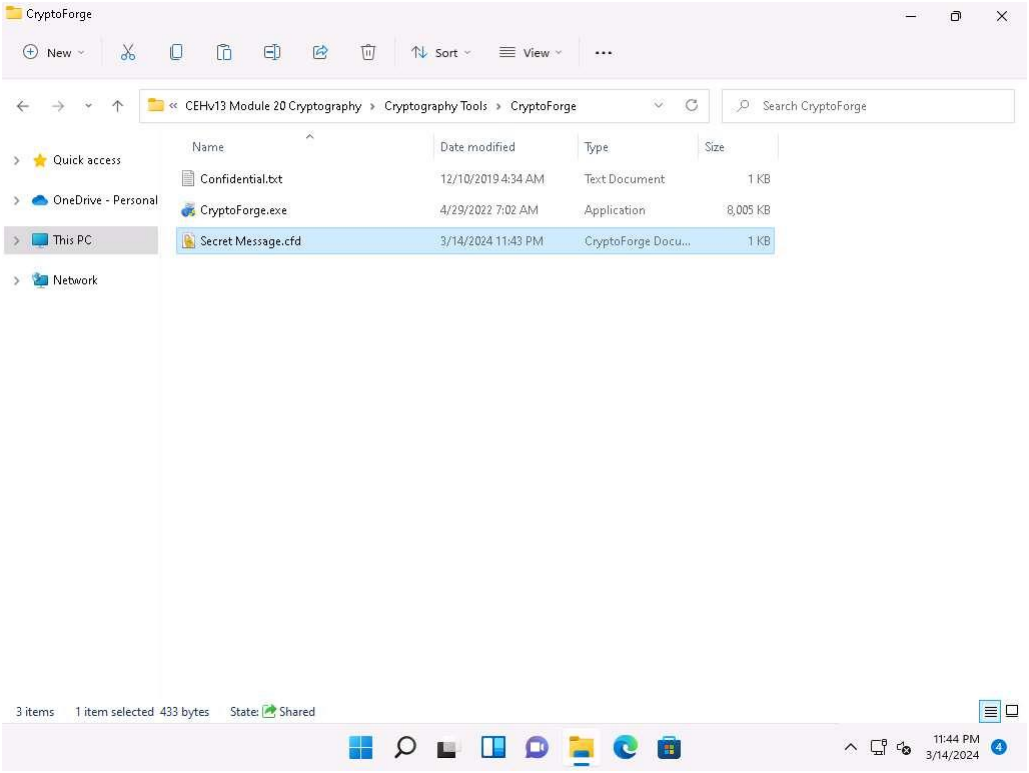


17. Close the **CryptoForge Text** window.

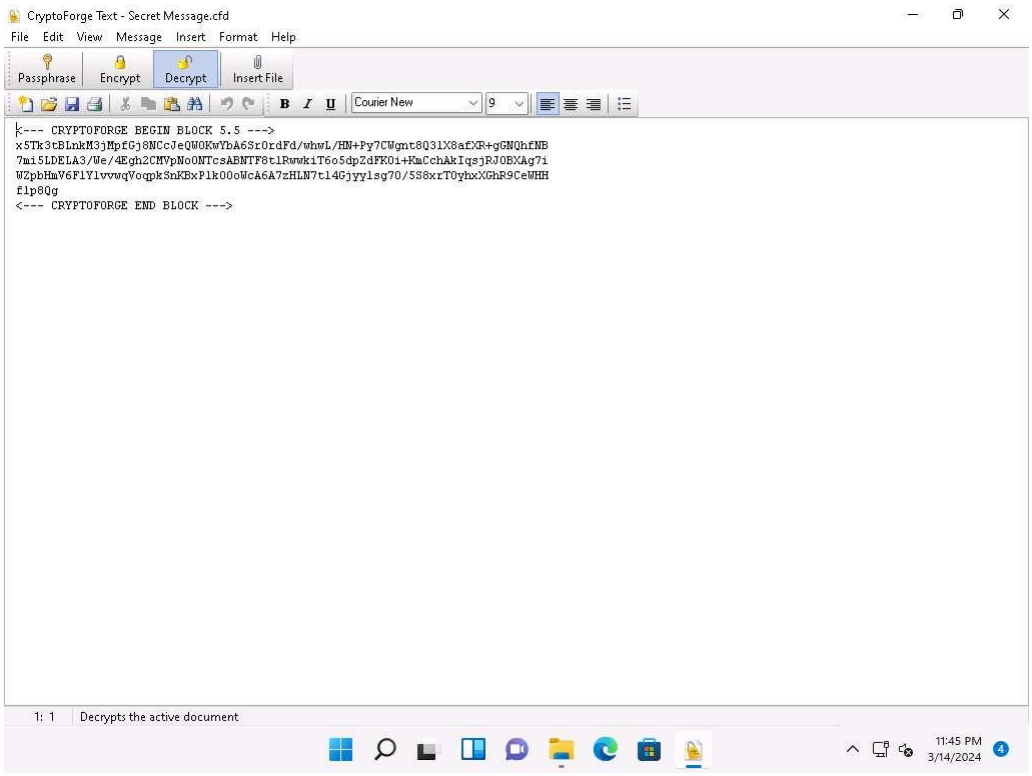
18. Now, let us assume that you shared the file through the mapped network drive and shared the password to decrypt the file in an email message or through some other means.

19. Click on Windows 11 to switch to the **Windows 11** machine and navigate to **E:\CEH-Tools\CEHv13 Module 20 Cryptography\Cryptography Tools\CryptoForge**.

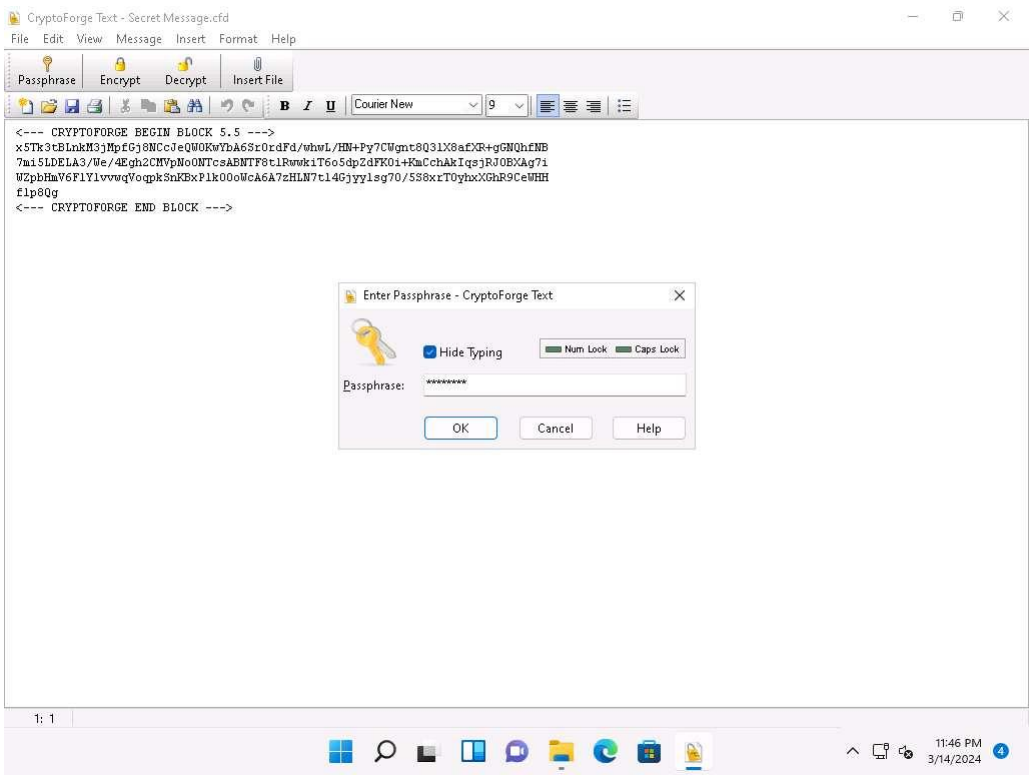
20. You will observe the encrypted file in this location; double-click the file **Secret Message.cfd**.



21. The **CryptoForge Text** window appears, displaying the message in an encrypted format. Click **Decrypt** from the toolbar to decrypt it.

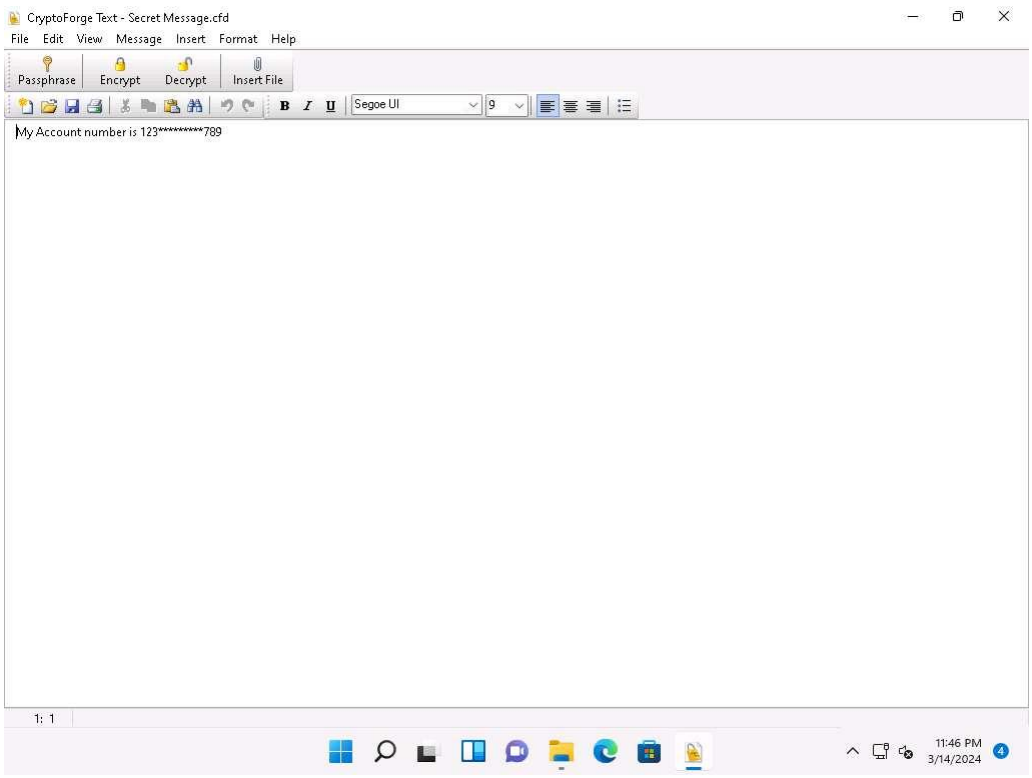


22. The **Enter Passphrase - CryptoForge Text** dialog-box appears; enter the password you provided in **Step#13** to decrypt the message in the **Passphrase** field and click **OK**.



23. The **CryptoForge Text** window appears, displaying the message in plain-text format, as shown in the screenshot.

In real-time, you may share sensitive information through email by encrypting data using CryptoForge.



24. This concludes the demonstration of performing file and text message encryption using CryptoForge.

25. Close all open windows and document all the acquired information.

Question 20.1.2.1

Use CryptoForge to encrypt the file E:\CEH-Tools\CEHv13 Module 20 Cryptography\Cryptography Tools\CryptoForge\Confidential.txt on the Windows 11 machine. What is the extension of the encrypted file?

Lab 2: Create a Self-signed Certificate

Lab Scenario

As a professional ethical hacker and penetration tester, you must possess a proper knowledge of creating this certificate as it validates the public key contained within the certificate belonging to the person, company, server, or other entity mentioned. The labs in this exercise demonstrate the creation of a self-signed certificate.

Lab Objectives

- Create and use self-signed certificates

Overview of Self-signed Certificate

In cryptography and computer security, a self-signed certificate is an identity certificate signed by the same entity whose identity it verifies. However, the term is unrelated to the identity of the person or organization that actually performs the signing procedure.

Task 1: Create and Use Self-signed Certificates

Self-signed certificates are widely used for testing servers. In self-signed certificates, a user creates a pair of public and private keys using a certificate creation tool such as Adobe Acrobat Reader, Java's keytool, Apple's Keychain, etc. and signs the document with the public key. The recipient requests the private key from the sender in order to verify the certificate. However, certificate verification rarely occurs due to the necessity of disclosing the private key: this makes self-signed certificates useful only in a self-controlled testing environment.

Here, we will create a self-signed certificate in Windows Server 2019.

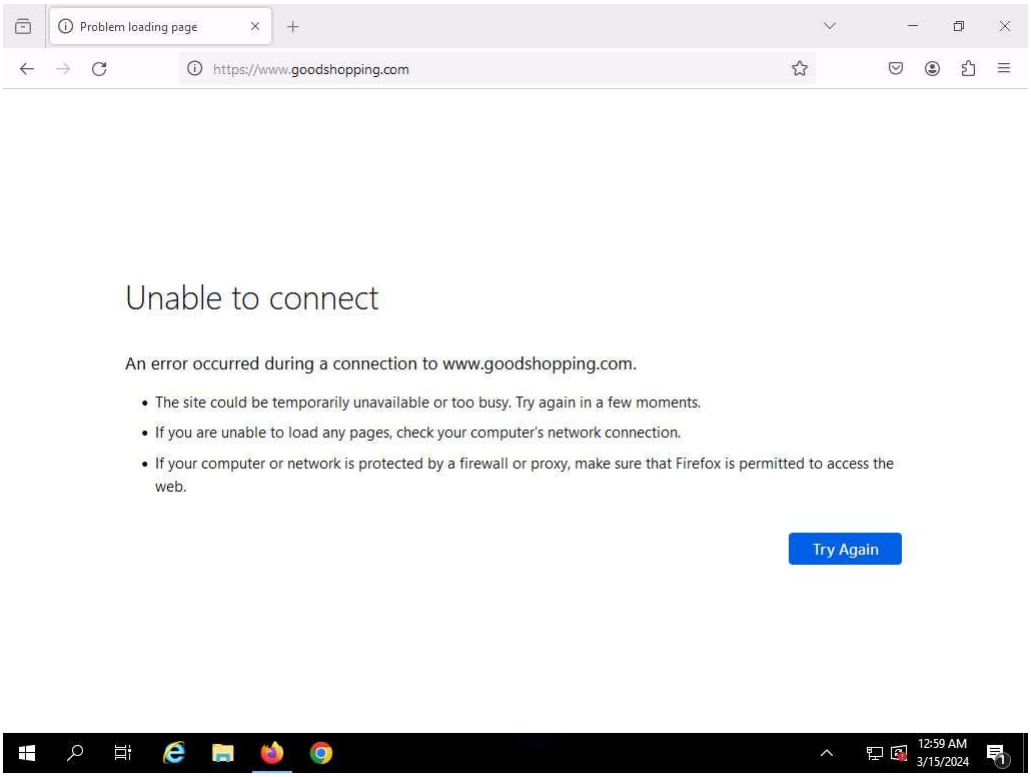
1. Click on Windows Server 2019 to switch to the **Windows Server 2019**.
   Click Ctrl+Alt+Delete to activate the machine and login with **Administrator**/**Pa$$w0rd**.

2. Before you start this task, you will need to check with your local sites whether they include a self-signed certificate.

3. Launch any web browser, and go to **https://www.goodshopping.com** (here, we are using **Mozilla Firefox**).

4. As you are using an https channel to browse the website, it displays a page stating that **Unable to connect**.

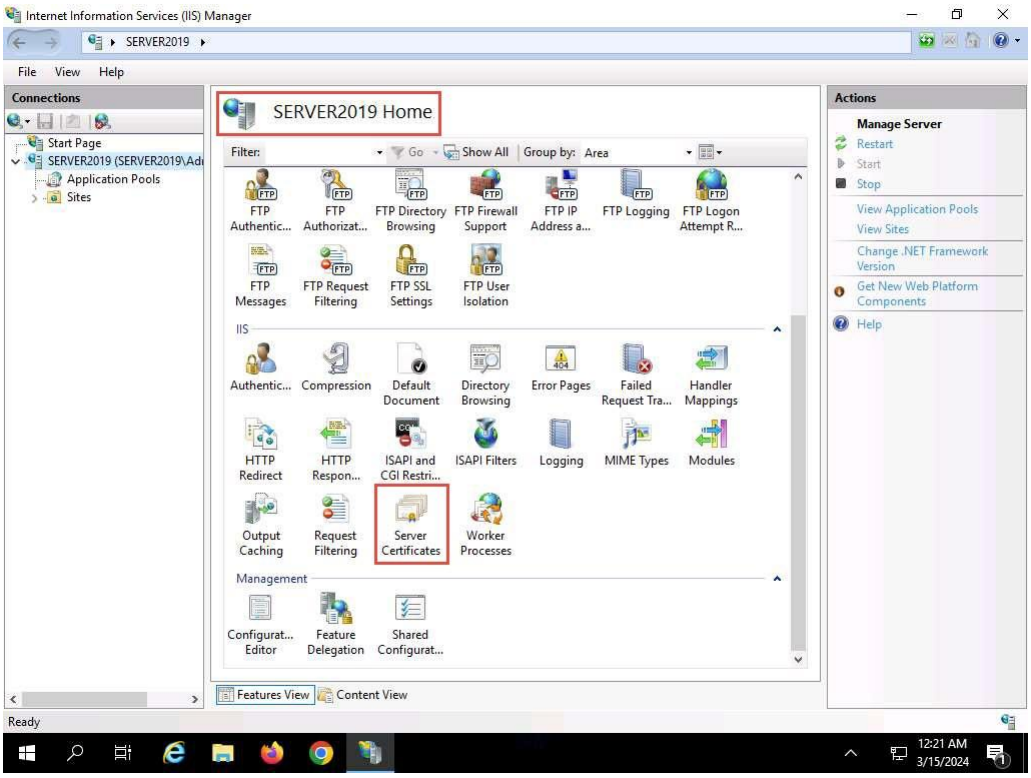
5. As the site does not have a self-signed certificate, it displays a error message, as shown in the screenshot. Close the web browser.



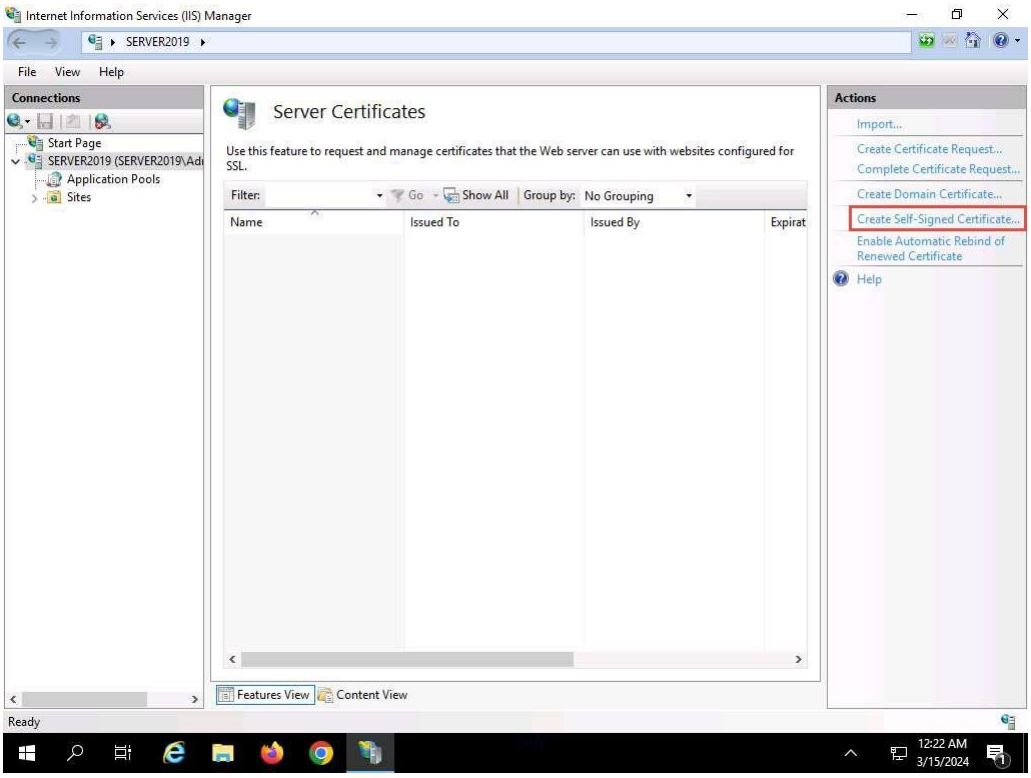
6. Click the **Type here to search** icon present in the bottom-left of **Desktop** and type **iis**. Select **Internet Information Services (IIS) Manager** from the results.

7. The **Internet Information Services (IIS) Manager** window appears; click the machine name (**SERVER2019 (SERVER2019\Administrator**)) under the **Connections** section from the left-hand pane.

8. In **SERVER2019 Home**, double-click **Server Certificates** in the **IIS** section.

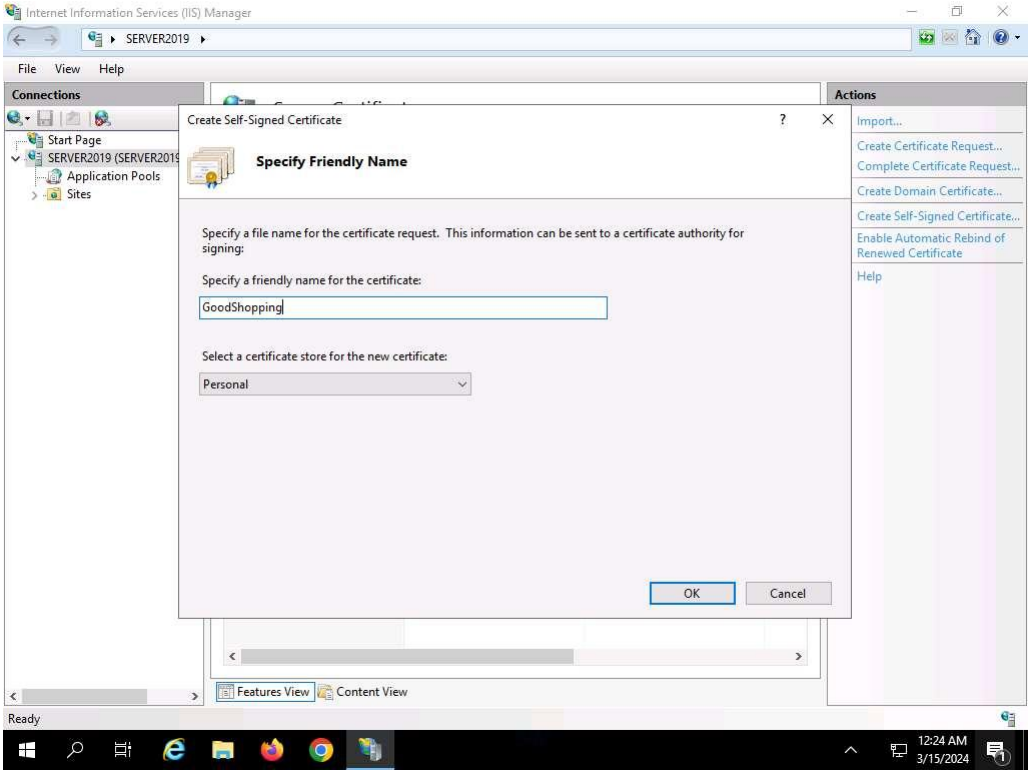


9. The **Server Certificates** wizard appears; click **Create Self-Signed Certificate…** from the right-hand pane in the **Actions** section.

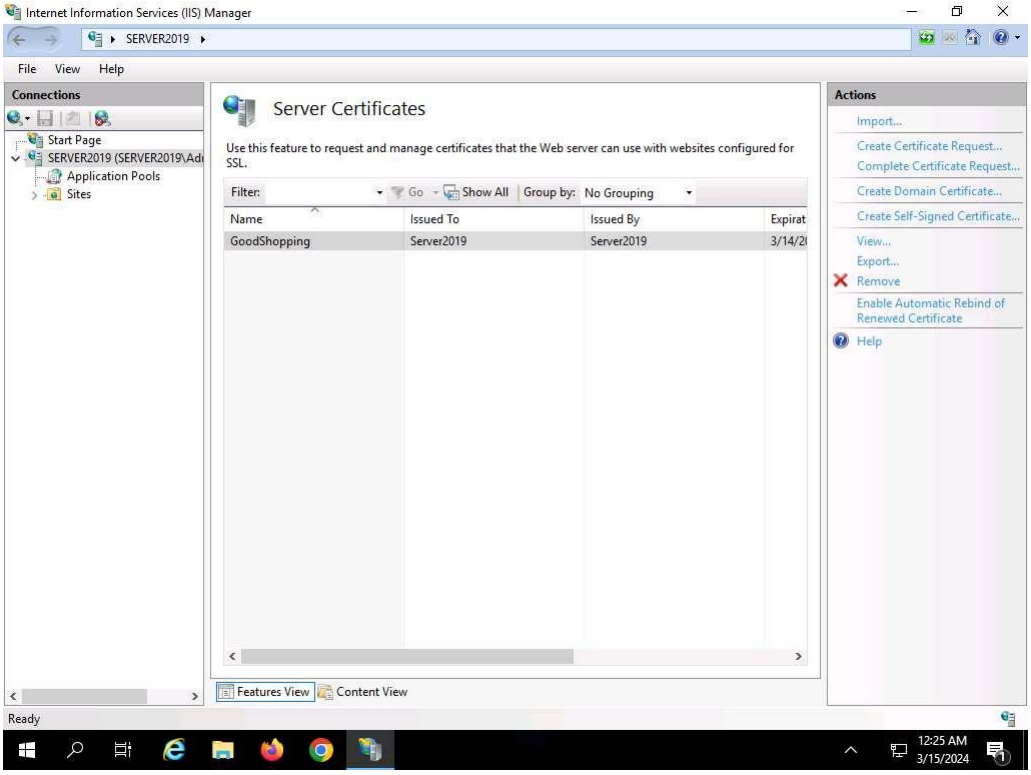


10. The **Create Self-Signed Certificate** window appears; type **GoodShopping** in the **Specify a friendly name for the certificate** field. Ensure that

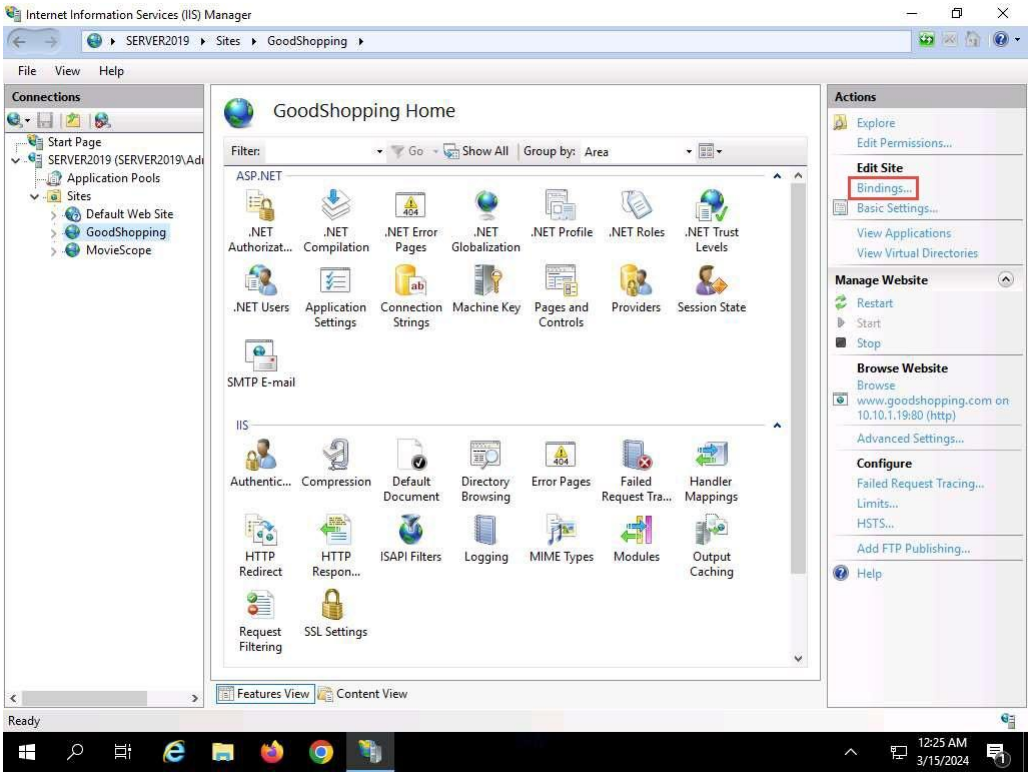
the **Personal** option is selected in the **Select a certificate store for the new certificate** field; then, click **OK**.



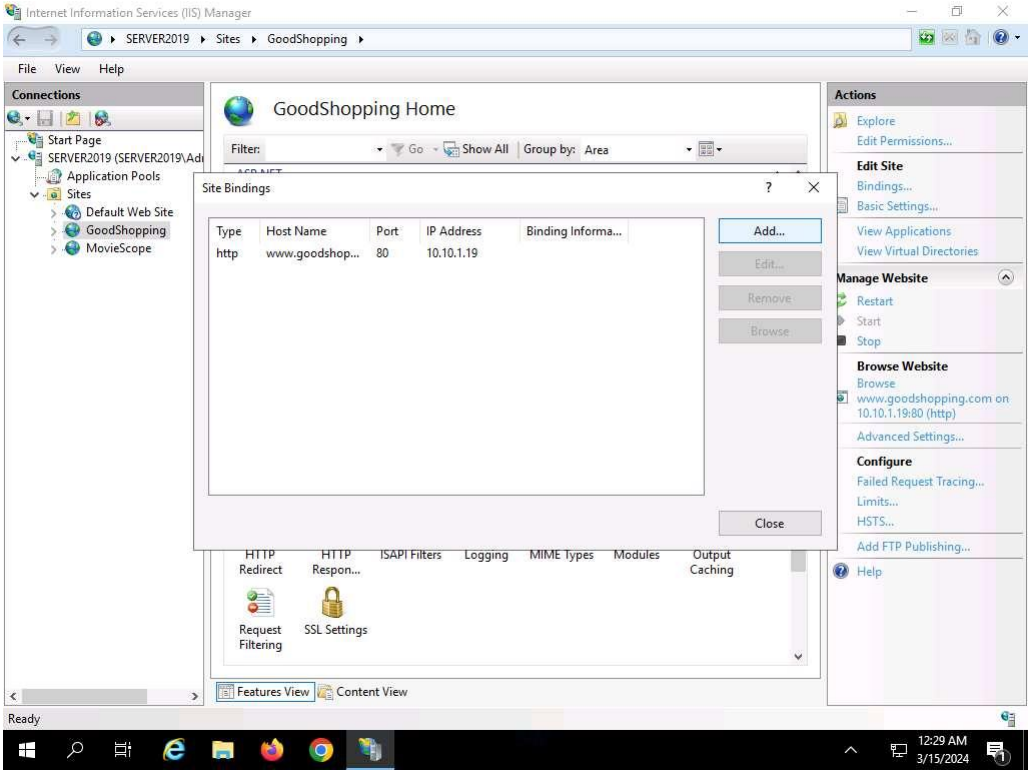
11. A newly created self-signed certificate will be displayed in the **Server Certificates** pane, as shown in the screenshot.



12. Expand the **Sites** node from the left-hand pane, and select **GoodShopping** from the available sites. Click **Bindings…** from the right-hand pane in the **Actions** section.



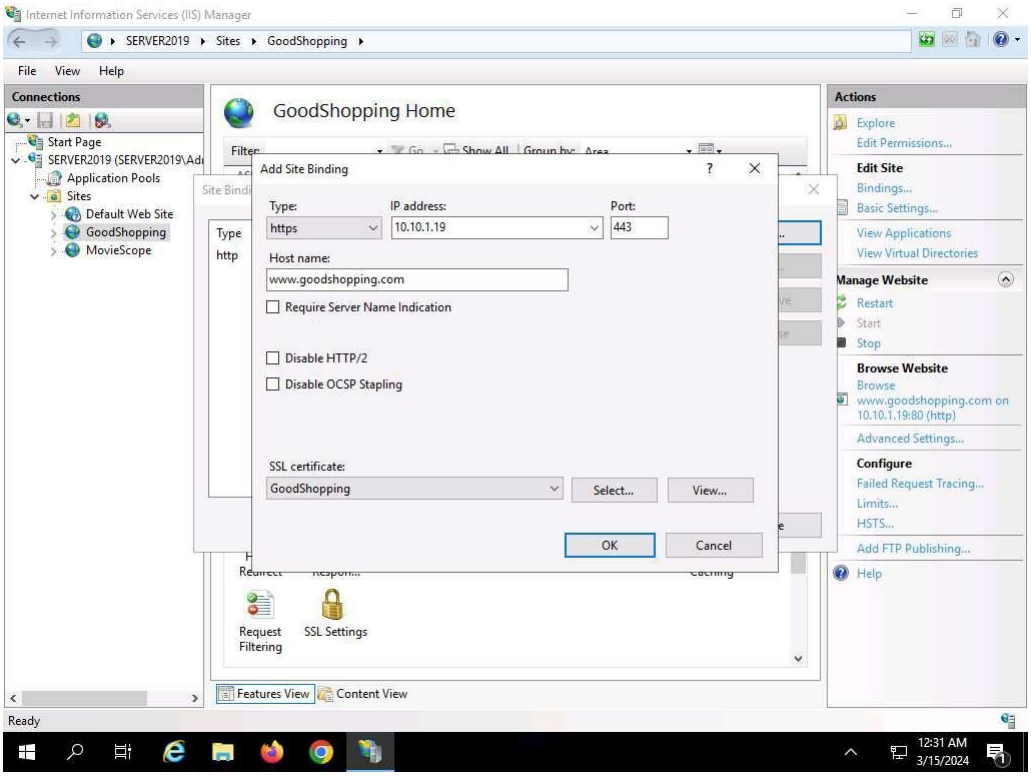
13. The **Site Bindings** window appears; click **Add…**.



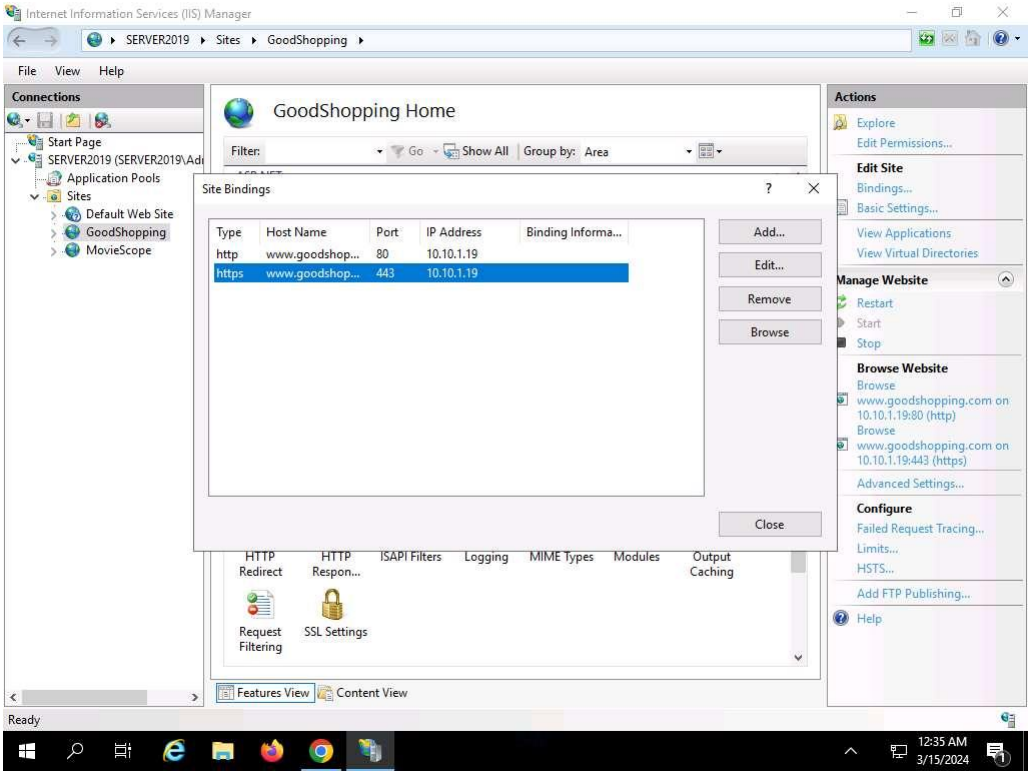
14. The **Add Site Binding** window appears; choose **https** from the **Type** field drop-down list. Once you choose the https type, the port number in the **Port** field automatically changes to **443** (the channel on which HTTPS runs).

15. Choose the **IP address** on which the site is hosted (here, **10.10.1.19**).

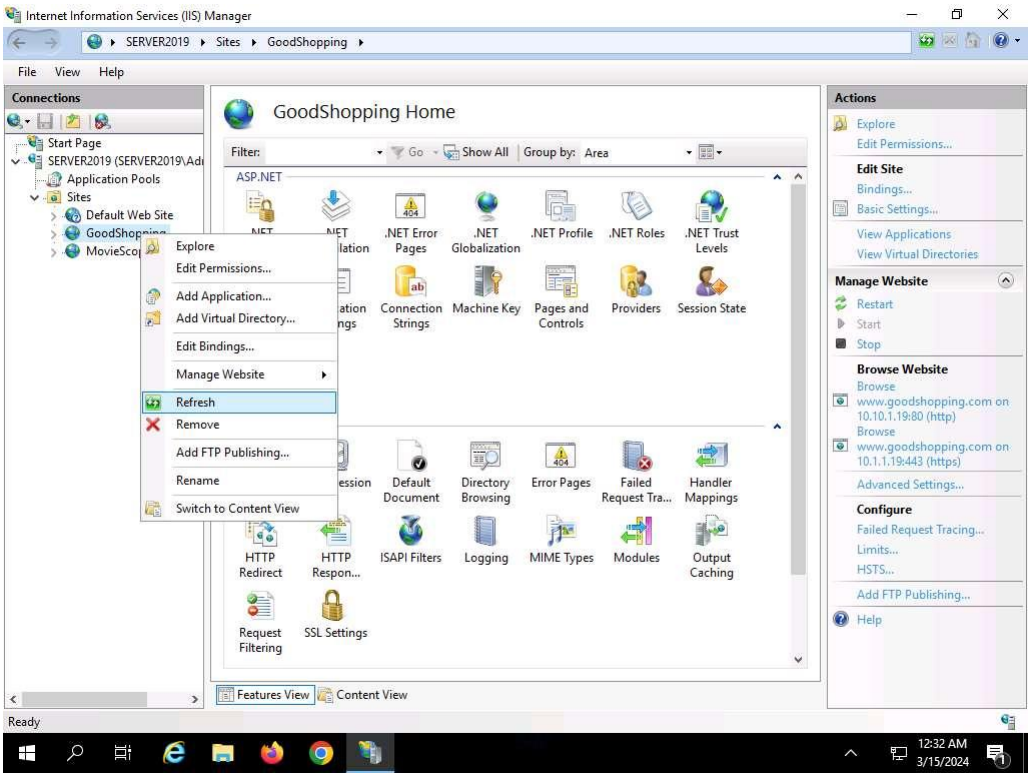
16. Under the **Host name** field, type **www.goodshopping.com**. Under the **SSL certificate** field, select **GoodShopping** from the drop-down list, and click **OK**.



17. The newly created SSL certificate is added to the **Site Bindings** window; then, click **Close**.



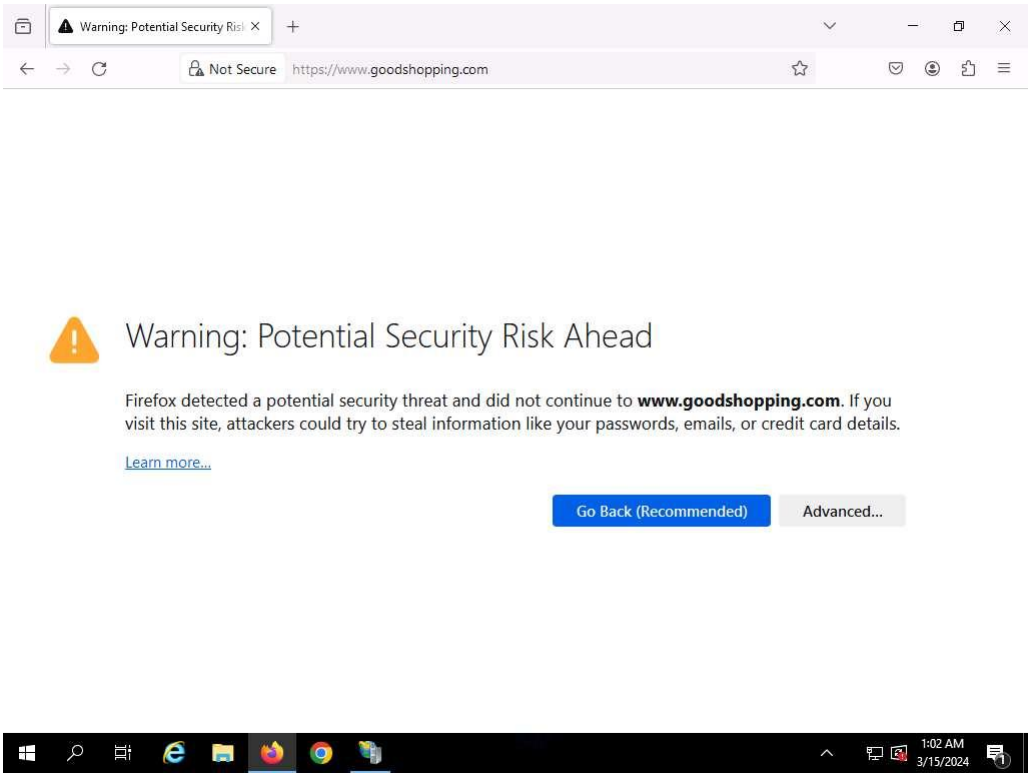
18. Now, right-click the name of the site for which you have created the self-signed certificate (here, **GoodShopping**) and click **Refresh** from the context menu.



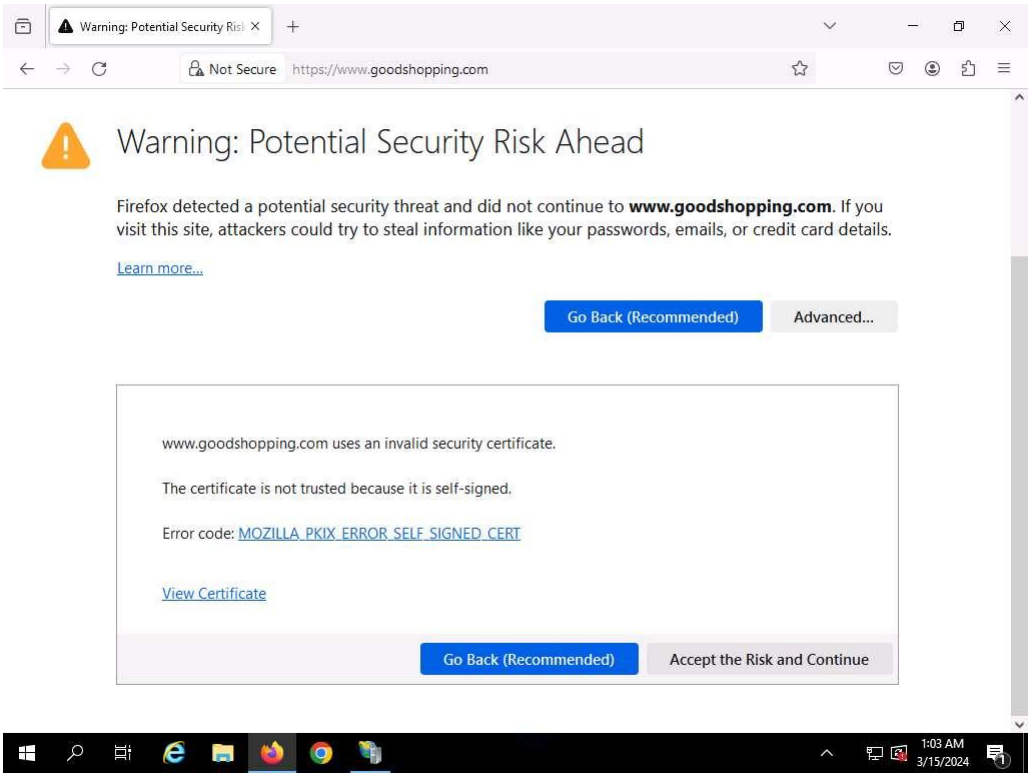
19. Minimize the **Internet Information Services (IIS) Manager** window.

20. Open the **Mozilla Firefox** browser and go to **https://www.goodshopping.com**.

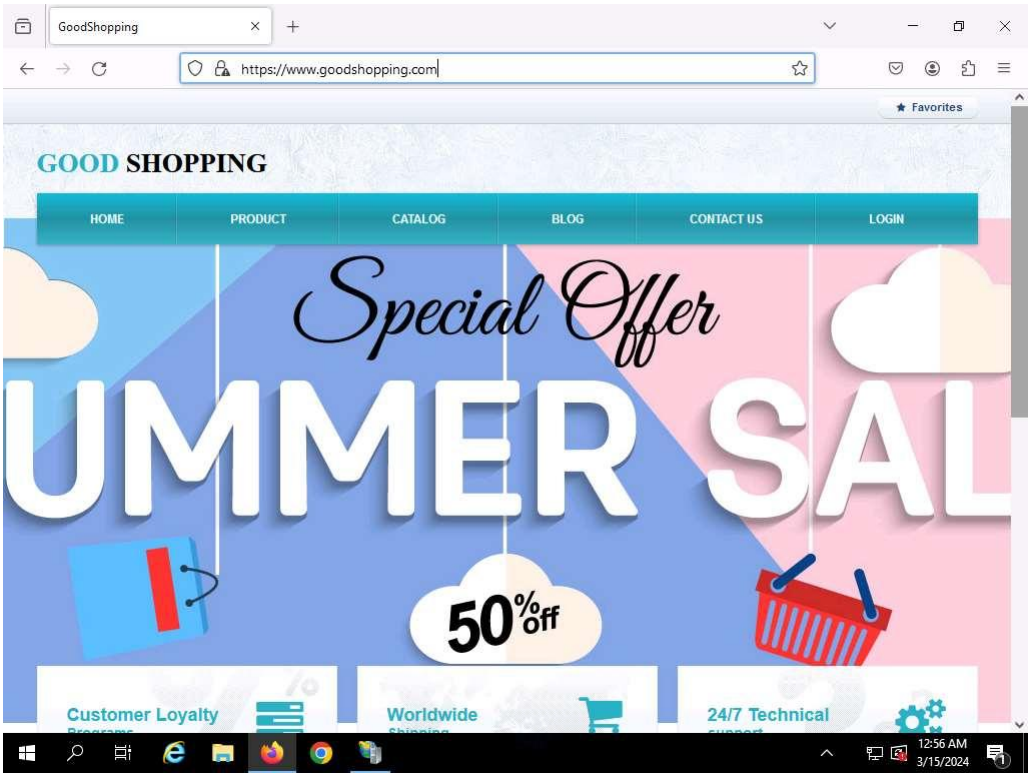
21. The **Warning:Potential Security Risk Ahead** message appears, click **Advanced…** to proceed.



22. Click **Accept the Risk and Continue**.



23. Now you can see **Goodshopping webpage** with **ssl certificate** assigned to it, as shown in the screenshot.



24. This concludes the demonstration of creating and using a self-signed certificate.

25. Close all open windows and document all the acquired information.

Question 20.2.1.1

Create and use a self-signed certificate for the website www.goodshopping.com hosted on the machine at 10.10.1.19. Write the port number on which HTTPS is running in this task.

Lab 3: Perform Disk Encryption

Lab Scenario

Disk encryption is a technology that protects the confidentiality of the data stored on a disk by converting it into an unreadable code using disk encryption software or hardware, thus preventing unauthorized users from accessing it. Disk encryption provides confidentiality and privacy using passphrases and hidden volumes. As a professional ethical hacker or pen tester, you should perform disk encryption in order to prevent sensitive information from unauthorized access.

Disk encryption works in a manner similar to text-message encryption and protects data even when the OS is not active. By using an encryption program for the user's disk (Blue Ray, DVD, USB flash drive, External HDD, and Backup), the user can safeguard any or all

information burned onto the disk and thus prevent it from falling into the wrong hands. Disk-encryption software scrambles the information burned on the disk into an illegible code. It is only after decryption of the disk information that one can read and use it.

This lab will demonstrate the use of various disk encryption tools to perform this technique.

Lab Objectives

- Perform disk encryption using VeraCrypt

Overview of Disk Encryption

Disk encryption is useful when the user needs to send sensitive information through email. In addition, disk encryption can prevent the real-time exchange of information from threats. When users exchange encrypted information, it minimizes the chances of compromising the data; the only way an attacker could access the information is by decrypting the message. Furthermore, encryption software installed on a user's system ensures the security of the system. Install encryption software on any systems that hold valuable information or on those exposed to unlimited data transfer.

Task 1: Perform Disk Encryption using VeraCrypt

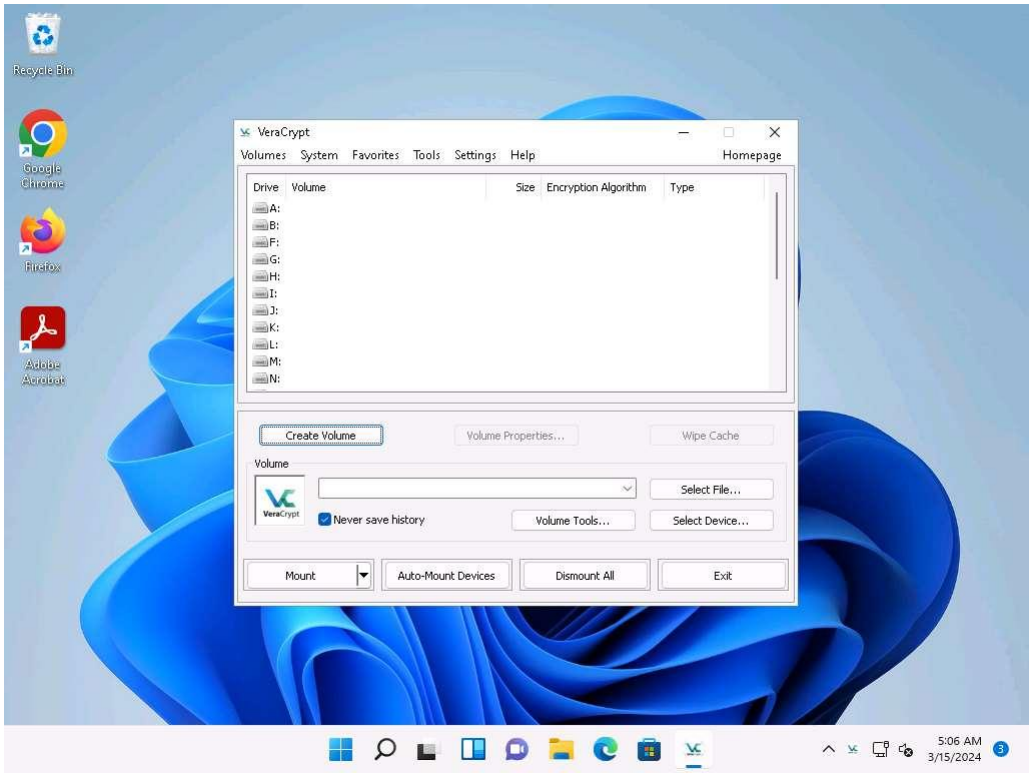
VeraCrypt is a software used for establishing and maintaining an on-the-fly-encrypted volume (data storage device). On-the-fly encryption means that data is automatically encrypted just before it is saved, and decrypted just after it is loaded, without any user intervention. No data stored on an encrypted volume can be read (decrypted) without using the correct password/keyfile(s) or correct encryption keys. The entire file system is encrypted (e.g., file names, folder names, free space, metadata, etc.).

Here, we will use the VeraCrypt tool to perform disk encryption.

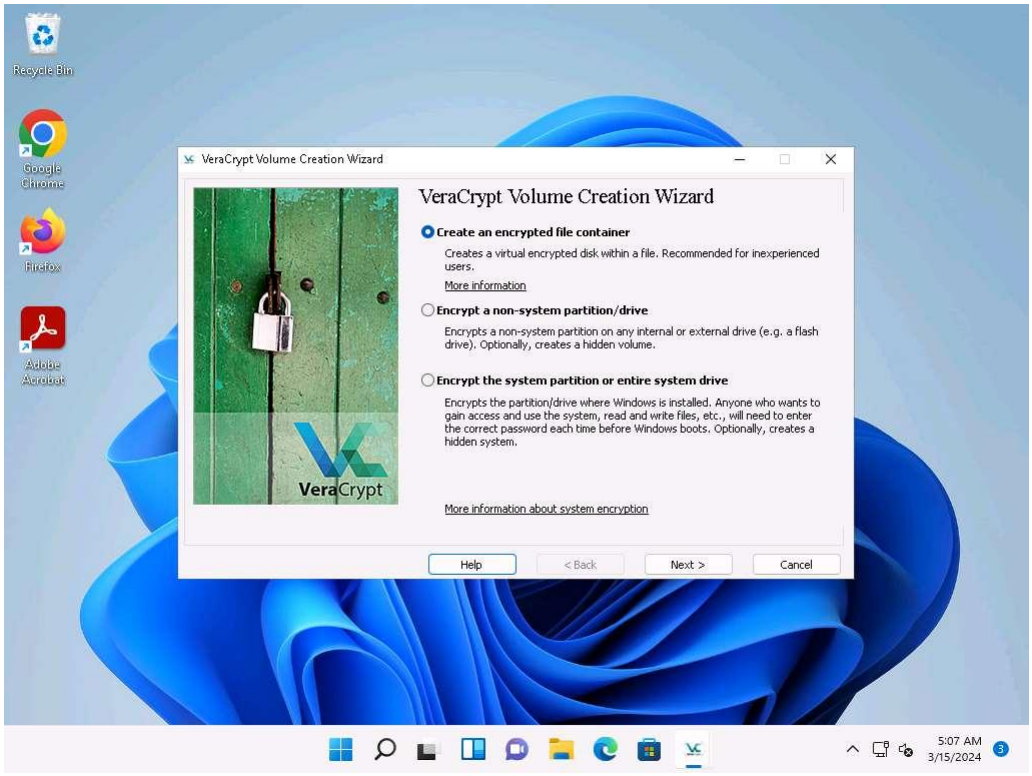
1. Click Windows 11 to switch to the **Windows 11** machine.

2. Click **Search** icon () on the **Desktop**, search for **vera** in the search field, the **VeraCrypt** appears in the results, click **Open** to launch it.

3. The **VeraCrypt** main window appears; click the **Create Volume** button.

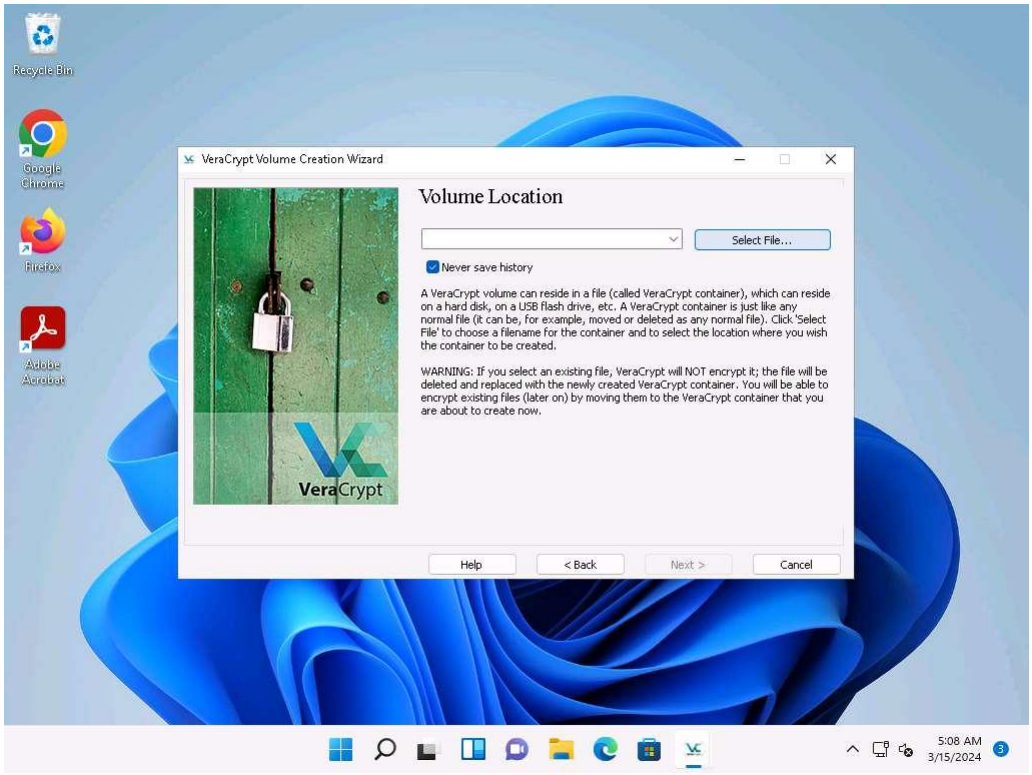


4. The **VeraCrypt Volume Creation Wizard** window appears. Ensure that the **Create an encrypted file container** radio-button is selected and click **Next** to proceed.

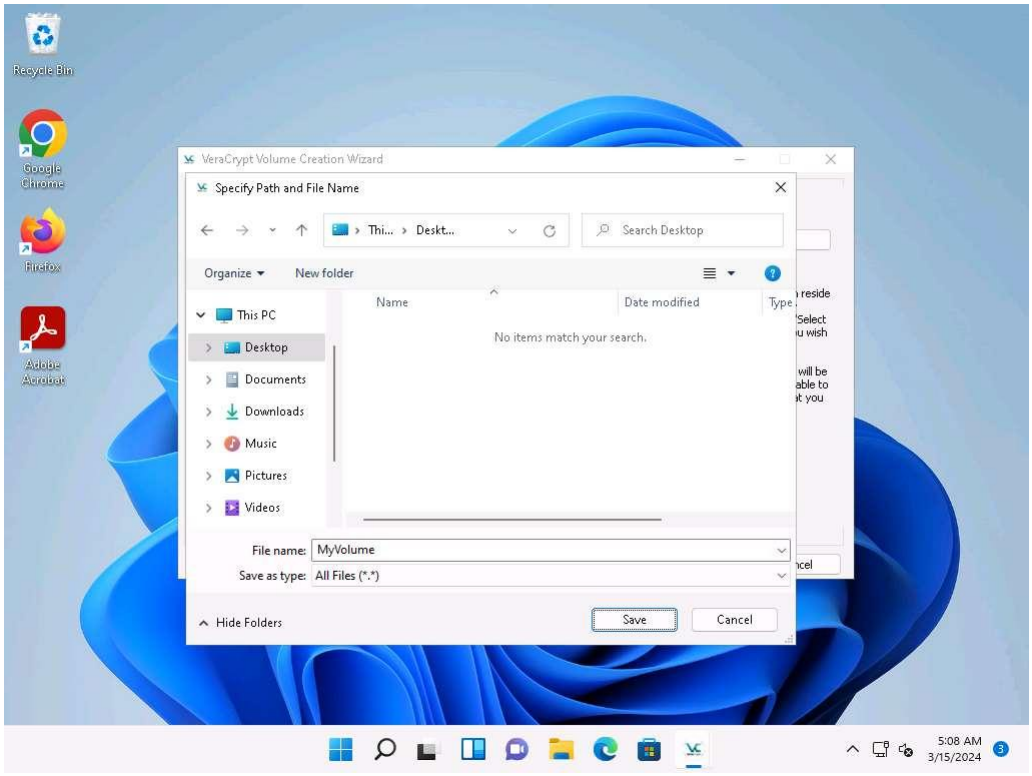


5. In the **Volume Type** wizard, keep the default settings and click **Next**.

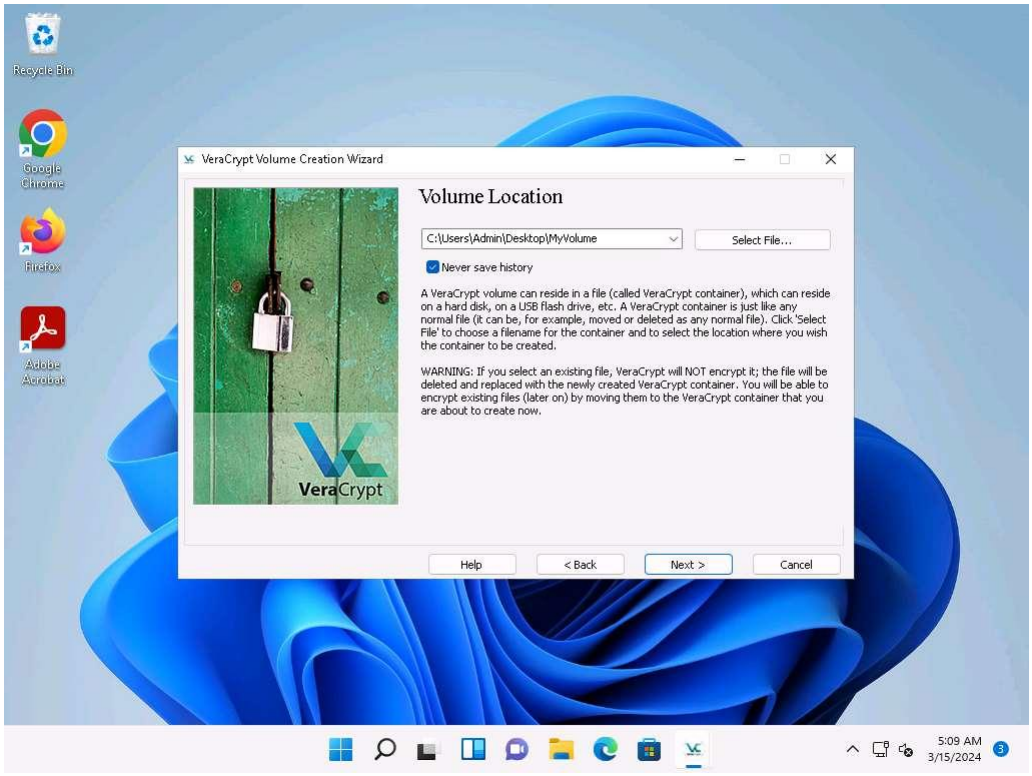
6. In the **Volume Location** wizard, click **Select File…**.



7. The **Specify Path and File Name** window appears; navigate to the desired location (here, **Desktop**), provide the **File name** as **MyVolume**, and click **Save**.

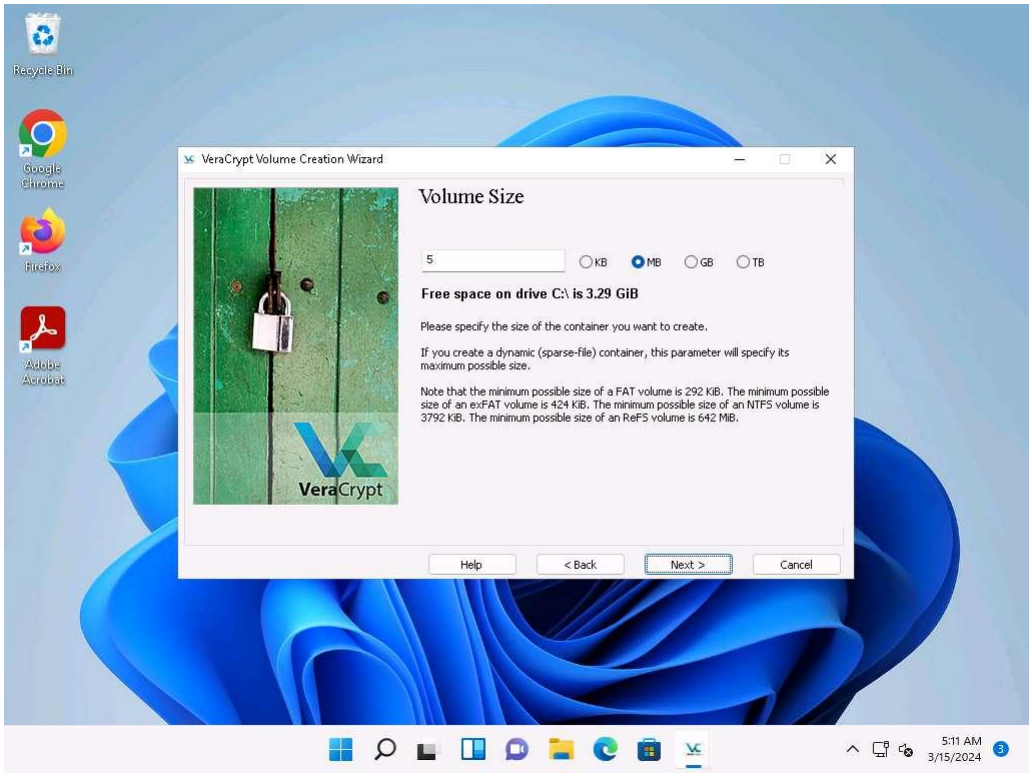


8. After saving the file, the location of a file containing the **VeraCrypt** volume appears under the **Volume Location** field; then, click **Next**.

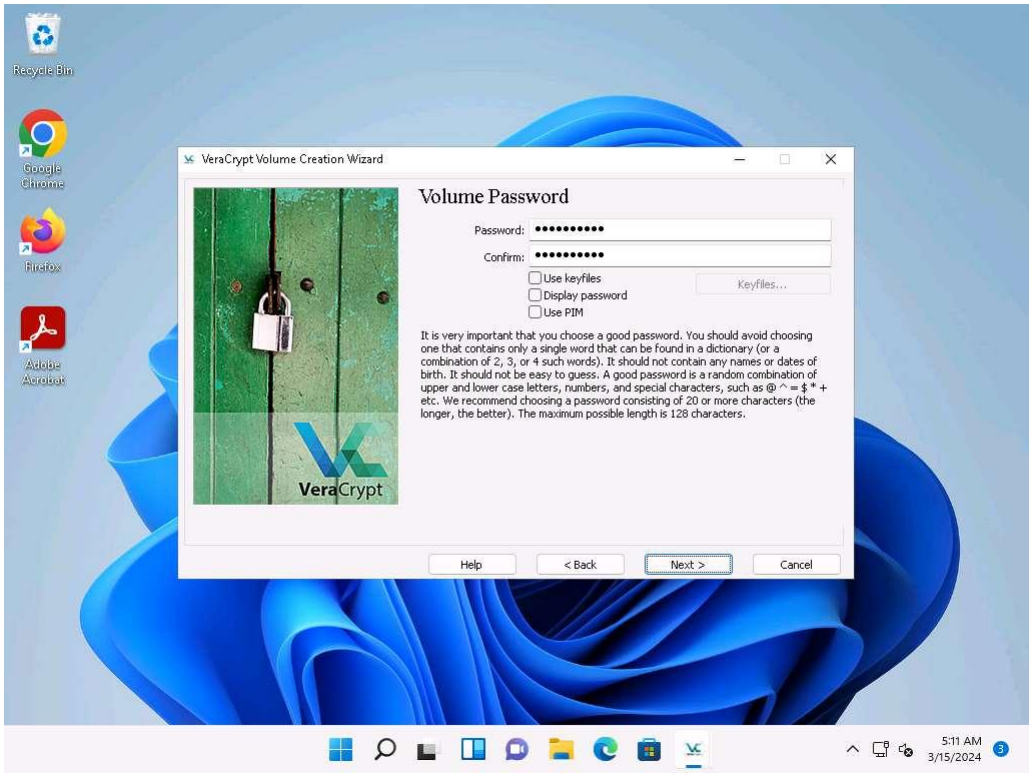


9. In the **Encryption Options** wizard, keep the default settings and click **Next**.

10. In the **Volume Size** wizard, ensure that the **MB** radio-button is selected and specify the size of the VeraCrypt container as **5**; then, click **Next**.



11. The **Volume Password** wizard appears; provide a strong password in the **Password** field, retype in the **Confirm** field, and click **Next**. The password provided in this lab is **qwerty@123**.

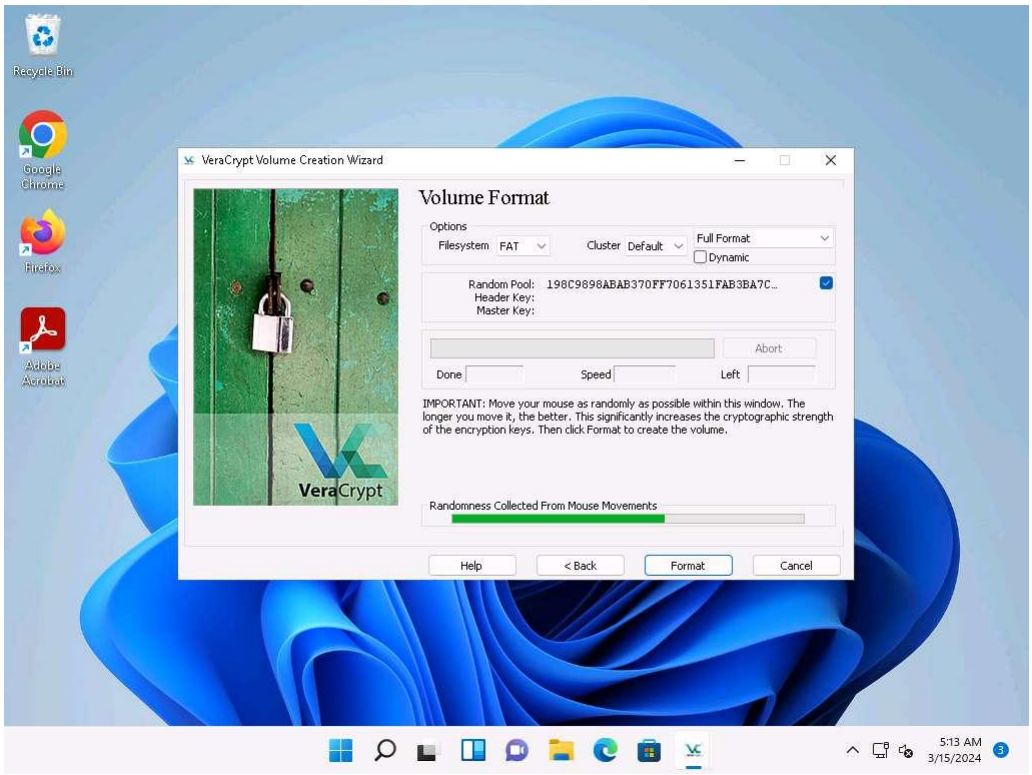


A **VeraCrypt Volume Creation Wizard** warning pop-up appears; then, click **Yes**.

12. The **Volume Format** wizard appears; ensure that **FAT** is selected in the **Filesystem** option and **Default** is selected in **Cluster** option.

13. Check the checkbox under the **Random Pool, Header Key**, and **Master Key** section.

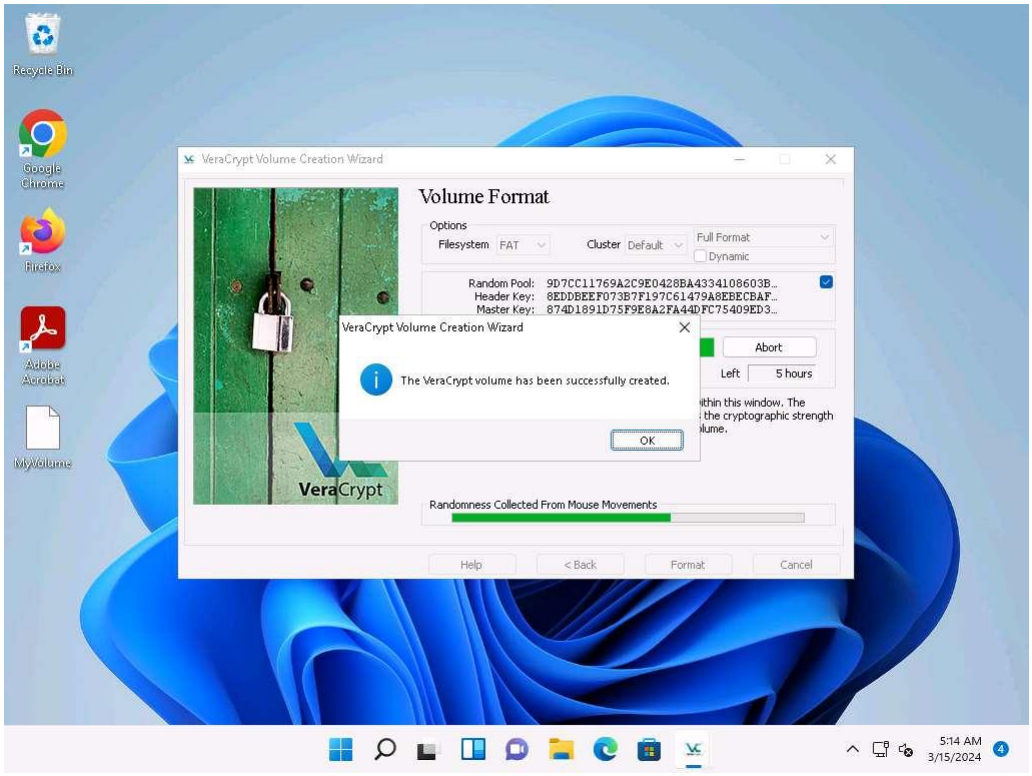
14. Move your mouse as randomly as possible within the **Volume Creation Wizard** window for at least **30 seconds** and click the **Format** button.



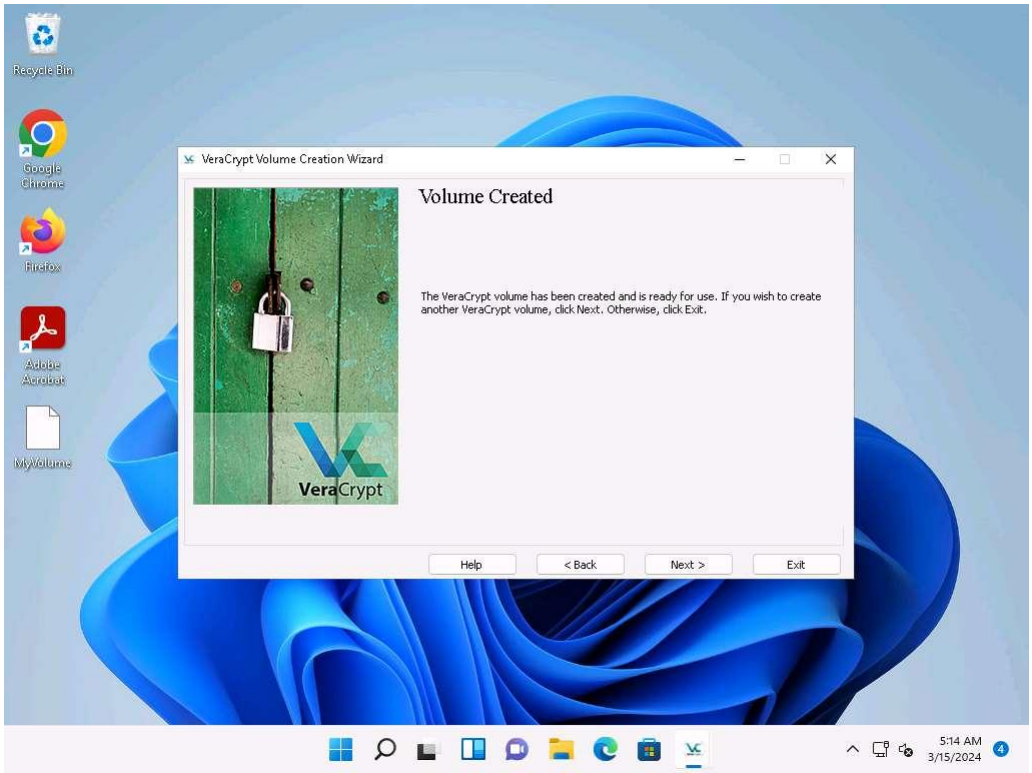
15. After clicking **Format**, VeraCrypt will create a file called **MyVolume** in the provided folder. This file depends on the VeraCrypt container (it will contain the encrypted VeraCrypt volume).

16. Depending on the size of the volume, volume creation may take some time.

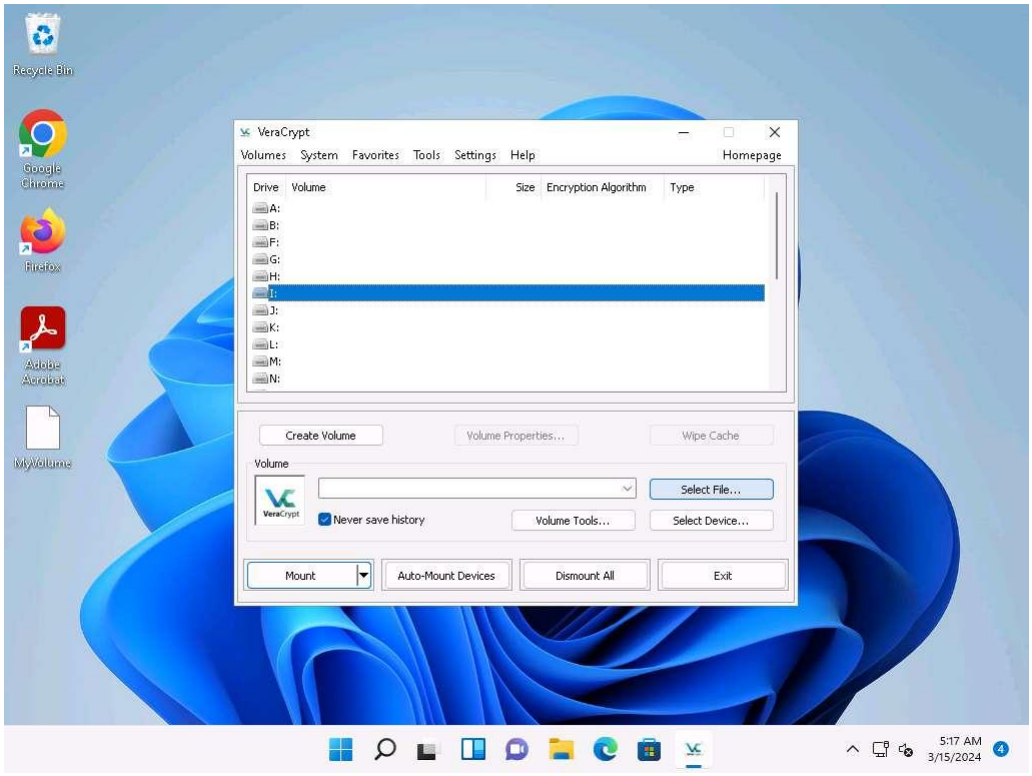
17. Once the volume is created, a **VeraCrypt Volume Creation Wizard** dialog-box appears; click **OK**.



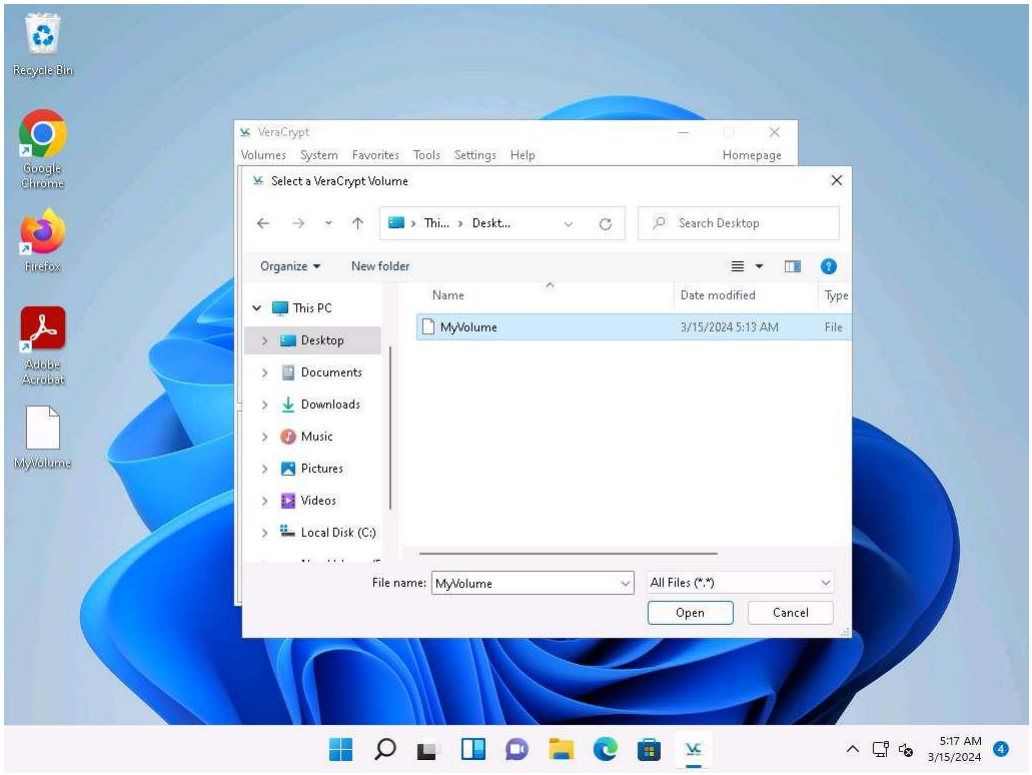
18. In the **VeraCrypt Volume Creation Wizard** window, a **Volume Created** message appears; then, click **Exit**.



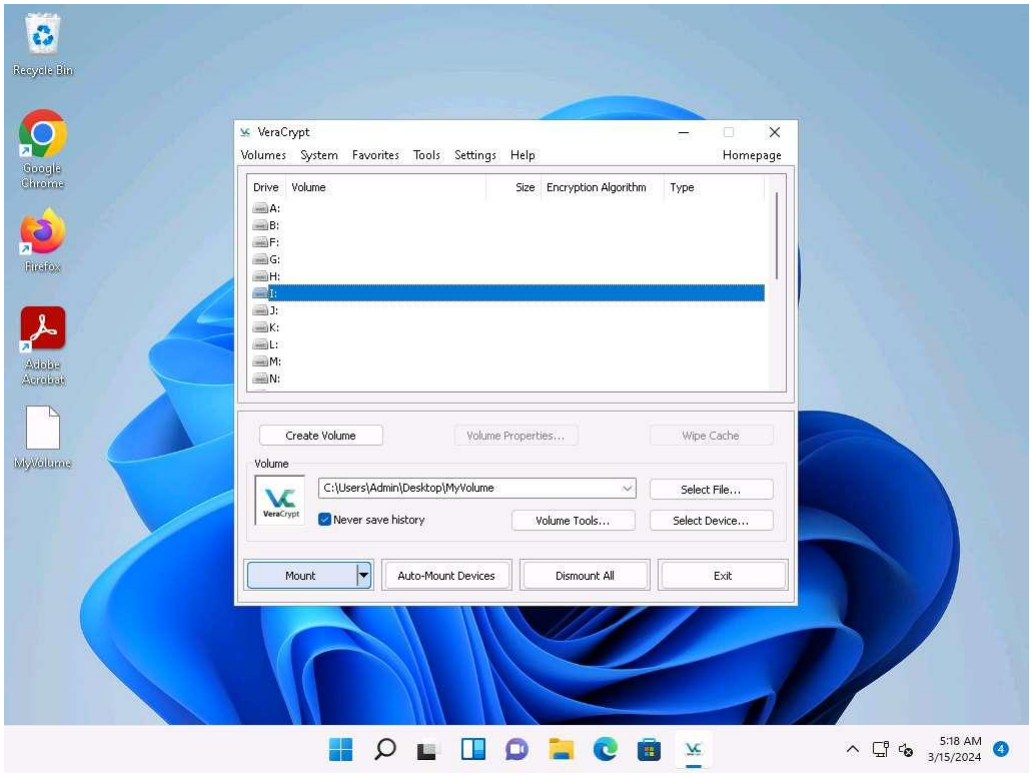
19. The **VeraCrypt** main window appears; select a drive (here, **I:**) and click **Select File…**.



20. The **Select a VeraCrypt Volume** window appears; navigate to **Desktop**, click **MyVolume**, and click **Open**.

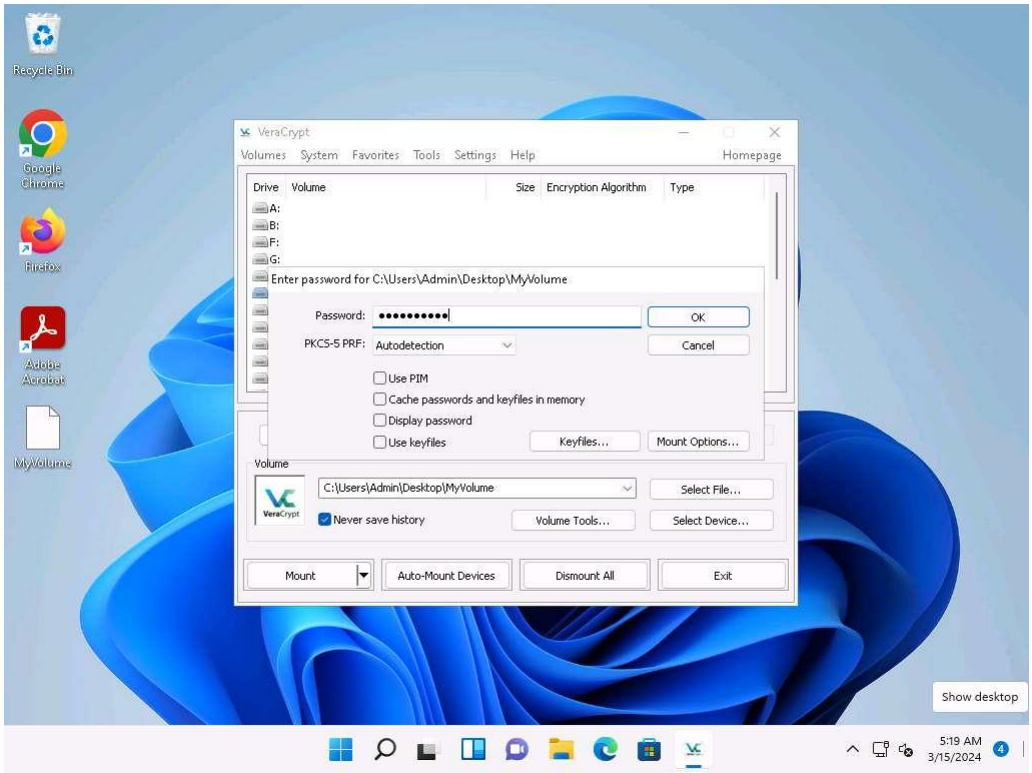


21. The window closes, and the **VeraCrypt** window appears displaying the location of selected **volume** under the Volume field; then, click **Mount**.

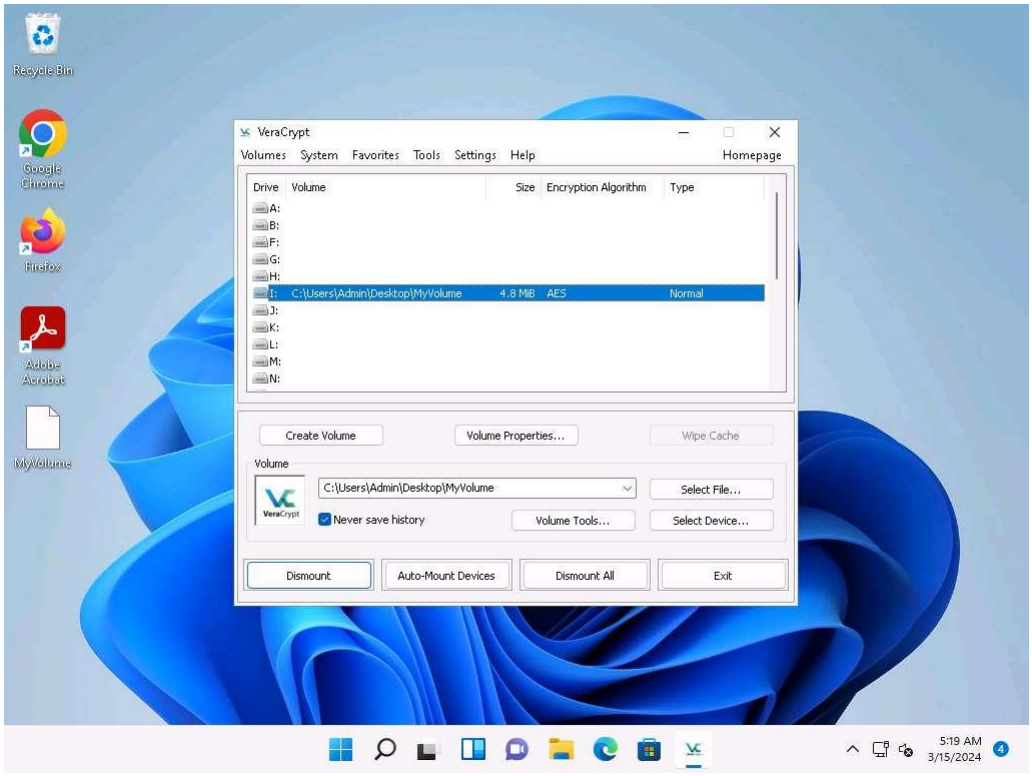


22. The **Enter password** dialog-box appears; type the password you specified in **Step#11** into the **Password** field and click **OK**.

    The password specified in this task is **qwerty@123**.



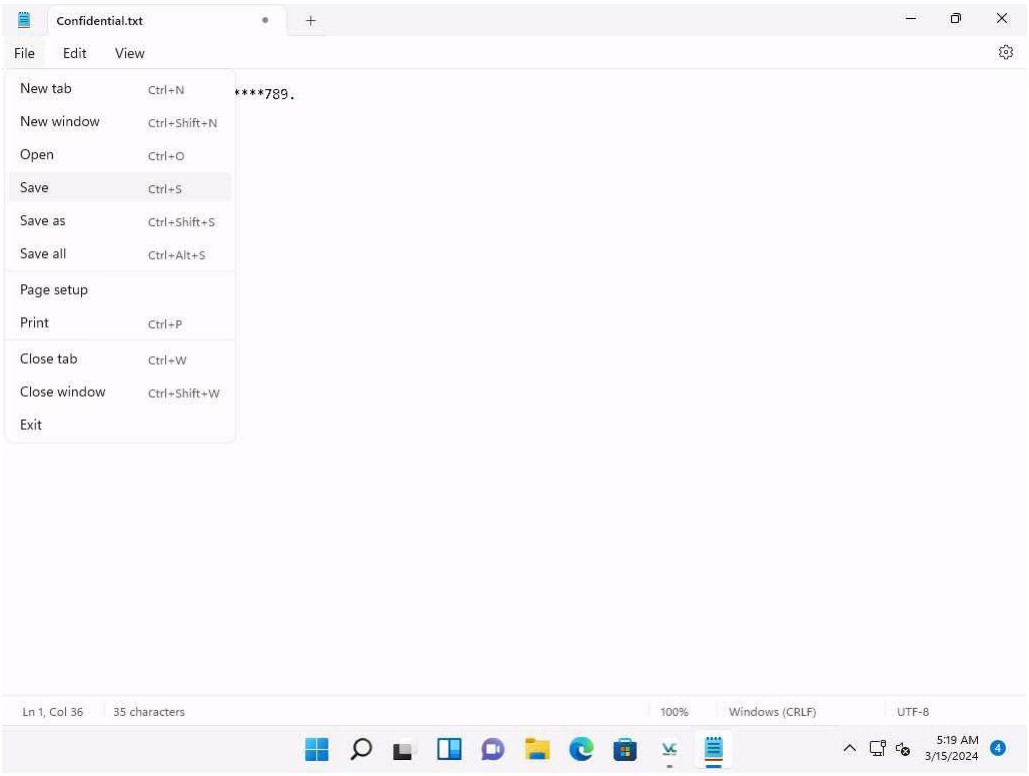
23. After the password is verified, **VeraCrypt** will mount the volume in **I:** drive, as shown in the screenshot.



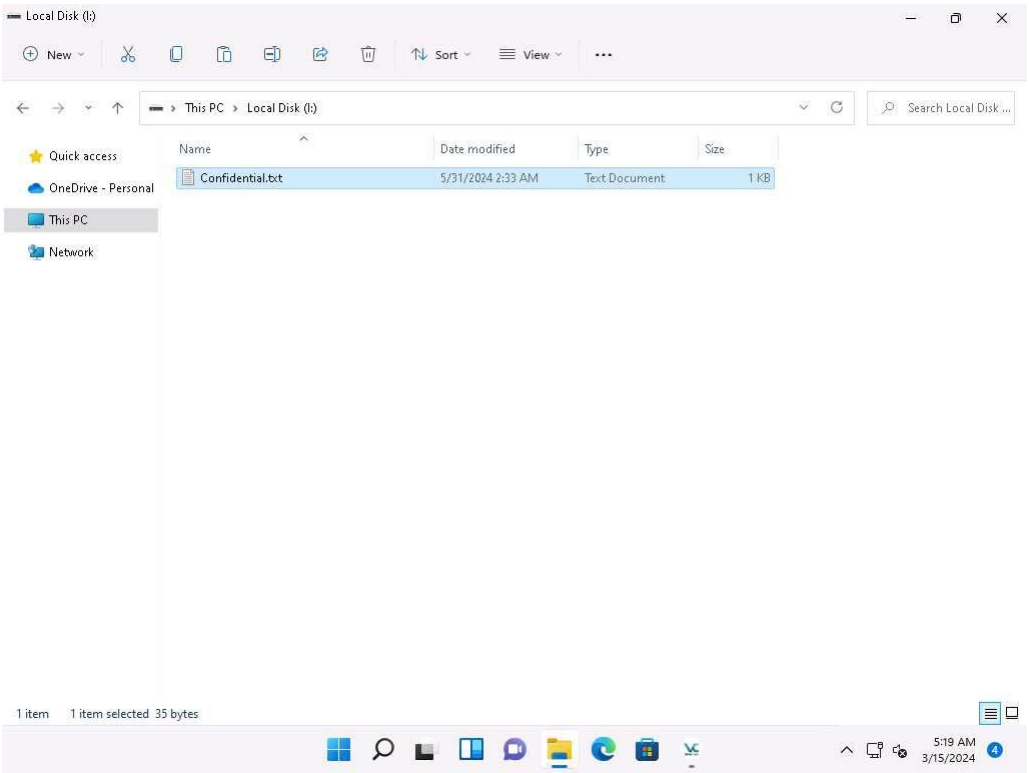
24. **MyVolume** has successfully mounted the container as a virtual disk (**I:**). The virtual disk is entirely encrypted (including file names, allocation tables, free space, etc.) and behaves similarly to a real disk. You can copy or move files to this virtual disk to encrypt them.

25. Create a text file on **Desktop** and name it **Test**. Open the text file and insert text.

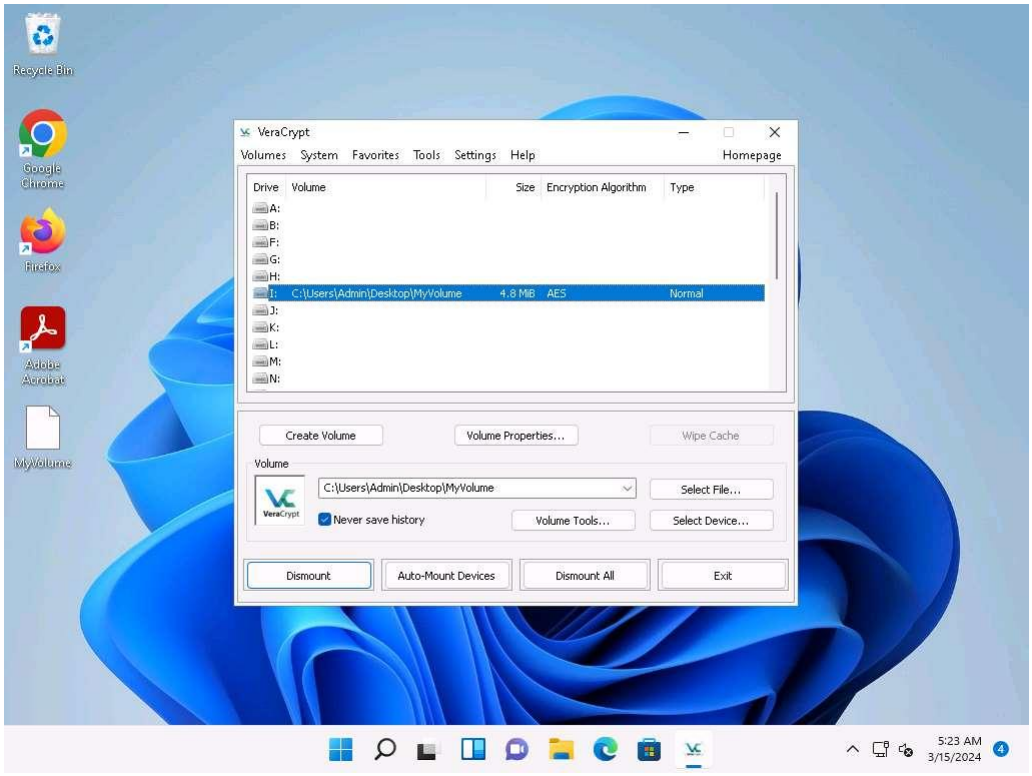
26. Click **File** in the menu bar and click **Save**.



27. Copy the file from **Desktop** and paste it into **Local Disk** (**I:**). Close the window.



28. Switch to the **VeraCrypt** window, click **Dismount**, and then click **Exit**.



29. The **I:** drive located in **This PC** disappears.

    This lab is used to demonstrate that, in cases of system hacks, if an attacker manages to gain remote access or complete access to the machine, he/she will not be able to find the encrypted volume—including its files—unless he/she is able to obtain the password. Thus, all sensitive information located on the encrypted volume is safeguarded.

30. This concludes the demonstration of performing disk encryption using VeraCrypt.

31. Close all open windows and document all the acquired information.

Question 20.3.1.1

Use VeraCrypt to create an encrypted volume. The block size of encryption algorithm used to encrypt the volume is 128-bit block. Name the encryption algorithm used in this task.

Lab 4: Perform Cryptography using AI

Lab Scenario

AI-enhanced cryptography empowers ethical hackers with advanced tools for securing data through complex algorithms and neural networks. It enables robust encryption, decryption, and anomaly detection, ensuring privacy and integrity in digital security practices.

The labs in this exercise demonstrate how you can use AI to perform various cryptographic functions.

Lab Objectives

- Perform cryptographic techniques using ShellGPT

Overview of Cryptography using AI

AI enhances cryptography with advanced algorithms such as AES, RSA, and quantum-resistant methods. Machine learning aids in generating secure keys, detecting anomalies in encrypted data, and optimizing cryptographic protocols. Neural networks refine encryption speed and strength, protecting data from cyber threats. AI-driven cryptography evolves to safeguard sensitive information in an increasingly digital world.

Task 1: Perform Cryptographic Techniques using ShellGPT

ShellGPT augments cryptography through innovative techniques. It leverages shell commands to encrypt data, execute cryptographic operations, and manage key distribution securely. ShellGPT integrates with existing shell environments, enhancing encryption efficiency and reliability. Its adaptability and automation streamline cryptographic processes, fortifying data protection in diverse computing environments.

Here, we will use ShellGPT to perform various cryptographic techniques.

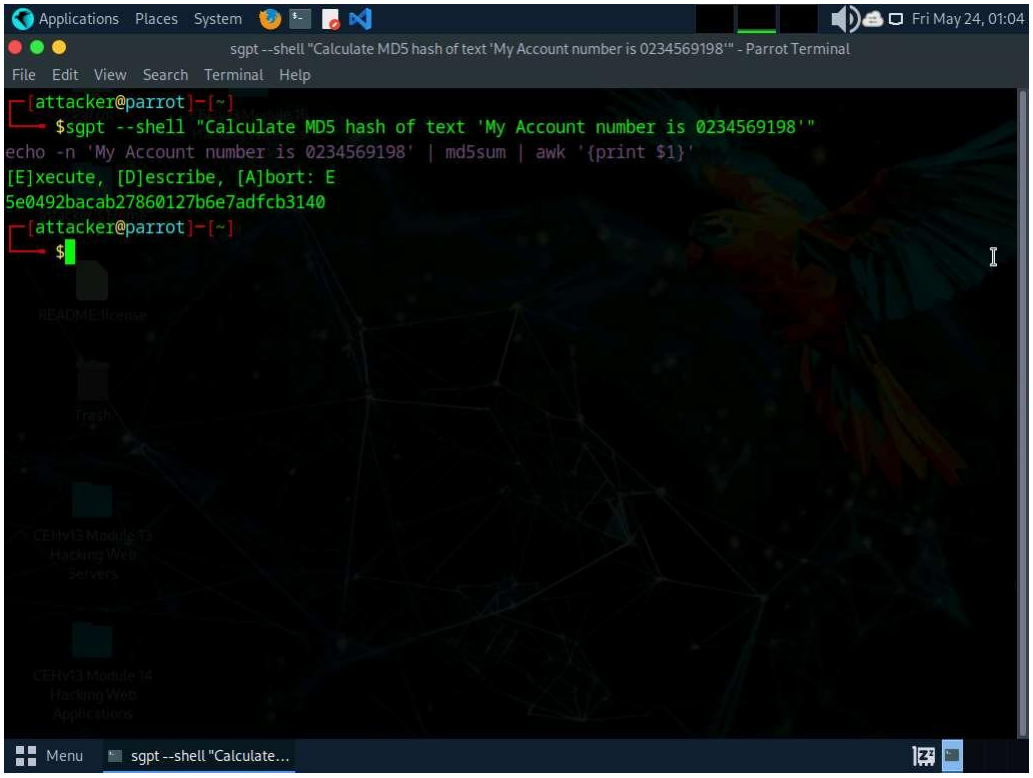
The commands generated by ShellGPT may vary depending on the prompt used and the tools available on the machine. Due to these variables, the output generated by ShellGPT might differ from what is shown in the screenshots. These differences arise from the dynamic nature of the AI's processing and the diverse environments in which it operates. As a result, you may observe differences in command syntax, execution, and results while performing this lab task.

1. Before starting this lab, click Parrot Security to switch to the **Parrot Security** machine and incorporate ShellGPT by following steps provided in Integrate ShellGPT in Parrot Security Machine.pdf.

   Alternatively, you can follow the steps to integrate **ShellGPT** provided in **Module 00: Integrate ShellGPT in Parrot Security Machine**.

2. After incorporating the ShellGPT API in Parrot Security machine, run **sgpt --shell "Calculate MD5 hash of text 'My Account number is 0234569198'"** command to calculate MD5 value of the given text.

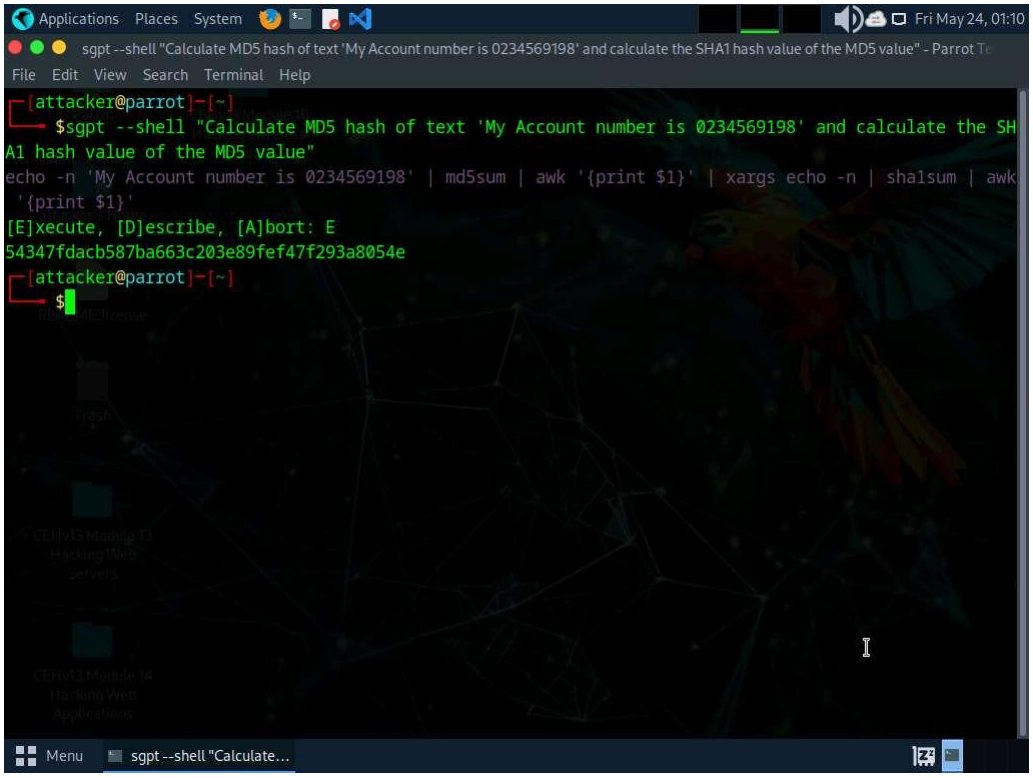
   In the prompt type **E** and press **Enter** to execute the command.



MD5 (Message Digest Algorithm 5) is a widely-used cryptographic hash function producing a 128-bit hash value, typically rendered as a 32-character hexadecimal number. It is commonly used to verify data integrity.

3. Now, we will perform multi-layer hashing using ShellGPT to do so, run **sgpt --shell "Calculate MD5 hash of text 'My Account number is 0234569198' and calculate the SHA1 hash value of the MD5 value"** command.

   In the prompt type **E** and press **Enter** to execute the command.



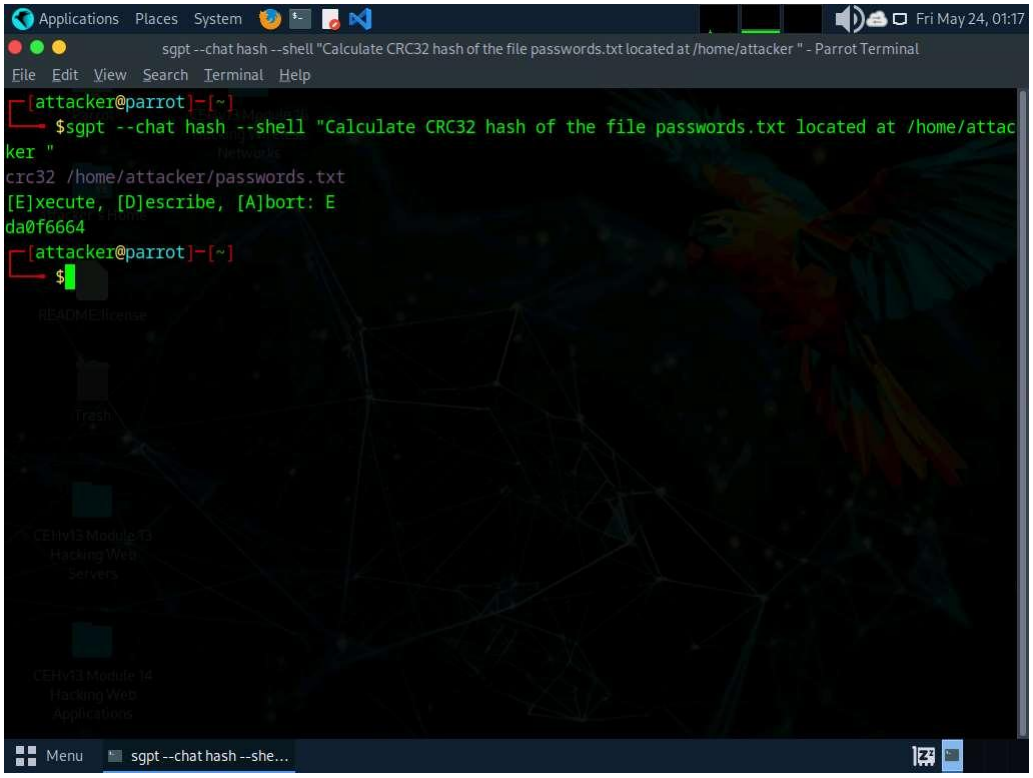
SHA-1 (Secure Hash Algorithm 1) is a cryptographic hash function that generates a 160-bit hash value, often represented as a 40-digit hexadecimal number.

4. Similarly, you can combine various hash functions in the ShellGPT prompt to perform multi-layer hashing.

5. We will now calculate hash of a file using ShellGPT, to do so, run **sgpt --chat hash --shell "Calculate CRC32 hash of the file passwords.txt located at /home/attacker"** command.

   In the prompt type **E** and press **Enter** to execute the command.

   You can use any file and hashing algorithm of your choice.

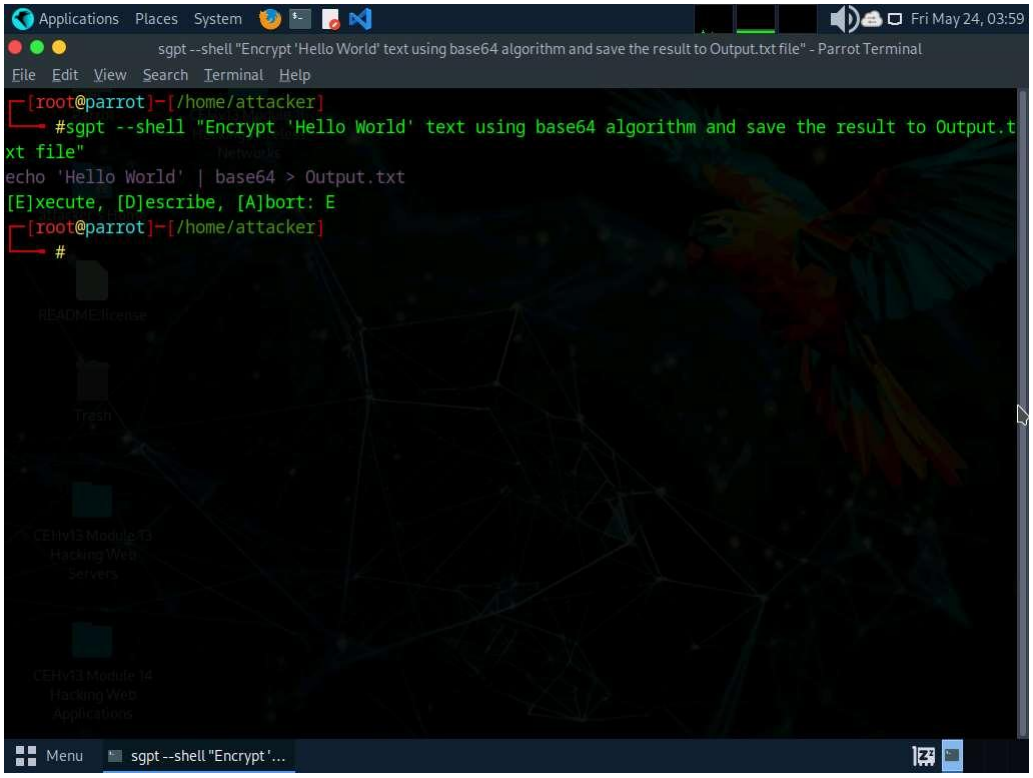


CRC32 (Cyclic Redundancy Check 32) is a widely used hash function to detect accidental changes to raw data. It generates a 32-bit checksum, providing a quick and efficient way to verify data integrity in storage and communication protocols like Ethernet, ZIP files, and various digital formats.

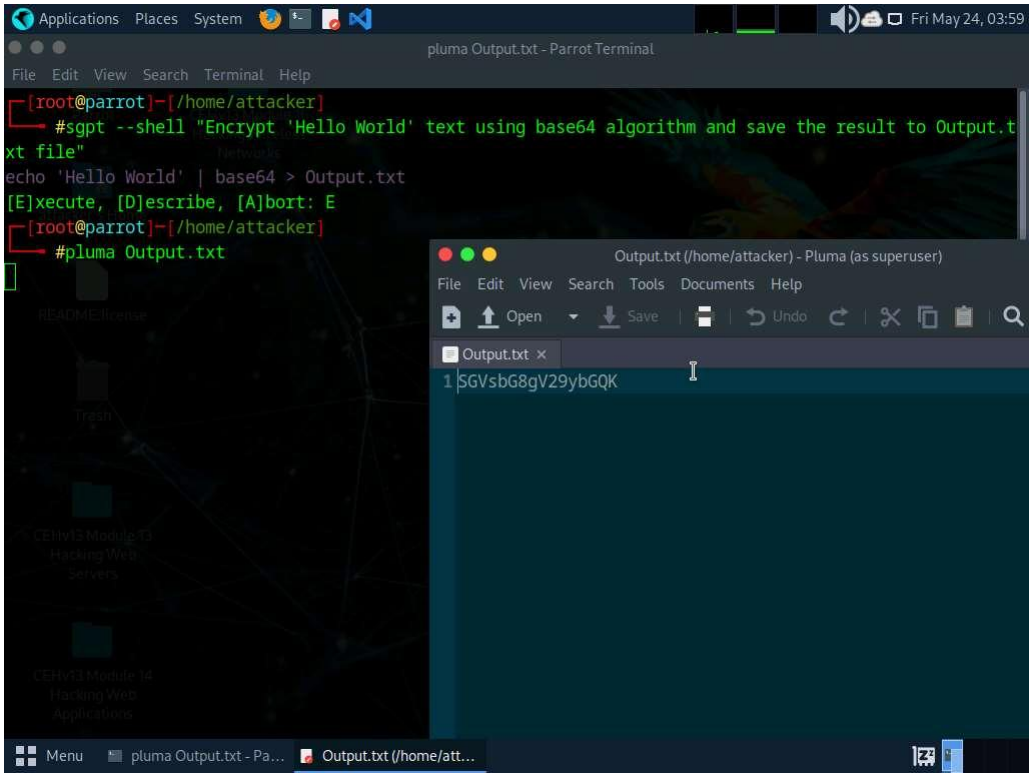
more...

6. To perform basic encryption using ShellGPT run **sgpt --shell "Encrypt 'Hello World' text using base64 algorithm and save the result to Output.txt file"**.

   In the prompt type **E** and press **Enter** to execute the command.



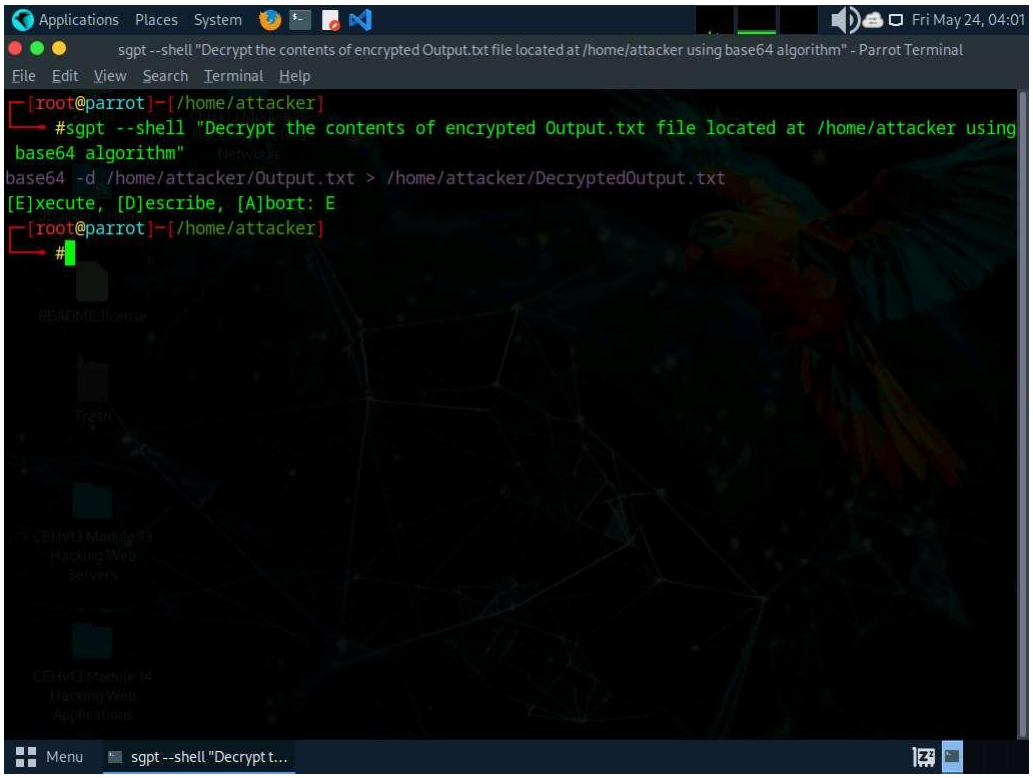
7. In the terminal run **pluma Output.txt** command to view the encrypted data.



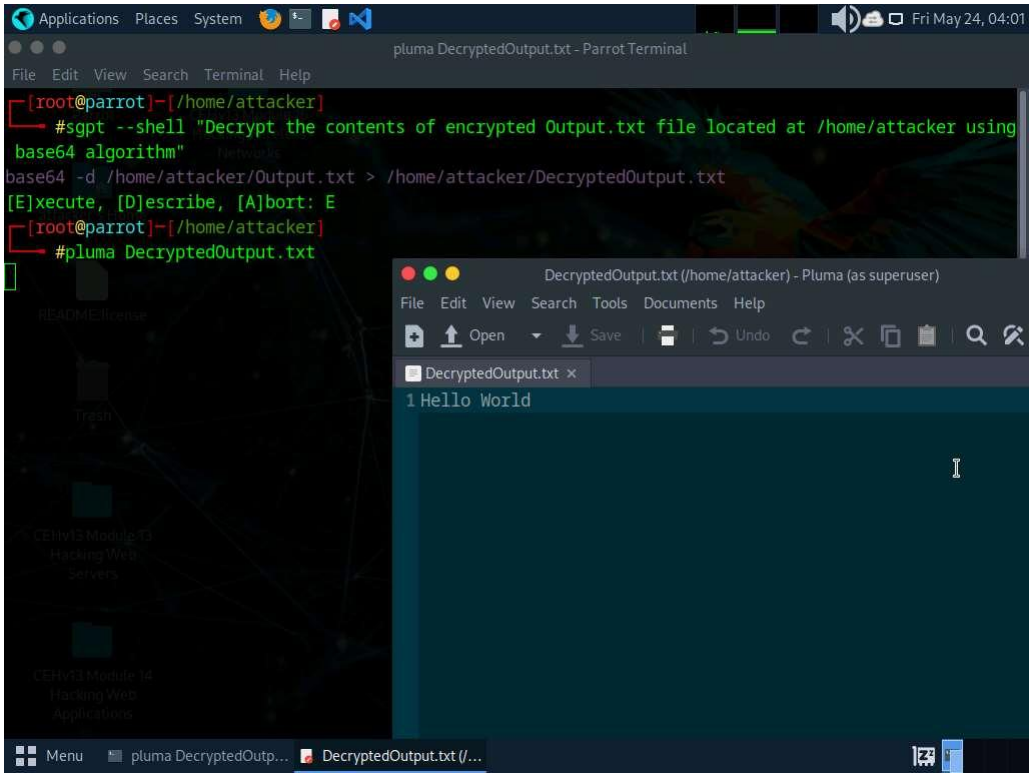
8. Close the text editor window.

9. Now we will decrypt the encrypted data using ShellGPT to do so, run **sgpt --shell "Decrypt the contents of encrypted Output.txt file located at /home/attacker using base64 algorithm"**.

   In the prompt type **E** and press **Enter** to execute the command.



10. In the terminal run **pluma DecryptedOutput.txt** command to view the decrypted data.



11. Close the text editor window.
12. Apart from the aforementioned commands, you can further explore additional options within the ShellGPT tool and utilize various other tools to perform various cryptography techniques.
13. This concludes the demonstration of performing footprinting using the ShellGPT.
14. Close all open windows and document all the acquired information.

Question 20.4.1.1

Use ShellGPT on Parrot Security machine and write a prompt to encrypt Hello World text using base64 algorithm. Enter the base64 encoded form of Hello World.