CySA+ Lab Series

Lab 18: Securing Data Using Encryption Software

Document Version: 2022-10-10

Material in this Lab Aligns to the Following	
CompTIA CySA+ (CS0-002) Exam Objectives	2.1 - Explain software assurance best practices 5.1 - Understand the importance of data privacy and protection
All-In-One CompTIA CySA+ Second Edition ISBN-13: 978-1260464306 Chapters	8: Security Solutions for Infrastructure Management 19: The Importance of Data Privacy and Protection

Contents

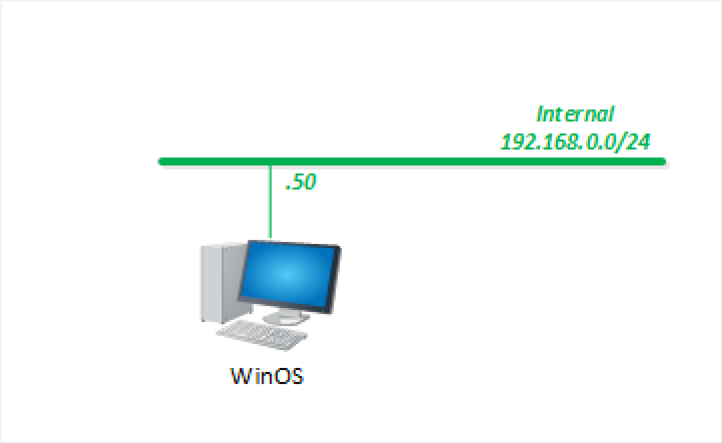
Introduction

Encryption is one of the best ways to protect an organization’s data whether in transit over a network or stored in files. Tools, such as VPNs, and protocols like SSH and SSL are used to encrypt data in transit, and in this lab, cybersecurity analysts will use the *VeraCrypt* tool for file and folder encryption and will then work with attacks against the encryption folder.

Objectives

- Creating a *VeraCrypt* Container
- Opening and Viewing Data within a *VeraCrypt* Container
- Crack the password on a *VeraCrypt* Container

Lab Topology



Lab Settings

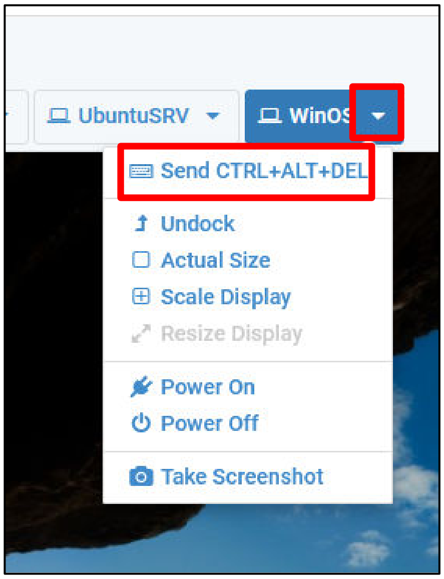
The information in the table below will be needed in order to complete the lab. The task sections below provide details on the use of this information.

Virtual Machine	IP Address	Account	Password
WinOS (Server 2019)	192.168.0.50	Administrator	NDGlabpass123!
MintOS (Linux Mint)	192.168.0.60	sysadmin	NDGlabpass123!
OSSIM (AlienVault)	172.16.1.2	root	NDGlabpass123!
UbuntuSRV (Ubuntu Server)	172.16.1.10	sysadmin	NDGlabpass123!
Kali	203.0.113.2	sysadmin	NDGlabpass123!
pfSense	203.0.113.1 172.16.1.1 192.168.0.1	admin	NDGlabpass123!

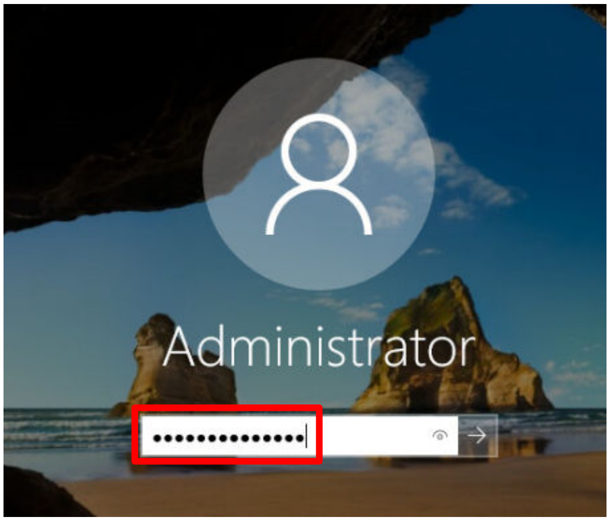
1 Creating a VeraCrypt Container

VeraCrypt is a utility that performs "On-the-Fly-Encryption" which is a transparent encryption technique that allows files and folders to be encrypted. It creates a file that is configured as a virtual encrypted disk. The file is mounted and used just like a regular disk, but any folders and files contained in the file will be encrypted. *VeraCrypt* is very versatile, using a variety of ciphers, such as *AES*, *Serpent,* and *Twofish,* and hash algorithms, such as *SHA-256*, *SHA-512,* and *RIPEMD*.

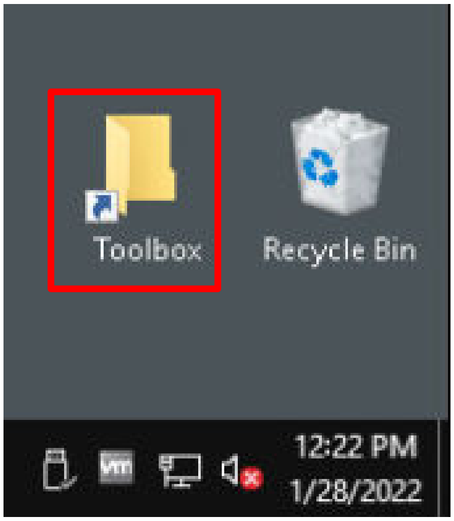
1. Set the focus on the **WinOS** computer.
2. Bring up the login window by sending a Ctrl + Alt + Delete. To do this, click the **WinOS** dropdown menu and click **Send CTRL+ALT+DEL**.



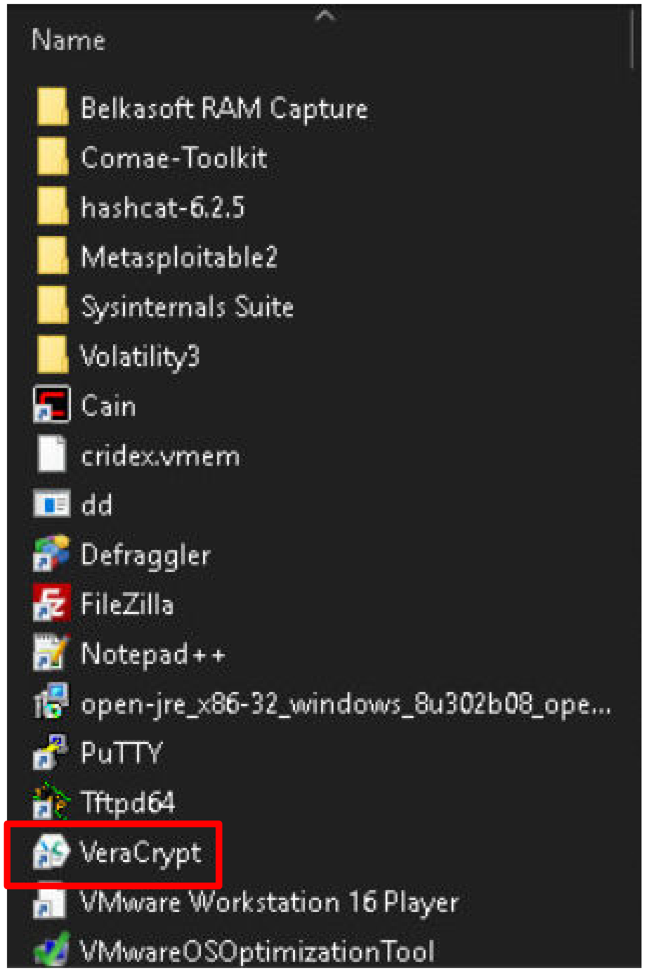
3. Log in as *Administrator* using the password: `NDGlabpass123!`



4. Double-click the **Toolbox** directory on the desktop to open the folder:



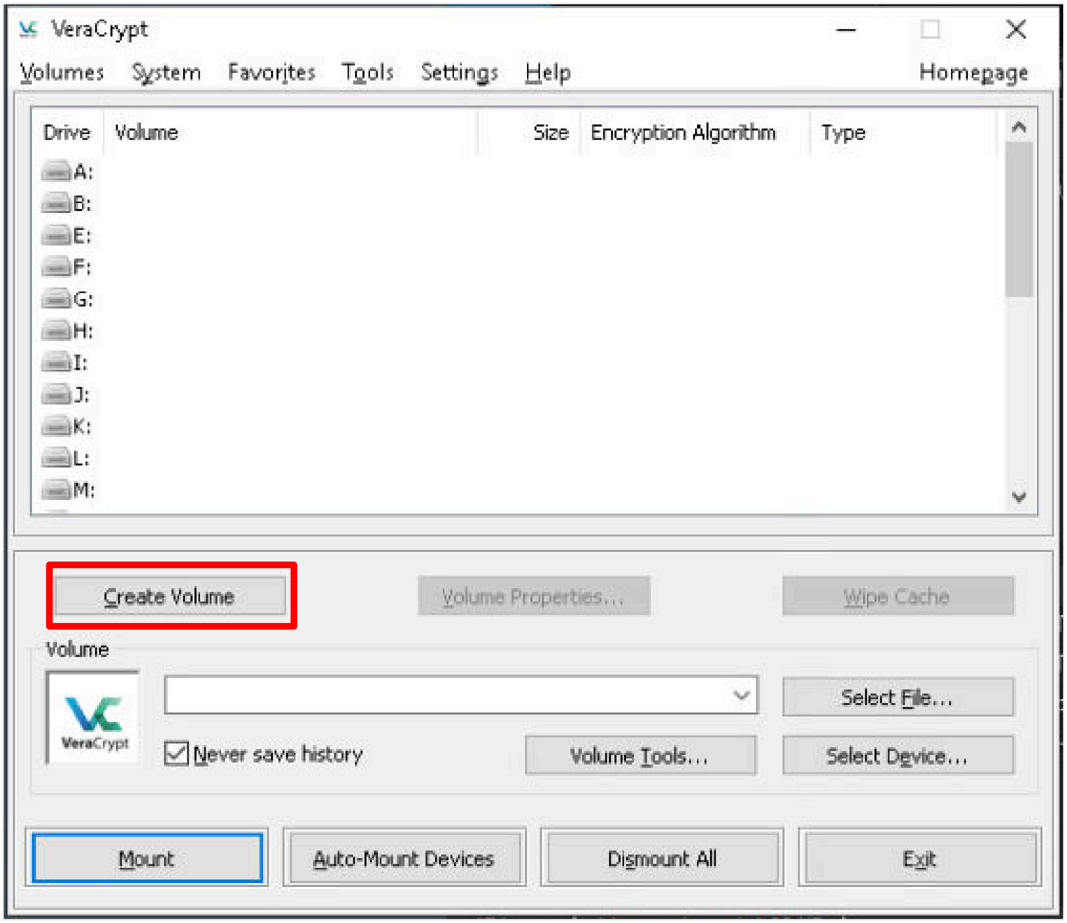
5. Double-click on **VeraCrypt.**



Name

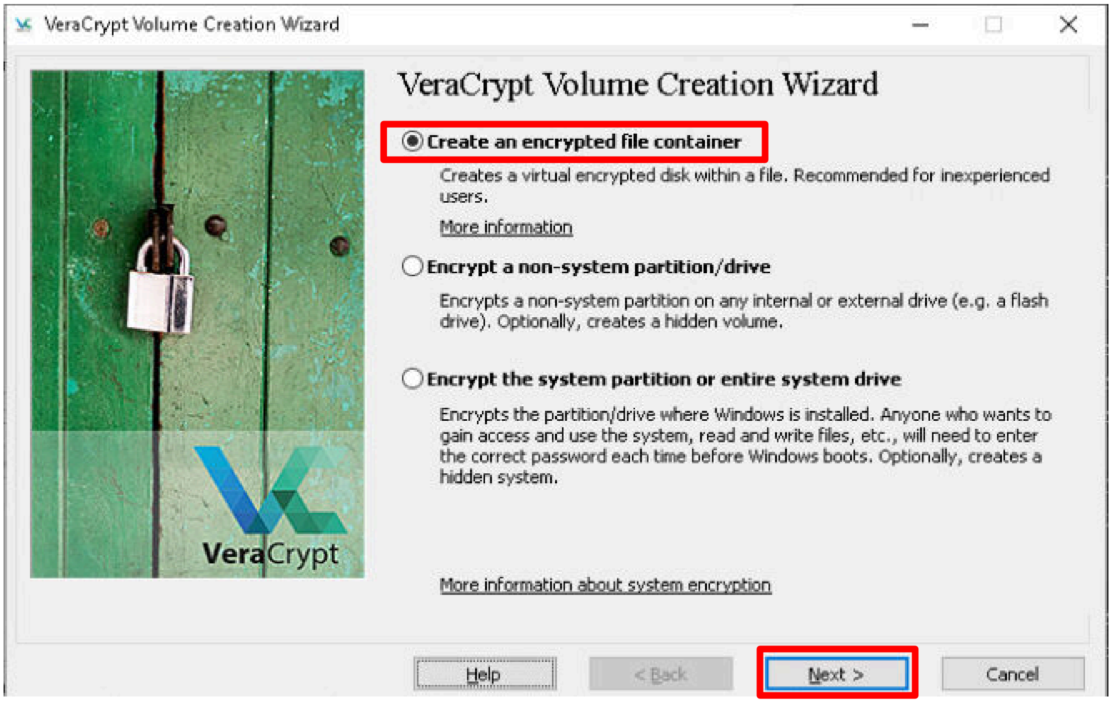
Belkasoft RAM Capture
Comae-Toolkit
hashcat-6.2.5
Metasploitable2
Sysinternals Suite
Volatility3
Cain
cridex.vmem
dd
Defraggler
FileZilla
Notepad++
open-jre_x86-32_windows_8u302b08_ope...
PuTTY
Tftpd64
VeraCrypt
VMware Workstation 16 Player
VMwareOSOptimizationTool

6. On the *VeraCrypt* window, click the **Create Volume** button to begin the *Volume Creation Wizard.*



☑ Never save history

7. Proceed with the wizard to create an encrypted file container by clicking the **Create an Encrypted File Container** radio button and then clicking the **Next** button.



VeraCrypt Volume Creation Wizard

● Create an encrypted file container

Creates a virtual encrypted disk within a file. Recommended for inexperienced users.

More information

○ Encrypt a non-system partition/drive

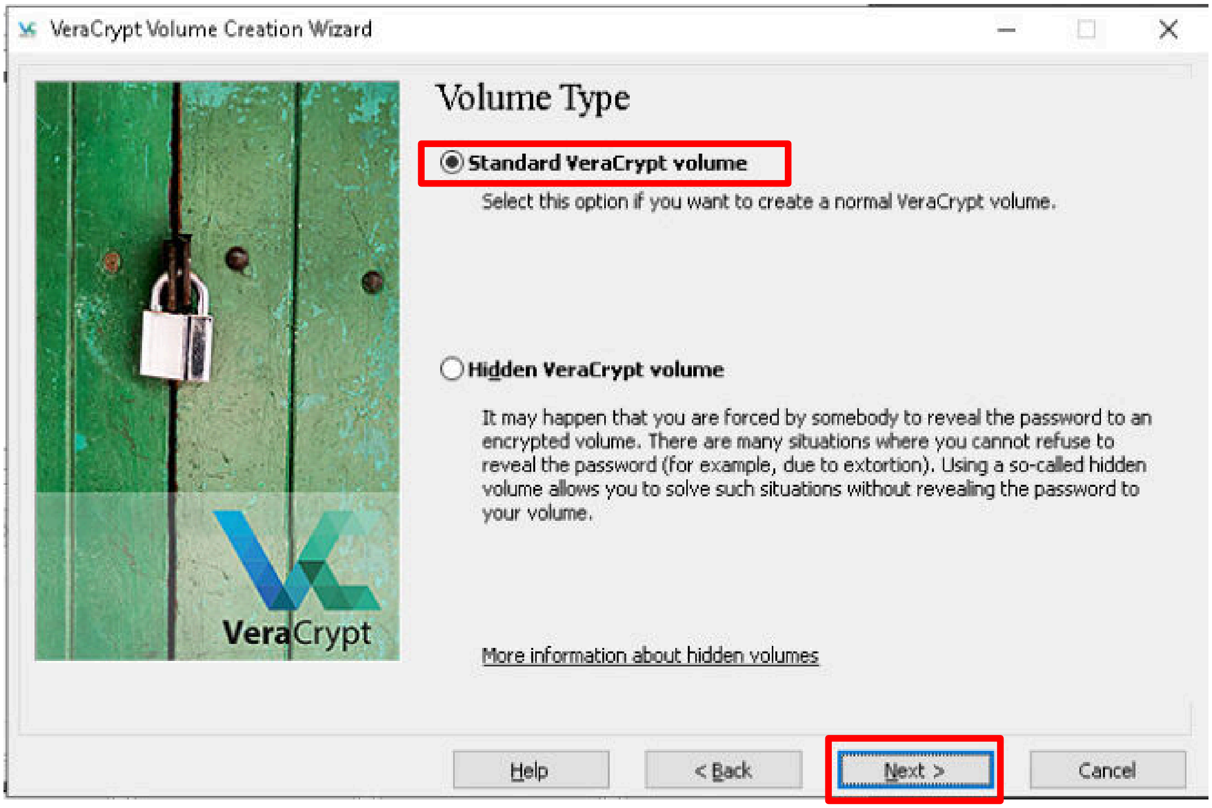
Encrypts a non-system partition on any internal or external drive (e.g. a flash drive). Optionally, creates a hidden volume.

○ Encrypt the system partition or entire system drive

Encrypts the partition/drive where Windows is installed. Anyone who wants to gain access and use the system, read and write files, etc., will need to enter the correct password each time before Windows boots. Optionally, creates a hidden system.

More information about system encryption

8. On the *Volume Type* screen, make sure the **Standard VeraCrypt Volume** radio button is selected and click the **Next** button.



VeraCrypt Volume Creation Wizard

Volume Type

● Standard VeraCrypt volume

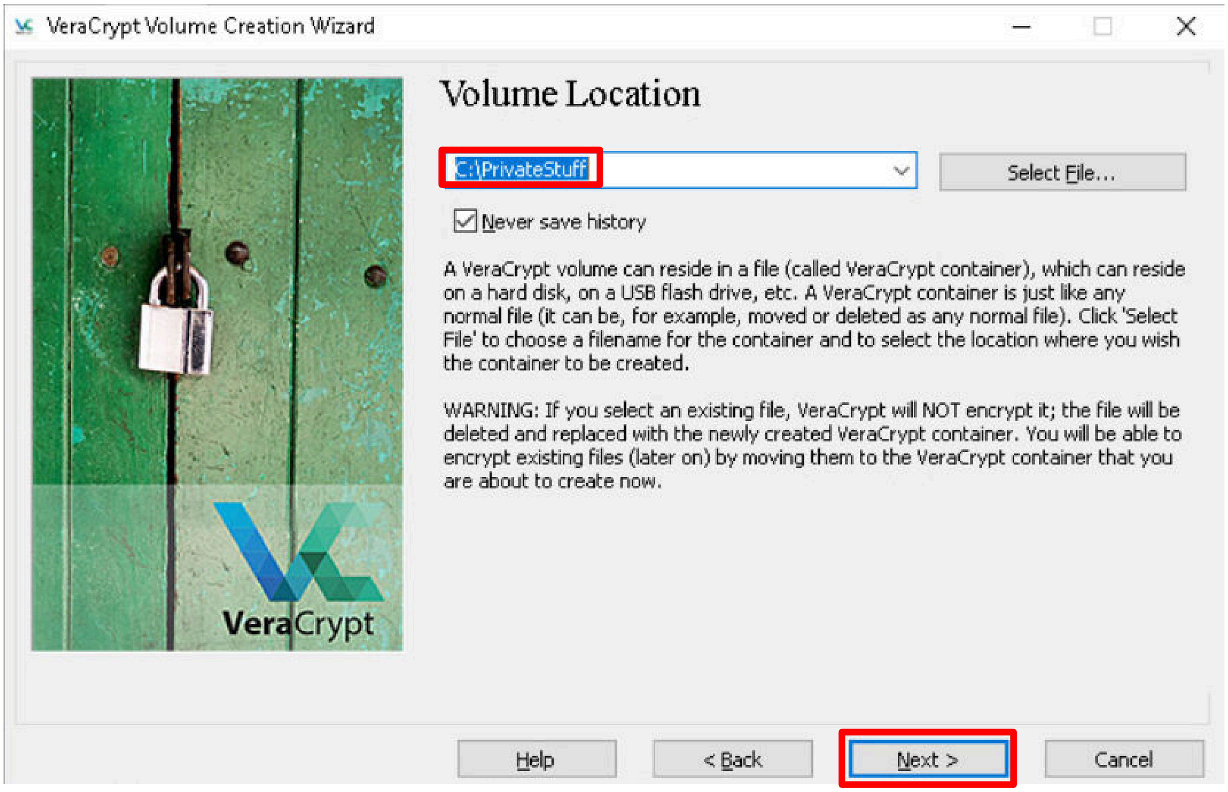
Select this option if you want to create a normal VeraCrypt volume.

○ Hidden VeraCrypt volume

It may happen that you are forced by somebody to reveal the password to an encrypted volume. There are many situations where you cannot refuse to reveal the password (for example, due to extortion). Using a so-called hidden volume allows you to solve such situations without revealing the password to your volume.

More information about hidden volumes

9. On the **Volume Location screen**, type **C:\PrivateStuff** to create the *VeraCrypt* container and click the **Next** button.



VeraCrypt Volume Creation Wizard

Volume Location

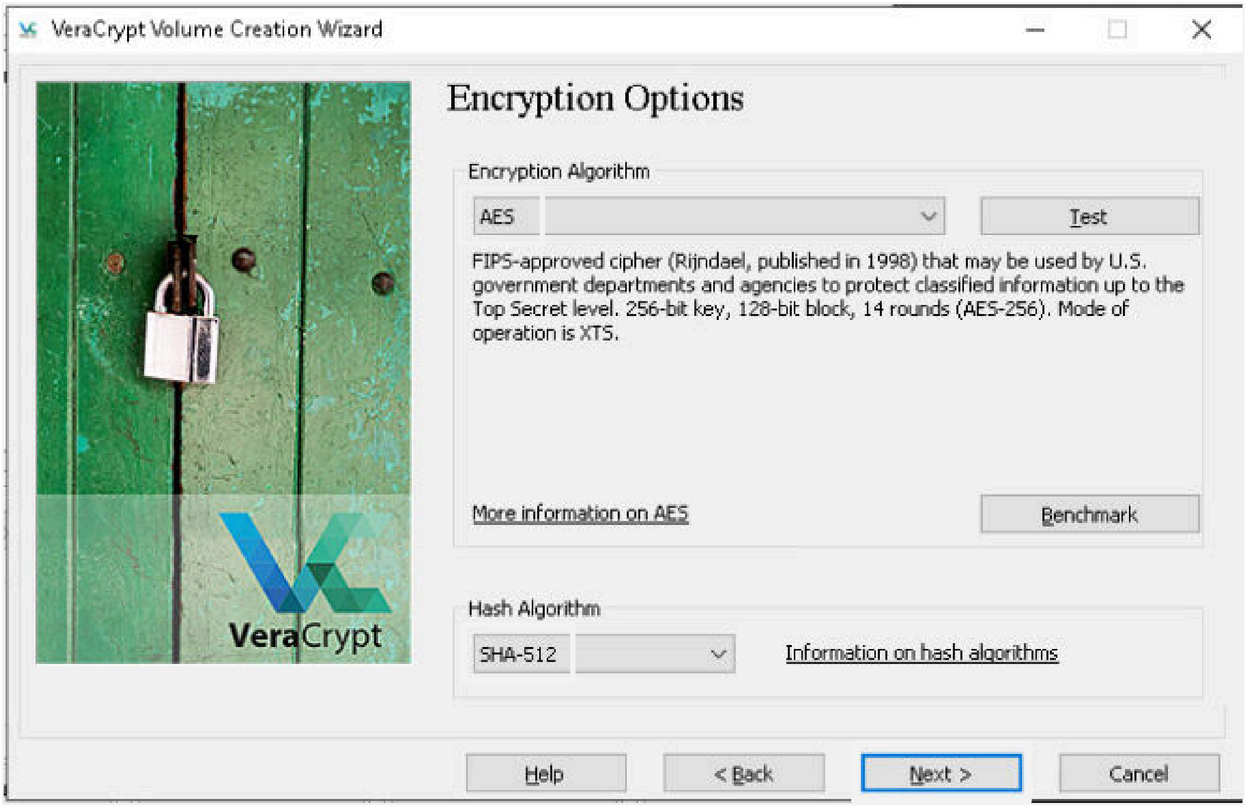
C:\PrivateStuff

☑ Never save history

A VeraCrypt volume can reside in a file (called VeraCrypt container), which can reside on a hard disk, on a USB flash drive, etc. A VeraCrypt container is just like any normal file (it can be, for example, moved or deleted as any normal file). Click 'Select File' to choose a filename for the container and to select the location where you wish the container to be created.

WARNING: If you select an existing file, VeraCrypt will NOT encrypt it; the file will be deleted and replaced with the newly created VeraCrypt container. You will be able to encrypt existing files (later on) by moving them to the VeraCrypt container that you are about to create now.

10. On the *Encryption Options* screen, leave the *Encryption Algorithm* as **AES** and the *Hash Algorithm* as **SHA-512** and click **Next**.



Encryption Options

Encryption Algorithm: AES

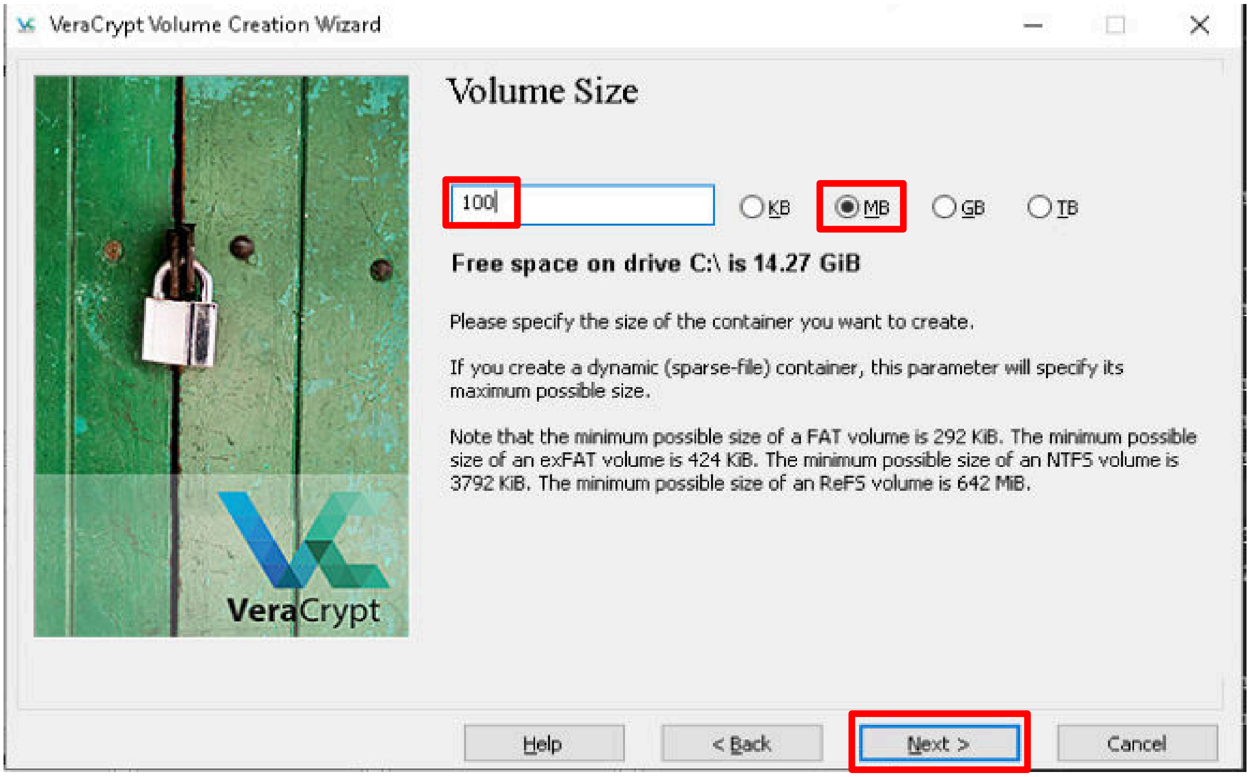
FIPS-approved cipher (Rijndael, published in 1998) that may be used by U.S. government departments and agencies to protect classified information up to the Top Secret level. 256-bit key, 128-bit block, 14 rounds (AES-256). Mode of operation is XTS.

More information on AES

Hash Algorithm: SHA-512

Information on hash algorithms

11. On the *Volume Size* window, type the value of `100` and make sure the **MB** radio button is selected, then click **Next**.



Volume Size

100 ○ KB ● MB ○ GB ○ TB

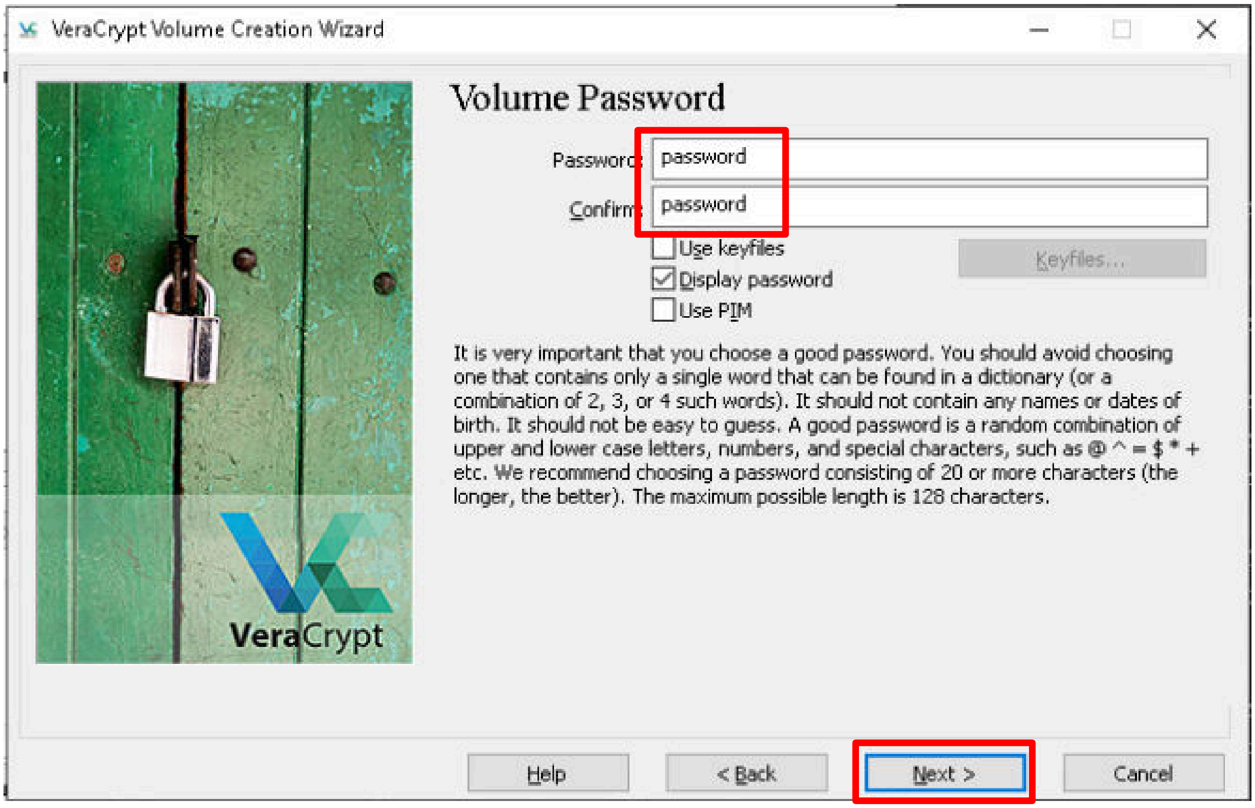
Free space on drive C:\ is 14.27 GiB

Please specify the size of the container you want to create.

If you create a dynamic (sparse-file) container, this parameter will specify its maximum possible size.

Note that the minimum possible size of a FAT volume is 292 KiB. The minimum possible size of an exFAT volume is 424 KiB. The minimum possible size of an NTFS volume is 3792 KiB. The minimum possible size of an ReFS volume is 642 MiB.

12. On the *Volume Password* window, enter `password` for the *password* and confirm with `password`**.** Click the **Display password** checkbox to confirm the spelling and click **Next**.



VeraCrypt Volume Creation Wizard

Volume Password

Password: password

Confirm: password

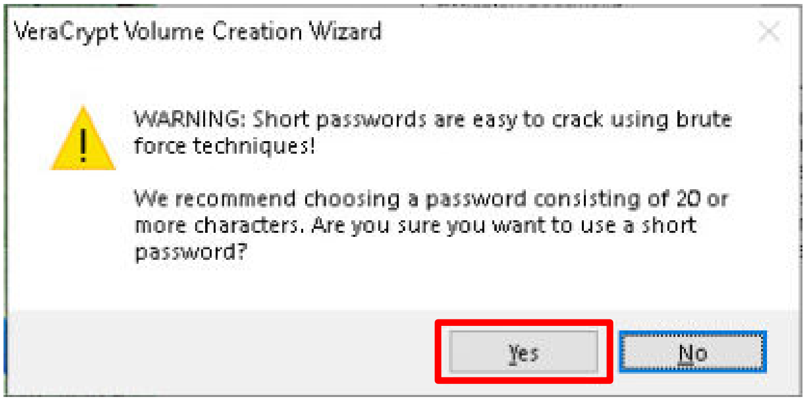
□ Use keyfiles

☑ Display password

□ Use PIM

It is very important that you choose a good password. You should avoid choosing one that contains only a single word that can be found in a dictionary (or a combination of 2, 3, or 4 such words). It should not contain any names or dates of birth. It should not be easy to guess. A good password is a random combination of upper and lower case letters, numbers, and special characters, such as @ ^ = $ * + etc. We recommend choosing a password consisting of 20 or more characters (the longer, the better). The maximum possible length is 128 characters.

13. On the *Warning Message* window, click on **Yes**.

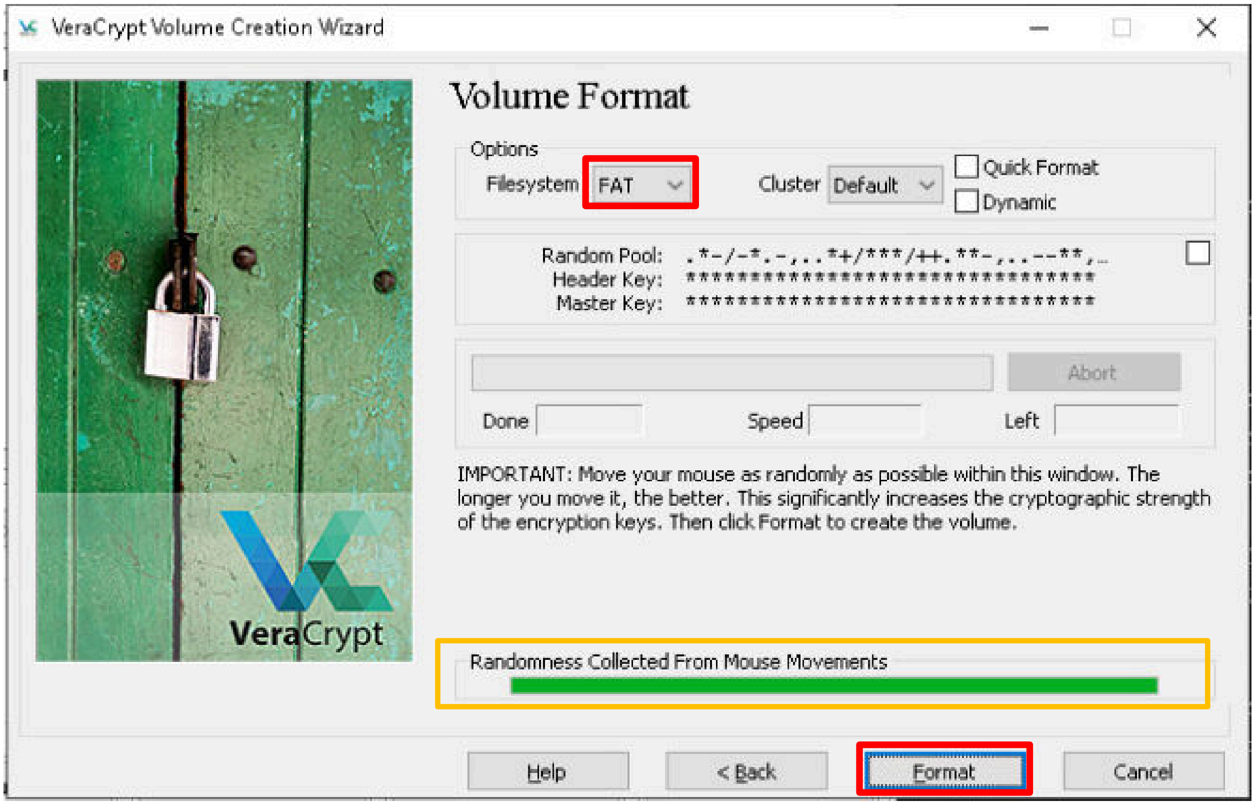


VeraCrypt Volume Creation Wizard

WARNING: Short passwords are easy to crack using brute force techniques!

We recommend choosing a password consisting of 20 or more characters. Are you sure you want to use a short password?

14. On the *Volume Format* window, leave the default *Filesystem* as **FAT,** then start moving the mouse around randomly to increase the strength of the encryption key. Watch the **progress bar** as it travels to the right. When you get to the far-right side, click on the **Format** button:



VeraCrypt Volume Creation Wizard

Volume Format

Options

Filesystem FAT Cluster Default ☐ Quick Format ☐ Dynamic

Random Pool: .*-/-*.-,..*+/***/++.**-,..--**,..
Header Key: ********************************
Master Key: ********************************

Done Speed Left

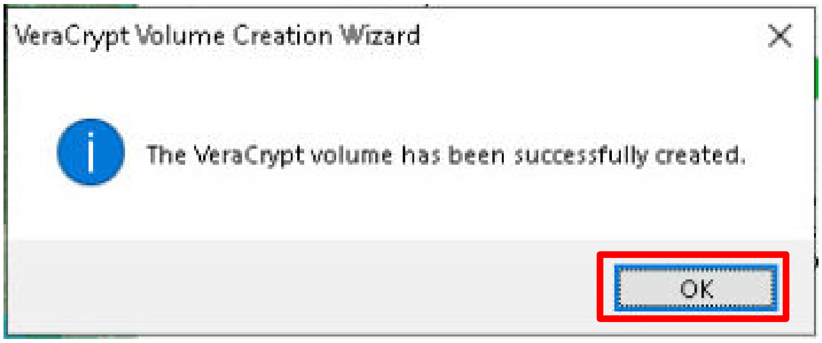
IMPORTANT: Move your mouse as randomly as possible within this window. The longer you move it, the better. This significantly increases the cryptographic strength of the encryption keys. Then click Format to create the volume.

Randomness Collected From Mouse Movements

The choice of file system is important when considering in which operating systems the *VeraCrypt* container will reside.

FAT is the most compatible with Windows, Linux and Mac, but files must be less than 4GB. **exFAT** allows for files over 4GB and very large partitions.

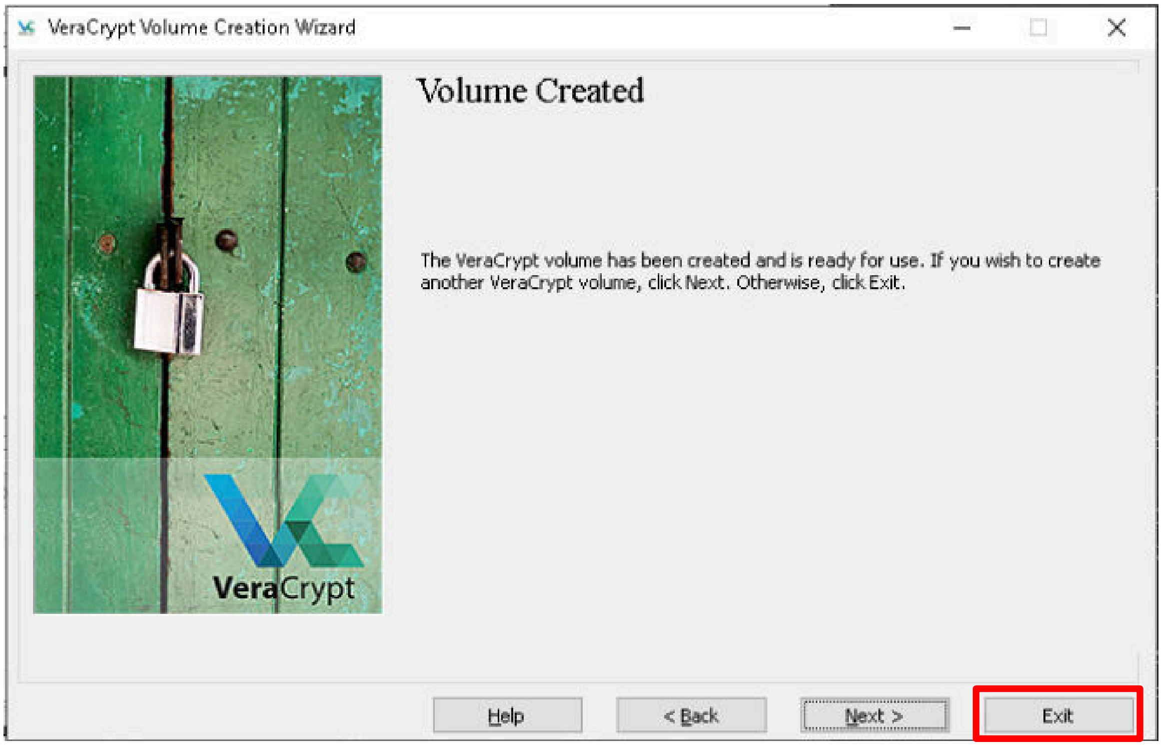
15. Click on **OK** in response to the message, *The VeraCrypt volume has been successfully created*.



VeraCrypt Volume Creation Wizard

The VeraCrypt volume has been successfully created.

16. On the *Volume Created* window, click the **Exit** button.



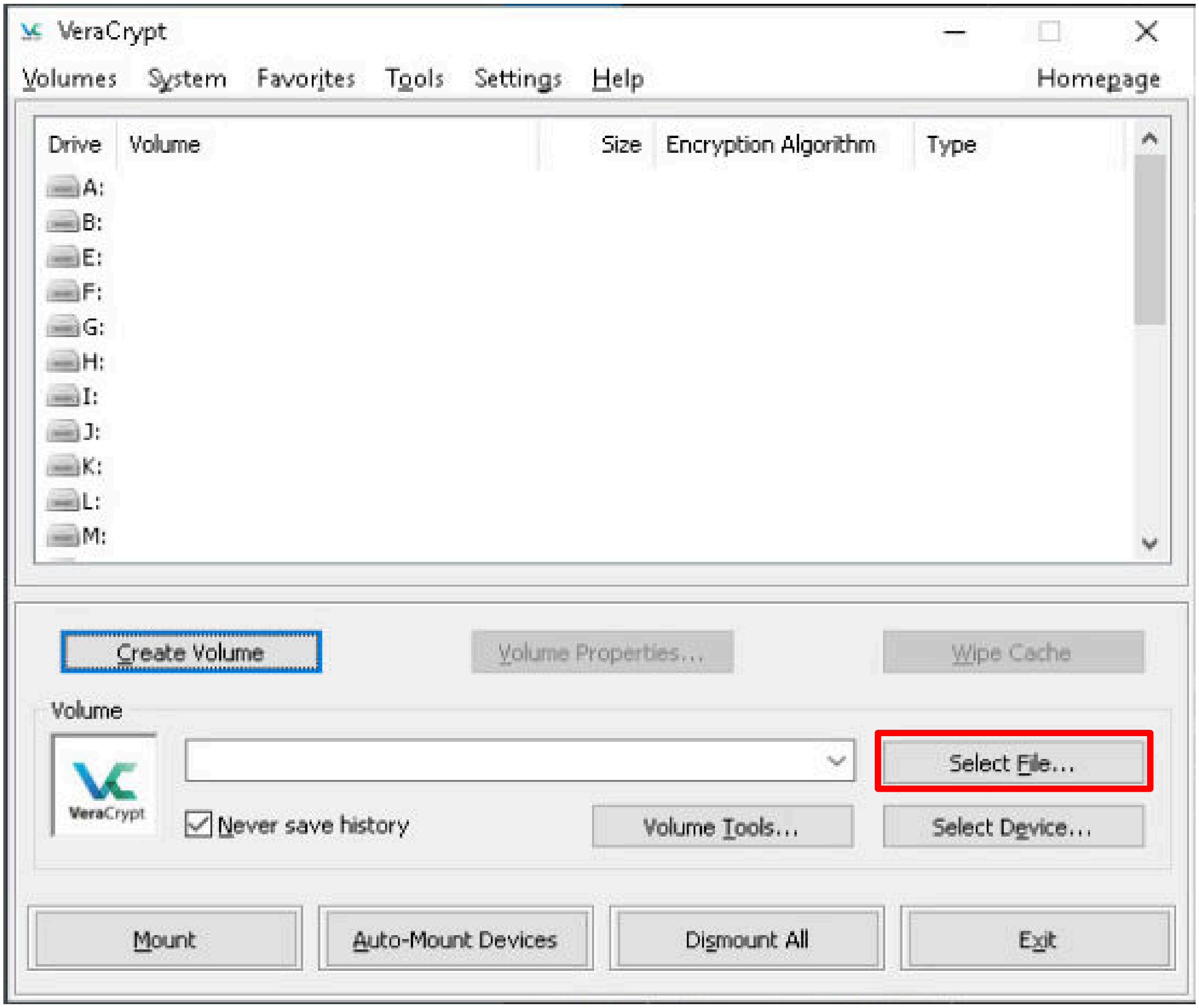
Volume Created

The VeraCrypt volume has been created and is ready for use. If you wish to create another VeraCrypt volume, click Next. Otherwise, click Exit.

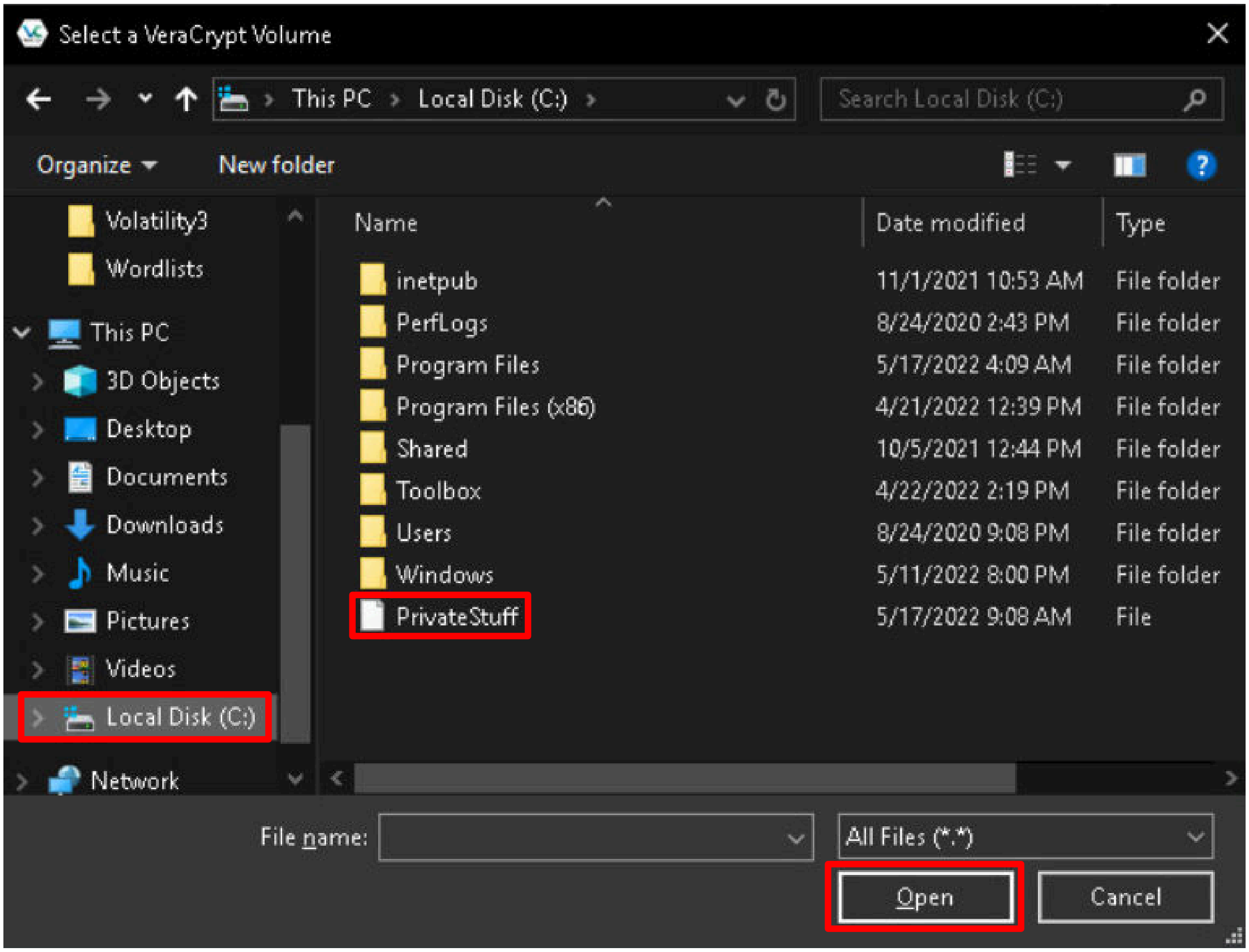
17. Leave the *VeraCrypt* application open and continue to the next section.

2 Opening and Viewing Data Within a VeraCrypt Container

1. Click on the **Select File** button to locate your *VeraCrypt* container.

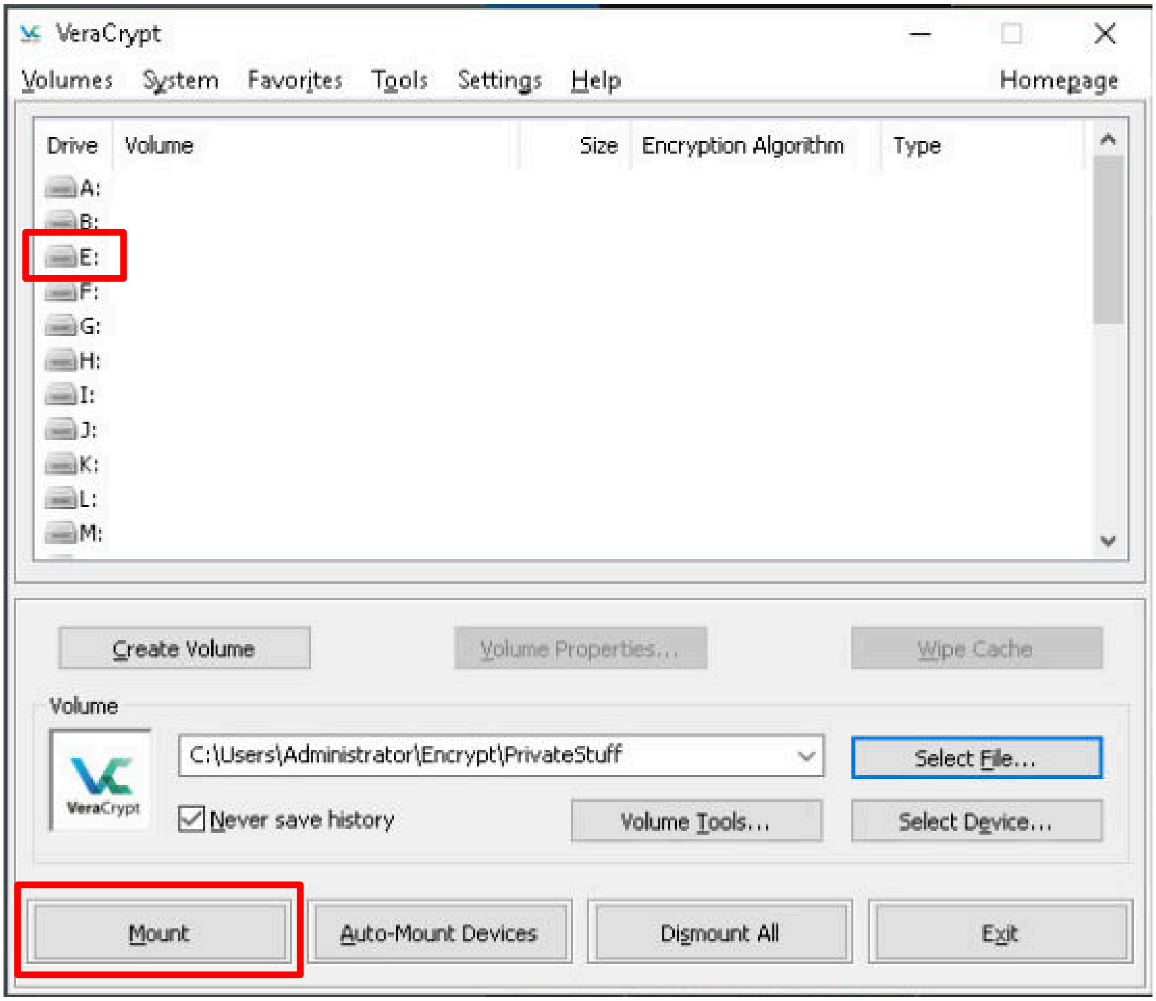


2. In the *Select a VeraCrypt Volume* window, click on **Local Disk (C:)** on the left pane and the **PrivateStuff** file on the right pane, then click the **Open** button.



Name	Date modified	Type
inetpub	11/1/2021 10:53 AM	File folder
PerfLogs	8/24/2020 2:43 PM	File folder
Program Files	5/17/2022 4:09 AM	File folder
Program Files (x86)	4/21/2022 12:39 PM	File folder
Shared	10/5/2021 12:44 PM	File folder
Toolbox	4/22/2022 2:19 PM	File folder
Users	8/24/2020 9:08 PM	File folder
Windows	5/11/2022 8:00 PM	File folder
PrivateStuff	5/17/2022 9:08 AM	File

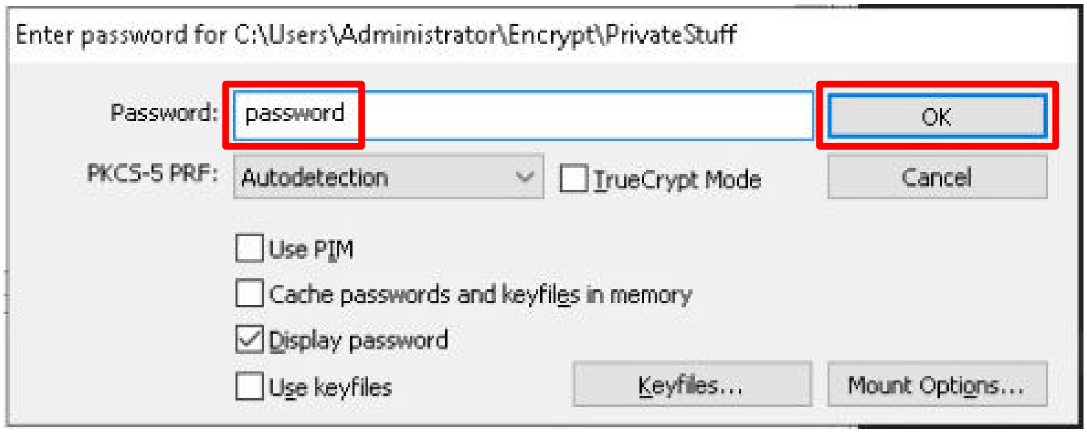
3. Select the **E:** Drive from the list and click the **Mount** Button.



C:\Users\Administrator\Encrypt\PrivateStuff

☑ Never save history

4. When prompted for a password, type `password` and click the **OK** button.



Enter password for C:\Users\Administrator\Encrypt\PrivateStuff

Password: password

PKCS-5 PRF: Autodetection

□ TrueCrypt Mode

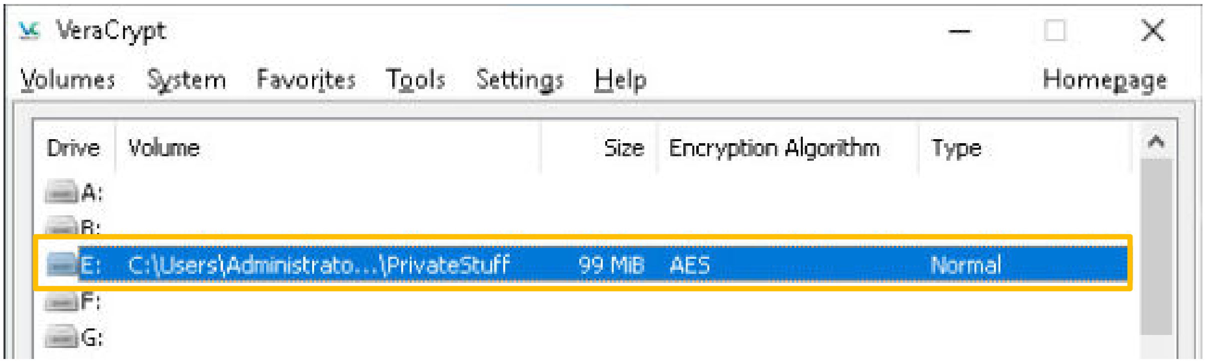
□ Use PIM

□ Cache passwords and keyfiles in memory

☑ Display password

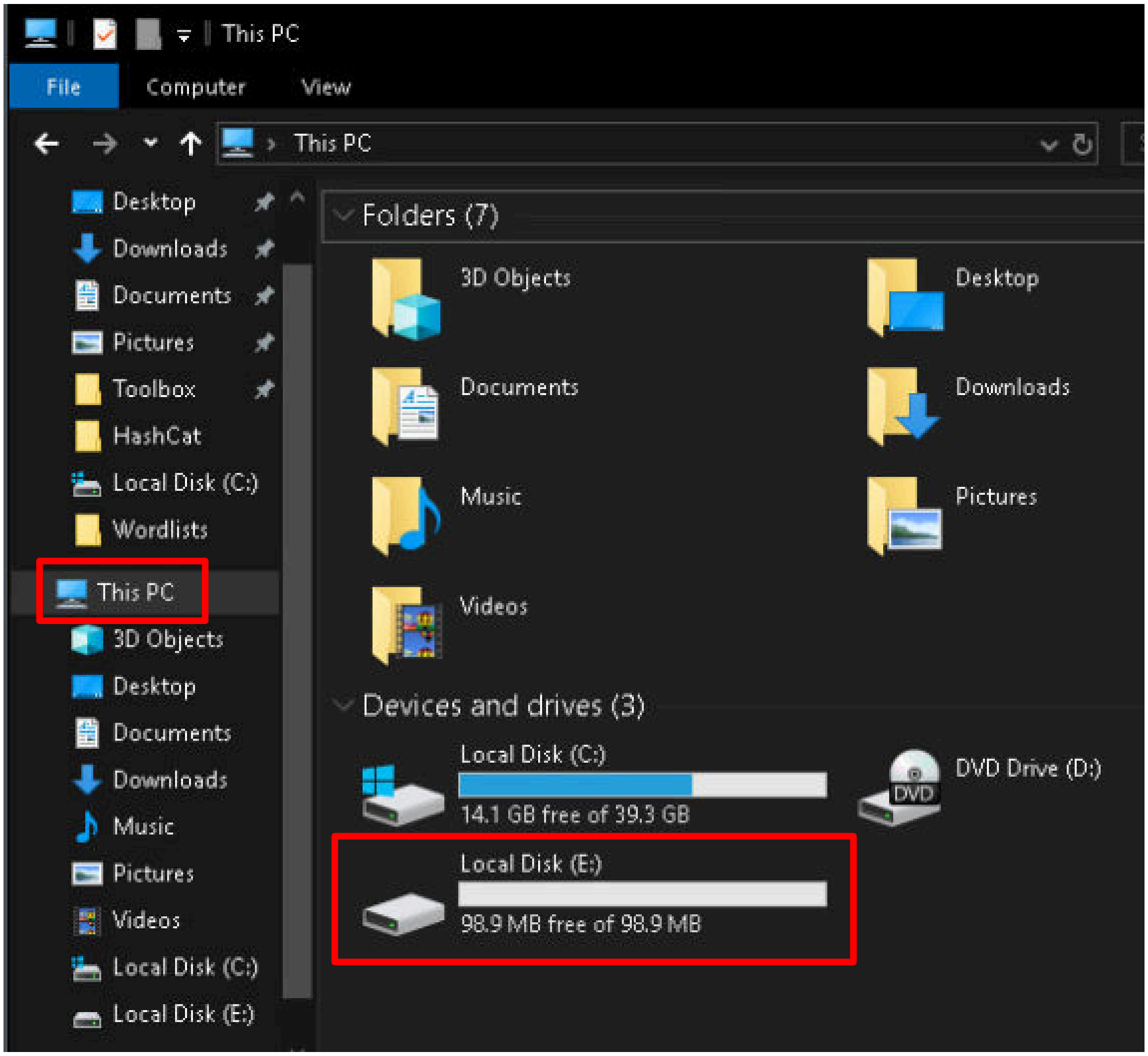
□ Use keyfiles

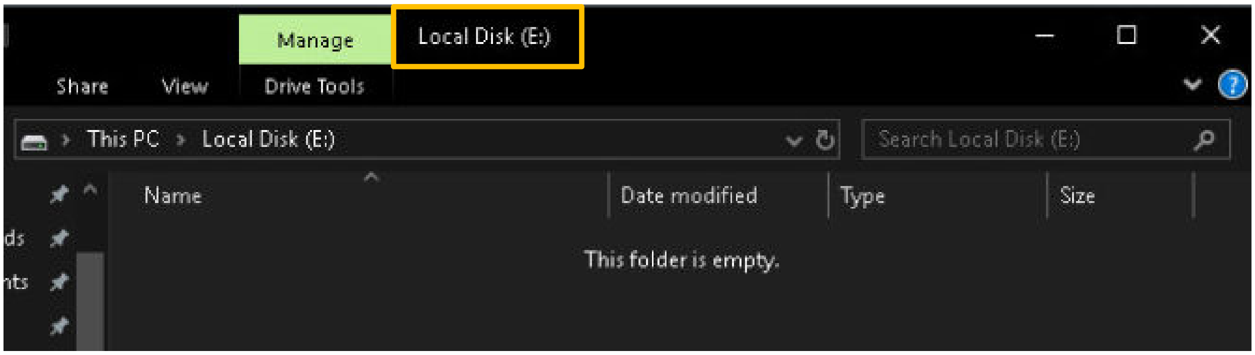
5. The container is now mounted as a **Volume** when looking at the *VeraCrypt* window.



Drive	Volume	Size	Encryption Algorithm	Type
A:				
B:				
E:	C:\Users\Administrato...\PrivateStuff	99 MiB	AES	Normal
F:				
G:				

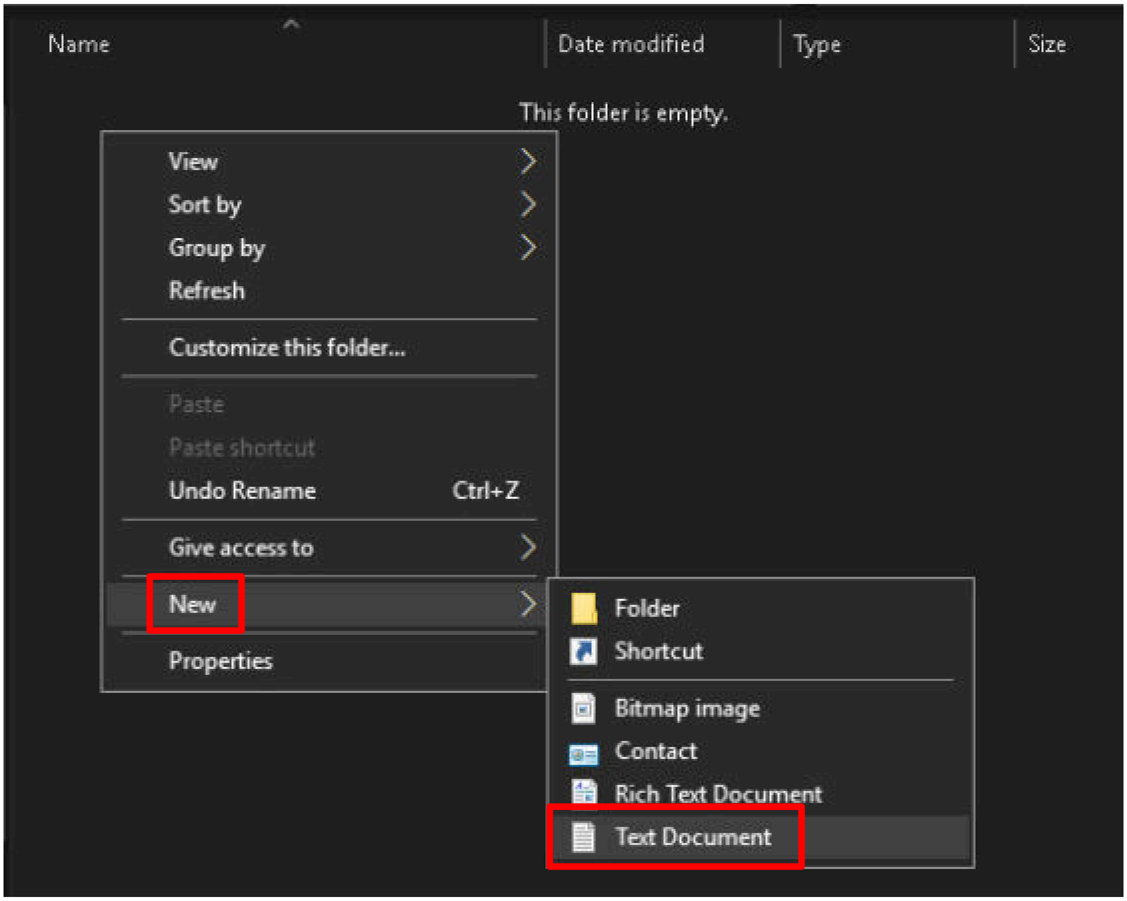
6. Minimize the **VeraCrypt** window.
7. Set the focus back to the *File Explorer* window. Click on the **This PC** icon in the left pane, and on the right pane under *Device and Drives,* double-click on the **Local Disk (E:)** icon.



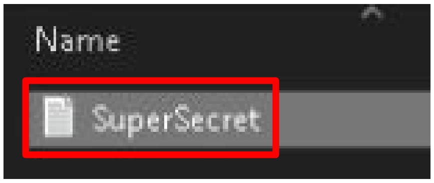


The *VeraCrypt* volume is displayed as a drive letter. When the volume is unmounted, no one will be able to see the files stored within the container unless they successfully mount the volume with the correct password.

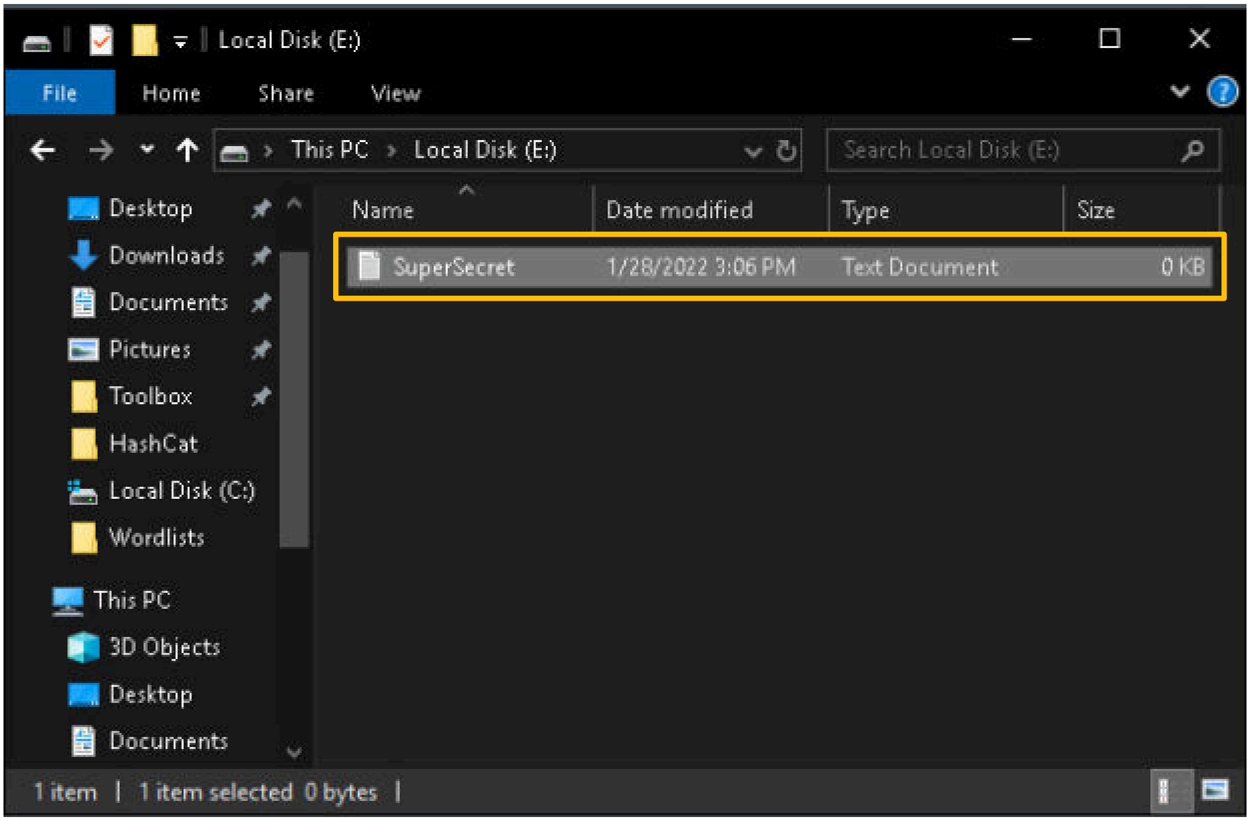
8. In the *File Explorer* window, right-click on the empty space in the right pane and click on **New,** then click on **Text Document.**



9. Name the file `SuperSecret` and press **Enter**.



10. You should now see the *SuperSecret* file in the *File Explorer* window.

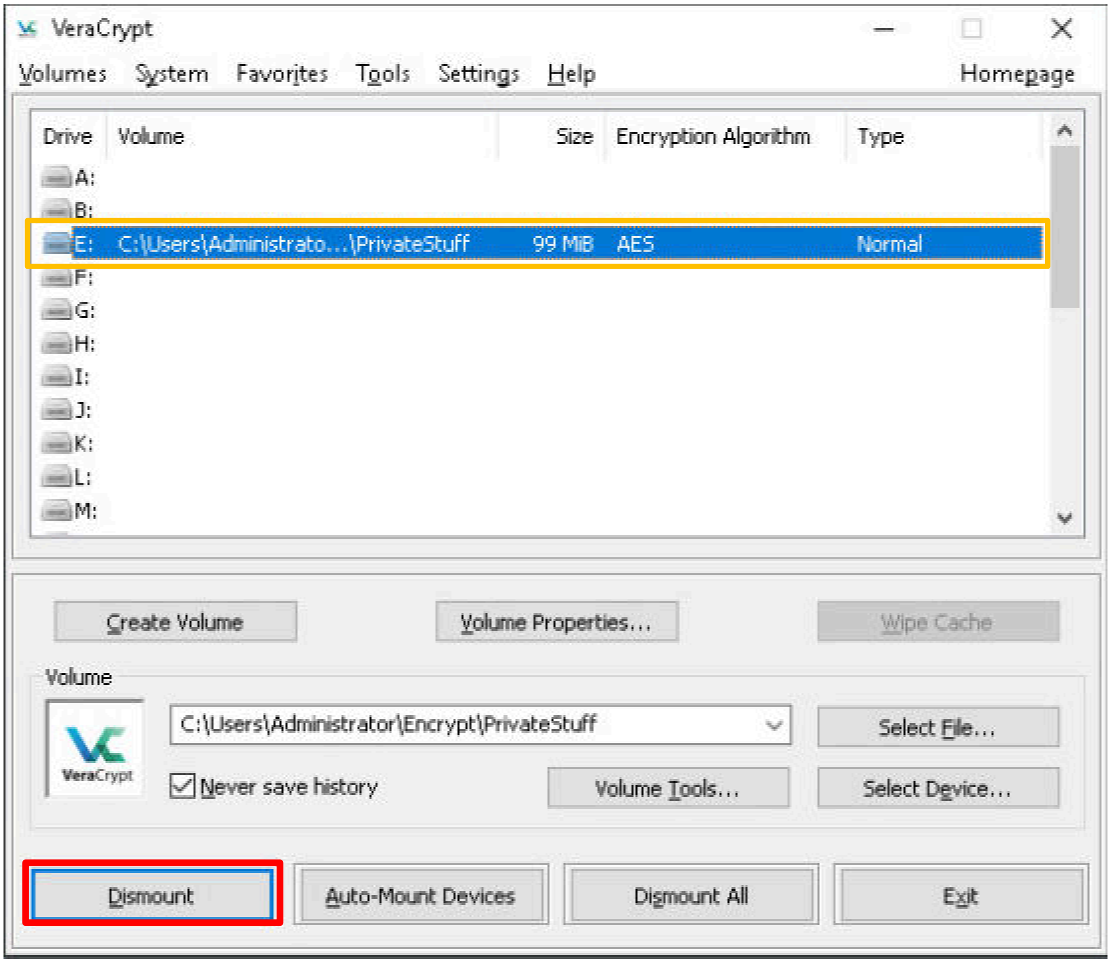


Name	Date modified	Type	Size
SuperSecret	1/28/2022 3:06 PM	Text Document	0 KB

1 item | 1 item selected 0 bytes |

11. Close the *File Explorer* window.

12. Restore the *VeraCrypt* application window; make sure the volume in **Drive E:** is selected, and click the **Dismount** button to unmount the *VeraCrypt* volume.



Drive	Volume	Size	Encryption Algorithm	Type
A:				
B:				
E:	C:\Users\Administrato...\PrivateStuff	99 MiB	AES	Normal
F:				
G:				
H:				
I:				
J:				
K:				
L:				
M:				

Volume

C:\Users\Administrator\Encrypt\PrivateStuff

☑ Never save history

13. Close the *VeraCrypt* window by clicking on the **Exit** button. Remain on the *WinOS* computer and continue to the next section.

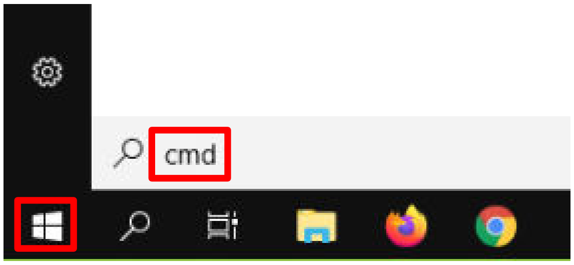
3 Pentest a VeraCrypt Container's Password Strength Using a Dictionary Attack

A very important task of the security analyst is *Pentesting* (Penetration Testing) with the goal of assessing security weaknesses that need to be strengthened. In this particular case, the strength of the password used to mount the *VeraCrypt* container will be tested. For the pentest, *HashCat* will be used.

A more complete description of the hash extraction rules can be found in the FAQ section of the HashCat wiki at"

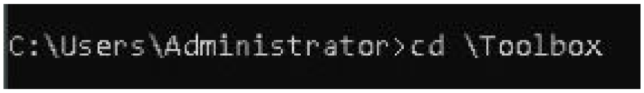
https://hashcat.net/wiki/doku.php?id=frequently_asked_questions under "**How do you extract the hashes from TrueCrypt volumes?**"

1. Open the *Command Prompt* window by clicking on the **Windows Start Button** in the bottom-left corner, type `cmd` and then press the **Enter** key to bring up the command prompt window.



2. At the *Command Prompt*, type the following command to access the *dd* utility directory.

```
cd \Toolbox
```

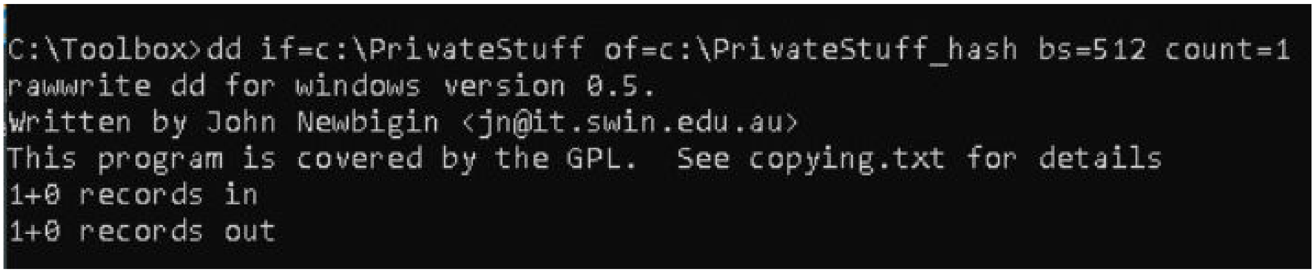


```
C:\Users\Administrator>cd \Toolbox
```

3. In order to crack the password, *HashCat* needs the binary of the file. There are several different types of *VeraCrypt* container files, and each one has a different location for the password's hash. In the case of the *VeraCrypt* container that you created earlier, you need the first **512 bytes** of the file that is used as the container.

   To extract the first **512 bytes** of the file, you will use the *dd* utility, which can copy and convert portions of a file. In the terminal, type the following command to extract the first **512 bytes** of the container file and save the resulting data to a new file which will be used by *HashCat* to crack.

```
dd if=c:\PrivateStuff of=c:\PrivateStuff_hash bs=512 count=1
```

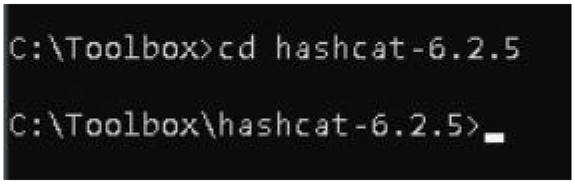


```
C:\Toolbox>dd if=c:\PrivateStuff of=c:\PrivateStuff_hash bs=512 count=1
rawwrite dd for windows version 0.5.
Written by John Newbigin <jn@it.swin.edu.au>
This program is covered by the GPL.  See copying.txt for details
1+0 records in
1+0 records out
```

Here's the command breakdown:
if= the input file, which is the VeraCrypt container
of= the output file, which is the file containing the 512 byte hash
bs= the number of bytes to be read from the input file
count= the number of input blocks to copy

4. At the *Command Prompt*, change to the *HashCat* utility directory by typing the following command:

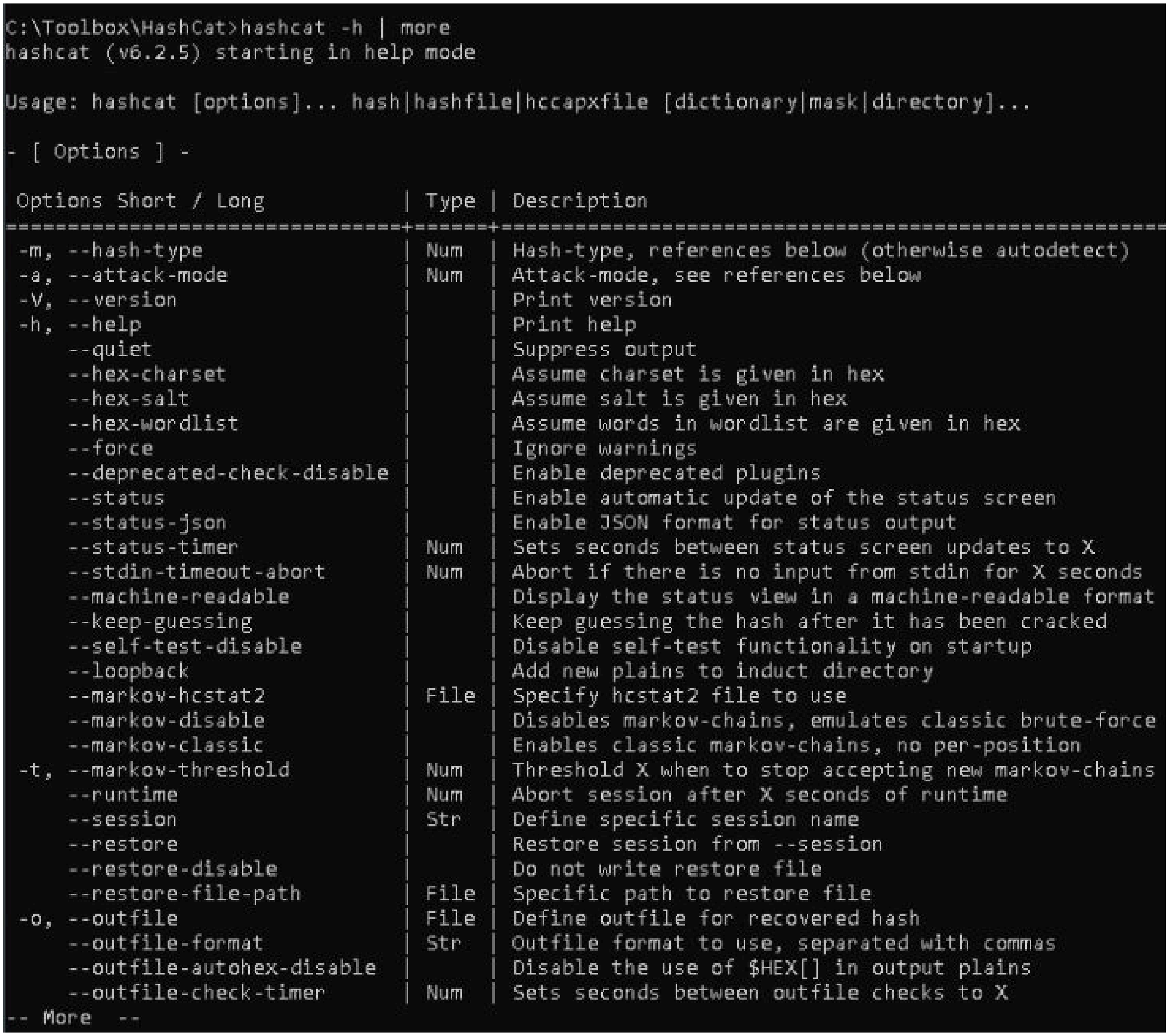
```
cd hashcat-6.2.5
```



```
C:\Toolbox>cd hashcat-6.2.5

C:\Toolbox\hashcat-6.2.5>_
```

5. Type the following command to display the help for *HashCat*:

```
hashcat -h | more
```



```
C:\Toolbox\HashCat>hashcat -h | more
hashcat (v6.2.5) starting in help mode

Usage: hashcat [options]... hash|hashfile|hccapxfile [dictionary|mask|directory]...

- [ Options ] -

 Options Short / Long           | Type | Description
================================+======+==========================================================
 -m, --hash-type                | Num  | Hash-type, references below (otherwise autodetect)
 -a, --attack-mode              | Num  | Attack-mode, see references below
 -V, --version                  |      | Print version
 -h, --help                     |      | Print help
     --quiet                    |      | Suppress output
     --hex-charset              |      | Assume charset is given in hex
     --hex-salt                 |      | Assume salt is given in hex
     --hex-wordlist             |      | Assume words in wordlist are given in hex
     --force                    |      | Ignore warnings
     --deprecated-check-disable |      | Enable deprecated plugins
     --status                   |      | Enable automatic update of the status screen
     --status-json              |      | Enable JSON format for status output
     --status-timer             | Num  | Sets seconds between status screen updates to X
     --stdin-timeout-abort      | Num  | Abort if there is no input from stdin for X seconds
     --machine-readable         |      | Display the status view in a machine-readable format
     --keep-guessing            |      | Keep guessing the hash after it has been cracked
     --self-test-disable        |      | Disable self-test functionality on startup
     --loopback                 |      | Add new plains to induct directory
     --markov-hcstat2           | File | Specify hcstat2 file to use
     --markov-disable           |      | Disables markov-chains, emulates classic brute-force
     --markov-classic           |      | Enables classic markov-chains, no per-position
 -t, --markov-threshold         | Num  | Threshold X when to stop accepting new markov-chains
     --runtime                  | Num  | Abort session after X seconds of runtime
     --session                  | Str  | Define specific session name
     --restore                  |      | Restore session from --session
     --restore-disable          |      | Do not write restore file
     --restore-file-path        | File | Specific path to restore file
 -o, --outfile                  | File | Define outfile for recovered hash
     --outfile-format           | Str  | Outfile format to use, separated with commas
     --outfile-autohex-disable  |      | Disable the use of $HEX[] in output plains
     --outfile-check-timer      | Num  | Sets seconds between outfile checks to X
-- More  --
```

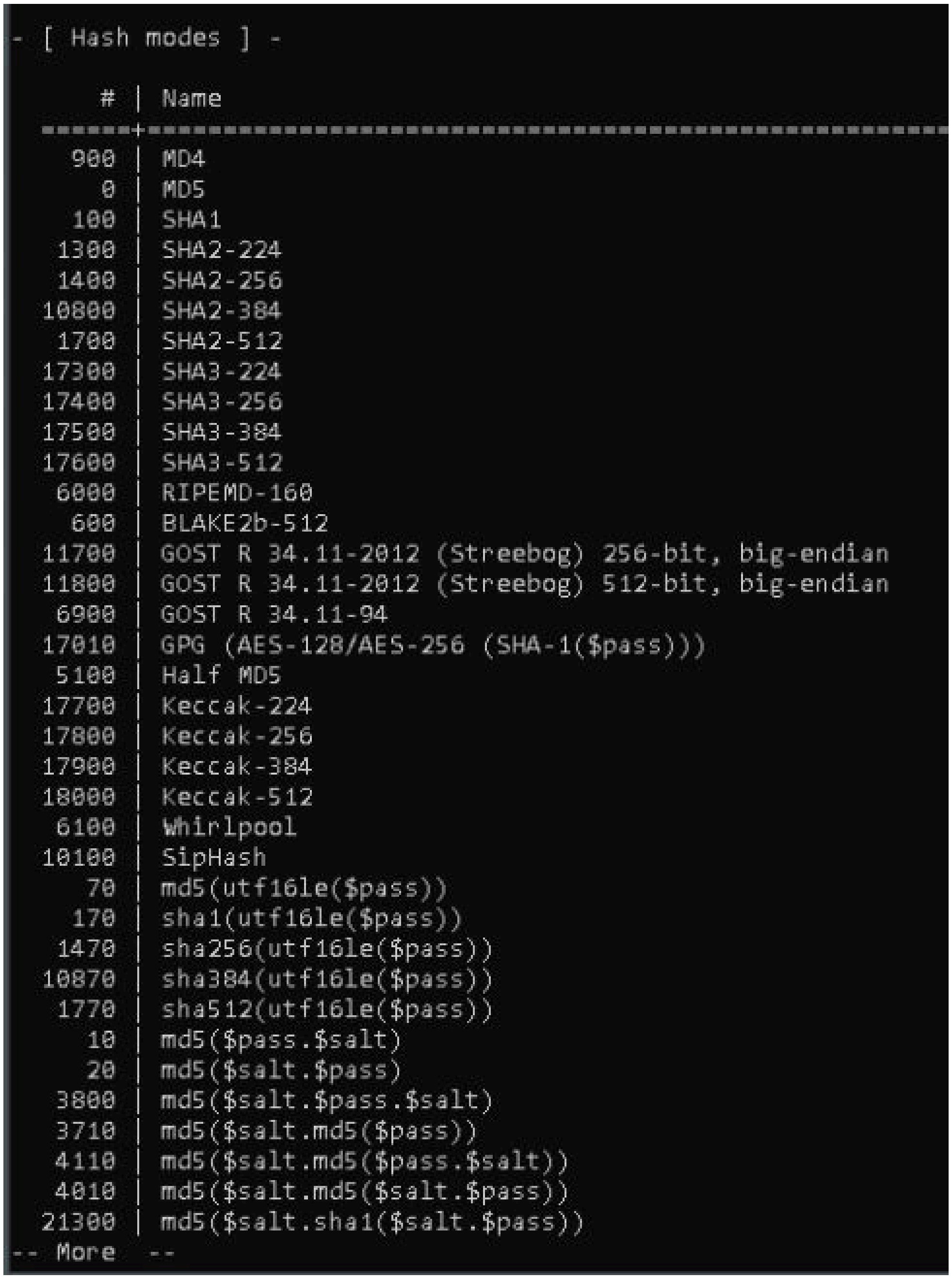
HashCat has several options that are needed to crack the hash:

-a Attack-Mode

-m Hash-Mode

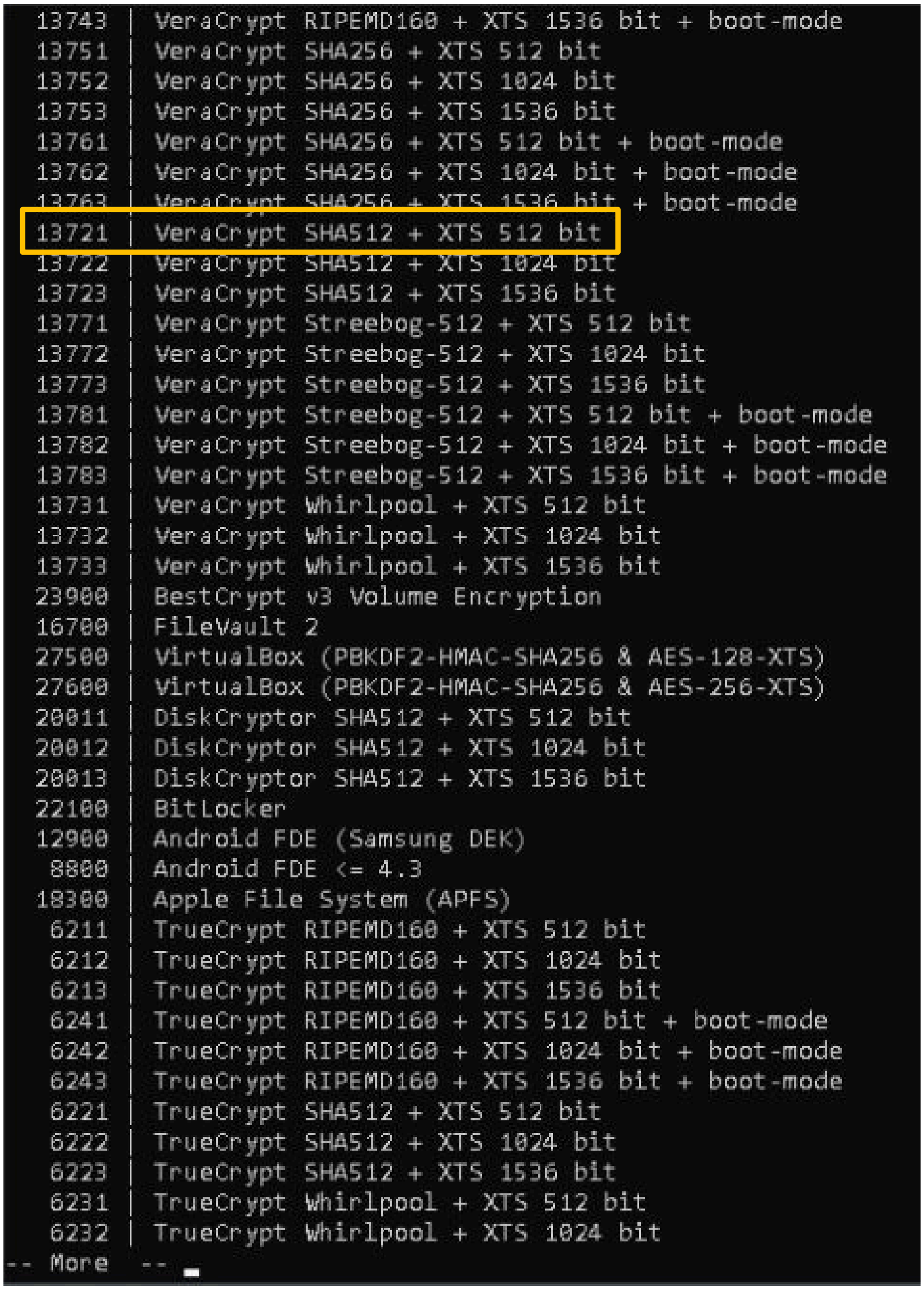
The specific values for these options can be found further down in the help file.

6. Press the **Spacebar** to advance one page at a time on the help screens and look for the *Hash Modes* section. This is where you will tell *HashCat* which hash algorithm was used to create the hash value.



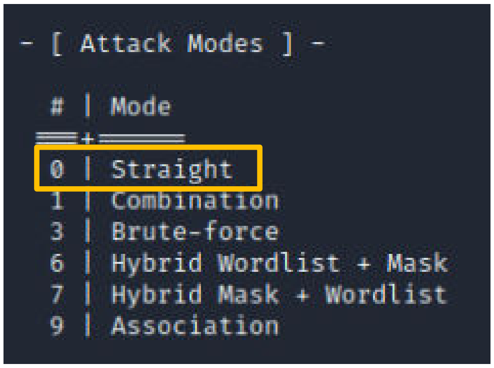
```
- [ Hash modes ] -

      # | Name
 ======+==========================================================
    900 | MD4
      0 | MD5
    100 | SHA1
   1300 | SHA2-224
   1400 | SHA2-256
  10800 | SHA2-384
   1700 | SHA2-512
  17300 | SHA3-224
  17400 | SHA3-256
  17500 | SHA3-384
  17600 | SHA3-512
   6000 | RIPEMD-160
    600 | BLAKE2b-512
  11700 | GOST R 34.11-2012 (Streebog) 256-bit, big-endian
  11800 | GOST R 34.11-2012 (Streebog) 512-bit, big-endian
   6900 | GOST R 34.11-94
  17010 | GPG (AES-128/AES-256 (SHA-1($pass)))
   5100 | Half MD5
  17700 | Keccak-224
  17800 | Keccak-256
  17900 | Keccak-384
  18000 | Keccak-512
   6100 | Whirlpool
  10100 | SipHash
     70 | md5(utf16le($pass))
    170 | sha1(utf16le($pass))
   1470 | sha256(utf16le($pass))
  10870 | sha384(utf16le($pass))
   1770 | sha512(utf16le($pass))
     10 | md5($pass.$salt)
     20 | md5($salt.$pass)
   3800 | md5($salt.$pass.$salt)
   3710 | md5($salt.md5($pass))
   4110 | md5($salt.md5($pass.$salt))
   4010 | md5($salt.md5($salt.$pass))
  21300 | md5($salt.sha1($salt.$pass))
-- More  --
```

7. Continue on to the next set of modes until you find the **VeraCrypt SHA512 + XTS 512 bit**. Note the number in the left column. This is the value of the *Hash-Mode*. In this case, it is **13721.**
   Make a note of it.



```
  13743 | VeraCrypt RIPEMD160 + XTS 1536 bit + boot-mode
  13751 | VeraCrypt SHA256 + XTS 512 bit
  13752 | VeraCrypt SHA256 + XTS 1024 bit
  13753 | VeraCrypt SHA256 + XTS 1536 bit
  13761 | VeraCrypt SHA256 + XTS 512 bit + boot-mode
  13762 | VeraCrypt SHA256 + XTS 1024 bit + boot-mode
  13763 | VeraCrypt SHA256 + XTS 1536 bit + boot-mode
  13721 | VeraCrypt SHA512 + XTS 512 bit
  13722 | VeraCrypt SHA512 + XTS 1024 bit
  13723 | VeraCrypt SHA512 + XTS 1536 bit
  13771 | VeraCrypt Streebog-512 + XTS 512 bit
  13772 | VeraCrypt Streebog-512 + XTS 1024 bit
  13773 | VeraCrypt Streebog-512 + XTS 1536 bit
  13781 | VeraCrypt Streebog-512 + XTS 512 bit + boot-mode
  13782 | VeraCrypt Streebog-512 + XTS 1024 bit + boot-mode
  13783 | VeraCrypt Streebog-512 + XTS 1536 bit + boot-mode
  13731 | VeraCrypt Whirlpool + XTS 512 bit
  13732 | VeraCrypt Whirlpool + XTS 1024 bit
  13733 | VeraCrypt Whirlpool + XTS 1536 bit
  23900 | BestCrypt v3 Volume Encryption
  16700 | FileVault 2
  27500 | VirtualBox (PBKDF2-HMAC-SHA256 & AES-128-XTS)
  27600 | VirtualBox (PBKDF2-HMAC-SHA256 & AES-256-XTS)
  20011 | DiskCryptor SHA512 + XTS 512 bit
  20012 | DiskCryptor SHA512 + XTS 1024 bit
  20013 | DiskCryptor SHA512 + XTS 1536 bit
  22100 | BitLocker
  12900 | Android FDE (Samsung DEK)
   8800 | Android FDE <= 4.3
  18300 | Apple File System (APFS)
   6211 | TrueCrypt RIPEMD160 + XTS 512 bit
   6212 | TrueCrypt RIPEMD160 + XTS 1024 bit
   6213 | TrueCrypt RIPEMD160 + XTS 1536 bit
   6241 | TrueCrypt RIPEMD160 + XTS 512 bit + boot-mode
   6242 | TrueCrypt RIPEMD160 + XTS 1024 bit + boot-mode
   6243 | TrueCrypt RIPEMD160 + XTS 1536 bit + boot-mode
   6221 | TrueCrypt SHA512 + XTS 512 bit
   6222 | TrueCrypt SHA512 + XTS 1024 bit
   6223 | TrueCrypt SHA512 + XTS 1536 bit
   6231 | TrueCrypt Whirlpool + XTS 512 bit
   6232 | TrueCrypt Whirlpool + XTS 1024 bit
-- More  -- _
```

8. Continue pressing the **Spacebar** to advance on the help screens and look for the *Attack Modes* section. This is where you would tell *HashCat* what attack mode to use.



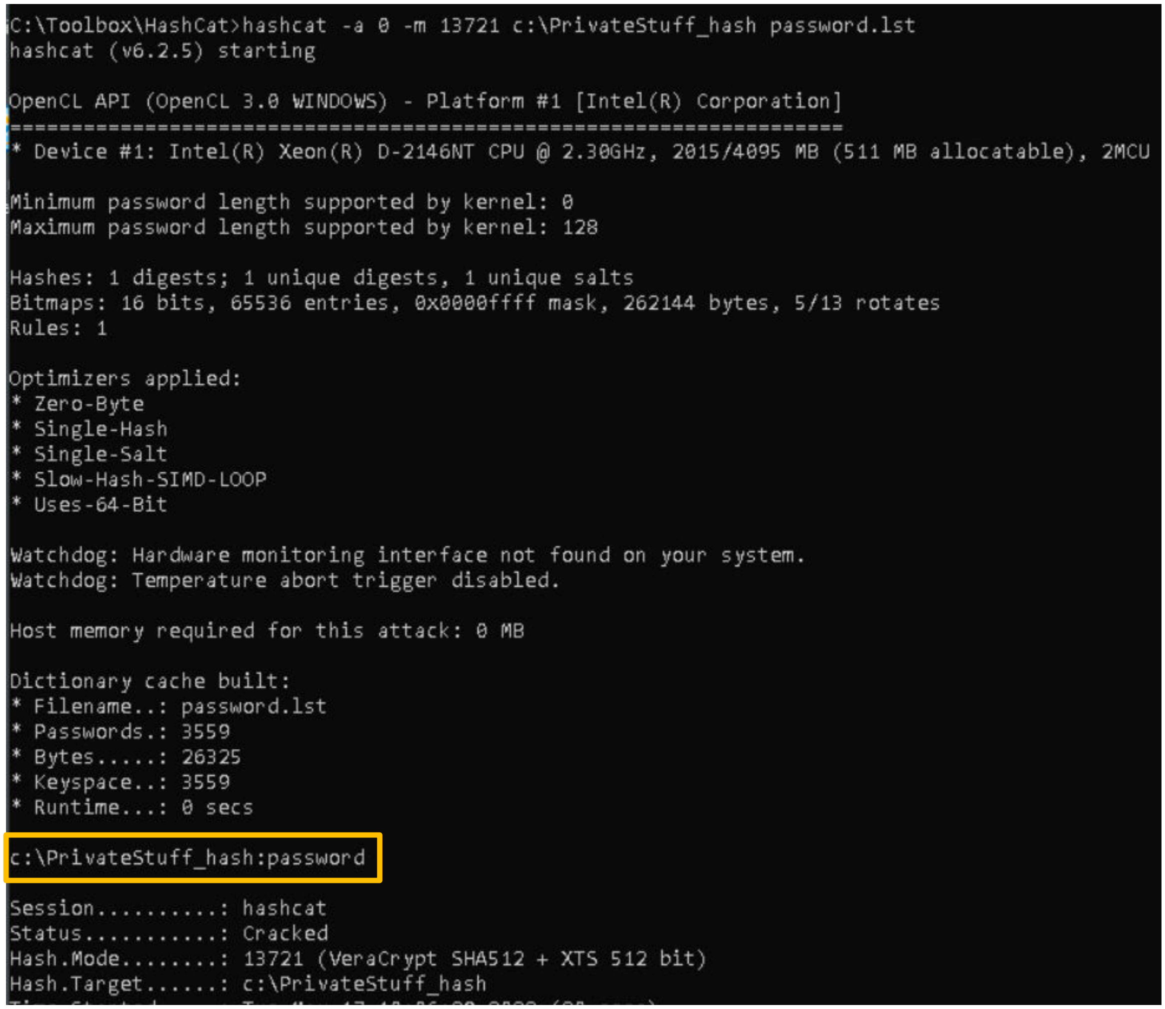
```
- [ Attack Modes ] -

  # | Mode
 ===+======
  0 | Straight
  1 | Combination
  3 | Brute-force
  6 | Hybrid Wordlist + Mask
  7 | Hybrid Mask + Wordlist
  9 | Association
```

Because you will use **Straight Mode** (dictionary), make a note of the value, which is **0**.

9. Once you have all of the option mode values, run *HashCat* against the password hash with the following command:

```
hashcat -a 0 -m 13721 c:\PrivateStuff_hash password.lst
```

It can take as long as 15 minutes for the scan to complete. When the scan completes, *HashCat* will show the password that was cracked from the hash file.



```
C:\Toolbox\HashCat>hashcat -a 0 -m 13721 c:\PrivateStuff_hash password.lst
hashcat (v6.2.5) starting

OpenCL API (OpenCL 3.0 WINDOWS) - Platform #1 [Intel(R) Corporation]
====================================================================
* Device #1: Intel(R) Xeon(R) D-2146NT CPU @ 2.30GHz, 2015/4095 MB (511 MB allocatable), 2MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 128

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Single-Hash
* Single-Salt
* Slow-Hash-SIMD-LOOP
* Uses-64-Bit

Watchdog: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.

Host memory required for this attack: 0 MB

Dictionary cache built:
* Filename..: password.lst
* Passwords.: 3559
* Bytes.....: 26325
* Keyspace..: 3559
* Runtime...: 0 secs

c:\PrivateStuff_hash:password

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 13721 (VeraCrypt SHA512 + XTS 512 bit)
Hash.Target......: c:\PrivateStuff_hash
```

If the scan shows that the password can be cracked in a short time, then a security analyst will report that the passwords need to be both longer and more complicated.

“Short” is, of course, relative. *HashCat* runs most effectively using high-powered GPUs for decryption. For example, using a Brute-Force scan on an 8 character password using just a single CPU is estimated to take 560 years.

10. The lab is now complete; you may now end the reservation.