Ramadan CTF Writeup

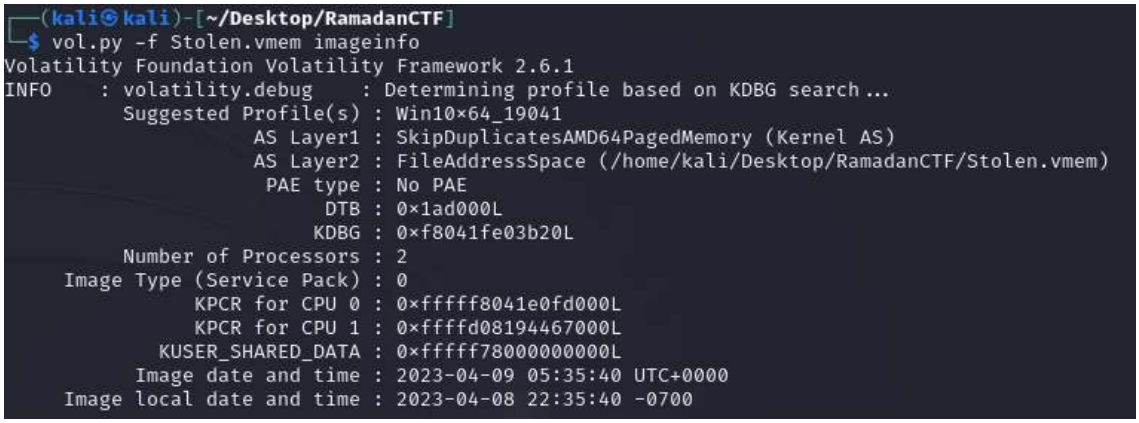
By : cicakberlari

History 101 (Digital Forensics)

- For this challenge, I don't manage to answer it during the competition, but I continue it for my learning purposes.
- For this challenge we receive a Stolen.vmem file from the zip. *.vmem* file is virtual machine's paging file, which usually belong to VMWare Workstation.

1. For this challenge, im using volatility2(I used volatility3 during competition but its not works haha). Its kinda hard anyways to install and setup it. Here some guide that I found and use it to setup my volatility2 :
    a. Installing Volatility 2 and 3 on Debian-based Linux - seanthegeek.net
    b. Volatility 3 CheatSheet - onfvpBlog [Ashley Pearson]
    c. Security Compass Internal CTF Write-Up | by Security Compass | Medium

2. After setup our volatility2, we start by doing some information gathering on the file. By running command ***vol.py -f "/path/to/file" imageinfo*** (make sure you are in bash if you follow setup on 1(a))



3. From (2) we know that this vmem file have Win10x64_19041 as its profile. We continue to do some other recon, one that I find is to get the cmdline for the OS.

```
┌──(kali㉿kali)-[~/Desktop/RamadanCTF]
└─$ vol.py -f Stolen.vmem --profile Win10x64_19041 cmdline
Volatility Foundation Volatility Framework 2.6.1
**************************************************
System pid:      4
**************************************************
Registry pid:     92
**************************************************
smss.exe pid:    316
**************************************************
csrss.exe pid:    444
Command line :
**************************************************
csrss.exe pid:    520
Command line :
**************************************************
wininit.exe pid:    568
**************************************************
winlogon.exe pid:    576
Command line : winlogon.exe
**************************************************
services.exe pid:    664
Command line : C:\Windows\system32\services.exe
**************************************************
lsass.exe pid:    684
Command line : C:\Windows\system32\lsass.exe
**************************************************
svchost.exe pid:    804
Command line : C:\Windows\system32\svchost.exe -k DcomLaunch -p
**************************************************
fontdrvhost.ex pid:    832
Command line :
**************************************************
fontdrvhost.ex pid:    828
**************************************************
svchost.exe pid:    924
```

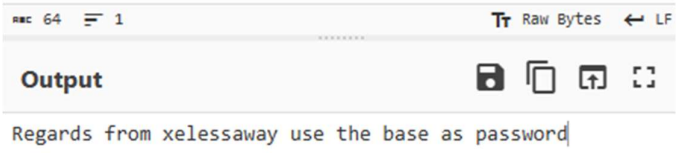
(The response is actually longer)

4. From the response we found one suspicious command :

```
cmd.exe pid:   2596
Command line : C:\Windows\system32\cmd.exe  echo "UmVnYXJkcyBmcm9tIHhlbGVzc2F3YXkgdXNlIHRoZSBiYXNlIGFzIHBhc3N3b3Jk"
**************************************************
```

5. The command echoing some base64. We try to decode it using CyberChef.

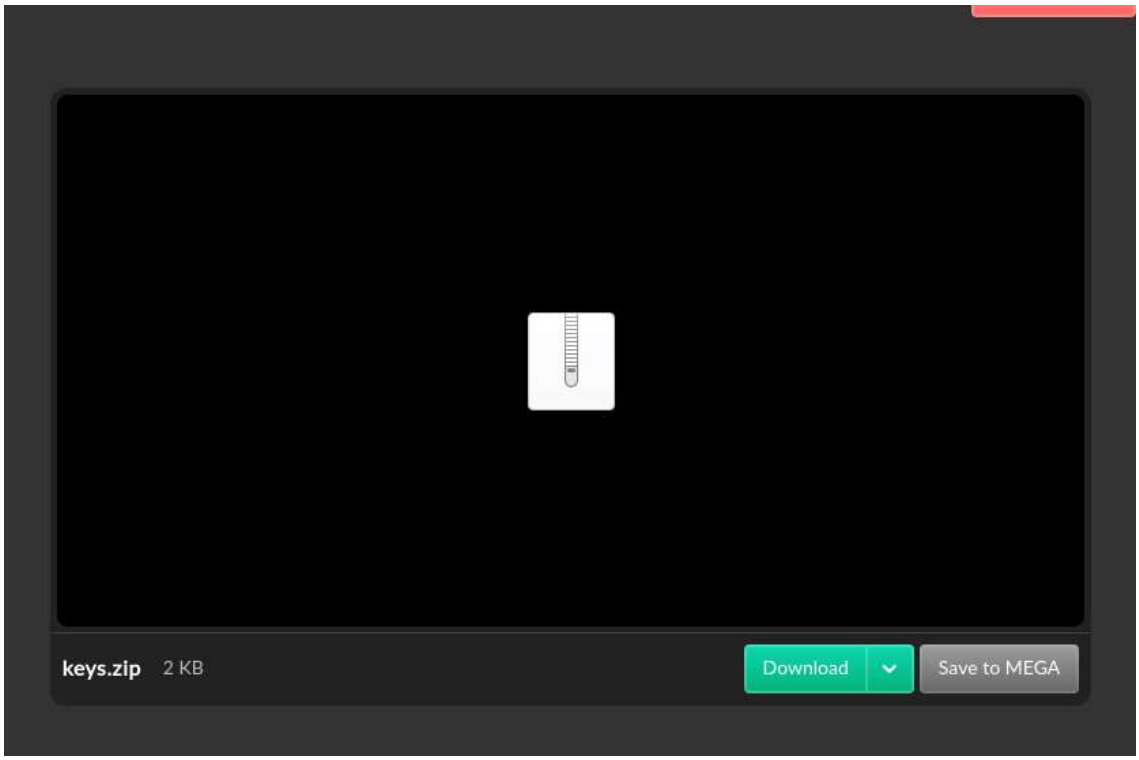
```
UmVnYXJkcyBmcm9tIHhlbGVzc2F3YXkgdXNlIHRoZSBiYXNlIGFzIHBhc
3N3b3Jk
```



6. At first, we thought this is its flag but when we try to submit it its wrong. So we assume that this base64(because xelessaway said that we need to use “base” as password) is going to be password for some other thing.
7. We search for some other clue and from the challenge name *History101,* we thought that it maybe have some relation with history. By doing some research we can found a plugin name **chromehistory**. This is the link for the github : [superponible/volatility-plugins: Plugins I've written for Volatility (github.com)]
8. We need to install it inside our “volatility/plugins” files. This part is little bit tricky as you need to search for volatility directory. If you using 1(a) setup you may need to run ***sudo find / -name "volatility" 2>/dev/null*** . After that I *ls* into plugins file and manually use vim to copy and paste the code haha. (I think there is command git clone that you can use with raw:github something like that)
9. Make sure you install sqlite_help.py too
10. Now we can use our plugins to get the history of the browser from the vmem file

11. Can you see something interesting? There is title that said “hello guys you are on the right path”!! Yeayy
12. From the urls that we found, there are two url that different from others which are Pastebin and mega url. We can click both link and we will go to Pastebin web and mega url

Pastebin



Mega

13. Oh noo the Pastebin is locked,, but worry not we remember right what xelesaway said? Used the base and we can unlocked the Pastebin. Now download both mega and Pastebin to our machine.

14. From mega we get ***keys.kdbx*** file and from Pastebin we can get wordlist. We use john to get the hash from our ***.kdbx*** file.

```
┌──(kali㉿kali)-[~/Desktop/RamadanCTF]
└─$ keepass2john keys.kdbx > keepasshash.txt

┌──(kali㉿kali)-[~/Desktop/RamadanCTF]
└─$ cat keepasshash.txt
keys:$keepass$*2*60000*0*7ba573f2a464b6f2fcd2d1f9ba458bce6fc0b1591bac91047f15a73f8bb79c3f*0fc4c9bcad9999902c5c360b3d
91bd65227eda6989*ebdec35450c1087b55117b00cb0d4284735a9fd2576b75ad23c6337f35c38e02
```

15. And then we use john again to compare the hash with wordlist that we got from Pastebin

```
┌──(kali㉿kali)-[~/Desktop/RamadanCTF/wc]
└─$ sudo john --wordlist=wordlist.txt keepasshash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (KeePass [SHA256 AES 32/64])
Cost 1 (iteration count) is 60000 for all loaded hashes
Cost 2 (version) is 2 for all loaded hashes
Cost 3 (algorithm [0=AES 1=TwoFish 2=ChaCha]) is 0 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
icanfindthishiddenpasswordforcrack (keys)
1g 0:00:00:57 DONE (2024-03-29 06:50) 0.01741g/s 106.4p/s 106.4c/s 106.4C/s manda321..manchester.
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

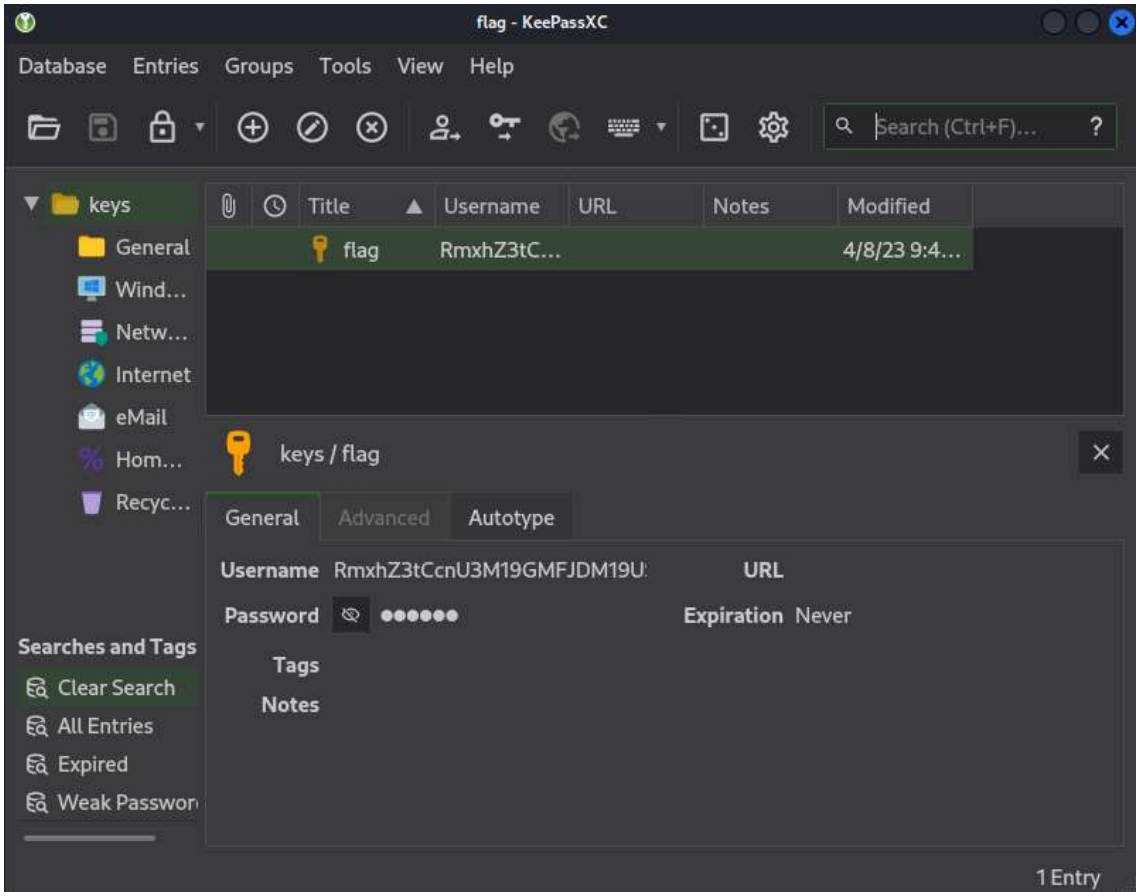
16. Boom we got the password, but its not the end. We need to open Keepass using KeepassXC. You can follow this tutorial from copilot haha :

1. **Install KeePassXC** (if not already installed):
   - Open a terminal and run the following command:

```
sudo apt-get install keepassxc
```

2. **Launch KeePassXC**:
   - You can start KeePassXC from the terminal by typing `keepassxc` or by finding it in your applications menu.
3. **Open the Database**:
   - In KeePassXC, go to `Database` → `Open Database...`.
   - Navigate to the location of your `.kdbx` file, select it, and click `Open`.
4. **Unlock the Database**:
   - KeePassXC will prompt you to enter the master password or use the key file to unlock the database. Since you have a key file, click on the button to browse for your key file and select it.
   - If your database also requires a password, enter it in the provided field.
5. **Access Your Entries**:
   - Once unlocked, you'll have access to all the entries stored in your KeePass database.

17. Run keepassxc on terminal and then open the keys.kdbx file using password that we crack before.



18. We can decode the username from base64 and then get the flag.