Stredná priemyselná škola elektrotechnická
Komenského 44, 040 01 Košice

Forenzná analýza operačnej pamäte Windows

Autor: Marek Horňak
Trieda: 4.B
Školský rok: 2023/2024
Odbor: 2561M – Informačné a sieťové technológie

Obsah

0. Úvod

Tento učebný materiál opisuje analýzu obrazu operačnej pamäte počítača s Windowsom. V prvých dvoch úlohách (laboch) budete podľa návodu analyzovať obrazy operačnej pamäte a hľadať v nich flagy. V každej úlohe je potrebné nájsť 3 flagy. Tretiu úlohu už budete analyzovať svojpomocne za použitia skúseností nadobudnutých v prvých dvoch úlohách.

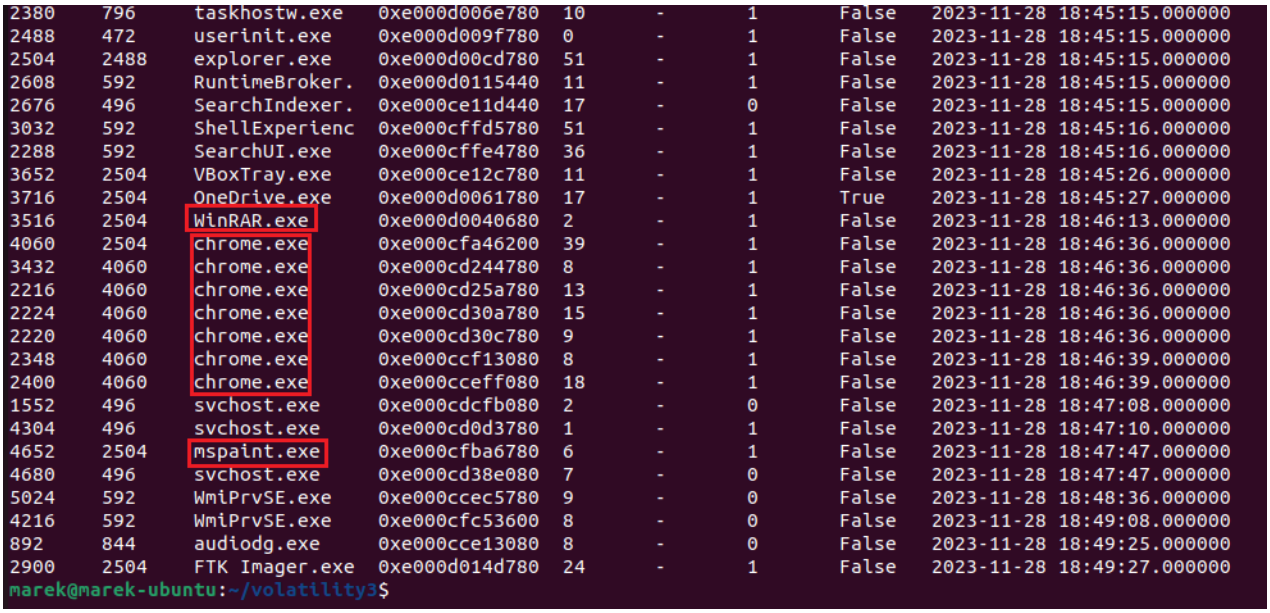
Pomôcky:

- Virtuálny počítač s Ubuntu alebo iným Linuxom
- Volatility3

1. Lab1 – Windows

Do virtuálneho počítača s Ubuntu si z Github stránky stiahnite súbor **lab1_win.raw** a premiestnite ho do priečinka s Volatility3. Nebudete tak musieť špecifikovať celú cestu k súboru. Presuňte sa do priečinka s Volatility3, aby bol vašim pracovným priečinkom. Začneme tým, že si vypíšeme zoznam bežiacich procesov, aby sme našli podozrivú aktivitu. Použijeme plugin *windows.pslist*:

python3 vol.py -f lab1_win.raw windows.pslist



2380	796	taskhostw.exe	0xe000d006e780	10	-	1	False	2023-11-28 18:45:15.000000
2488	472	userinit.exe	0xe000d009f780	0	-	1	False	2023-11-28 18:45:15.000000
2504	2488	explorer.exe	0xe000d00cd780	51	-	1	False	2023-11-28 18:45:15.000000
2608	592	RuntimeBroker.	0xe000d0115440	11	-	1	False	2023-11-28 18:45:15.000000
2676	496	SearchIndexer.	0xe000ce11d440	17	-	0	False	2023-11-28 18:45:15.000000
3032	592	ShellExperienc	0xe000cffd5780	51	-	1	False	2023-11-28 18:45:16.000000
2288	592	SearchUI.exe	0xe000cffe4780	36	-	1	False	2023-11-28 18:45:16.000000
3652	2504	VBoxTray.exe	0xe000ce12c780	11	-	1	False	2023-11-28 18:45:26.000000
3716	2504	OneDrive.exe	0xe000d0061780	17	-	1	True	2023-11-28 18:45:27.000000
3516	2504	WinRAR.exe	0xe000d0040680	2	-	1	False	2023-11-28 18:46:13.000000
4060	2504	chrome.exe	0xe000cfa46200	39	-	1	False	2023-11-28 18:46:36.000000
3432	4060	chrome.exe	0xe000cd244780	8	-	1	False	2023-11-28 18:46:36.000000
2216	4060	chrome.exe	0xe000cd25a780	13	-	1	False	2023-11-28 18:46:36.000000
2224	4060	chrome.exe	0xe000cd30a780	15	-	1	False	2023-11-28 18:46:36.000000
2220	4060	chrome.exe	0xe000cd30c780	9	-	1	False	2023-11-28 18:46:36.000000
2348	4060	chrome.exe	0xe000ccf13080	8	-	1	False	2023-11-28 18:46:39.000000
2400	4060	chrome.exe	0xe000cceff080	18	-	1	False	2023-11-28 18:46:39.000000
1552	496	svchost.exe	0xe000cdcfb080	2	-	0	False	2023-11-28 18:47:08.000000
4304	496	svchost.exe	0xe000cd0d3780	1	-	1	False	2023-11-28 18:47:10.000000
4652	2504	mspaint.exe	0xe000cfba6780	6	-	1	False	2023-11-28 18:47:47.000000
4680	496	svchost.exe	0xe000cd38e080	7	-	0	False	2023-11-28 18:47:47.000000
5024	592	WmiPrvSE.exe	0xe000ccec5780	9	-	0	False	2023-11-28 18:48:36.000000
4216	592	WmiPrvSE.exe	0xe000cfc53600	8	-	0	False	2023-11-28 18:49:08.000000
892	844	audiodg.exe	0xe000cce13080	8	-	0	False	2023-11-28 18:49:25.000000
2900	2504	FTK Imager.exe	0xe000d014d780	24	-	1	False	2023-11-28 18:49:27.000000

marek@marek-ubuntu:~/volatility3$

Obrázok 1: bežiace procesy

Na obrázku 1 je vidieť výstup s bežiacimi procesmi. Väčšina procesov ako napr. svchost.exe sú klasické windowsové procesy, ktoré bežia na pozadí. Červenou farbou sú vyznačené procesy, ktoré spustil používateľ. Sú to *WinRAR.exe, chrome.exe a mspaint.exe*. Tieto procesy musíme vyšetriť, pretože môžu ukrývať nejakú podozrivú aktivitu. Začnime programom mspaint.exe. Z toho vieme usúdiť, že používateľ niečo kreslil a mohlo by to byť dôležité. Všimnite si číslo procesu: 4652. Volatility3 nám umožňuje stiahnuť si všetky dáta patriace danému procesu. Použijeme plugin *windows.memmap*:

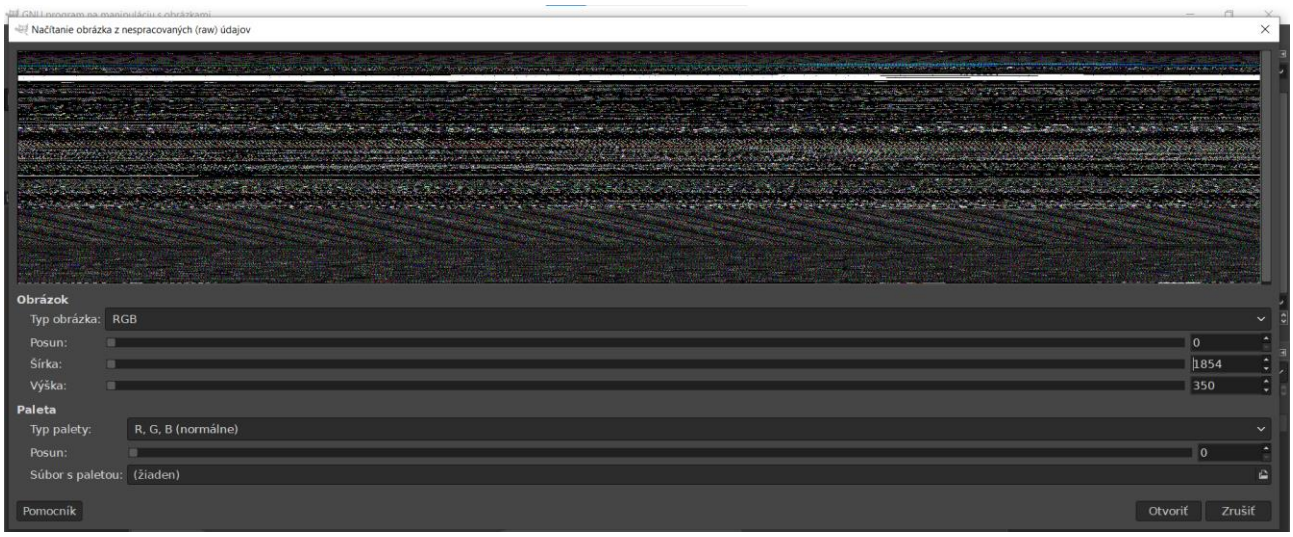
python3 vol.py -f lab1_win.raw -o output/ windows.memmap --dump --pid 4652

Prepínačom -o sme špecifikovali výstupný priečinok, kam sa má výstup uložiť. --dump znamená, že chceme vytvoriť pamäťový dump (extrakt) a --pid špecifikuje konkrétny proces.

```
0xfa8000000000  0x7fc00000      0x200000        0xf9f2000       pid.4652.dmp
0xfa8000200000  0x7fa00000      0x200000        0xfbf2000       pid.4652.dmp
0xfa8000400000  0x7f800000      0x200000        0xfdf2000       pid.4652.dmp
0xfa8000600000  0x7f600000      0x200000        0xfff2000       pid.4652.dmp
0xfa8000800000  0x7f400000      0x200000        0x101f2000      pid.4652.dmp
0xfa8000a00000  0x7f200000      0x200000        0x103f2000      pid.4652.dmp
0xfa8000c00000  0x7f000000      0x200000        0x105f2000      pid.4652.dmp
0xfa8000e00000  0x7ee00000      0x200000        0x107f2000      pid.4652.dmp
0xfa8001000000  0x7ec00000      0x200000        0x109f2000      pid.4652.dmp
0xfa8001200000  0x7ea00000      0x200000        0x10bf2000      pid.4652.dmp
0xfa8001400000  0x7e800000      0x200000        0x10df2000      pid.4652.dmp
0xfa8001600000  0x7e600000      0x200000        0x10ff2000      pid.4652.dmp
0xfa8001800000  0x7e400000      0x200000        0x111f2000      pid.4652.dmp
0xffffffd00000  0x1000  0x1000  0x113f2000      pid.4652.dmp
0xffffffd01000  0x3000  0x1000  0x113f3000      pid.4652.dmp
0xffffffd02000  0x2000  0x1000  0x113f4000      pid.4652.dmp
0xffffffd03000  0x4000  0x8000  0x113f5000      pid.4652.dmp
0xffffffd0b000  0xe000  0x1000  0x113fd000      pid.4652.dmp
0xffffffd0c000  0xc000  0x2000  0x113fe000      pid.4652.dmp
0xffffffd0e000  0xf000  0x1000  0x11400000      pid.4652.dmp
0xffffffd10000  0x60000 0x1000  0x11401000      pid.4652.dmp
0xffffffd12000  0x61000 0x1000  0x11402000      pid.4652.dmp
marek@marek-ubuntu:~/volatility3$
```

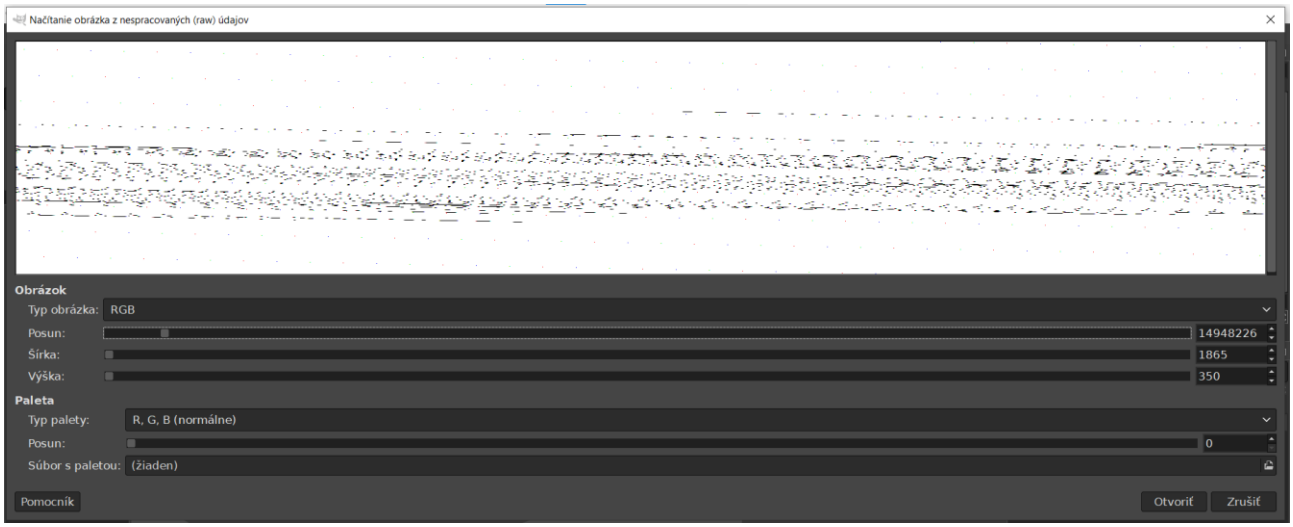
Obrázok 2: vytvorenie pamäťového extraktu

Na obrázku vidíme, že každý úsek (offset) pamäti, ktorý je pridelený procesu 4652 bol uložený do súboru *pid.4652.dmp*. Aby sme si súbor vedeli prezerať, je potrebné ho premenovať na súbor s príponou .data. Takto ho vieme otvoriť v programe Gimp. Keď to urobíme, uvidíme niečo takéto:



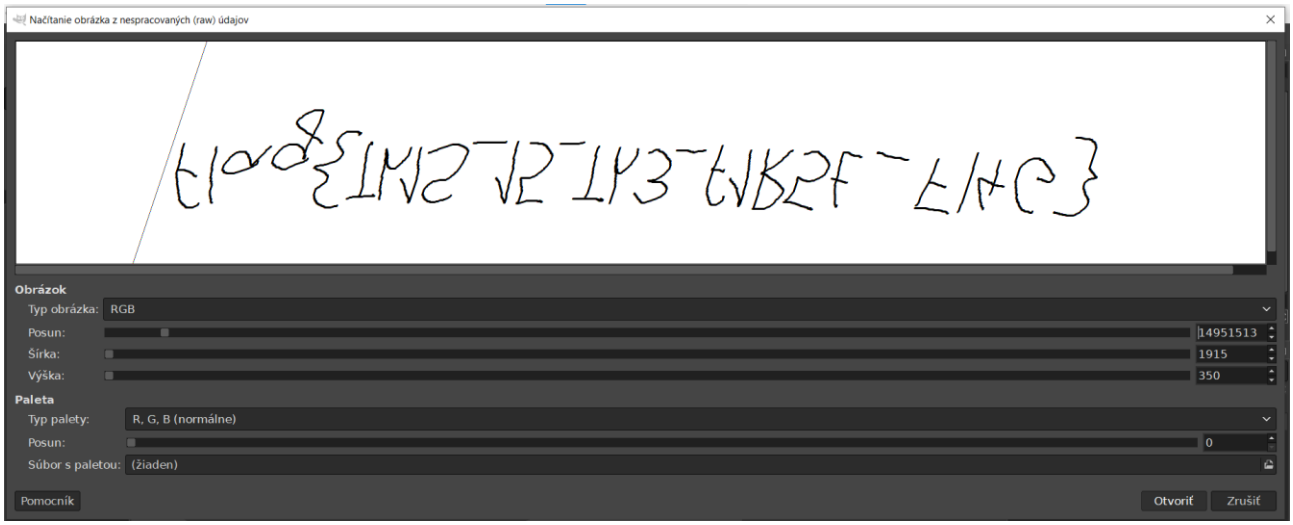
Obrázok 3: otvorenie súboru v Gimpe

Tento súbor sú zatiaľ len nespracované dáta, takže ho musíme trochu upraviť. Dobrá stratégia na to ako to urobiť je natiahnuť si okno s obrázkom cez celú obrazovku, tak ako to vidíte na obrázku 3 a upraviť šírku tak, aby obrázok pokrýval celú šírku okna. Potom sa trochu pohrajte s posunom až kým neuvidíte niečo takéto:



Obrázok 4: hľadanie súvislého obrazu

Teraz potrebujeme trochu posúvať šírku obrazu, až kým neuvidíme súvislý text.



Obrázok 5: prvý flag

Našli sme prvý flag. Kliknutím na *Nástroje>Transformačné nástroje>Otočiť* obrázok otočíme a kliknutím na *Nástroje>Transformačné nástroje>Preklopiť* obrázok preklopíme do čitateľnej polohy. Prvý flag je teda: flag{Th1S_15_Th3_fl1R5t_Fl4G} Presunieme sa na druhý podozrivý proces a tým je chrome.exe. Na získanie histórie navštívených stránok sa používa plugin *windows.vadyarascan*:

```
python3 vol.py -f lab1_win.raw windows.vadyarascan --pid 4060 --yara-rules "/http.{100}?/" | awk '
  /^0x/{
      S=""
      for(i=5; i<NF; i++) {
          S=S sprintf("%s", $i)
      }
      print "PID:" $2 "\tRule:" $3 "\tComponent:" $4
      system("echo " S " | xxd -r -ps | xxd -o " $1 " | sed \"s/^/0x/\"")
      print ""
      next
  }
```

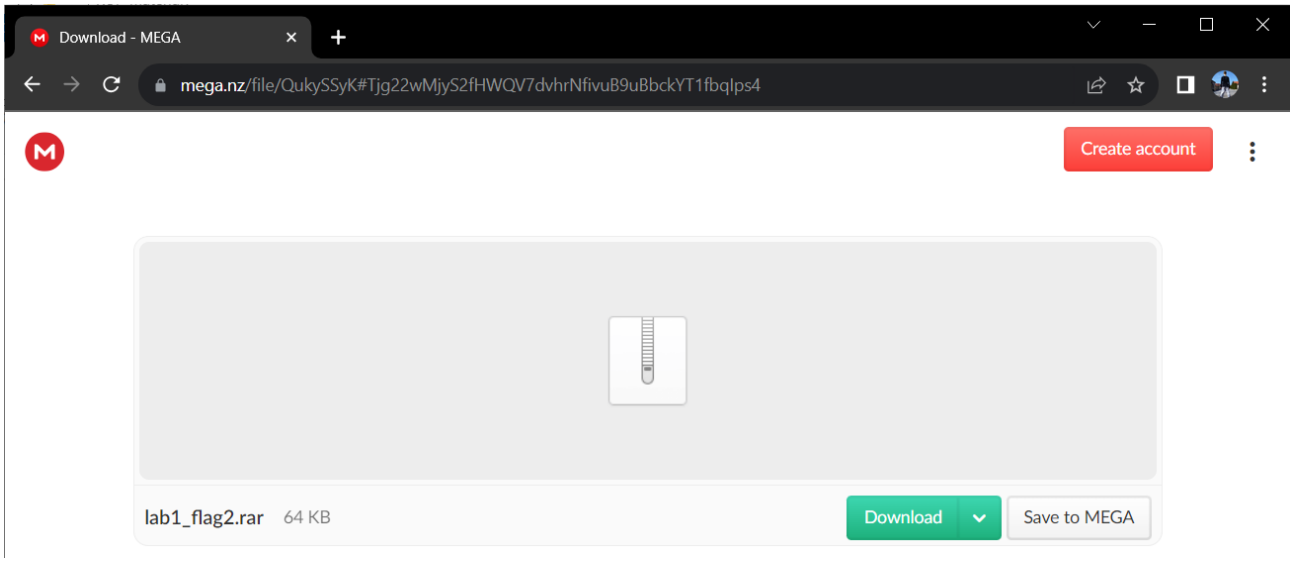
```
    /^Offset/{next}
    {print}
'
```

Keďže Chrome má spustených viacero procesov, použijeme číslo rodičovského procesu (Parental PID), teda 4060. Samotný plugin neposkytuje dostatočný výstup na zistenie odkazu, použijeme k nemu doplnkový program, ktorý trochu upraví jeho výstup. Plugin hľadá v procese 4060 reťazec http, aby našiel všetky hypertextové prepojenia aj tie šifrované a vypisuje sto nasledujúcich znakov. Predvolene plugin vypíše len daný reťazec ak ho našiel a vypisuje ho len v hexadecimálnom formáte:

```
PID:4060        Rule:r1 Component:$a
0x7ffe0f3c8b40: 6874 7470 733a 2f2f 6d65 6761 2e6e 7a2f  https://mega.nz/
0x7ffe0f3c8b50: 6669 6c65 2f51 756b 7953 5379 4b23 546a  file/QukySSyK#Tj
0x7ffe0f3c8b60: 6732 3277 4d6a 7953 3266 4857 5156 3764  g22wMjyS2fHWQV7d
0x7ffe0f3c8b70: 7668 724e 6669 7675 4239 7542 6263 6b59  vhrNfivuB9uBbckY
0x7ffe0f3c8b80: 5431 6662 7149 7073 3400 0000 0000 0000  T1fbqIps4.......
0x7ffe0f3c8b90: 0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffe0f3c8ba0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
```

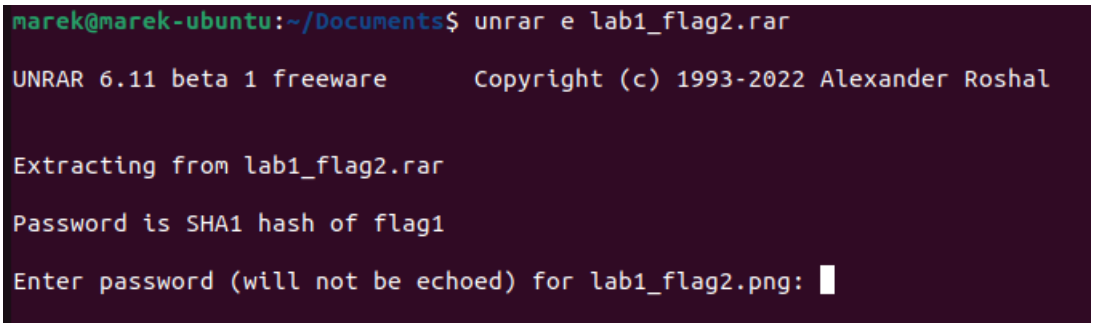
Obrázok 6: link na online cloud mega.nz

Po chvíli scrollovania nájdeme link na stránku Mega. Je to online cloud, čiže predpokladáme, že tam nájdeme nejaký užitočný súbor. Skopírujeme si link a otvoríme ho v prehliadači:



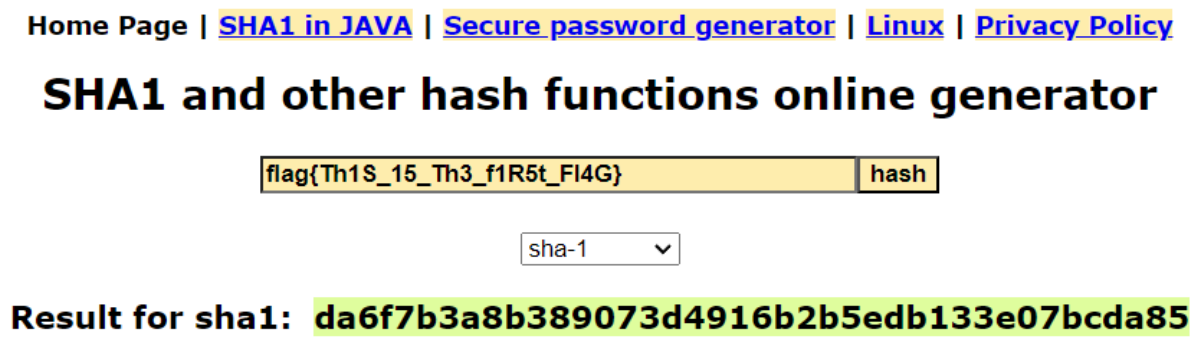
Obrázok 7: súbor v cloude

Dostali sme sa k WinRAR archívu. Pravdepodobne je v ňom niečo ukryté. Stiahneme si ho a skúsime ho otvoriť. Archív je však chránený heslom:



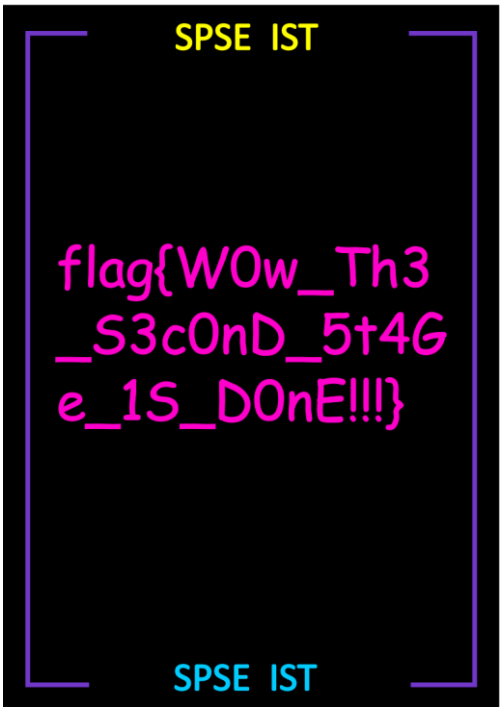
Obrázok 8: extrahovanie archívu

Dostali sme nápovedu, že heslo, ktorým odomkneme archív je SHA1 hash prvého flagu. Nájdeme si na internete nejaký online hash generátor a vygenerujeme si hash prvého flagu:



Obrázok 9: SHA1 hash flagu 1

Získaný hash použijeme na odheslovanie archívu a dostaneme obrázok s druhým flagom:



Obrázok 10: druhý flag

Tretím podozrivým procesom je WinRAR.exe. Potrebujeme zistiť, či otvoril nejaký súbor a ak áno, aký. Na to nám poslúži plugin *windows.cmdline*:

python3 vol.py -f lab1_win.raw windows.cmdline | grep WinRAR

```
marek@marek-ubuntu:~/volatility3$ python3 vol.py -f /media/sf_VMshare/lab1_win.raw windows.cmdline | grep WinRAR
3516ressWinRAR.exe      "C:\Program Files\WinRAR\WinRAR.exe" -iext "C:\Users\Marek\Documents\Important.rar"
marek@marek-ubuntu:~/volatility3$
```

Obrázok 11: hľadanie .rar archívov

Príkazom grep sme si vyfiltrovali výstup len na program WinRAR. Ostatné procesy nás momentálne nezaujímajú. Vidíme, že winrar.exe otvoril archív s názvom *Important.rar*. To znie dôležito. Teraz potrebujeme nájsť jeho adresu v pamäti. To docielime pluginom *windows.filescan*. Na uloženie súboru z pamäte použijeme plugin *windows.files*:

python3 vol.py -f lab1_win.raw windows.filescan | grep Important.rar

python3 vol.py -f lab1_win.raw -o output/ windows.dumpfiles --virtaddr 0xe000cdd5a800

```
marek@marek-ubuntu:~/volatility3$ python3 vol.py -f /media/sf_VMshare/lab1_win.raw windows.filescan | grep Important.rar
0xe000cdd5a800.0\Users\Marek\Documents\Important.rar    216
marek@marek-ubuntu:~/volatility3$ python3 vol.py -f /media/sf_VMshare/lab1_win.raw -o /media/sf_VMshare/ windows.dumpfiles --virtaddr 0xe000cdd5a800
Volatility 3 Framework 2.5.2
Progress:  100.00               PDB scanning finished
Cache   FileObject      FileName        Result

DataSectionObject       0xe000cdd5a800  Important.rar   file.0xe000cdd5a800.0xe000d02e97c0.DataSectionObject.Important.rar.dat
marek@marek-ubuntu:~/volatility3$
```

Obrázok 12: nájdenie a extrakcia archívu

Prvý plugin vypíše všetky súbory, ktoré sa nachádzali v počítači. Nie všetky sa však dajú extrahovať, pretože neotvorené súbory neboli nahrané do operačnej pamäte a tým pádom ich nebude možné získať. Tento archív bol počas behu počítača otvorený a nachádzal sa v operačnej pamäti, takže sme sa k nemu dostali. Súbor si premenujeme na Important.rar a otvoríme ho. Opäť je chránený heslom:

```
marek@marek-ubuntu:~/Documents$ unrar e Important.rar

UNRAR 6.11 beta 1 freeware      Copyright (c) 1993-2022 Alexander Roshal


Extracting from Important.rar

Password is one of environment variables

Enter password (will not be echoed) for lab1_flag3.png:
```

Obrázok 12: rozbalenie archívu

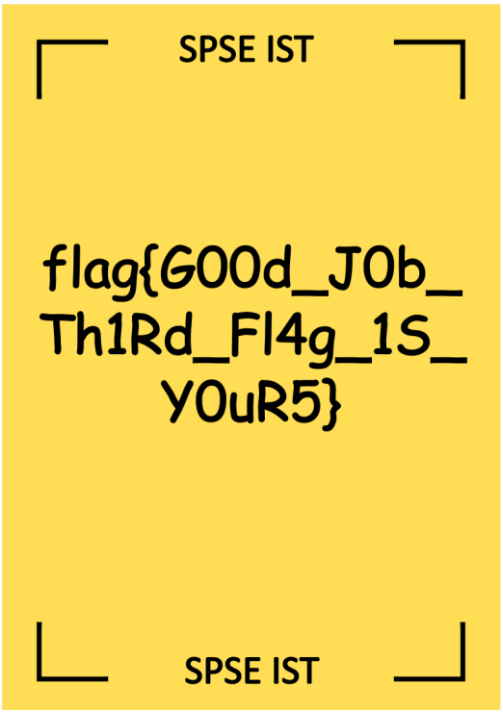
Pomôcka nám hovorí, že heslo sa skrýva medzi premennými prostredia tzv. environment variables. Sú to premenné, s ktorými operačný systém pracuje. Sú to nastavenia, ktorými sa dá ovplyvňovať fungovanie programov. Na zobrazenie premenných prostredia slúži plugin *windows.envars*:

python3 vol.py -f lab1_win.raw windows.envars | grep WinRAR

```
marek@marek-ubuntu:~/volatility3$ python3 vol.py -f /media/sf_VMshare/lab1_win.raw windows.envars | grep WinRAR
344gresscsrss.exe       0xa0143025b0scanWinRAR password Flag3Pa$$w0rd
416     wininit.exe     0xed32f0d90     WinRAR password Flag3Pa$$w0rd
424     csrss.exe       0xd1c47025b0    WinRAR password Flag3Pa$$w0rd
472     winlogon.exe    0x8dac980d90    WinRAR password Flag3Pa$$w0rd
496     services.exe    0x3533802900    WinRAR password Flag3Pa$$w0rd
504     lsass.exe       0xd07f502900    WinRAR password Flag3Pa$$w0rd
504     lsass.exe       0xd07f502900    WinRAR password Flag3Pa$$w0rd
592     svchost.exe     0x8814e02a50    WinRAR password Flag3Pa$$w0rd
624     svchost.exe     0x24f702ad0     WinRAR password Flag3Pa$$w0rd
```

Obrázok 13: premenné prostredia vo Windowse

Z výstupu je zjavné, že sme našli premennú s názvom WinRAR password s hodnotou Flag3Pa$$w0rd, teda heslo. Použijeme ho na odomknutie archívu a dostaneme tretí a posledný flag tejto úlohy:



Obrázok 14: tretí flag

Úspešne sme zozbierali všetky flagy. Môžeme sa presunúť na ďalšiu úlohu.

2. Lab2 – Windows

Do virtuálneho počítača s Ubuntu si z Github stránky stiahnite súbor lab2_win.raw a premiestnite ho do priečinka s Volatility3. Presuňte sa do priečinka s Volatility3, aby bol vašim pracovným priečinkom. Na začiatok si tak ako v predošlej úlohe vypíšeme zoznam bežiacich procesov, aby sme našli podozrivú aktivitu. Použijeme plugin *windows.pslist*:

```
1312    508     svchost.exe     0xe0011e4bb780  11      -       0       False   2023-11-30 17:52:40.000000
1504    508     svchost.exe     0xe0011e5a6780  4       -       0       False   2023-11-30 17:52:41.000000
1640    508     MsMpEng.exe     0xe0011e643780  24      -       0       False   2023-11-30 17:52:41.000000
2032    908     dasHost.exe     0xe0011e857580  3       -       0       False   2023-11-30 17:52:44.000000
1808    508     NisSrv.exe      0xe0011e81a780  7       -       0       False   2023-11-30 17:52:44.000000
2400    828     taskhostw.exe   0xe0011e9b1080  9       -       1       False   2023-11-30 17:52:56.000000
2456    828     sihost.exe      0xe0011e4ef780  7       -       1       False   2023-11-30 17:52:56.000000
2604    484     userinit.exe    0xe0011ea08780  0       -       1       False   2023-11-30 17:52:57.000000
2676    2604    explorer.exe    0xe0011ea24780  40      -       1       False   2023-11-30 17:52:57.000000
2888    600     RuntimeBroker.  0xe0011eab2640  10      -       1       False   2023-11-30 17:52:58.000000
2992    508     SearchIndexer.  0xe0011eb09780  15      -       0       False   2023-11-30 17:52:58.000000
2376    600     ShellExperienc  0xe0011eb7a780  36      -       1       False   2023-11-30 17:52:59.000000
3144    600     SearchUI.exe    0xe0011ec40780  29      -       1       False   2023-11-30 17:53:00.000000
3844    2676    VBoxTray.exe    0xe0011ee49080  11      -       1       False   2023-11-30 17:53:12.000000
3908    2676    OneDrive.exe    0xe0011ee4b780  16      -       1       True    2023-11-30 17:53:13.000000
3692    2676    WinRAR.exe      0xe0011eeba780  4       -       1       False   2023-11-30 17:53:33.000000
228     600     WmiPrvSE.exe    0xe0011bb1a780  10      -       0       False   2023-11-30 17:53:42.000000
3644    2676    cmd.exe 0xe0011bb4e300  1       -       1       False   2023-11-30 17:53:56.000000      N/A
1912    3644    conhost.exe     0xe0011dd6f300  2       -       1       False   2023-11-30 17:53:56.000000
1404    3644    cmd.exe 0xe0011ba2e080  1       -       1       False   2023-11-30 17:54:03.000000      N/A
2168    2676    chrome.exe      0xe0011ee7b080  40      -       1       False   2023-11-30 17:54:30.000000
3672    2168    chrome.exe      0xe0011ec77080  8       -       1       False   2023-11-30 17:54:30.000000
3960    2168    chrome.exe      0xe0011baaf780  13      -       1       False   2023-11-30 17:54:31.000000
3984    2168    chrome.exe      0xe0011badb780  15      -       1       False   2023-11-30 17:54:31.000000
3668    2168    chrome.exe      0xe0011ba0f780  9       -       1       False   2023-11-30 17:54:31.000000
3656    2168    chrome.exe      0xe0011bd7a080  16      -       1       False   2023-11-30 17:54:33.000000
4652    508     svchost.exe     0xe0011c9d5780  3       -       1       False   2023-11-30 17:54:50.000000
5092    508     TrustedInstall  0xe0011c269780  7       -       0       False   2023-11-30 17:55:40.000000
4108    600     TiWorker.exe    0xe0011c284780  4       -       0       False   2023-11-30 17:55:40.000000
4280    796     audiodg.exe     0xe0011c732780  7       -       0       False   2023-11-30 17:56:18.000000
4404    2676    FTK Imager.exe  0xe0011c740780  25      -       1       False   2023-11-30 17:56:23.000000
1884    828     WMIADAP.exe     0xe0011c6f2780  6       -       0       False   2023-11-30 17:56:48.000000
420     600     WmiPrvSE.exe    0xe0011c758640  9       -       0       False   2023-11-30 17:56:48.000000
marek@marek-ubuntu:~/volatility3$
```

Obrázok 15: bežiace procesy

Na obrázku 16 sú červenou farbou zvýraznené procesy, ktoré treba vyšetriť. Vidíme tu: *WinRAR.exe, cmd.exe* a *chrome.exe.* Začnime s *WinRAR.exe*. Aby sme zistili, aký súbor otvoril, použijeme plugin *windows.cmdline*:

python3 vol.py -f lab2_win.raw windows.cmdline | grep WinRAR

```
marek@marek-ubuntu:~/volatility3$ python3 vol.py -f /media/sf_VMshare/lab2_win.raw windows.cmdline | grep WinRAR
3692ressWinRAR.exe      "C:\Program Files\WinRAR\WinRAR.exe" -iext "C:\Users\Marek\Documents\Secret.rar"
marek@marek-ubuntu:~/volatility3$
```

Obrázok 16: archív Secret.rar

Na obrázku vidíme, že WinRAR otvoril archív Secret.rar. Stiahneme si ho a pozrieme sa čo je vnútri. Použijeme plugin *windows.filescan a windows.dumpfiles*:

python3 vol.py -f lab2_win.raw windows.filescan | grep Secret.rar

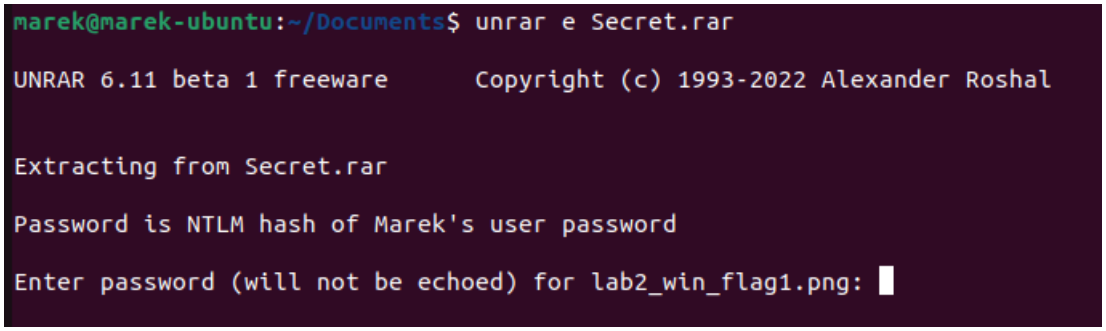
python3 vol.py -f lab2_win.raw -o output/ windows.dumpfiles –virtaddr 0xe0011ef4bb60

```
marek@marek-ubuntu:~/volatility3$ python3 vol.py -f /media/sf_VMshare/lab2_win.raw windows.filescan | grep Secret.rar
0xe0011ef4bb60.0\Users\Marek\Documents\Secret.rar       216
marek@marek-ubuntu:~/volatility3$ python3 vol.py -f /media/sf_VMshare/lab2_win.raw -o /home/marek/Documents/ windows.dumpfiles --virtaddr 0xe0011ef4bb60
Volatility 3 Framework 2.5.2
Progress:  100.00               PDB scanning finished
Cache   FileObject      FileName        Result

DataSectionObject       0xe0011ef4bb60  Secret.rar      Error dumping file
marek@marek-ubuntu:~/volatility3$
```

Obrázok 18: extrakcia archívu

Vo výstupe síce vypísalo chybovú hlášku „Error dumping file“, no operácia prebehla úspešne a môžeme sa pozrieť na obsah súboru. Extrahujeme súbor.



Obrázok 17: extrahovanie archívu

Súbor je však zaheslovaný. V komentári k súboru máme pomôcku, že heslo je NTLM hash Marekovho používateľského hesla. Windows uchováva dva hashe pre každé používateľské heslo. Prvý hash je extrémne nezabezpečený a zastaraný hash používajúci LANMAN algoritmus. Operačné systémy Windows od verzie Vista už nepoužívajú tieto hashe, takže ich miesto je vyplnené fiktívnou hodnotou začínajúcou písmenami „aad“. Druhý hash je NTLM hash, lepší ako LANMAN, ale stále pomerne nebezpečný, pretože sa dá oveľa ľahšie prelomiť ako napríklad hashe v Linuxe alebo Mac OS. Na extrakciu hashov používateľských hesiel slúži plugin *windows.hashdump*:

python3 vol.py -f lab2_win.raw windows.hashdump

Obrázok 18: hashe používateľských hesiel

Použijeme NTLM hash Marekovho hesla na odomknutie archívu a dostaneme obrázok s flagom:

Obrázok 19: prvý flag

Presunieme sa na proces *cmd.exe*. Cez CMD sa v operačnom systéme Windows dá robiť množstvo úkonov a v reálnej analýze by sme museli vyskúšať jednu možnosť za druhou. Teraz sa budeme sústrediť len na jeden úkon. Cez CMD sa dajú nastavovať premenné prostredia t.j. environment variables. Pozrieme sa teda či nenájdeme nejakú podozrivú premennú. Použijeme plugin *windows.envars*:

python3 vol.py -f lab2_win.raw windows.envars

```
3644    cmd.exe 0x251e401400    PROCESSOR_IDENTIFIER    AMD64 Family 25 Model 80 Stepping 0, AuthenticAMD
3644    cmd.exe 0x251e401400    PROCESSOR_LEVEL 25
3644    cmd.exe 0x251e401400    PROCESSOR_REVISION      5000
3644    cmd.exe 0x251e401400    ProgramData     C:\ProgramData
3644    cmd.exe 0x251e401400    ProgramFiles    C:\Program Files
3644    cmd.exe 0x251e401400    ProgramFiles(x86)       C:\Program Files (x86)
3644    cmd.exe 0x251e401400    ProgramW6432    C:\Program Files
3644    cmd.exe 0x251e401400    PROMPT  $P$G
3644    cmd.exe 0x251e401400    PSModulePath    C:\Windows\system32\WindowsPowerShell\v1.0\Modules\
3644    cmd.exe 0x251e401400    PUBLIC  C:\Users\Public
3644    cmd.exe 0x251e401400    RANDOM  ZmxhZ3tXM2xjMG0zX1QwXyRUNGczXzJfT2ZfTDRCXzJ9Cg==
3644    cmd.exe 0x251e401400    SESSIONNAME     Console
3644    cmd.exe 0x251e401400    SystemDrive     C:
3644    cmd.exe 0x251e401400    SystemRoot      C:\Windows
3644    cmd.exe 0x251e401400    TEMP    C:\Users\Marek\AppData\Local\Temp
3644    cmd.exe 0x251e401400    TMP     C:\Users\Marek\AppData\Local\Temp
3644    cmd.exe 0x251e401400    USERDOMAIN      DESKTOP-A5BSAIF
3644    cmd.exe 0x251e401400    USERDOMAIN_ROAMINGPROFILE       DESKTOP-A5BSAIF
3644    cmd.exe 0x251e401400    USERNAME        Marek
```

Obrázok 20: premenné prostredia

Vo výstupe si všimnite premennú s názvom RANDOM a jej hodnotu. Vyzerá ako base64 text. Skúsime ho dekódovať. V Linuxe sa to dá urobiť takto:

```
marek@marek-ubuntu:~$ echo ZmxhZ3tXM2xjMG0zX1QwXyRUNGczXzJfT2ZfTDRCXzJ9Cg== | base64 --decode
flag{W3lc0m3_T0_$T4g3_2_Of_L4B_2}
marek@marek-ubuntu:~$
```

Obrázok 21: dekódovanie hodnoty premennej – druhý flag

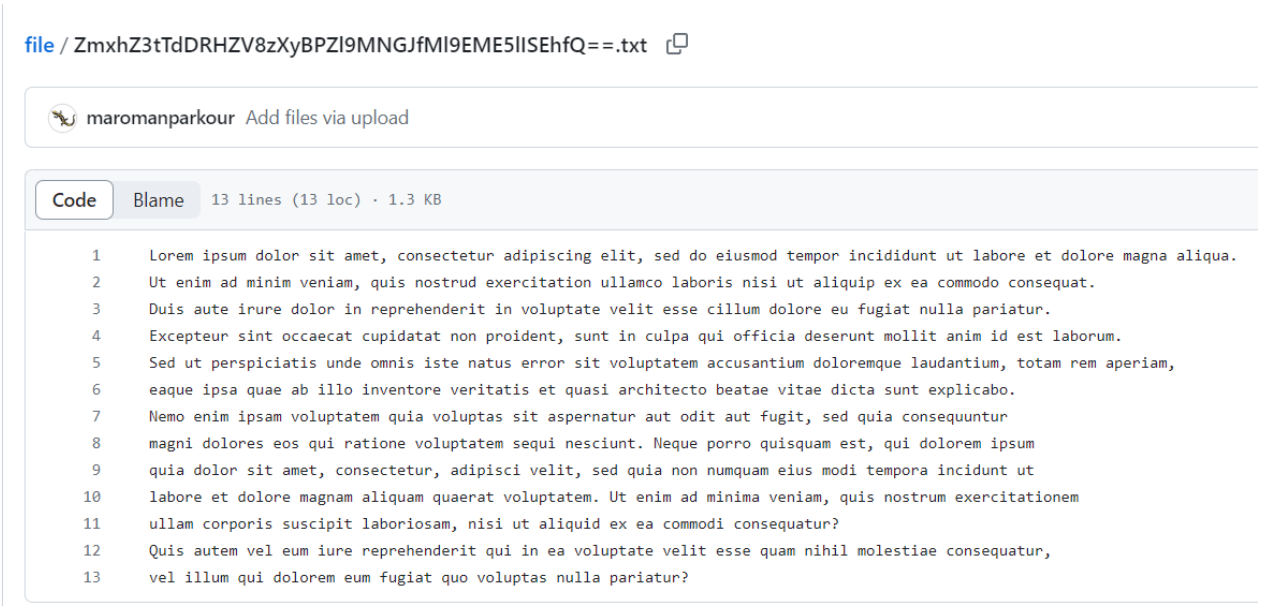
A máme druhý flag. Už chýba len posledný. Ten sa skrýva niekde v procese *chrome.exe*. Použijeme plugin *windows.vadyarascan* spolu so skriptom na editovanie výstupu, aby bol prehľadnejší:

```
python3 vol.py -f lab2_win.raw windows.vadyarascan --pid 2168 --yara-rules "/http.{100}?/" | awk '
  /^0x/{
      S=""
      for(i=5; i<NF; i++) {
          S=S sprintf("%s", $i)
      }
      print "PID:" $2 "\tRule:" $3 "\tComponent:" $4
      system("echo " S " | xxd -r -ps | xxd -o " $1 " | sed \"s/^/0x/\"")
      print ""
      next
  }
  /^Offset/{next}
  {print}
'
```

```
PID:2168        Rule:r1 Component:$a
0x7ffc91a83b40: 6874 7470 733a 2f2f 6769 7468 7562 2e63  https://github.c
0x7ffc91a83b50: 6f6d 2f6d 6172 6f6d 616e 7061 726b 6f75  om/maromanparkou
0x7ffc91a83b60: 722f 6669 6c65 2f62 6c6f 622f 6d61 696e  r/file/blob/main
0x7ffc91a83b70: 2f5a 6d78 685a 3374 5464 4452 485a 5638  /ZmxhZ3tTdDRHZV8
0x7ffc91a83b80: 7a58 7942 505a 6c39 4d4e 474a 664d 6c39  zXyBPZl9MNGJfMl9
0x7ffc91a83b90: 454d 4535 6c49 5345 6866 513d 3d2e 7478  EME5lISEhfQ==.tx
0x7ffc91a83ba0: 7400 0000 0000 0000 0000 0000 0000 0000  t...............
0x7ffc91a83bb0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7ffc91a83bc0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
```

Obrázok 22: chrome história

Našli sme link na Github stránku. Link sa odkazuje na nejaký textový súbor. Skopírujeme si link a pozrieme sa, čo tam je:



Obrázok 23: súbor na Git hube

Vyzerá to, že sme sa dostali do slepej uličky ale nie je to celkom tak. Súbor síce obsahuje len Lorem ipsum text, ale všimnite si jeho názov. Vo forenznej analýze je potrebné si všímať detaily a ja keď to vyzerá, že táto cesta nikam nevedie, v skutočnosti to môže byť celkom inak. Vyzerá ako base64 text. Skopírujeme si ho a dekódujeme ho:

```
marek@marek-ubuntu:~$ echo ZmxhZ3tTdDRHZV8zXyBPZl9MNGJfMl9EME5lISEhfQ== | base64 --decode
flag{St4Ge_3_ Of_L4b_2_D0Ne!!!}marek@marek-ubuntu:~$
```

Obrázok 23: dekódovanie názvu súboru – tretí flag

3. Lab3 – Windows

Tretiu a poslednú úlohu tohto materiálu už budete riešiť každý sám. Do virtuálneho počítača s Ubuntu si z Github stránky stiahnite súbor **lab3_win.raw** a premiestnite ho do priečinka s Volatility3. Presuňte sa do priečinka s Volatility3, aby bol vašim pracovným priečinkom.

Znenie úlohy:

Bežnému používateľovi Windowsu z neznámych dôvodov zlyhal počítač. Bol však nastavený tak, aby pri neočakávanej poruche automaticky vytvoril obraz operačnej pamäte. Používateľ oznámil, že na počítači mal správcu hesiel k jeho súkromným účtom a potrebuje sa k nim dostať. Taktiež povedal, že posielal nejaké e-maily. Vašou úlohou bude pracovať ako forenzný analytik a analyzovať obraz jeho operačnej pamäte.

4. Referencia a syntax príkazov

Tu nájdete krátky popis a syntax každého pluginu, ktorý budete potrebovať pri analýze jednotlivých obrazov operačnej pamäte:

- python3 vol.py -f memory_dump.raw windows.pslist
  - vypíše zoznam bežiacich procesov a ich PID a PPID
- python3 vol.py -f memory_dump.raw windows.cmdline
  - vypíše postupne aktiváciu procesov a ich manipuláciu so súbormi
- python3 vol.py -f memory_dump.raw windows.envars
  - vypíše zoznam premenných prostredia
- python3 vol.py -f memory_dump.raw windows.hashdump
  - vypíše zoznam používateľov a ich hash hodnoty
- python3 vol.py -f memory_dump.raw windows.filescan
  - vypíše zoznam všetkých súborov v systéme
- python3 vol.py -f memory_dump.raw windows.vadyarascan --pid 0000 --yara-rules „string“
  - vyhľadá zadaný reťazec v yara-rules v pamäti daného procesu
- python3 vol.py -f memory_dump.raw -o out_dir/ windows.memmap --dump --pid 0000
  - extrahuje celý pamäťový úsek patriaci danému procesu do súboru
- python3 vol.py -f memory_dump.raw -o out_dir/ windows.dumpfiles --virtaddr 0xe00000000000
  - extrahuje celý pamäťový úsek patriaci virtuálnej pamäťovej adrese do súboru