

Put Student Name(s) ↓		Put Student IDs ↓	Due Date	Grade Weight
Ishan Aakash Patel		146151238	As Posted	6%
Name	Lab3: Memory Forensics Challenge			
Instructions	<ul style="list-style-type: none"> <li>• It is an Individual assignment. Put your name + Student ID in the empty spaces above.</li> <li>• Submit via the BB relevant link ONLY. NO submission via email please. Be sure to submit the final version file ONLY.</li> <li>• Show your genuine signs of your work is done on your machine. This includes: <ul style="list-style-type: none"> <li>◦ Screenshots that show your desktop background with Date/Time.</li> <li>◦ Show a pop-up bx that shows "your name + IP".</li> <li>◦ Show your logged account when applicable. Optional: Your photo.</li> </ul> </li> <li>• Submit your report name: CYT215-Lab3-Student Name &amp; ID</li> </ul>			
Challenge Scenario	<p>You are a memory forensic analyst. One of the clients of your company, lost the access to their system due to an unknown error. He is supposedly a very popular "environmental" activist. As a part of the investigation, he told us that his go to applications are browsers, his password managers etc. We hope that you can dig into this memory dump and find his important stuff and give it back to us.</p>			
Challenge Questions To be Answered	<p>In this challenge, there are 3 flags to find. You will need to use Volatility to perform a memory dump and investigate to locate the flags. This is a Capture the Flag (CTF) exercise, where the objective is to discover the hidden flags.</p> <p>To help you get started, there are several write-ups available online. Here is one useful resource:</p> <p><a href="https://mox.me/posts/writeups/memlabs-lab2/">https://mox.me/posts/writeups/memlabs-lab2/</a>  <a href="https://www.youtube.com/watch?v=c7DTCH2ajqM">https://www.youtube.com/watch?v=c7DTCH2ajqM</a></p> <p>Using the write-up linked above and the YouTube as a reference, you will create your own documentation on how you solved the challenge and found the 3 flags. Your write-up should include clear screenshots showing each step you took to find all 3 flags on your system (with time stamps from the Linux command line). This will ensure that you are not copying write ups from online and trying the challenge yourself on your machine. Any copied screenshots or writeups will receive an automatic ZERO. Please run the tools on your own machine/ Virtual Machine. Utilize the provided write-up and other online resources to learn how to use Volatility and analyze memory files for artifacts.</p>			

At the end of your report, answer the following question:

1. What is the purpose of Volatility?
2. Why is it important to analyze memory dumps?
3. How difficult did you find this challenge?

Prerequisites –

Install a Linux VM

On this VM, install Volatility. Guides below:

<https://www.youtube.com/watch?v=AJ0denKsXyw&t=528s> (volatility 3)

<https://github.com/volatilityfoundation/volatility3> (Volatility 3)

<https://github.com/volatilityfoundation/volatility> (Volatility 2)

<https://blog.onfvp.com/post/volatility-cheatsheet/> - Cheat sheet for commands Volatility 2 vs Volatility 3

Students Work	<ul style="list-style-type: none"><li>• Go to the challenge <a href="https://github.com/stuxnet999/MemLabs/tree/master/Lab%202">https://github.com/stuxnet999/MemLabs/tree/master/Lab%202</a></li><li>• Download the raw image on your VIRTUAL MACHINE. <a href="https://mega.nz/file/ChoDHaja#1XvuQd49c7-7kgJvPXIEAst-NXi8L3ggwienE1uoZTk">https://mega.nz/file/ChoDHaja#1XvuQd49c7-7kgJvPXIEAst-NXi8L3ggwienE1uoZTk</a></li></ul>
---------------	---

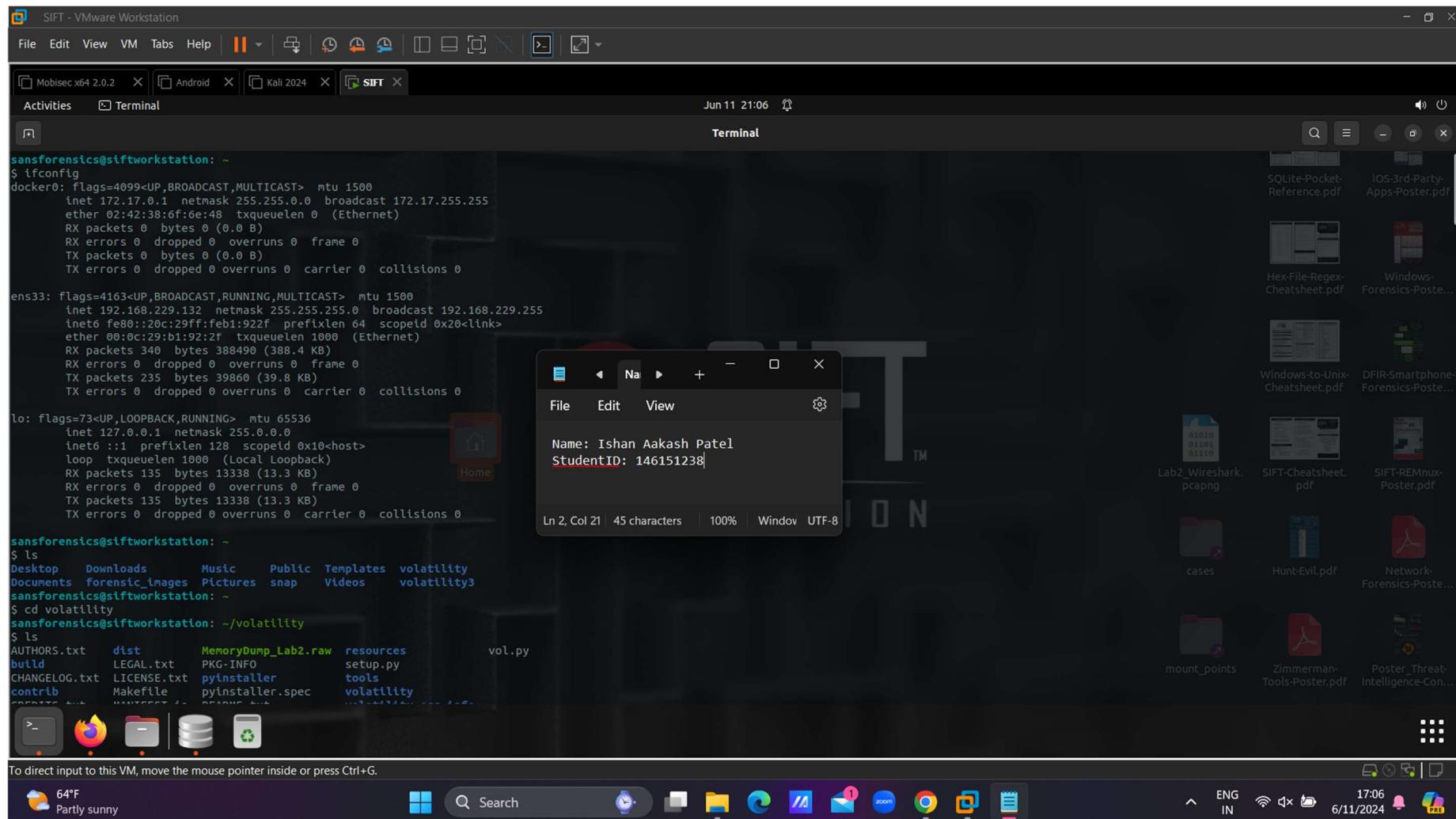
required for this activity	<ul style="list-style-type: none"> <li>Follow the following writeups and tutorials to find the 3 flags to the challenge:             <a href="https://mox.me/posts/writeups/memlabs-lab2/">https://mox.me/posts/writeups/memlabs-lab2/</a>   <a href="https://www.youtube.com/watch?v=c7DTCH2ajqM">https://www.youtube.com/watch?v=c7DTCH2ajqM</a> </li> </ul> <p>There are several other resources on google as well for this challenge. Use them all as a reference to learn.</p> <ul style="list-style-type: none"> <li>Document each step (WITH command line screenshots) to show steps how you retrieved each flag.</li> </ul> <p>At the end of your report, answer the following questions:</p> <ol style="list-style-type: none"> <li>What is the purpose of Volatility?</li> <li>Why is it important to analyze memory dumps?</li> <li>How difficult did you find this challenge?</li> </ol> <p><b>Your write up/report can go below this table.</b></p> <p><b>ENSURE THAT YOU ARE DOING THIS LAB IN A VIRTUAL MACHINE.</b></p>
Grading Alerts	<ul style="list-style-type: none"> <li>If you do NOT use this template or delete any part of it or use any other template, you will be degraded.</li> <li>If you do NOT follow the file naming convention, you will be degraded.</li> <li>If you do NOT submit your file in PDF; you will be degraded.</li> <li>If you do NOT show your account real name (when applicable); you will be degraded.</li> <li>If you do NOT show your machine desktop background (with date &amp; time) and IP, you will be degraded.</li> <li>If you do NOT write (in your own words) your learning experience for the activity practices, you will be degraded.</li> </ul>

**START WRITE UP BELOW THIS**

Memlabs Lab-2

I am using SIFT virtual machine for this particular Lab and I installed volatility in the SIFT VM.

My IP address.



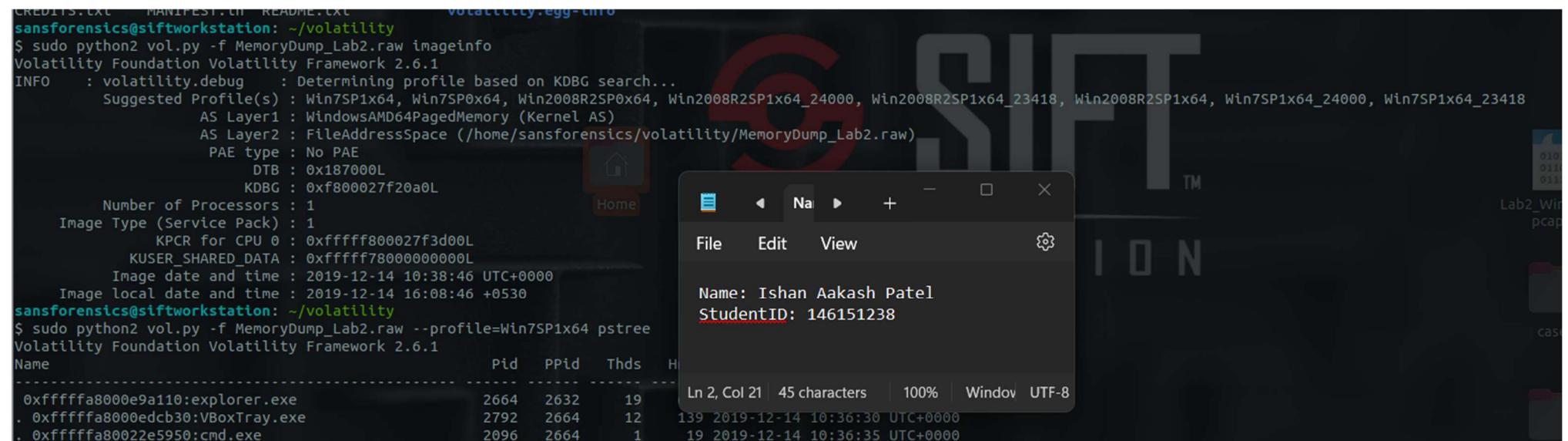
## Stage 1

### Step 1: Identify the Memory Profile

First, we need to identify the correct profile for Volatility to use with the memory dump.

```
vol.py -f MemoryDump_Lab2.raw imageinfo
```

Based on the output, we determine that `Win7SP1x64` is the appropriate profile.



The screenshot shows a terminal window with the following text:

```
CREDITS.LLC  MANIFEST.LLC  README.LLC      volatility.egg-info
sansforensics@siftworkstation: ~/volatility
$ sudo python2 vol.py -f MemoryDump_Lab2.raw imageinfo
Volatility Foundation Volatility Framework 2.6.1
INFO    : volatility.debug    : Determining profile based on KDBG search...
Suggested Profile(s) : Win7SP1x64, Win7SP0x64, Win2008R2SP0x64, Win2008R2SP1x64_24000, Win2008R2SP1x64_23418, Win2008R2SP1x64, Win7SP1x64_24000, Win7SP1x64_23418
          AS Layer1 : WindowsAMD64PagedMemory (Kernel AS)
          AS Layer2 : FileAddressSpace (/home/sansforensics/volatility/MemoryDump_Lab2.raw)
          PAE type : No PAE
          DTB : 0x187000L
          KDBG : 0xf800027f20a0L
          Number of Processors : 1
          Image Type (Service Pack) : 1
          KPCR for CPU 0 : 0xfffffff800027f3d00L
          KUSER_SHARED_DATA : 0xfffffff78000000000L
          Image date and time : 2019-12-14 10:38:46 UTC+0000
          Image local date and time : 2019-12-14 16:08:46 +0530
sansforensics@siftworkstation: ~/volatility
$ sudo python2 vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 pstree
Volatility Foundation Volatility Framework 2.6.1
Name          Pid  PPid  Thds  H
0xfffffa8000e9a110:explorer.exe    2664  2632  19
. 0xfffffa8000edcb30:VBoxTray.exe  2792  2664  12
. 0xfffffa80022e5950:cmd.exe      2096  2664  1

```

A file dialog is overlaid on the terminal window, containing the following information:

Name: Ishan Aakash Patel  
StudentID: 146151238

File dialog details:  
Name: Ishan Aakash Patel  
StudentID: 146151238  
Ln 2, Col 21 | 45 characters | 100% | Window | UTF-8

## Step 2: Check Command-Line Output

We use the cmdscan plugin to look for unusual command-line activity.

```
vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 cmdscan
```

The screenshot shows the SIFT - VMware Workstation interface. In the top bar, there are tabs for Mobisec x64 2.0.2, Android, Kali 2024, and SIFT. Below the tabs, the Activities and Terminal tabs are visible. The date and time are Jun 11 22:17. A note window titled "Terminal" is open in the bottom right corner, containing student information.

```
ubuntu-desktop-minimal ~ https://www.petermstewart.net/memlabs-memory-forensics-challenges-lab-2-write-up/
needrestart is being skipped since dpkg has failed
E: Sub-process /usr/bin/dpkg returned an error code (1)
sansforensics@siftworkstation: ~/volatility
$ sudo python2 vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 cmdscanner of the system is an environmental activist;
Volatility Foundation Volatility Framework 2.6.1
*****
CommandProcess: conhost.exe Pid: 2068
CommandHistory: 0x3deb10 Application: cmd.exe Flags: Allocated, Reset
CommandCount: 1 LastAdded: 0 LastDisplayed: 0
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x60
Cmd #0 @ 0x3db330: Nothing here kidsP(): -f MemoryDump_Lab2.raw --profile
Cmd #15 @ 0x3a0158: =
Cmd #16 @ 0x3ddc80: >
*****
CommandProcess: conhost.exe Pid: 3852
CommandHistory: 0x16eba0 Application: DumpIt.exe Flags: Allocated
CommandCount: 0 LastAdded: -1 LastDisplayed: -1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x60
Cmd #15 @ 0x130158:
Cmd #16 @ 0x16dd00:
```

Value
C:\Program Files\Common
C:\Program Files (x86)\Common
C:\Program Files\Comon
C:\Windows\System32\cmd
cmd
C:\Windows\cmd
1
Windows_NT
C:\Windows\System32\Windows\;C:\Windows\System32\Windows\;C:\Windows\System32\WindowsPowerShell\v1.0\

**Name:** Ishan Aakash Patel  
**StudentID:** 146151238

Ln 2, Col 21 | 45 characters | 100% | Window | UTF-8

### **Step 3: Examine Environment Variables**

The challenge description mentions environmental activism. We check environment variables for suspicious strings using the `envvars` plugin.

```
vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 envvars -p 2068
```

```
Cmd #16 @ 0x16dd00:  
sansforensics@siftworkstation:~$ /volatility  
$ sudo python2 vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 envars -p 2068  
Volatility Foundation Volatility Framework 2.6.1  
Pid      Process          Block           Variable          Value  
-----  
2068  conhost.exe        0x00000000003bd810 CommonProgramFiles          C:\Program Files\Common Files  
2068  conhost.exe        0x00000000003bd810 CommonProgramFiles(x86)    C:\Program Files (x86)\Common Files  
2068  conhost.exe        0x00000000003bd810 CommonProgramW6432     C:\Program Files\Common Files  
2068  conhost.exe        0x00000000003bd810 ComSpec            C:\Windows\system32\cmd.exe  
2068  conhost.exe        0x00000000003bd810 FP_NO_HOST_CHECK      NO  
2068  conhost.exe        0x00000000003bd810 NEW_TMP           C:\Windows\ZmxhZt3M2xjMG0zX1QwXYRUNGczXyFFT2ZfTDRCXzJ9  
2068  conhost.exe        0x00000000003bd810 NUMBER_OF_PROCESSORS    1  
2068  conhost.exe        0x00000000003bd810 OS                Windows_NT  
2068  conhost.exe        0x00000000003bd810 Path              C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbm;C:\Windows\System32\WindowsPowerShell\v1.0\  
2068  conhost.exe        0x00000000003bd810 PATHEXT         .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC  
2068  conhost.exe        0x00000000003bd810 PROCESSOR_ARCHITECTURE   AMD64  
2068  conhost.exe        0x00000000003bd810 PROCESSOR_IDENTIFIER    Intel64 Family 6 Model 142 Stepping 9, GenuineIntel  
2068  conhost.exe        0x00000000003bd810 PROCESSOR_LEVEL       6  
2068  conhost.exe        0x00000000003bd810 PROCESSOR_REVISION     8e09  
2068  conhost.exe        0x00000000003bd810 ProgramFiles       C:\Program Files  
2068  conhost.exe        0x00000000003bd810 ProgramFiles(x86)    C:\Program Files (x86)  
2068  conhost.exe        0x00000000003bd810 ProgramW6432      C:\Program Files  
2068  conhost.exe        0x00000000003bd810 PSModulePath      C:\Windows\system32\WindowsPowerShell\v1.0\Modules\  
2068  conhost.exe        0x00000000003bd810 SystemDrive       C:  
2068  conhost.exe        0x00000000003bd810 SystemRoot        C:\Windows  
2068  conhost.exe        0x00000000003bd810 TEMP             C:\Windows\TEMP  
T0x00000000003bd810bTMPNEW_TMPcontains what appearC:\Windows\TEMPstring:  
2068  conhost.exe        0x00000000003bd810 USERNAME        SYSTEM  
2068  conhost.exe        0x00000000003bd810 windir          C:\Windows  
2068  conhost.exe        0x00000000003bd810 windows_tracing_flags  3  
2068  conhost.exe        0x00000000003bd810 windows_tracing_logfile  C:\BVTBin\Tests\installpackage\csilogfile.log  
sansforensics@siftworkstation: ~$ /volatility
```

## Step 4: Decode Base64 String

We find a base64 string in the environment variables, specifically in `NEW_TMP`. We decode it using CyberChef to get our first flag.

Base64 String: `ZmxhZ3t3M2xjMG0zX1QwXyRUNGczXyFft2ZfTDRCXzJ9`

Decoded Flag: `flag{w3lc0m3_T0_$T4g3_!_Of_L4B_2}`

The screenshot shows the CyberChef interface with the following details:

- Operations:** A sidebar on the left containing various tools like To Base64, From Base64, To Hex, From Hex, To Hexdump, From Hexdump, URL Decode, Regular expression, Entropy, Fork, Magic, Data format, and Encryption / Encoding.
- Recipe:** Set to "From Base64".
  - Alphabet dropdown: "A-Za-zA-Z0-9+=".
  - Checkboxes: "Remove non-alphabet chars" (checked) and "Strict mode".
- Input:** The base64 string: `ZmxhZ3t3M2xjMG0zX1QwXyRUNGczXyFft2ZfTDRCXzJ9`.
- Output:** The decoded flag: `flag{w3lc0m3_T0_$T4g3_!_Of_L4B_2}`.
- Bottom Panel:** A terminal-like window showing the decoded flag and some user information:

```
Name: Ishan Aakash Patel
StudentID: 146151238
```

## Stage - 2

### Step 1: Analyze Process Tree

We use the `pstree` plugin to display a hierarchical list of processes running on the system at the time of the memory dump. This helps us identify key applications such as web browsers and password managers.

```
vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 pstree
```

The process tree shows that Chrome and the KeePass password manager were running, providing leads for further investigation.

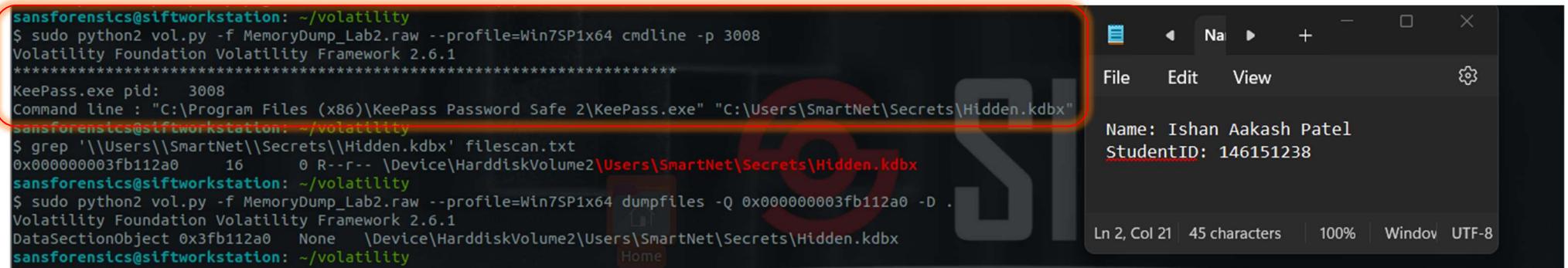
The terminal window displays the output of the `vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 pstree` command. The process tree shows various system processes and two application processes highlighted with a red box: `KeePass.exe` (PID 3008) and `VBoxTray.exe` (PID 1895). A floating window in the foreground contains the student's name and ID: **Name: Ishan Aakash Patel** and **StudentID: 146151238**.

```
IMAGE LOCAL DATE AND TIME : 2019-12-14 10:08:40 +0530
sansforensics@siftworkstation: ~/volatility
$ sudo python2 vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 pstree
Volatility Foundation Volatility Framework 2.6.1
Name          Pid  PPid  Thds  Hnds Time
-----+-----+-----+-----+-----+
0xfffffa8000e9a110:explorer.exe      2664  2632   19   632 2019-12-14 10:36:29 UTC+0000
. 0xfffffa8000edcb30:VBoxTray.exe    2792  2664   12   139 2019-12-14 10:36:30 UTC+0000
. 0xfffffa80022e5950:cmd.exe        2096  2664   1    19 2019-12-14 10:36:35 UTC+0000
. 0xfffffa8002109b30:chrome.exe     2296  2664   27   658 2019-12-14 10:36:45 UTC+0000
.. 0xfffffa8001cc7a90:chrome.exe    2304  2296   8    71 2019-12-14 10:36:45 UTC+0000
.. 0xfffffa8000ea2b30:chrome.exe    2964  2296   13   295 2019-12-14 10:36:47 UTC+0000
.. 0xfffffa8000fae6a0:chrome.exe    2572  2296   8    177 2019-12-14 10:36:56 UTC+0000
.. 0xfffffa800230eb30:chrome.exe    1632  2296   14   219 2019-12-14 10:37:12 UTC+0000
.. 0xfffffa8000eea7a0:chrome.exe    2476  2296   2    55 2019-12-14 10:36:46 UTC+0000
0xfffffa8001c5b2b0:wininit.exe      424   312   3    75 2019-12-14 10:35:30 UTC+0000
. 0xfffffa8001c95320:services.exe   484   424   8    206 2019-12-14 10:35:31 UTC+0000
.. 0xfffffa80020f2b30:taskhost.exe  1928  484   9    154 2019-12-14 10:36:04 UTC+0000
.. 0xfffffa8001dacb30:svchost.exe   876   484   39   962 2019-12-14 10:35:38 UTC+0000
.. 0xfffffa8002105b30:taskeng.exe   1996  876   5    79 2019-12-14 10:36:04 UTC+0000
0xfffffa8001d13060:VBoxService.exe  652   484   14   135 2019-12-14 10:35:36 UTC+0000
. 0xfffffa8001e1eb30:svchost.exe   472   484   12   301 2019-12-14 10:35:42 UTC+0000
. 0xfffffa8001e47740:svchost.exe   1044  484   16   361 2019-12-14 10:35:43 UTC+0000
. 0xfffffa8001f7c060:TCPsvcs.EXE   1412  484   97   97 2019-12-14 10:35:53 UTC+0000
. 0xfffffa800100c060:WmiApSrv.exe  2004  484   6    115 2019-12-14 10:37:05 UTC+0000
.. 0xfffffa8002131340:explorer.exe 1064  2004   37   989 2019-12-14 10:36:05 UTC+0000
.... 0xfffffa800224a8c0:KeePass.exe 3008  1064   12   316 2019-12-14 10:37:56 UTC+0000
.... 0xfffffa80021c0b30:VBoxTray.exe 1895  1064   13   130 2019-12-14 10:36:13 UTC+0000
. 0xfffffa80011aa060:DumpIt.exe    3844  1064   2    45 2019-12-14 10:38:43 UTC+0000
. 0xfffffa8001d76320:svchost.exe   812   484   21   474 2019-12-14 10:35:38 UTC+0000
. 0xfffffa8001df65f0:audiodg.exe   268   812   7    131 2019-12-14 10:35:41 UTC+0000
. 0xfffffa80010e5b30:svchost.exe   1076  484   17   337 2019-12-14 10:38:02 UTC+0000
. 0xfffffa8000cf9220:spoolsv.exe   1208  484   13   279 2019-12-14 10:35:47 UTC+0000
. 0xfffffa8002230b30:sppsvc.exe   2764  484   5    151 2019-12-14 10:38:00 UTC+0000
. 0xfffffa8001cec790:svchost.exe   588   484   12   354 2019-12-14 10:35:35 UTC+0000
. 0xfffffa8001189b30:WmiPrvSE.exe  4004  588   9    15...4 2019-12-14 10:39:00 UTC+0000
. 0xfffffa800105c060:WmiPrvSE.exe  2636  588   12   293 2019-12-14 10:37:02 UTC+0000
. 0xfffffa800101e640:dhcpcsvc.exe 2376  588   9    250 2019-12-14 10:37:40 UTC+0000
. 0xfffffa8001d4ab30:svchost.exe   720   484   7    275 2019-12-14 10:35:37 UTC+0000
. 0xfffffa8001da6930:svchost.exe   852   484   20   417 2019-12-14 10:35:38 UTC+0000
```

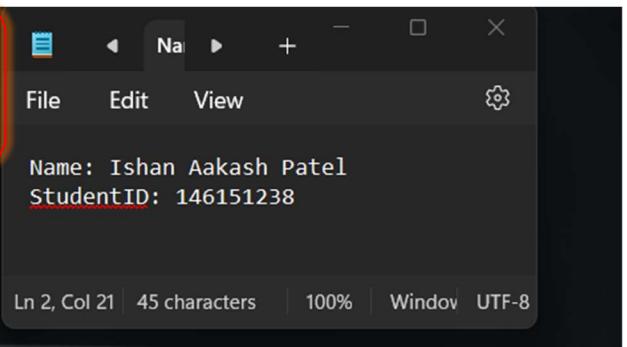
## Step 2: Analyze KeePass Process

We identified the KeePass process (PID 3008) from the process tree. Now, we find the command line associated with it.

```
vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 cmdline -p 3008
```



```
sansforensics@siftworkstation: ~/volatility
$ sudo python2 vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 cmdline -p 3008
Volatility Foundation Volatility Framework 2.6.1
*****
KeePass.exe pid: 3008
Command line : "C:\Program Files (x86)\KeePass Password Safe 2\KeePass.exe" "C:\Users\SmartNet\Secrets\Hidden.kdbx"
sansforensics@siftworkstation: ~/volatility
$ grep '\\Users\\SmartNet\\Secrets\\Hidden.kdbx' filescan.txt
0x00000003fb112a0 16 0 R--r-- \Device\HarddiskVolume2\Users\SmartNet\Secrets\Hidden.kdbx
sansforensics@siftworkstation: ~/volatility
$ sudo python2 vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 dumpfiles -Q 0x00000003fb112a0 -D .
Volatility Foundation Volatility Framework 2.6.1
DataSectionObject 0x3fb112a0 None \Device\HarddiskVolume2\Users\SmartNet\Secrets\Hidden.kdbx
sansforensics@siftworkstation: ~/volatility
```

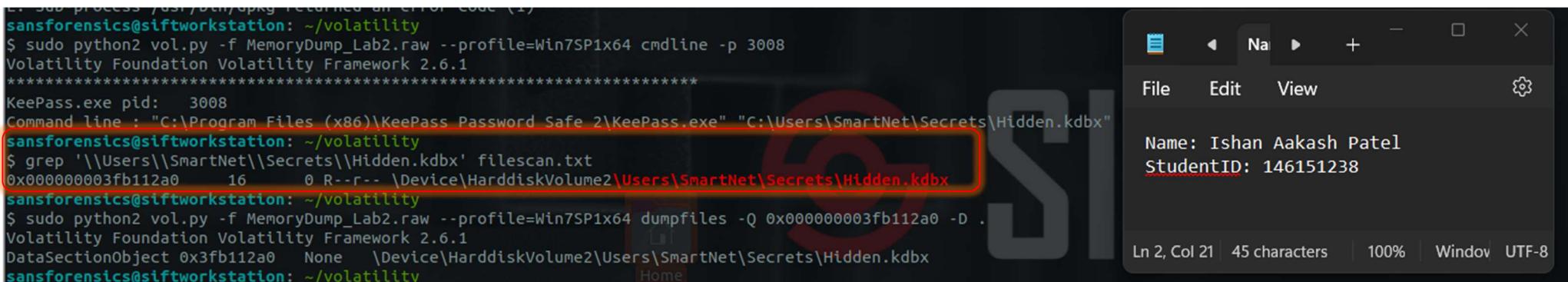


```
Name: Ishan Aakash Patel
StudentID: 146151238
Ln 2, Col 21 | 45 characters | 100% | Window UTF-8
```

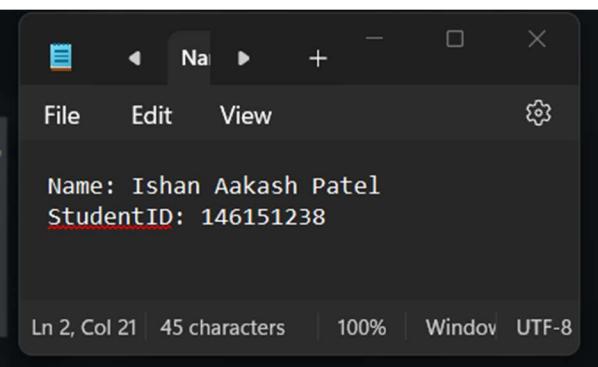
## Step 3: Locate KeePass Database File

We search for the KeePass database file using grep on the filescan output.

```
grep '\\Users\\SmartNet\\Secrets\\Hidden.kdbx' filescan.txt
```



```
E: Sub process /usr/bin/apkg returned an error code (1)
sansforensics@siftworkstation: ~/volatility
$ sudo python2 vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 cmdline -p 3008
Volatility Foundation Volatility Framework 2.6.1
*****
KeePass.exe pid: 3008
Command line : "C:\Program Files (x86)\KeePass Password Safe 2\KeePass.exe" "C:\Users\SmartNet\Secrets\Hidden.kdbx"
sansforensics@siftworkstation: ~/volatility
$ grep '\\Users\\SmartNet\\Secrets\\Hidden.kdbx' filescan.txt
0x00000003fb112a0 16 0 R--r-- \Device\HarddiskVolume2\Users\SmartNet\Secrets\Hidden.kdbx
sansforensics@siftworkstation: ~/volatility
$ sudo python2 vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 dumpfiles -Q 0x00000003fb112a0 -D .
Volatility Foundation Volatility Framework 2.6.1
DataSectionObject 0x3fb112a0 None \Device\HarddiskVolume2\Users\SmartNet\Secrets\Hidden.kdbx
sansforensics@siftworkstation: ~/volatility
```



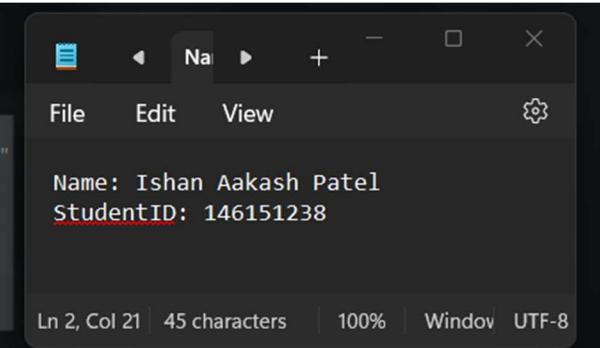
```
Name: Ishan Aakash Patel
StudentID: 146151238
Ln 2, Col 21 | 45 characters | 100% | Window UTF-8
```

#### Step 4: Extract KeePass Database File

We extract the database file from the memory dump using its offset.

```
vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 dumpfiles -Q 0x000000003fb112a0 -D .
```

```
E: Sub-process /usr/bin/dpkg returned an error code (1)
sansforensics@siftworkstation: ~/volatility
$ sudo python2 vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 cmdline -p 3008
Volatility Foundation Volatility Framework 2.6.1
*****
KeePass.exe pid: 3008
Command line : "C:\Program Files (x86)\KeePass Password Safe 2\KeePass.exe" "C:\Users\SmartNet\Secrets\Hidden.kdbx"
sansforensics@siftworkstation: ~/volatility
$ grep '\\Users\\SmartNet\\Secrets\\Hidden.kdbx' filescan.txt
0x000000003fb112a0      16    0 R--r-- \Device\HddiskVolume2\Users\SmartNet\Secrets\Hidden.kdbx
sansforensics@siftworkstation: ~/volatility
$ sudo python2 vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 dumpfiles -Q 0x000000003fb112a0 -D .
Volatility Foundation Volatility Framework 2.6.1
DataSectionObject 0x3fb112a0  None  \Device\HddiskVolume2\Users\SmartNet\Secrets\Hidden.kdbx
sansforensics@siftworkstation: ~/volatility
```



## Step 5: Find KeePass Password

We search for a password using grep on the filescan output and locate a file named `Password.png`.

```
grep -i 'password' filescan.txt
```

## Step 6: Extract Password Image

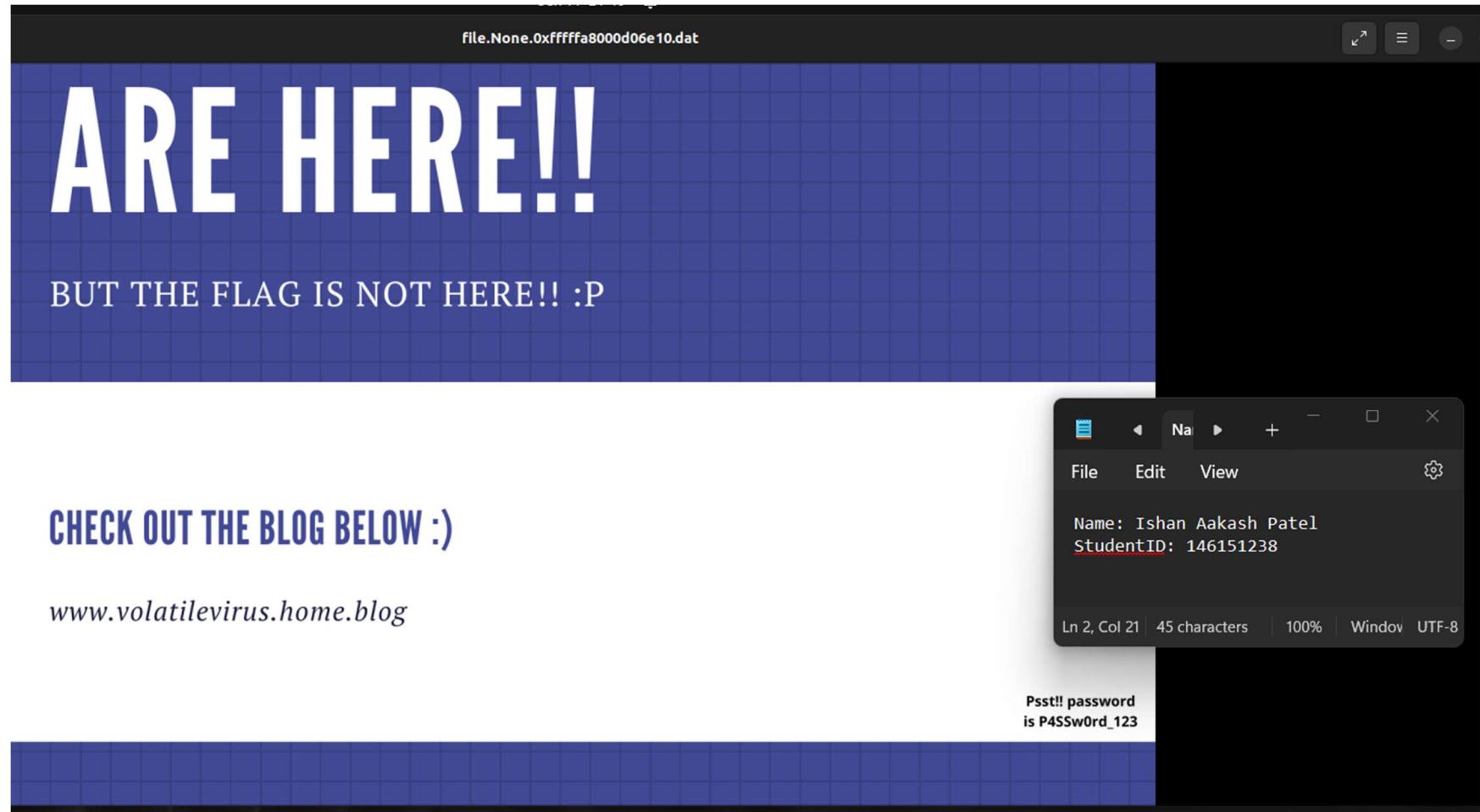
We extract the `Password.png` file from the memory dump using its offset.

```
vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 dumpfiles -Q 0x000000003fce1c70 -D .
Volatility Foundation Volatility Framework 2.6.1
DataSectionObject 0x3fb112a0  None  \Device\HarddiskVolume2\Users\SmartNet\Secrets\Hidden.kdbx
sansforensics@siftworkstation: ~/volatility
$ grep -i 'password' filescan.txt
0x000000003e868370    16      0 R--r-d \Device\HarddiskVolume2\Program Files (x86)\KeePass Password Safe 2\KeePass.exe.config
0x000000003e873070    8       0 R--r-d \Device\HarddiskVolume2\Program Files (x86)\KeePass Password Safe 2\KeePass.exe
0x000000003e8ef2d0   13      0 R--r-d \Device\HarddiskVolume2\Program Files (x86)\KeePass Password Safe 2\KeePass.exe
0x000000003e8f0360    4       0 R--r-d \Device\HarddiskVolume2\Program Files (x86)\KeePass Password Safe 2\KeePass.XmlSerializers.dll
0x000000003eaf7880   15      1 R--r-d \Device\HarddiskVolume2\Program Files (x86)\KeePass Password Safe 2\KeePass.XmlSerializers.dll
0x000000003fb0abc0   10      0 R--r-d \Device\HarddiskVolume2\Program Files (x86)\KeePass Password Safe 2\KeePassLibC64.dll
0x000000003fce1c70    1      0 R--r-d \Device\HarddiskVolume2\Users\Alissa Simpson\Pictures\Password.png
0x000000003fd62f20    2      0 R--r-- \Device\HarddiskVolume2\Program Files (x86)\KeePass Password Safe 2\KeePass.config.xml
0x000000003fecf820   15      0 R--r-d \Device\HarddiskVolume2\Program Files (x86)\KeePass Password Safe 2\unins000.exe
sansforensics@siftworkstation: ~/volatility
$ sudo python2 vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 dumpfiles -Q 0x000000003fce1c70 -D .
Volatility Foundation Volatility Framework 2.6.1
DataSectionObject 0x3fce1c70  None  \Device\HarddiskVolume2\Users\Alissa Simpson\Pictures\Password.png
sansforensics@siftworkstation: ~/volatility
$
```

## Step 7: Retrieve Password from Image

We open the `Password.png` file and find the password in the lower right-hand corner.

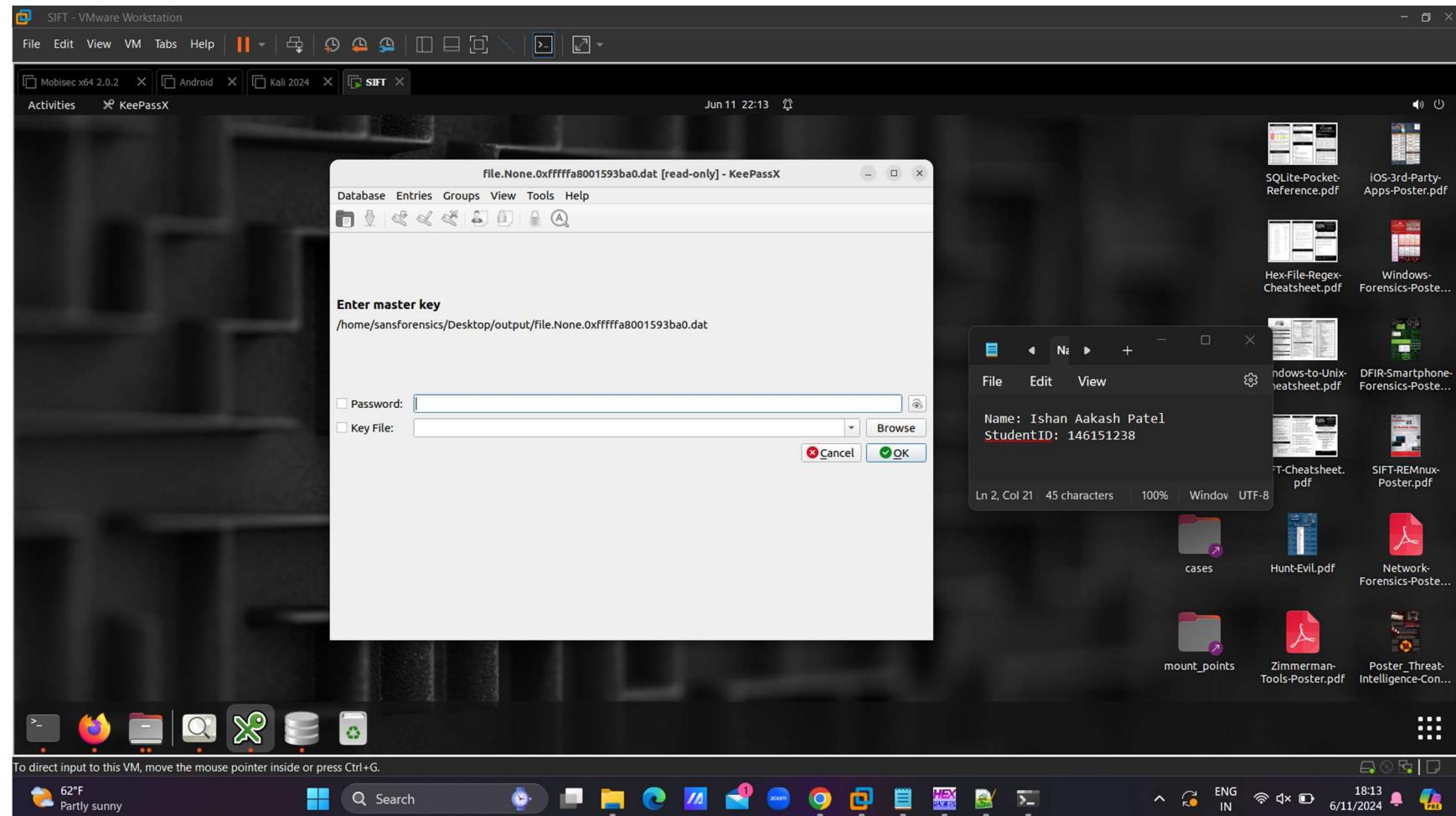
Password: P4SSw0rd\_123

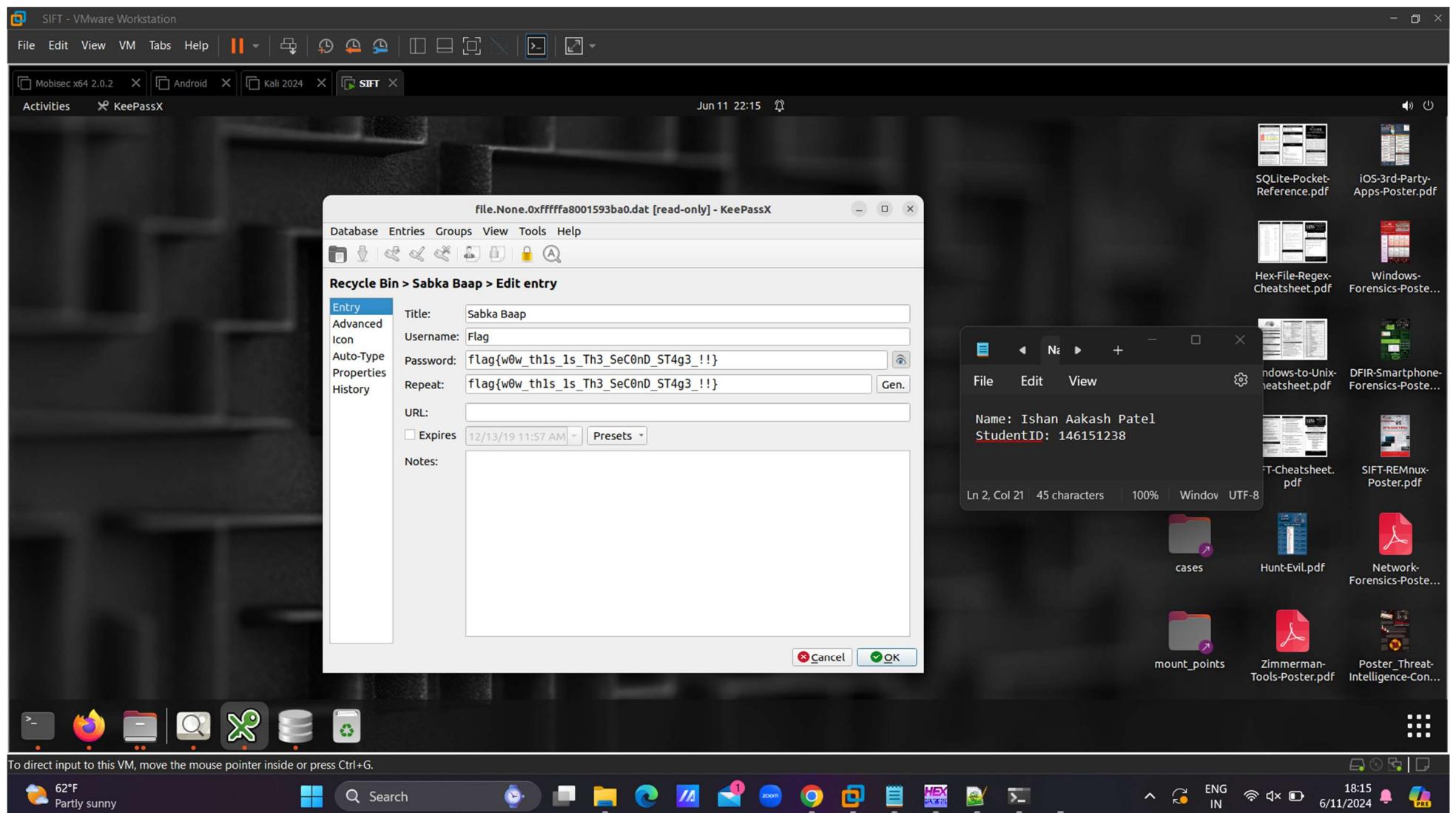


## Step 8: Open KeePass Database

We use the password to open the KeePass database and find the flag for Stage 2. (Note : Install keepassX)

Flag: flag{w0w\_th1s\_1s\_Th3\_SeC0nD\_ST4g3\_! ! }





## Stage 3

### Step 1: Locate Chrome History File

We use grep on the filescan output to find the location of the Chrome History file.

```
grep '\\Chrome\\User Data\\Default\\History' filescan.txt
```

### Step 2: Extract Chrome History File

We extract the History file from the memory dump using its offset.

```
vol.py -f MemoryDump_Lab2.raw --profile=Win7SP1x64 dumpfiles -Q 0x000000003fcfb1d0 -D .
```

The terminal session shows the following steps:

- Execution of `vol.py` to generate a filescan.txt report.
- Use of `grep` to filter the filescan.txt report for the 'History' file.
- Execution of `vol.py` again to dump the specific History file from the memory dump at offset 0x000000003fcfb1d0.
- Execution of `sqlitebrowser` to open the dumped SQLite database.
- Display of the SQLite browser interface showing the extracted data.

The SQLite browser interface displays the following data:

Name	Value
Name	Ishan Aakash Patel
StudentID	146151238

Terminal status bar: Ln 2, Col 21 | 45 characters | 100% | Window UTF-8

### Step 3: Analyze Chrome History File

We open the History file, which is a SQLite database, using the sqlite3 command-line tool or a GUI application.

```
sqlite3 History
```

The screenshot shows the DB Browser for SQLite interface with the following details:

- Title Bar:** DB Browser for SQLite - /home/sansforensics/volatility/file.None.0xfffffa8000efd1d0.dat
- Menu Bar:** File Edit View Tools Help
- Toolbar:** New Database Open Database Write Changes Revert Changes Open Project Attach Database
- Table Selection:** Database Structure, Browse Data, Edit Pragmas, Execute SQL, Table: urls
- Data Grid:** Shows a list of URLs from the history database. The columns are id, url, and title. The grid contains approximately 33 rows of data, with the last row (id 32) highlighted in blue.
- Right Panel (Edit Database Cell):**
  - Mode:** Text
  - Content:** https://mega.nz/#F!TrgSQQTS!H0ZrUzF0B-ZK NM3y9E76lg
  - Type of data currently in cell:** Text / Numeric
  - Character Count:** 50 character(s)
  - Buttons:** Apply, Remote, Identity (Select an identity to connect), DBHub.io, Local, Current Database
- Bottom Status Bar:** Ln 2, Col 21 | 45 characters | 100% | Window | UTF-8 | SQL Log | Plot | DB Schema | Remote | Read only | UTF-8

## Step 4: Investigate MEGA Site

We find that the MEGA file-sharing website was visited and download a ZIP file named MemLab\_Lab2\_Stage3.zip.

Activities Firefox Web Browser

Memlabs Memory Forens X MEGA

https://mega.nz/folder/TrgSQQTS#H0ZrUzF0B-ZKNM3y9E76lg

Download Save to MEGA Search EN Create account Log in

MemLabs\_Lab2\_Stage3

Name	Size	Type	Date added
Important.zip	56 KB	ZIP compressed	12/23/2019, 09:48

Name: Ishan Aakash Patel  
StudentID: 146151238

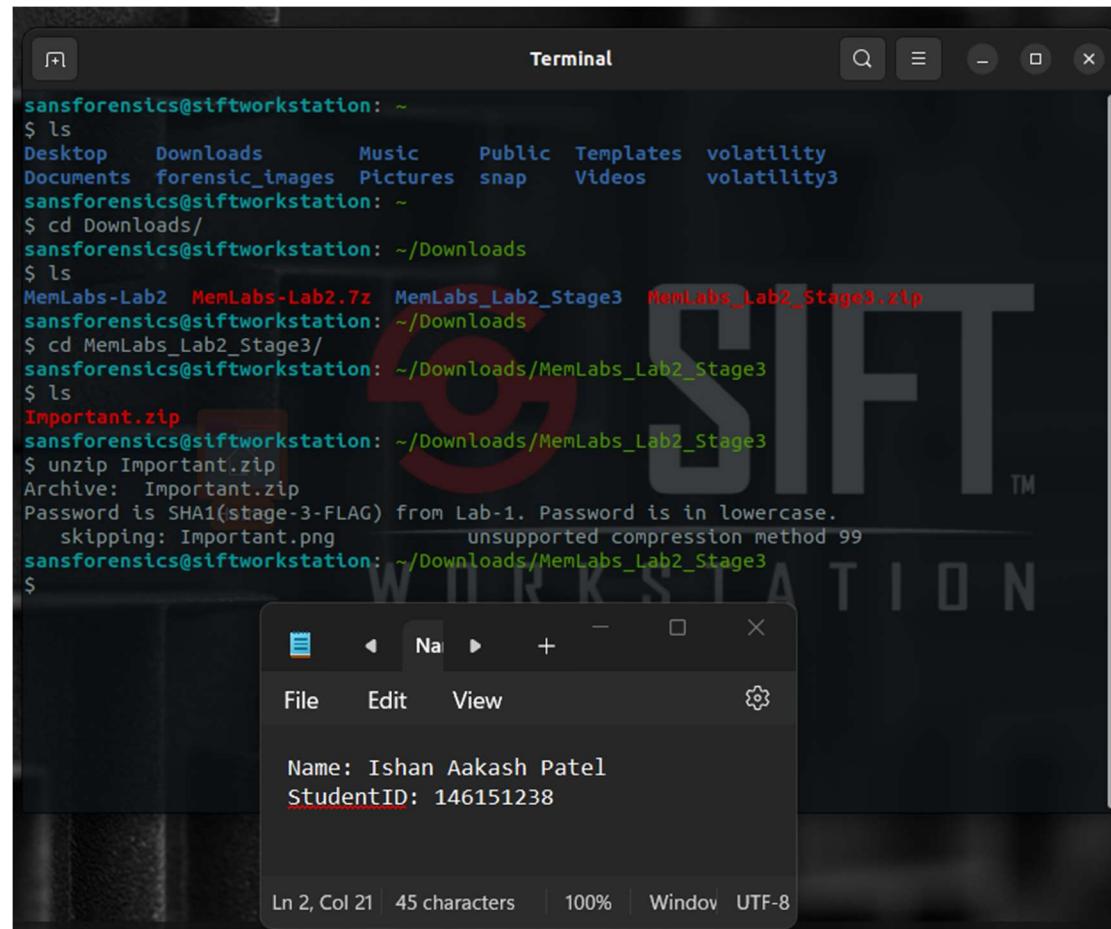
Ln 2, Col 21 | 45 characters | 100% | Window | UTF-8

## Step 5: Extract ZIP File

We use the SHA1 hash from Lab 1 as the password to extract the ZIP file using CyberChef.

Password SHA1 Hash: 6045dd90029719a039fd2d2ebcca718439dd100a

Extracted File: Important.png



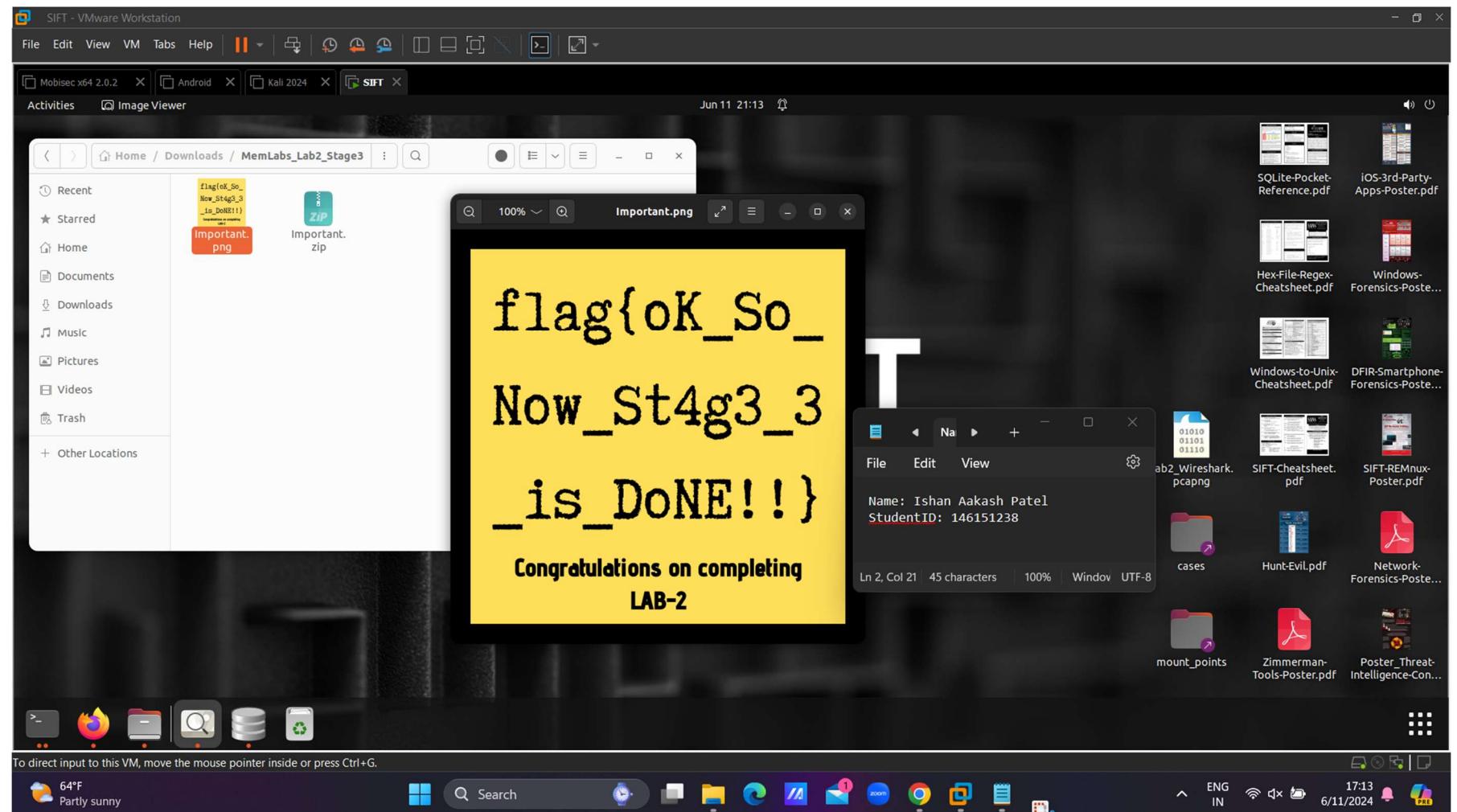
```
sansforensics@siftworkstation: ~
$ ls
Desktop Downloads Music Public Templates volatility
Documents forensic_images Pictures snap Videos volatility3
sansforensics@siftworkstation: ~
$ cd Downloads/
sansforensics@siftworkstation: ~/Downloads
$ ls
MemLabs-Lab2 MemLabs-Lab2.7z MemLabs_Lab2_Stage3 MemLabs_Lab2_Stage3.zip
sansforensics@siftworkstation: ~/Downloads
$ cd MemLabs_Lab2_Stage3/
sansforensics@siftworkstation: ~/Downloads/MemLabs_Lab2_Stage3
$ ls
Important.zip
sansforensics@siftworkstation: ~/Downloads/MemLabs_Lab2_Stage3
$ unzip Important.zip
Archive: Important.zip
Password is SHA1(stage-3-FLAG) from Lab-1. Password is in lowercase.
  skipping: Important.png      unsupported compression method 99
sansforensics@siftworkstation: ~/Downloads/MemLabs_Lab2_Stage3
$
```

The terminal window shows the user's directory path (~Downloads/MemLabs\_Lab2\_Stage3), the command 'unzip Important.zip', and the resulting output where it asks for a password (SHA1(stage-3-FLAG) from Lab-1). The password is entered as 'stage-3-FLAG'. The terminal then extracts the file 'Important.png' which contains the text 'Name: Ishan Aakash Patel' and 'StudentID: 146151238'.

## Step 7: Retrieve Flag from Image

We open `Important.png` and find our third flag.

Flag: `flag{oK_So_Now_St4g3_3_is_DoNE!!}`



## **Questions and Answers**

1. **What is the purpose of Volatility?** Volatility is a tool used for analyzing memory dumps to extract digital artifacts, helping in digital forensics and incident response.
2. **Why is it important to analyze memory dumps?** Analyzing memory dumps is important because it provides insights into the state of a system at a specific point in time, revealing running processes, open files, network connections, and other critical information that can aid in investigations.
3. **How difficult did you find this challenge?** I found this challenge moderately difficult. It required careful attention to detail and a good understanding of Volatility commands, but following the provided resources and write-ups made it manageable.

## **Learning Experience**

This memory forensics challenge was a great learning experience for me. I had to use Volatility to analyze a memory dump and uncover hidden flags. Throughout the process, I learned how to use various Volatility plugins to investigate system processes, browser history, and password managers. It was interesting to see how much information can be retrieved from a memory dump and how it can be used to solve real-world problems. The challenge also helped me improve my skills in using command-line tools and understanding digital forensics concepts.