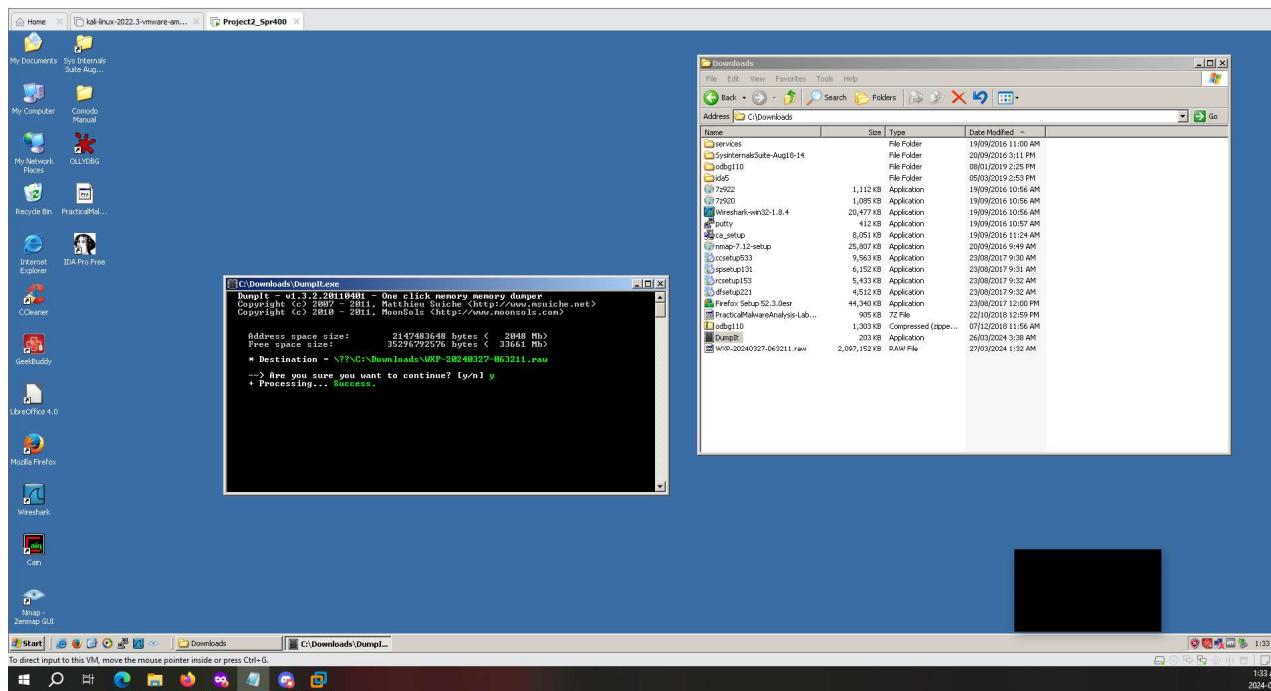


Name: Abdulfatah Abdillahi

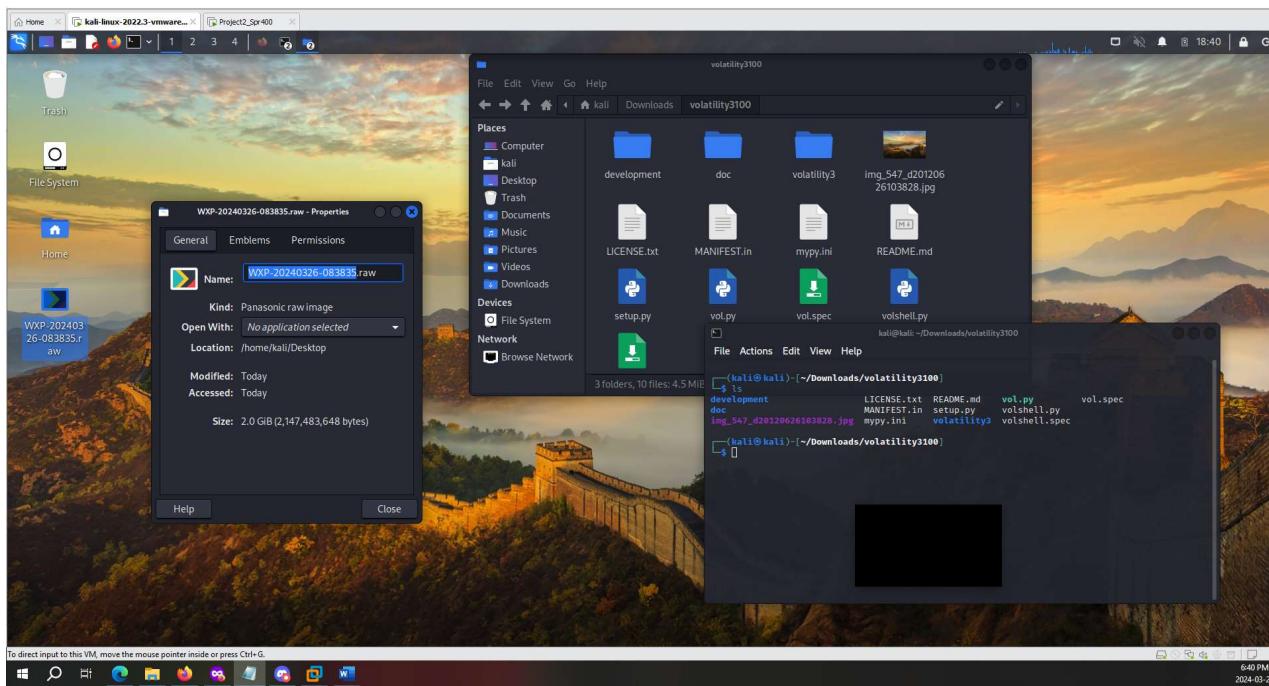
Project: Dumping The Memory

Part 1

Here, you can see me using the dumpit tool after downloading it. Then I used dumpit to dump the memory of my WindowsXP virtual machine. As you can see the memory dump file was saved as WXP-20240327-063211.raw

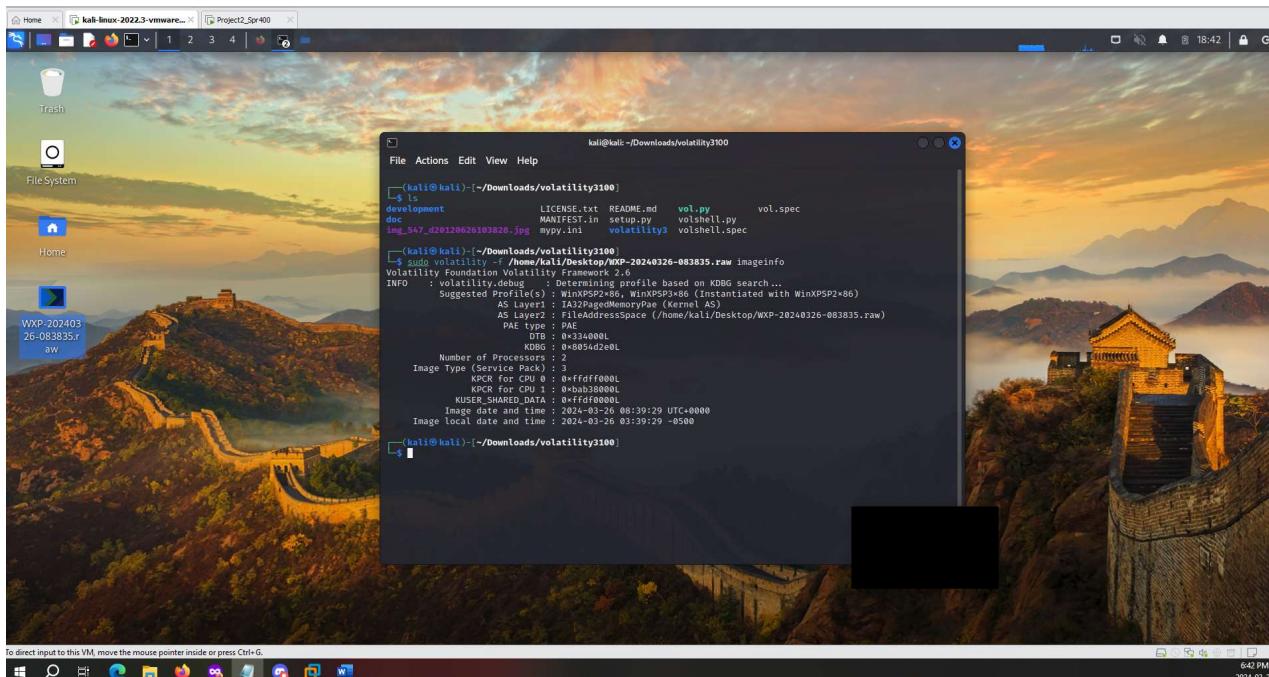


Here, you can see me moving the memory dump to Kali to analyze it using volatility.



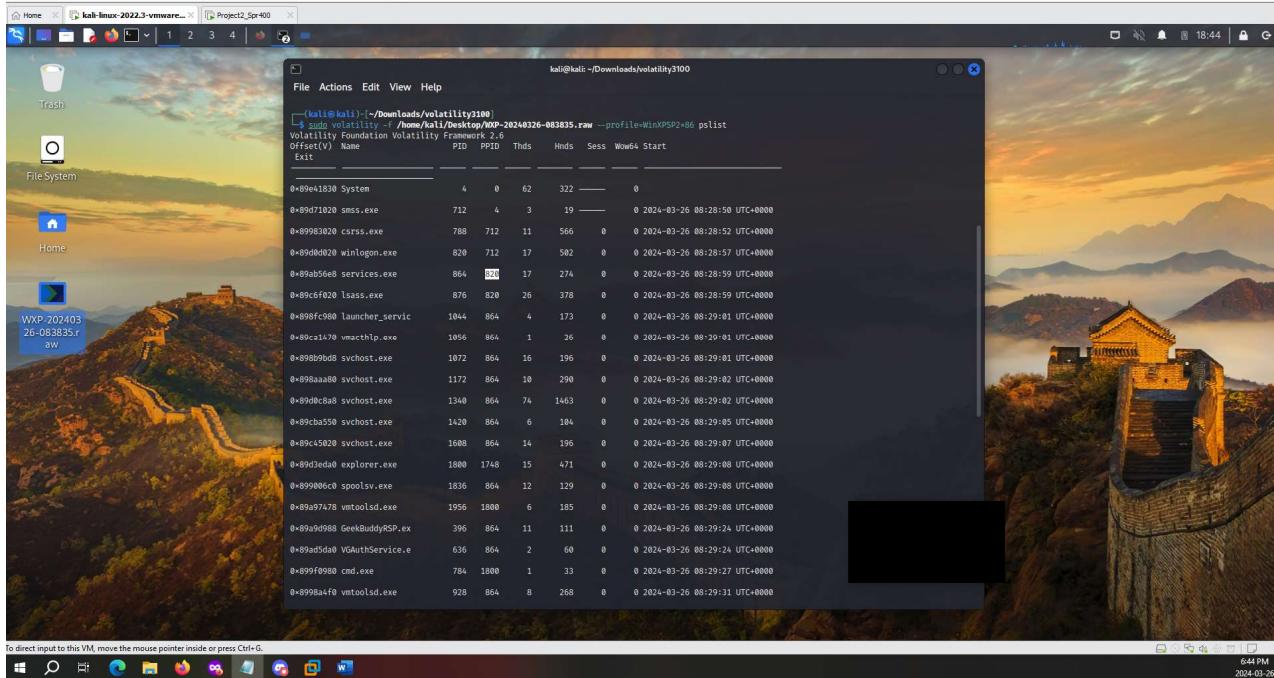
Part 2

Here, you can see me using the **imageinfo** plugin to determine the OS profile of the memory dump. This is necessary because you cannot start the Volatility investigation without knowing the OS profile. Volatility has determined the profile to be WinXPSP2x86.

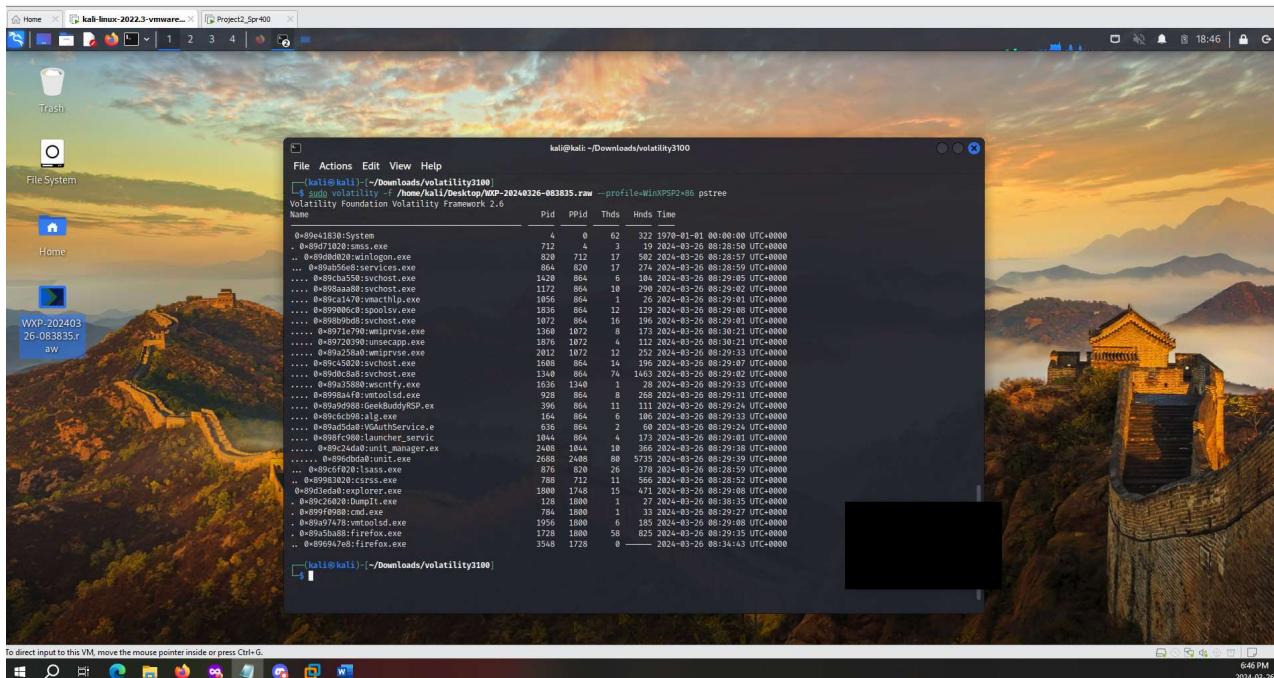


Processes and DLLs

- **Plist:** This command lists the running processes in the memory dump. Meaning it will give you a list of processes that were running when the memory dump was taken. For example, it shows **firefox.exe** process which is responsible for launching and managing the Firefox web browser on the computer.

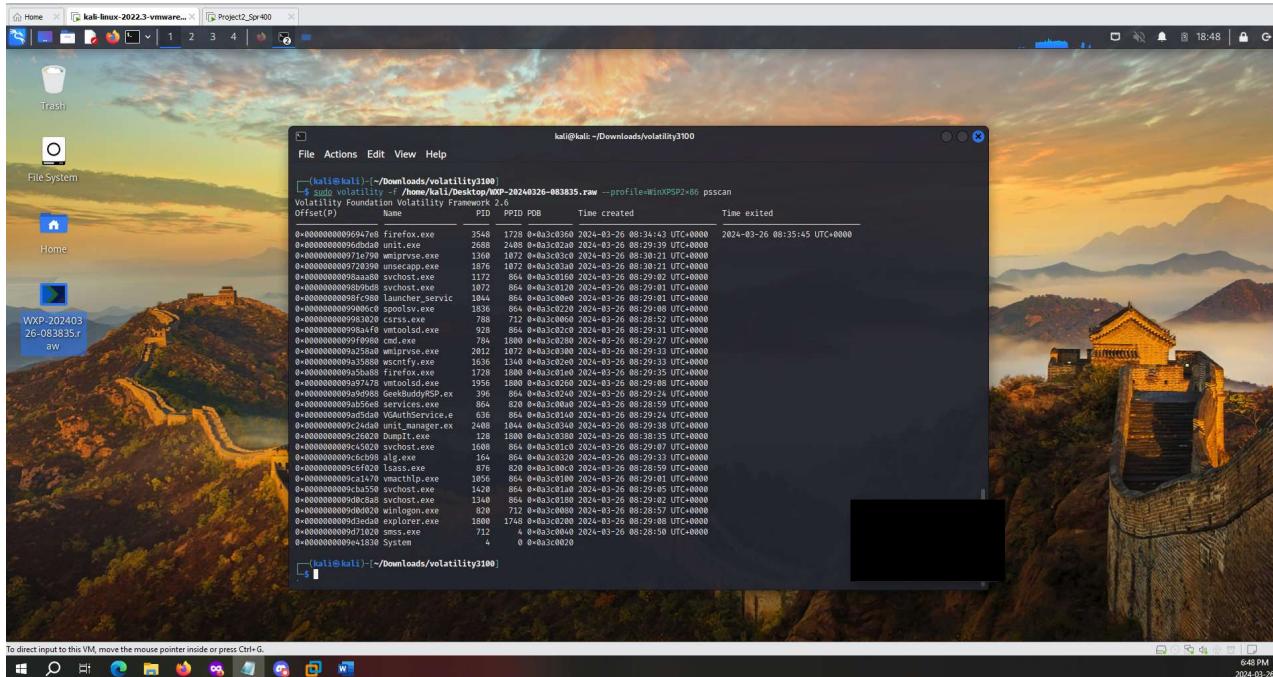


- **Pstree:** This command provides a tree-based view of the running processes. Meaning it will give you a tree view of the processes, showing parent-child relationships. For example, smss.exe (PID 712) is a child of System (PID 4), and winlogon.exe (PID 820) is a child of smss.exe (PID 712). This continues down the tree, showing the hierarchy of processes.



- **Psscan:** This command scans for EPROCESS blocks in memory, which can sometimes find processes that are not in the active process list. This can be useful for finding hidden or

terminated processes. For example, it shows you various system and user processes running, such as System, smss.exe, csrss.exe, etc.



File System

- **mftparser:** This command is used to parse the Master File Table (MFT) from a memory dump. This can provide valuable information about files on the system. For example, it shows several attributes:
 - **\$STANDARD_INFORMATION:** This contains standard information about the file, such as its creation time, modification time, MFT altered time, and access time. It also shows the type of the file (e.g., Archive).
 - **\$FILE_NAME:** This contains the name of the file and its path. It also shows the same timestamps as in **\$STANDARD_INFORMATION**. There can be multiple **\$FILE_NAME** attributes for a single file, representing different names of the file (e.g., short name and long name).
 - **\$DATA:** This attribute contains the actual data of the file. In this case, it seems to be empty.
 - **\$OBJECT_ID:** This attribute contains the object ID of the file, which is a unique identifier that can be used to track the file even if it's moved to another volume.

```

/home kali-kali:~$ ./Downloads/volatility3100
(kali㉿kali)-[~/Downloads/volatility3100]
└─$ sudo volatility -f /home/kali/Desktop/WXP-20240326-083835.raw --profile=WinXPSP2x86 mftparser
Volatility Foundation Volatility Framework 2.6
Scanning for MFT entries and building directory, this can take a while
*****MFT entry found at offset 0x45:000
Attribute: In Use & File
Record Number: 22108
Link count: 2

$STANDARD_INFORMATION
Creation Modified MFT Altered Access Date Type
2024-03-26 08:35:20 UTC+0000 2024-03-26 08:35:32 UTC+0000 2024-03-26 08:35:32 UTC+0000 2024-03-26 08:35:32 UTC+0000 Archive

$FILE_NAME
Creation Modified MFT Altered Access Date Name/Path
2024-03-26 08:35:20 UTC+0000 2024-03-26 08:35:20 UTC+0000 2024-03-26 08:35:20 UTC+0000 2024-03-26 08:35:20 UTC+0000 DOCUMENT\1\student\LOCALS-1\APPLIC-C\1\Mozilla\Firefox\Profiles\U13MBH-1.DEF\cache2\entries\40C5E27E7E90168F1E117E771D2D0B4B10096CA

$FILE_NAME
Creation Modified MFT Altered Access Date Name/Path
2024-03-26 08:35:20 UTC+0000 2024-03-26 08:35:20 UTC+0000 2024-03-26 08:35:20 UTC+0000 2024-03-26 08:35:20 UTC+0000 DOCUMENT\1\student\LOCALS-1\APPLIC-C\1\Mozilla\Firefox\Profiles\U13MBH-1.DEF\cache2\entries\40C5E27E7E90168F1E117E771D2D0B4B10096CA

$DATA
$OBJECT_ID
Object ID: 40000000-0000-0000-0000-000000000000
Birth Volume ID: 3af70000-0000-3a5f-0000-000000000000
Birth Object ID: 3106a0ed-0000-51cf-ffff-ffff82794711
Birth Domain ID: 00000000-0000-0000-0000-000000000000
*****
```

Registry

- **Hivescan:** This command locates the registry hives in memory. Meaning it will give you a list of registry hives found in the memory dump. For example, some common hives include *SAM*, *SECURITY*, *SOFTWARE*, *SYSTEM*, and user profile hives. As you can see below the output shows the offsets of each hive.

```

/home kali-kali:~$ ./Downloads/volatility3100
(kali㉿kali)-[~/Downloads/volatility3100]
└─$ sudo volatility -f /home/kali/Desktop/WXP-20240326-083835.raw --profile=WinXPSP2x86 mftparser --output=mftparser.txt
Volatility Foundation Volatility Framework 2.6
Plugin MFTParser is unable to produce output in format -file=mftparser.txt. Supported formats are ['body', 'dot', 'greptext', 'html', 'json', 'split ext', 'xlsx']. Please send a feature request

(kali㉿kali)-[~/Downloads/volatility3100]
└─$ sudo volatility -f /home/kali/Desktop/WXP-20240326-083835.raw --profile=WinXPSP2x86 mftparser > mftparser.txt
Volatility Foundation Volatility Framework 2.6
Offset(P)
0x0a2ca008
0x0a430008
0x0a75d608
0x0e5fc908
0x134a4008
0x1373e008
0x1373e060
0x1699758
0x17568580
0x181dd000
0x181dd060
0x183ec600
0x21a52b60

(kali㉿kali)-[~/Downloads/volatility3100]
```

- **Hivelist:** This command lists the virtual addresses of the registry hives in memory. Meaning it will give you a list of registry hives and their locations in memory. From the output, you can see various system and user registry hives (this is normally combined with

hivescan plugin). For example, **\Device\HarddiskVolume1\WINDOWS\system32\config\software** is the *SOFTWARE* hive, which contains software configuration data for the system. **\Device\HarddiskVolume1\Documents and Settings\student\NTUSER.DAT** is the *NTUSER.DAT* hive for the student user, which contains user-specific registry data.

```

kali@kali:~/Downloads/volatility3100
File Actions Edit View Help
Volatility Foundation Volatility Framework 2.6
[kali㉿kali] ~[~/Downloads/volatility3100]
$ sudo volatility -f /home/kali/Desktop/WXP-20240326-083835.raw --profile=WinXPSP2x86 hivescan
Volatility Foundation Volatility Framework 2.6
Offset(P)
0x0a3ca008
0x0a4e03008
0x0a75d0008
0x0e5c03008
0x118061400
0x1373e0008
0x1373e0008
0x169075000
0x170050000
0x180030008
0x1818d0000
0x183ec0000
0x21a620000
[kali㉿kali] ~[~/Downloads/volatility3100]
$ sudo volatility -f /home/kali/Desktop/WXP-20240326-083835.raw --profile=WinXPSP2x86 hivelist
Volatility Foundation Volatility Framework 2.6
Virtual      Physical   Name
0x1a7758 0x1699c758 \Device\HarddiskVolume1\Documents and Settings\student\Local Settings\Application Data\Microsoft\Windows\UsrClass.dat
0x1bd6b000 0x21a62b60 \Device\HarddiskVolume1\Documents and Settings\student\NTUSER.DAT
0x1ba00000 0x183ec000 \Device\HarddiskVolume1\Documents and Settings\NetworkService\Local Settings\Application Data\Microsoft\Windows\UsrClass.dat
0x180030000 0x1818d000 \Device\HarddiskVolume1\Documents and Settings\NetworkService\Local Service\NTUSER.DAT
0x1fc00000 0x1818d000 \Device\HarddiskVolume1\Documents and Settings\NetworkService\Local Settings\Application Data\Microsoft\Windows\UsrClass.dat
0x1a1f3008 0x180030000 \Device\HarddiskVolume1\Documents and Settings\NetworkService\NTUSER.DAT
0x19759000 0x0efc9000 \Device\HarddiskVolume1\WINDOWS\system32\config\software
0x12000000 0x1373eb00 \Device\HarddiskVolume1\WINDOWS\system32\config\default
0x19da0000 0x1373eb00 \Device\HarddiskVolume1\WINDOWS\system32\config\Network
0x19da0000 0x1373eb00 [no name]
0x1456b000 0x0a76db00 [no name]
0x10370000 0x0a3ca000 [no name]
0x102f0000 0x0a03000 [no name]

[kali㉿kali] ~[~/Downloads/volatility3100]

```

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

5:01 PM
2024-03-26

Networking

- **connections:** This command views TCP connections that were active at the time of the memory acquisition. For example, you can see various network connections made by different processes. For example, the process with PID 1728 has made several connections to different IP addresses on various ports.

A screenshot of a Kali Linux desktop environment. The desktop background is a photograph of the Great Wall of China at sunset, with misty mountains in the background. On the left, there's a vertical dock with icons for Home, File System, Home, and WXP_2024... A terminal window is open in the foreground, showing a command-line session with the Volatility Framework 2.6. The terminal output includes:

```
kali@kali: ~/Downloads/volatility3100
File Actions Edit View Help
└─$ sudo volatility -f /home/kali/Desktop/WXP-20240326-083835.raw --profile=WinXPSP2x86 hivelist
Volatility Foundation Volatility Framework 2.6
Virtual Physical Name

0xe1a7a758 0x1699c758 \Device\HarddiskVolume1\Documents and Settings\student\Local Settings\Application Data\Microsoft\Windows\UsrClass.dat
0xebd6b660 0x21a62b60 \Device\HarddiskVolume1\Documents and Settings\student\NTUSER.DAT
0xbab8b660 0x183c6b60 \Device\HarddiskVolume1\Documents and Settings\Local Settings\Temporary Internet Files\Temporary Internet Service\WINNTER.DAT
0x1fc6b660 0x1818d660 \Device\HarddiskVolume1\Documents and Settings\NetworkService\local Settings\Application Data\Microsoft\Windows\UsrClass.dat
0x1af36008 0x18030008 \Device\HarddiskVolume1\Documents and Settings\NetworkService\NTUSER.DAT
0x19759008 0x169fc960 \Device\HarddiskVolume1\WINDOWS\system32\config\software
0x197ab0b8 0x1373e6b8 \Device\HarddiskVolume1\WINDOWS\system32\config\default
0x197ab0b8 0x1373e6b8 \Device\HarddiskVolume1\WINDOWS\system32\config\SAM
0x191ab0b8 0x1373e6b8 \Device\HarddiskVolume1\WINDOWS\system32\config\SECURITY
0x1456b660 0x0a7fd6b6 [no name]
0x10370008 0x0a3c0008 \Device\HarddiskVolume1\WINDOWS\system32\config\system
0x102f0008 0x0a3c0008 [no name]

[ kali@kali: ~] └─$ ./Downloads/volatility3100
└─$ sudo volatility -f /home/kali/Desktop/WXP-20240326-083835.raw --profile=WinXPSP2x86 connections
Volatility Foundation Volatility Framework 2.6
Offset(V) Local Address      Remote Address      Pid

0x896434b8 192.168.160.209:1143    142.251.33.162:443   1728
0x896fe68 192.168.160.209:1215    142.251.33.161:443   1728
0x89637808 192.168.160.209:1216    77.77.77.77:443     1728
0x89637808 192.168.160.209:1209    142.251.33.160:443   1728
0x89637888 192.168.160.209:1222    142.251.33.162:443   1728
0x89657f8 192.168.160.209:1243     172.64.149.23:80    1728
0x89cc0e68 192.168.160.209:1247     72.21.81.240:80    1800
0x896a9558 192.168.160.209:1248     104.24.144.123:80   1728
0x896a9558 192.168.160.209:1249     142.251.33.174:443   1728
0x896e3408 127.0.0.1:1036       127.0.0.1:1035      1728
0x8902f008 127.0.0.1:1035       127.0.0.1:1036      1728
0x89834248 192.168.160.209:1244     185.199.1.103:443    1728
0x89834248 192.168.160.209:1245     192.168.160.209:1261  1800
0x8982ad00 192.168.160.209:1171    172.217.1.3:443     1728
0x896a0088 192.168.160.209:1236    142.251.32.66:443   1728
0x896ab008 192.168.160.209:1237    142.251.32.66:443   1728
```

- **connscan:** This command is similar to the previous one, but it can find artifacts from previous connections that have since been terminated, in addition to the active ones. For example, the first row indicates a connection was made by the process with PID 1728 from local address 192.168.160.209:1232 to remote address 142.251.33.162:443.

A screenshot of a Kali Linux desktop environment. The terminal window in the foreground shows the output of the volatility3100 tool, specifically the 'netstat' command, against a WinXPSP2x86 profile. The table lists numerous network connections, each with a unique offset, remote address, and process ID (Pid). The background of the desktop is a scenic image of the Great Wall of China at sunset, with mountains and clouds visible. The desktop interface includes icons for Home, File System, and Trash, along with a taskbar at the top.

- sockets:** This command is used to detect listening sockets for any protocol (TCP, UDP, RAW, etc). For example, the first row indicates a TCP socket was created by the process with PID 4 at local address 192.168.160.209 on port 139 at the specified create time.

Offset(V)	PID	Port	Proto	Protocol	Address	Create Time
0x89e41c38	4	139	6	TCP	192.168.160.209	2024-03-26 08:29:48 UTC+0000
0x89fc998	1420	1151	17	UDP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x89e72450	1728	1243	6	TCP	0.0.0.0	2024-03-26 08:38:10 UTC+0000
0x89c7d10	4	47	GRE		0.0.0.0	2024-03-26 08:29:34 UTC+0000
0x89e26e08	1800	1247	6	TCP	0.0.0.0	2024-03-26 08:38:31 UTC+0000
0x89e26e10	876	1020	17	UDP	0.0.0.0	2024-03-26 08:29:30 UTC+0000
0x89e6a008	396	5981	6	TCP	127.0.0.1	2024-03-26 08:29:24 UTC+0000
0x89e6bca0	1420	1136	17	UDP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x89e95798	1728	1139	6	TCP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x89e99908	1728	1232	6	TCP	0.0.0.0	2024-03-26 08:35:16 UTC+0000
0x89e99900	1420	1024	17	UDP	0.0.0.0	2024-03-26 08:29:42 UTC+0000
0x89e9c300	4	137	17	UDP	192.168.160.209	2024-03-26 08:29:08 UTC+0000
0x89d0dc08	4	445	6	TCP	0.0.0.0	2024-03-26 08:28:50 UTC+0000
0x89e43b08	1728	1236	6	TCP	0.0.0.0	2024-03-26 08:35:20 UTC+0000
0x89e43b10	1728	1161	6	TCP	0.0.0.0	2024-03-26 08:34:34 UTC+0000
0x89e99909	1420	1149	17	UDP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x89e9c208	1728	1135	6	TCP	0.0.0.0	2024-03-26 08:29:02 UTC+0000
0x89e7938	1420	1152	17	UDP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x89d12a08	1728	1244	6	TCP	0.0.0.0	2024-03-26 08:38:11 UTC+0000
0x89e99908	1728	1245	6	TCP	0.0.0.0	2024-03-26 08:35:10 UTC+0000
0x89e991720	1420	1040	17	UDP	0.0.0.0	2024-03-26 08:29:42 UTC+0000
0x8980d5d0	1340	123	17	UDP	127.0.0.1	2024-03-26 08:29:32 UTC+0000
0x89a969320	876	0	255	Reserved	0.0.0.0	2024-03-26 08:29:24 UTC+0000
0x89a96930	1728	1171	6	TCP	0.0.0.0	2024-03-26 08:34:49 UTC+0000
0x89e99909	1728	1173	17	UDP	192.168.160.209	2024-03-26 08:34:40 UTC+0000
0x89e94520	1728	1237	6	TCP	0.0.0.0	2024-03-26 08:35:20 UTC+0000
0x89a41558	1420	1025	17	UDP	0.0.0.0	2024-03-26 08:29:09 UTC+0000
0x89e9d9b8	1728	1245	6	TCP	0.0.0.0	2024-03-26 08:38:12 UTC+0000
0x89922a08	1340	123	17	UDP	192.168.160.209	2024-03-26 08:29:32 UTC+0000
0x89922a09	1728	1242	6	TCP	0.0.0.0	2024-03-26 08:35:23 UTC+0000
0x89cf7e98	1728	1249	6	TCP	0.0.0.0	2024-03-26 08:39:10 UTC+0000
0x89c5e5b0	1728	1036	6	TCP	0.0.0.0	2024-03-26 08:29:39 UTC+0000
0x89c9d18	1728	1038	6	TCP	127.0.0.1	2024-03-26 08:29:33 UTC+0000
0x89e99909	1728	1159	6	TCP	0.0.0.0	2024-03-26 08:30:00 UTC+0000
0x89e92b08	1728	1215	6	TCP	0.0.0.0	2024-03-26 08:35:05 UTC+0000
0x899f3228	1688	1980	17	UDP	127.0.0.1	2024-03-26 08:29:33 UTC+0000
0x89808008	876	4508	17	UDP	0.0.0.0	2024-03-26 08:29:24 UTC+0000
0x89e92948	1420	1135	17	UDP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x89e99908	1728	1247	6	TCP	0.0.0.0	2024-03-26 08:35:07 UTC+0000
0x89d5a548	4	445	17	UDP	0.0.0.0	2024-03-26 08:28:50 UTC+0000
0x89a3d08	1688	1980	17	UDP	192.168.160.209	2024-03-26 08:29:33 UTC+0000
0x89c5e608	1728	1035	6	TCP	127.0.0.1	2024-03-26 08:29:39 UTC+0000

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

5:06 PM
2024-03-26

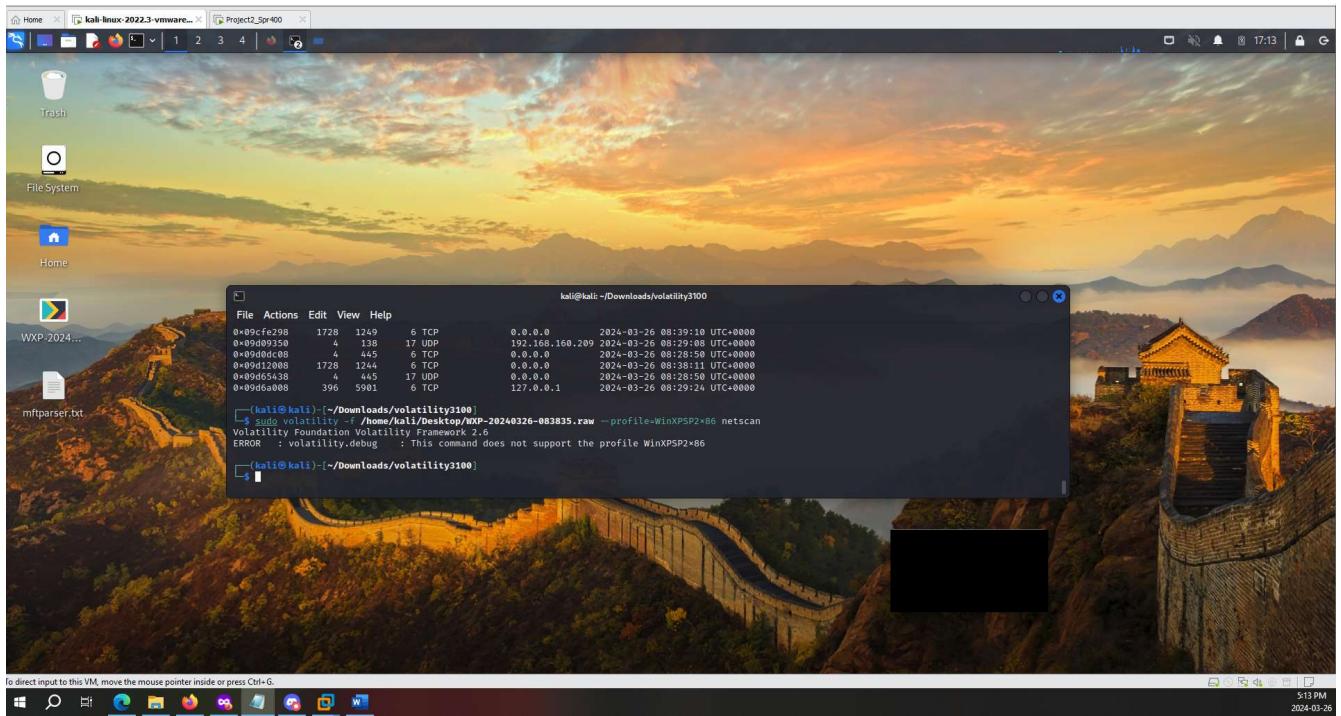
- sockscan:** This command is used to find the ADDRESS_OBJECT structures. Each row represents a network socket in the system. For example, the first row indicates a TCP socket was created by the process with PID 1728 at local address 0.0.0.0 on port 1216 at the specified create time.

Offset(V)	PID	Port	Proto	Protocol	Address	Create Time
0x0958e98	1728	1216	6	TCP	0.0.0.0	2024-03-26 08:35:06 UTC+0000
0x09643b0	1728	1236	6	TCP	0.0.0.0	2024-03-26 08:35:20 UTC+0000
0x09643b10	1728	1159	6	TCP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x0966a030	1728	1143	6	TCP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x0966bc40	1420	1136	17	UDP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x09672450	1728	1243	6	TCP	0.0.0.0	2024-03-26 08:38:10 UTC+0000
0x0967ae98	1728	1217	6	TCP	0.0.0.0	2024-03-26 08:35:00 UTC+0000
0x0968a030	1728	1158	6	TCP	0.0.0.0	2024-03-26 08:34:40 UTC+0000
0x09691e88	1728	1165	6	TCP	0.0.0.0	2024-03-26 08:34:49 UTC+0000
0x09691e30	1728	1246	6	TCP	0.0.0.0	2024-03-26 08:38:12 UTC+0000
0x09694520	1728	1237	6	TCP	0.0.0.0	2024-03-26 08:35:20 UTC+0000
0x09695798	1728	1139	6	TCP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x09695798	1728	1152	6	TCP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x09691008	1728	1168	6	TCP	0.0.0.0	2024-03-26 08:34:49 UTC+0000
0x0966d600	1728	1128	6	TCP	0.0.0.0	2024-03-26 08:34:44 UTC+0000
0x0966d608	1728	1167	6	TCP	0.0.0.0	2024-03-26 08:34:33 UTC+0000
0x09980808	876	4508	17	UDP	0.0.0.0	2024-03-26 08:29:24 UTC+0000
0x09845d50	1340	123	17	UDP	127.0.0.1	2024-03-26 08:29:32 UTC+0000
0x09825b08	1728	1196	6	TCP	0.0.0.0	2024-03-26 08:34:59 UTC+0000
0x09849d08	1728	1159	17	UDP	0.0.0.0	2024-03-26 08:29:00 UTC+0000
0x09858208	1728	1215	6	TCP	0.0.0.0	2024-03-26 08:35:05 UTC+0000
0x0982e008	1800	1247	6	TCP	0.0.0.0	2024-03-26 08:38:31 UTC+0000
0x0984bb20	1728	1158	6	TCP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x098807e0	876	500	17	UDP	0.0.0.0	2024-03-26 08:29:24 UTC+0000
0x098807e0	1340	1233	17	UDP	192.168.160.209	2024-03-26 08:34:40 UTC+0000
0x0992a008	1420	1032	6	TCP	0.0.0.0	2024-03-26 08:29:34 UTC+0000
0x0998ee98	1420	1148	17	UDP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x0998e928	1688	1980	17	UDP	127.0.0.1	2024-03-26 08:29:33 UTC+0000
0x09841c58	1420	1025	17	UDP	0.0.0.0	2024-03-26 08:29:00 UTC+0000
0x09841c58	1728	1139	6	TCP	192.168.160.209	2024-03-26 08:29:24 UTC+0000
0x0993d008	876	0	255	Reserved	0.0.0.0	2024-03-26 08:29:24 UTC+0000
0x0993d008	1688	1900	17	UDP	192.168.160.209	2024-03-26 08:29:33 UTC+0000
0x09afce98	1420	1151	17	UDP	0.0.0.0	2024-03-26 08:34:47 UTC+0000
0x09bf1820	1728	1177	6	TCP	0.0.0.0	2024-03-26 08:34:40 UTC+0000
0x09bf1820	1728	1040	17	UDP	0.0.0.0	2024-03-26 08:29:42 UTC+0000
0x09b349a8	2408	1037	6	TCP	0.0.0.0	2024-03-26 08:29:39 UTC+0000
0x09b44c0f0	1728	1169	6	TCP	0.0.0.0	2024-03-26 08:34:49 UTC+0000
0x09c5e5b0	1728	1036	6	TCP	0.0.0.0	2024-03-26 08:29:39 UTC+0000
0x09c5e5b0	1728	1053	6	TCP	127.0.0.1	2024-03-26 08:29:00 UTC+0000
0x09c70008	1420	1051	17	UDP	0.0.0.0	2024-03-26 08:29:43 UTC+0000
0x09c82e98	1727	135	6	TCP	0.0.0.0	2024-03-26 08:29:02 UTC+0000

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

5:07 PM
2024-03-26

- **netscan:** This command is used to list network-related objects, like sockets and connections. This command did not work because it is designed to work with newer versions of Windows (Vista and later) that include certain networking features unlike my WindowsXP machine. In other words it is not compatible.



Part 3: Malware Memory Images

This section is similar to the previous one but this time we compare outputs based on the provided clean and infected memory dumps.

- **imageinfo:** Here we can see that both machines have the same suggested profile by Volatility, indicating that we're dealing with the same operating system in both memory dumps.

```
(kali㉿kali)-[~/Downloads/volatility3100]
$ sudo volatility -f /home/kali/Desktop/xp-clean.bin imageinfo
Volatility Foundation Volatility Framework 2.6
INFO : volatility.debug : Determining profile based on KDBG search...
INFO : volatility.debug : Suggested Profile(s) : WinXPSP2x86, WinXPSP3x86 (Instantiated with WinXPSP2x86)
INFO : volatility.debug : 
AS Layer1 : FileAddressSpace (/home/kali/Desktop/xp-clean.bin)
AS Layer2 : FileAddressSpace (/home/kali/Desktop/xp-clean.bin)
PAE type : PAE
DTB : 0x200000L
KUSER_SHARED_DATA : 0x60545a00L
Number of Processors : 1
Image Type (Service Pack) : 3
KPCR for CPU 0 : 0xffffdf000L
KUSER_SHARED_DATA : 0x60545a00L
Image date and time : 2011-04-10 21:05:48 UTC+0000
Image local date and time : 2011-04-10 14:05:48 -0700

(kali㉿kali)-[~/Downloads/volatility3100]
$ [4]

(kali㉿kali)-[~/Downloads/volatility3100]
$ sudo volatility -f /home/kali/Desktop/xp-infected.bin imageinfo
Volatility Foundation Volatility Framework 2.6
INFO : volatility.debug : Determining profile based on KDBG search...
INFO : volatility.debug : Suggested Profile(s) : WinXPSP2x86, WinXPSP3x86 (Instantiated with WinXPSP2x86)
INFO : volatility.debug : 
AS Layer1 : FileAddressSpace (/home/kali/Desktop/xp-infected.bin)
AS Layer2 : FileAddressSpace (/home/kali/Desktop/xp-infected.bin)
PAE type : PAE
DTB : 0x200000L
KUSER_SHARED_DATA : 0x60545a00L
Number of Processors : 1
Image Type (Service Pack) : 3
KPCR for CPU 0 : 0xffffdf000L
KUSER_SHARED_DATA : 0x60545a00L
Image date and time : 2011-04-10 21:29:25 UTC+0000
Image local date and time : 2011-04-10 14:29:25 -0700

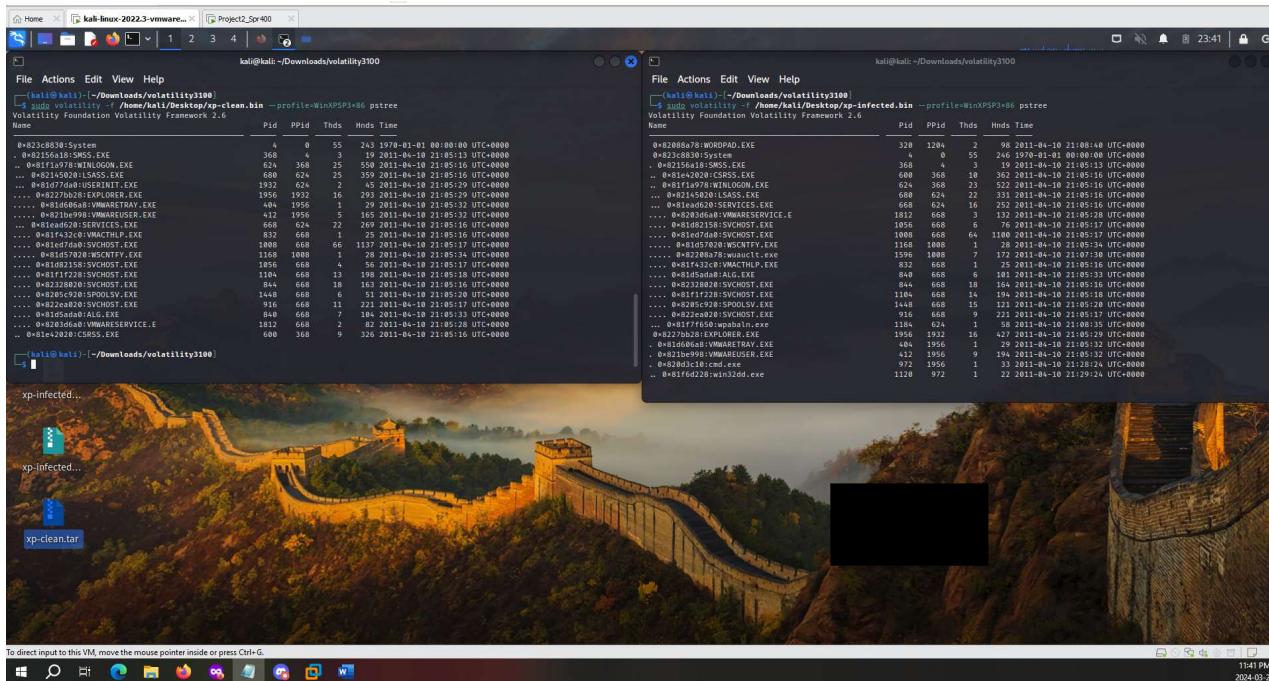
(kali㉿kali)-[~/Downloads/volatility3100]
$ [4]
```

- Plist:** Here, you can see me utilizing the pslist plugin to look at the list of processes running on both machines when the memory dump was taken. When comparing both outputs I notice 5 more processes running on the infected machine. These processes are **wuauctl.exe**, **wpabalg.exe**, **WORDPAD.EXE**, **cmd.exe**, and **win32dd.exe**. We know already that **cmd.exe**, and **win32dd.exe** are used as part of the process of creating the memory dump, so we can ignore those two. Thus, we're now left with the three other processes to suspect of being malicious: **wuauctl.exe**, **wpabalg.exe**, and **WORDPAD.EXE**.

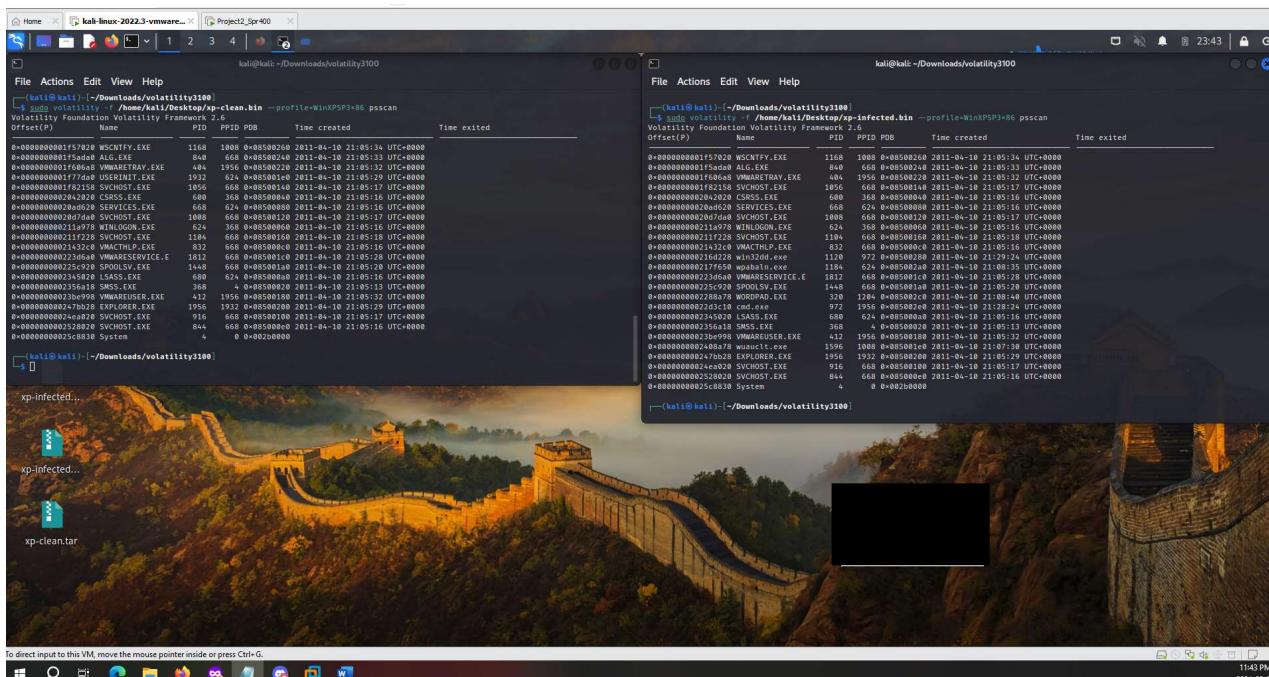
Offset(V)	Name	PID	PPID	Thds	Hnds	Sess	Wow64	Start	Exit
0x823c8830	System	4	0	55	243	—	0		
0x82356108	SMSS.EXE	368	4	3	19	—	0	2011-04-10 21:05:13 UTC+0000	
0x81e12020	CSRSS.EXE	600	368	9	326	0	0	2011-04-10 21:05:16 UTC+0000	
0x81f1a978	WINLOGON.EXE	624	368	25	558	0	0	2011-04-10 21:05:16 UTC+0000	
0x81ed6d20	SERVICES.EXE	668	624	22	269	0	0	2011-04-10 21:05:16 UTC+0000	
0x82158020	LSASS.EXE	688	624	25	359	0	0	2011-04-10 21:05:16 UTC+0000	
0x81f32c00	VMACTHLP.EXE	832	668	1	25	0	0	2011-04-10 21:05:16 UTC+0000	
0x82328820	SVCHOST.EXE	844	668	18	163	0	0	2011-04-10 21:05:16 UTC+0000	
0x822ea020	SVCHOST.EXE	916	668	11	221	0	0	2011-04-10 21:05:16 UTC+0000	
0x81ed7d00	SVCHOST.EXE	1008	668	66	1137	0	0	2011-04-10 21:05:17 UTC+0000	
0x81d82158	SVCHOST.EXE	1056	668	4	56	0	0	2011-04-10 21:05:17 UTC+0000	
0x81ff7228	SVCHOST.EXE	1104	668	13	198	0	0	2011-04-10 21:05:18 UTC+0000	
0x8205c920	SPOLSLV.EXE	1448	668	6	51	0	0	2011-04-10 21:05:20 UTC+0000	
0x8203d6a0	VMWARESERVICE.E	1812	668	2	82	0	0	2011-04-10 21:05:28 UTC+0000	
0x81d7d6a0	USERINIT.EXE	1932	624	2	45	0	0	2011-04-10 21:05:29 UTC+0000	
0x8227fb28	EXPLORER.EXE	1956	1932	16	293	0	0	2011-04-10 21:05:29 UTC+0000	
0x81d06e68	VMWARETRAY.EXE	404	1956	1	29	0	0	2011-04-10 21:05:32 UTC+0000	
0x82109980	VMWAREUSER.EXE	412	1956	5	165	0	0	2011-04-10 21:05:32 UTC+0000	
0x81d5a000	ALG.EXE	840	668	7	104	0	0	2011-04-10 21:05:33 UTC+0000	
0x81d57020	WSCHTFFY.EXE	1168	1088	1	28	0	0	2011-04-10 21:05:34 UTC+0000	

Offset(V)	Name	PID	PPID	Thds	Hnds	Sess	Wow64	Start	Exit
0x823c8830	System	4	0	55	246	—	0	2011-04-10 21:05:13 UTC+0000	
0x82356108	SMSS.EXE	368	4	3	19	—	0	2011-04-10 21:05:13 UTC+0000	
0x81e12020	CSRSS.EXE	600	368	10	362	0	0	2011-04-10 21:05:16 UTC+0000	
0x81f1a978	WINLOGON.EXE	624	368	23	522	0	0	2011-04-10 21:05:16 UTC+0000	
0x81ed6d20	SERVICES.EXE	668	624	16	252	0	0	2011-04-10 21:05:16 UTC+0000	
0x82158020	LSASS.EXE	688	624	22	331	0	0	2011-04-10 21:05:16 UTC+0000	
0x81f32c00	VMACTHLP.EXE	832	668	1	25	0	0	2011-04-10 21:05:16 UTC+0000	
0x82328820	SVCHOST.EXE	844	668	18	164	0	0	2011-04-10 21:05:16 UTC+0000	
0x822ea020	SVCHOST.EXE	916	668	9	221	0	0	2011-04-10 21:05:17 UTC+0000	
0x81ed7d00	SVCHOST.EXE	1008	668	64	1180	0	0	2011-04-10 21:05:17 UTC+0000	
0x81d82158	SVCHOST.EXE	1056	668	6	76	0	0	2011-04-10 21:05:17 UTC+0000	
0x81ff7228	SVCHOST.EXE	1104	668	14	194	0	0	2011-04-10 21:05:18 UTC+0000	
0x8205c920	SPOLSLV.EXE	1448	668	15	121	0	0	2011-04-10 21:05:20 UTC+0000	
0x8203d6a0	VMWARESERVICE.E	1812	668	3	132	0	0	2011-04-10 21:05:28 UTC+0000	
0x8227fb28	EXPLORER.EXE	1956	1932	16	427	0	0	2011-04-10 21:05:29 UTC+0000	
0x81d5b6a8	VMWARETRAY.EXE	404	1956	1	29	0	0	2011-04-10 21:05:32 UTC+0000	
0x82109980	VMWAREUSER.EXE	412	1956	9	194	0	0	2011-04-10 21:05:32 UTC+0000	
0x81d5a000	ALG.EXE	840	668	6	181	0	0	2011-04-10 21:05:33 UTC+0000	
0x81d57020	WSCHTFFY.EXE	1168	1088	1	28	0	0	2011-04-10 21:05:34 UTC+0000	
0x81f6d228	WORDPAD.EXE	320	1204	2	98	0	0	2011-04-10 21:08:40 UTC+0000	
0x8205c920	cmd.exe	972	1956	1	33	0	0	2011-04-10 21:28:24 UTC+0000	
0x81f6d228	win32dd.exe	1120	972	1	22	0	0	2011-04-10 21:29:24 UTC+0000	

- Pstree:** After looking at both outputs for this plugin. I do not see anything standing out to help me narrow down the malicious process.



- Psscan:** Here, you can see that I utilized the psscan plugin with the intention of finding any processes with a “Time Exited” value which would indicate possible hidden processes being created and thus narrowing down our list of suspicious malwares. Unfortunately, there was not any “Time Exited” fields found in the output.



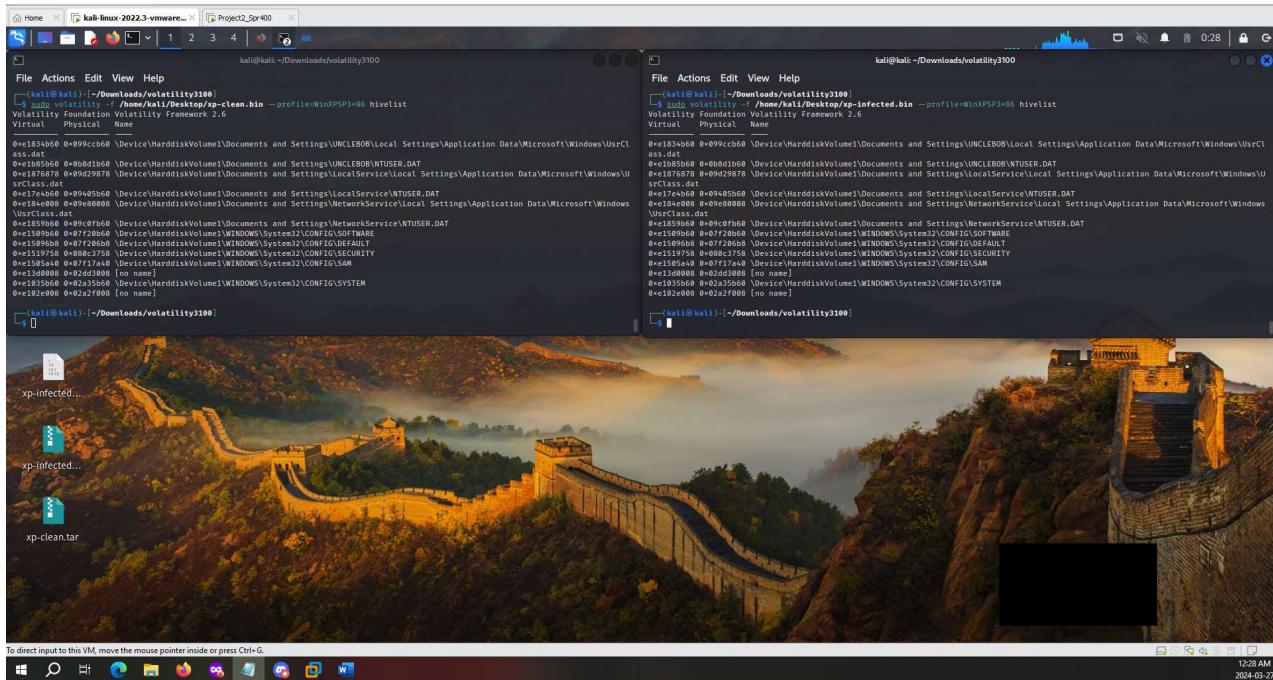
- Mbrparser:** Here, I initially attempted to run mfptparser (as it worked in part2), however this time it did not work in both clean and infected memory dumps, so I used the mbrparser

plugin instead. Despite seeing some output with this plug in, there was once again nothing concrete that can be used to better make the decision of isolating a single malware process.

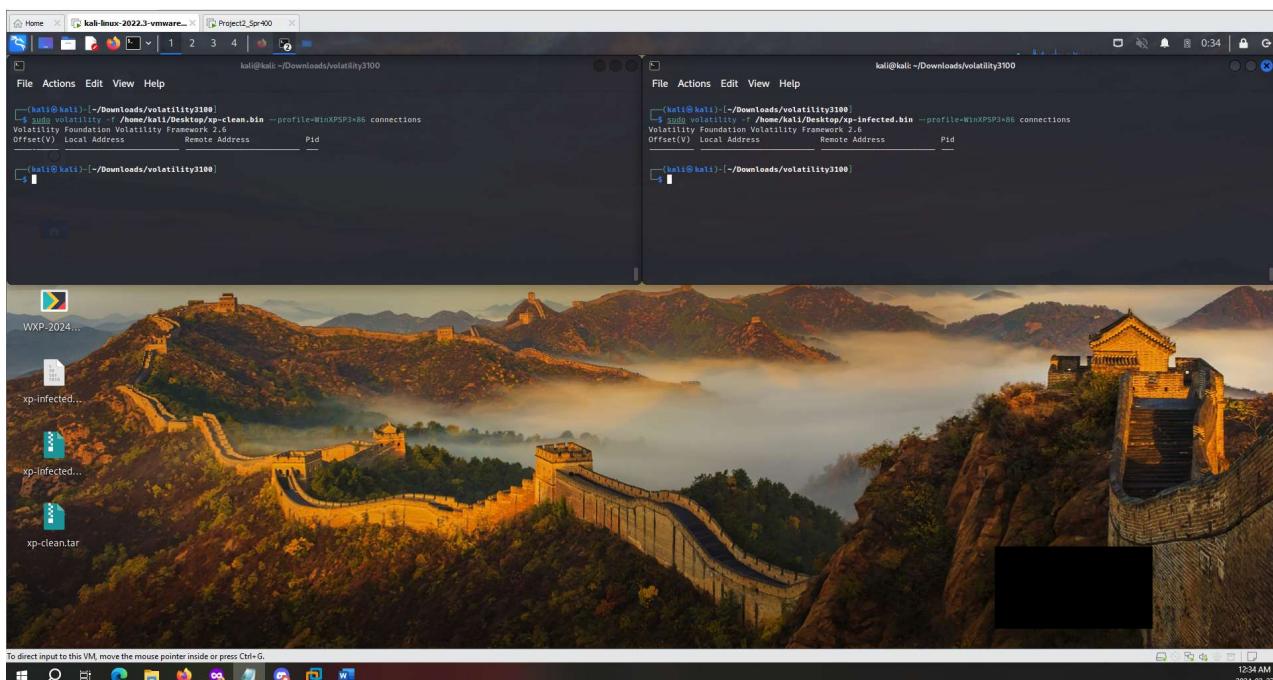
- **Hivescan:** Here, you can see me utilizing the hivescan plugin on both dumps to locate the registry hives in memory. Again, this didn't lead to any concrete conclusions.

A screenshot of a Kali Linux virtual machine interface. The desktop background features a scenic view of the Great Wall of China winding through misty mountains. Two terminal windows are open at the top of the screen. The left terminal window, titled 'Project2_Srv400', shows volatility 2.6 running against a memory dump from 'xp-clean'. The right terminal window shows volatility 2.6 running against a memory dump from 'xp-infected'. Both windows display memory dump offsets and their corresponding hex values. On the desktop, there are several icons: 'xp-infected...', 'xp-infected...', and 'xp-clean.tar'. The system tray at the bottom shows standard icons like network, battery, and volume. The status bar at the bottom right indicates the date and time as '2024-02-27 12:26 AM'.

- **Hivelist:** Here, you can see me utilizing the hivelist plugin on both dumps to find the virtual addresses of the registry hives in memory. Again, this didn't lead to any concrete conclusions.



- **Connection:** Here, we can see the output of this plugin on both clean and infected memory dumps showing that there were no active connection at the time these memory dumps were created.



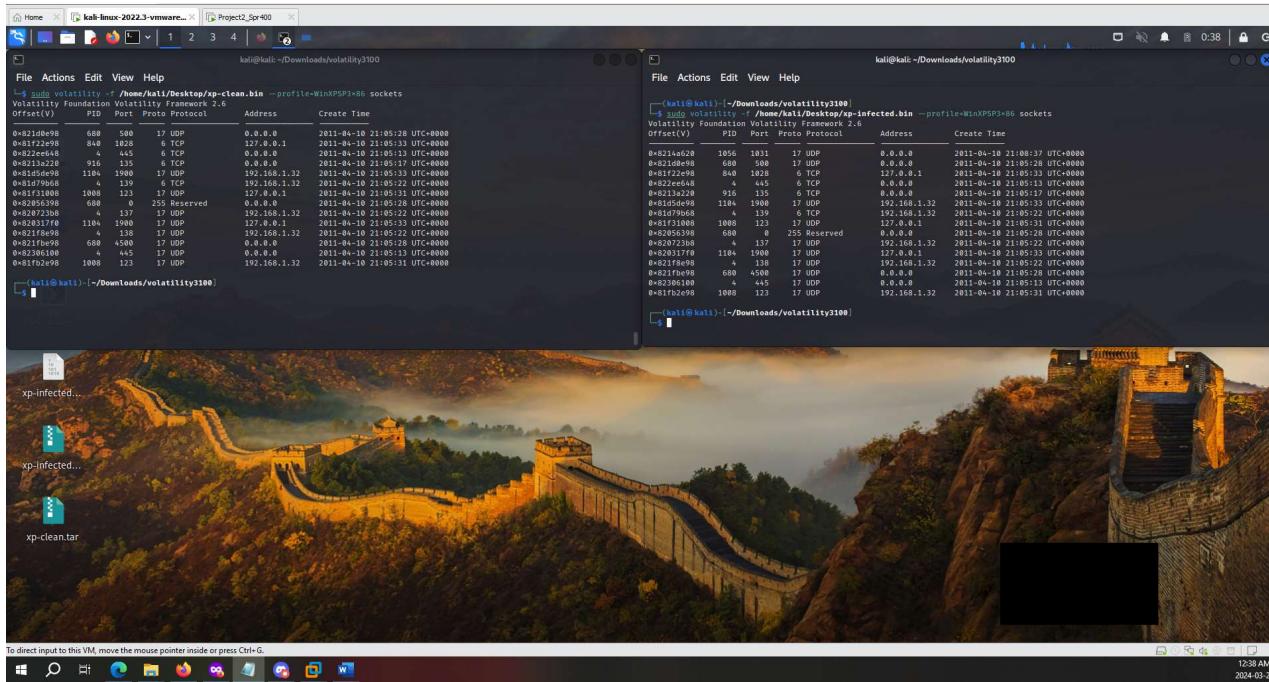
- **Connscan:** Here, we utilized the connscan plugin to determine if there are any connections that we're previously terminated in either machine from which the dump was taken. On the infected machine we realize that we have 2 connections that were created (and later terminated) – one with a process of ID (PID) of 1204 and the other with a PID 4. These processes were created because of that connection. If we take the PIDs and compare them with the list of running processes (pslist) above, we will find that they come back as the Parent process ID (PPID) of two processes – **WORDPAD.EXE** and **SMSS.EXE**, respectively. Recall that WORDPAD.EXE was already one of the processes we were suspecting earlier – with that in mind we can now move forward to utilize the next plugin.

```

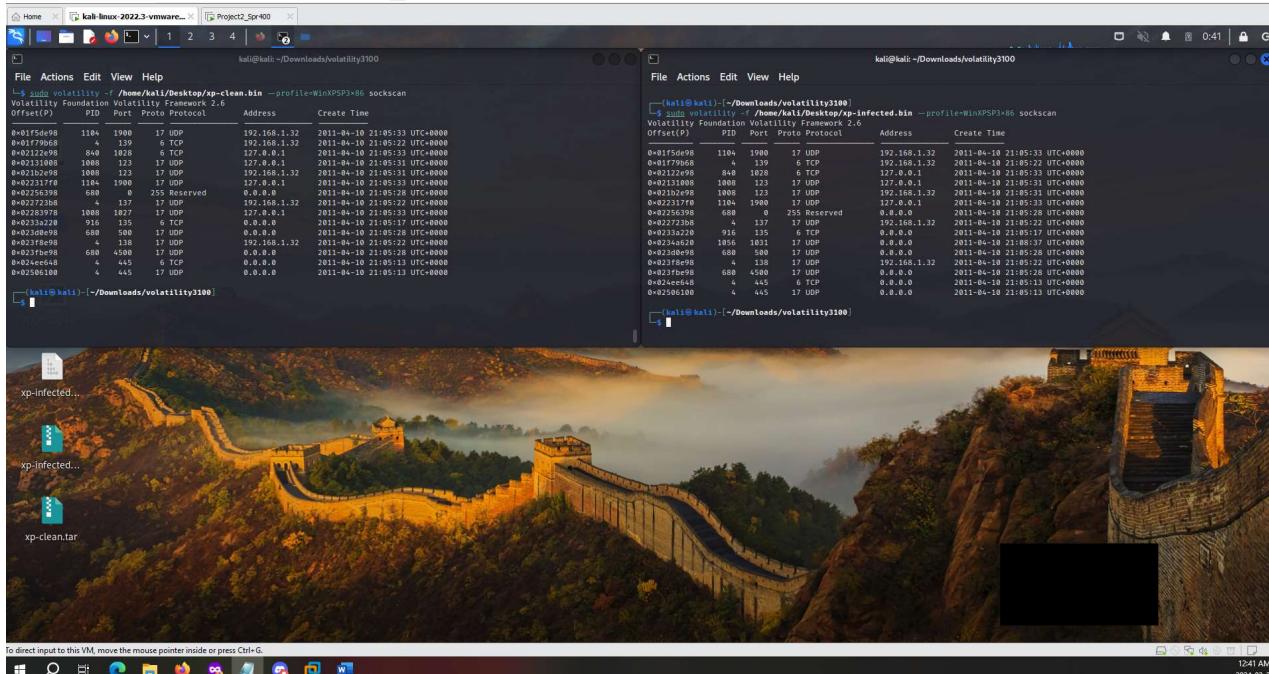
File Actions Edit View Help
Volatility Foundation Volatility Framework 2.6
Offset(V) Local Address Remote Address Pid
(kali㉿kali)-[~/Downloads/volatility3100]
$ sudo volatility -f /home/kali/Desktop/xp-infected.bin --profile=WinXPSP3x86 connscan
Volatility Foundation Volatility Framework 2.6
Offset(P) Local Address Remote Address Pid
(kali㉿kali)-[~/Downloads/volatility3100]
$ sudo volatility -f /home/kali/Desktop/xp-clean.bin --profile=WinXPSP3x86 connscan
Volatility Foundation Volatility Framework 2.6
Offset(V) Local Address Remote Address Pid
(kali㉿kali)-[~/Downloads/volatility3100]
$ sudo volatility -f /home/kali/Desktop/xp-infected.bin --profile=WinXPSP3x86 connscan
Volatility Foundation Volatility Framework 2.6
Offset(P) Local Address Remote Address Pid
0+0x2350cd8 192.168.1.32:1044 91.199.75.77:80 1204
0+0x2408838 192.168.1.32:1047 192.168.1.150:139 4

```

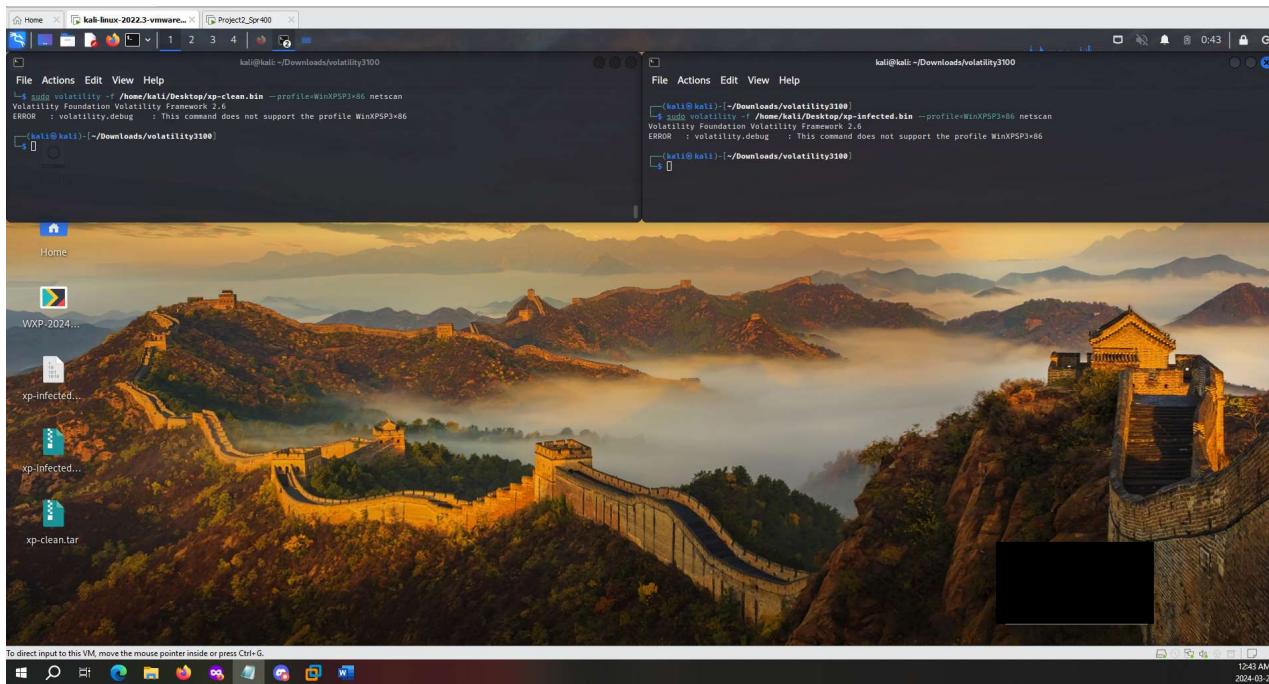
- **Sockets:** Here, I am utilizing the sockets plugin on both clean and infected memory dumps to determine the live sockets that existed at the time of the memory dump. When comparing both outputs they seem almost identical. The only difference is the infected one has an additional process that traces back to svchost.exe. In other words, there are no new leads here we could use to narrow down the list of malware processes.



- **Sockscan:** Here, I am utilizing the sockscan plugin on both clean and infected memory dumps to determine the previously existing sockets. Again, there are no new leads here we could use to narrow down the list of malware processes.



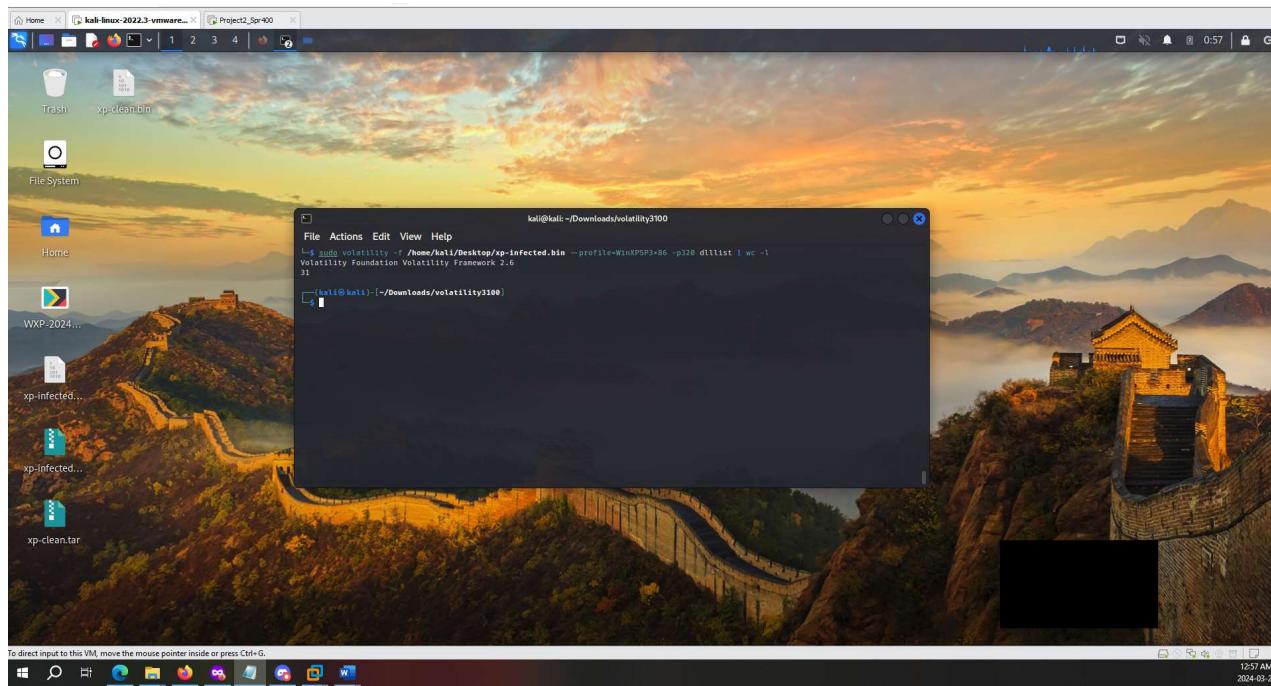
- **Netscan**: like previously explained in part 2, this plugin is not compatible with Windows XP, so we got an error.



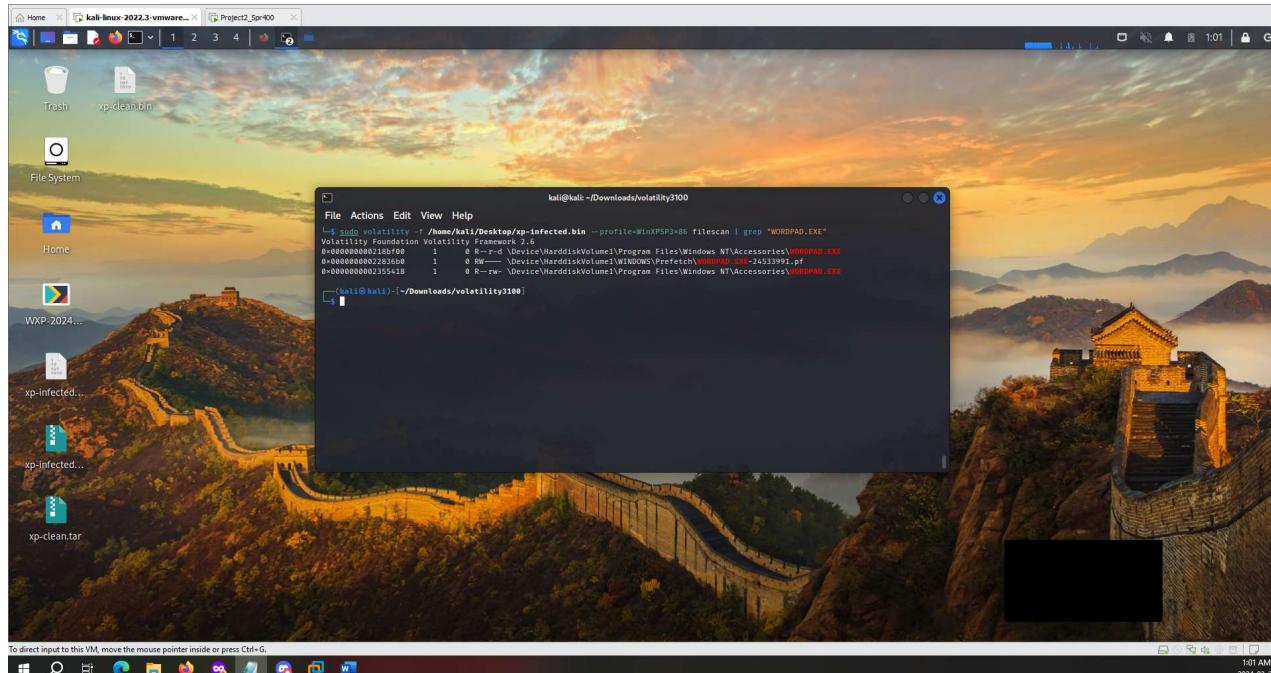
Task 2

Thus far we have suspected a few malicious processes (**wuauctl.exe**, **wpabalg.exe**, **WORDPAD.EXE**), but only one of them was created from a previous connection (WORDPAD.EXE). As per the instructions, in this task we will dig in deeper to find more proof on what makes this process seem malicious. To do this I'm thinking of checking the characteristics of this process against those of the legitimate program to determine if there is anything interesting or worth noting.

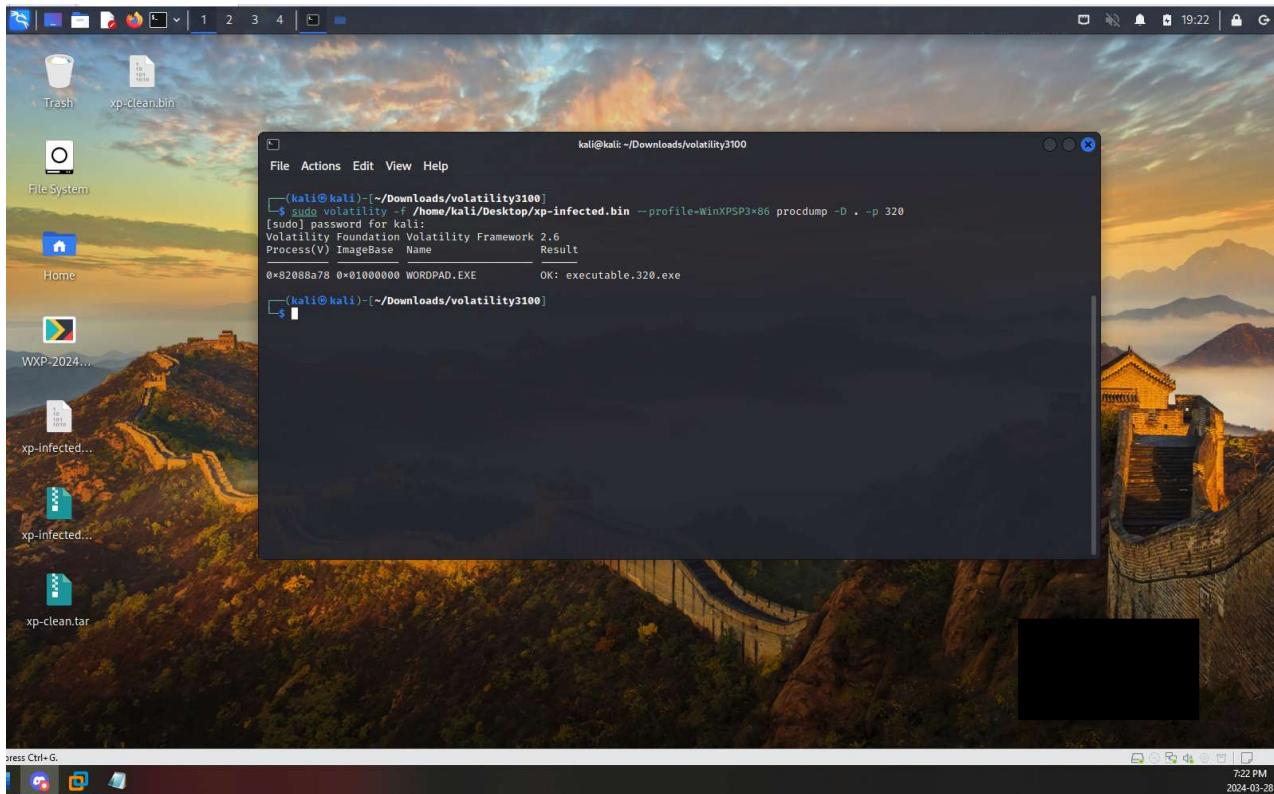
A common practice in digital forensics and incident response is to examine the Dynamic Link Libraries (DLL) of the files associated with a suspect process. We can do this using volatility. Knowing the Process ID of WORDPAD.EXE (320) we can examine the DLLs loaded by this specific process. After running that we see that **31 DLLs** were loaded by this specific process. This is concerning because after doing some research I found that on average, WordPad loads **a few hundred DLLs** during its execution. This finding suggests that WORDPAD.EXE is a suspicious process.



Another thing we can do is utilize the filescan plugin to determine if there were any open files that contain the term WORDPAD.EXE at the time the memory dump was taken. The output shows that we have 3 files open, 2 of which are located in the same directory (\Windows NT\Accessories). This is concerning because there is no reason why there should be two files with the same name (but with different permission) open in the same directory. The third file, (located in \WINDOWS\Prefetch) is not suspicious as it is known to be used to be typically used by the operating system to boost the loading speed of running programs. This is another finding that further confirms our suspicions that WORDPAD.EXE is a malicious process.



Now that our suspicions are leading towards a single process (wordpad.exe), we can utilize the procdump plugin to export this process in an executable format. We can later utilize this executable to upload to an online service like VirusTotal to check it against 70 antivirus scanners and services.



After uploading the executable, we can see in the below screenshot that it was indeed recognized as malicious by two security vendors (“Bkav Pro” and “Sangfor Engine Zero”); which finally concludes our investigation with WORDPAD.EXE being the malicious process.

VirusTotal - File - 52ab6... | Project2_Spr400

https://www.virustotal.com/gui/file/52ab621671e472e6e14c092fd21594a9dff4cae83991b9d11af21175d43

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

2 / 73 security vendors and no sandboxes flagged this file as malicious

52ab621671e472e6e14c092fd21594a9dff4cae83991b9d11af21175d43 wordpad

Size: 209.50 KB Last Modification Date: a moment ago

Community Score: 2 / 73

Detection DETAILS RELATIONS BEHAVIOR TELEMETRY COMMUNITY

Join the VT Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Do you want to automate checks?

Security vendor's analysis			
Bkav Pro	W32.A!DdetectMalware	Sangfor Engine Zero	Trojan.WIn32.Save.a
Acronis (Static ML)	Undetected	AhnLab-V3	Undetected
Alibaba	Undetected	AllCloud	Undetected
AVG	Undetected	Anti-AVL	Undetected
Baidu	Undetected	Avast	Undetected
BitDefenderTheta	Undetected	Avira (no cloud)	Undetected
CMC	Undetected	BitDefender	Undetected
		ClamAV	Undetected
		CrowdStrike Falcon	Undetected

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

Windows taskbar icons: Home, File Explorer, Mail, Firefox, etc.

Bottom right corner: 11:15 AM, 2024-03-27