Critical - Memory Forensics

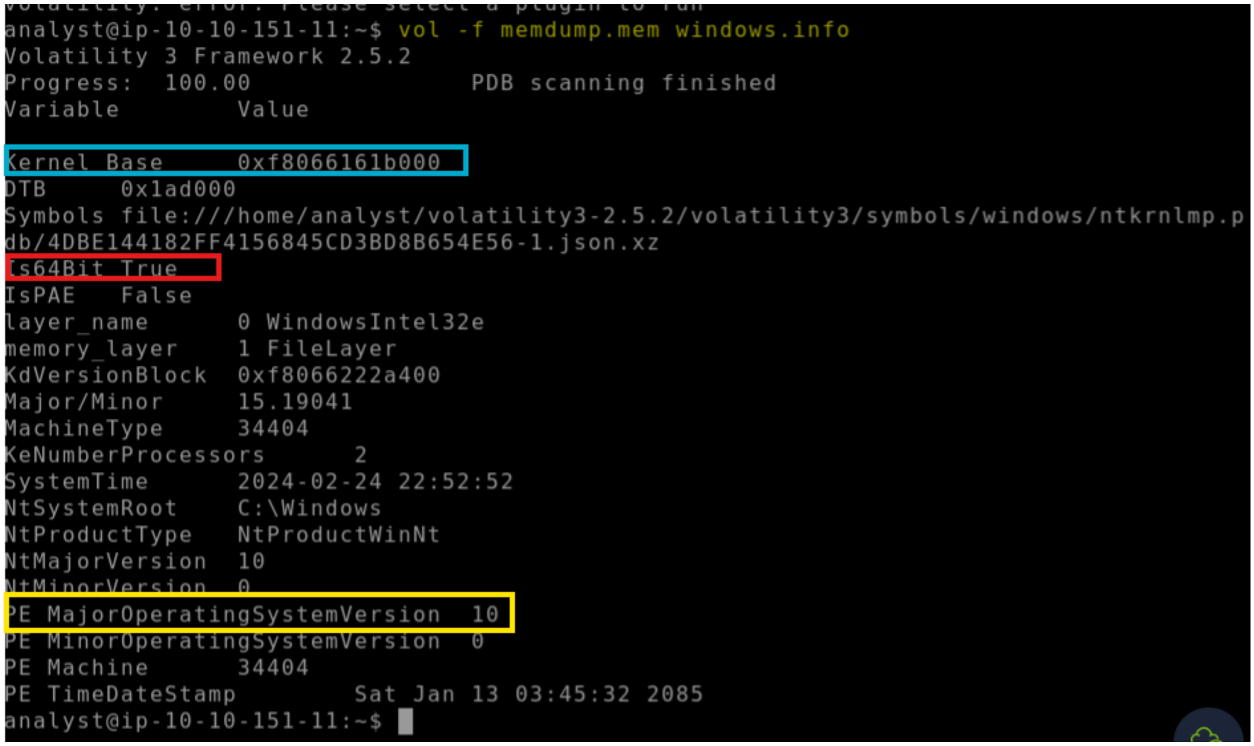
Obtaining Information

Getting information about the target is crucial to our investigation since it ensures we're analyzing the correct context and environment of the evidence. This step helps us understand specific architecture and operating systems, ensuring our findings' accuracy, relevance, and legitimacy.

Is the architecture of the machine x64 (64bit) Y/N? Answer: Y

What is the Verison of the Windows OS? Answer: 10

What is the base address of the kernel? Answer: 0xf8066161b000



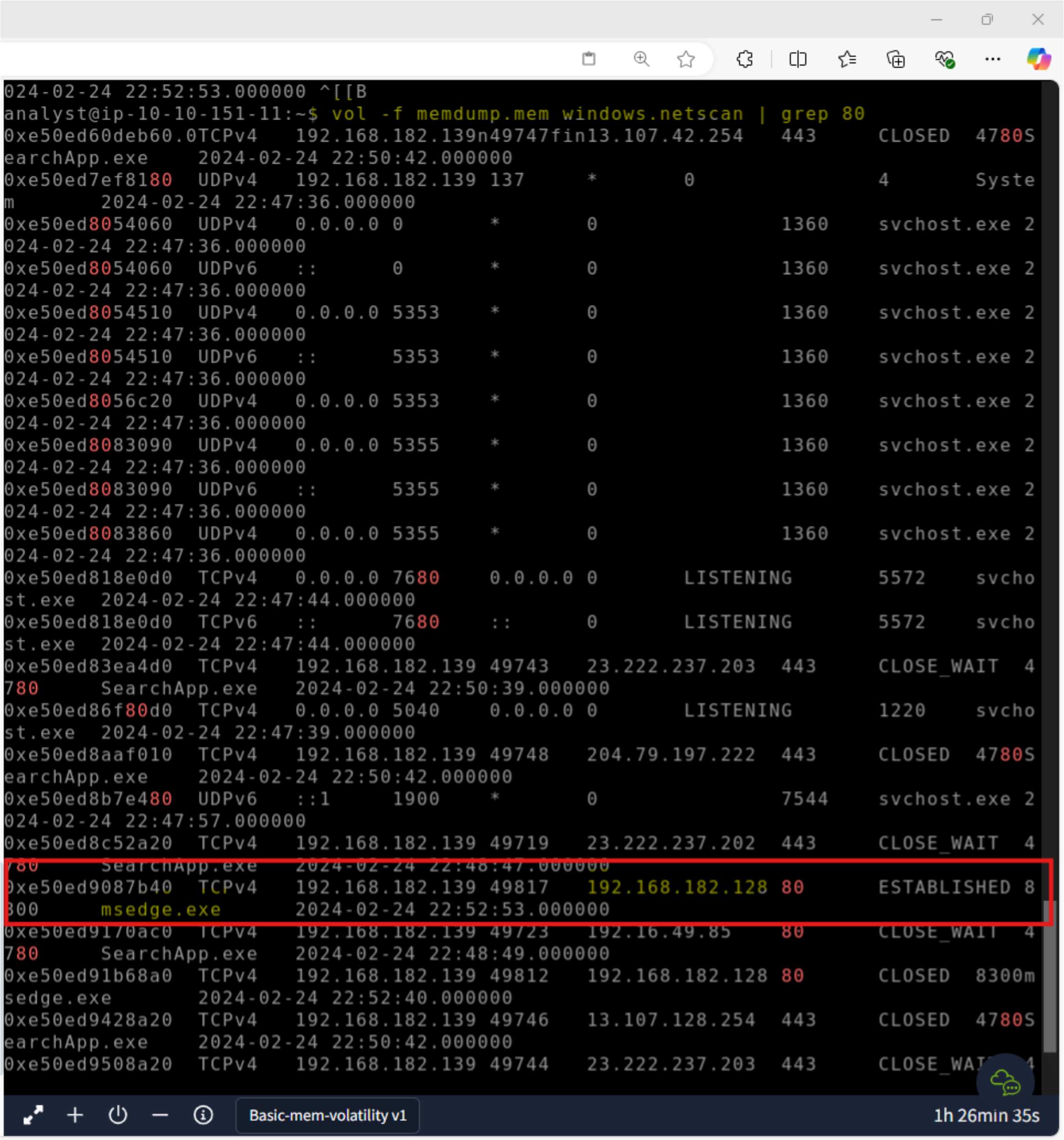
```
volatility: error: Please select a plugin to run
analyst@ip-10-10-151-11:~$ vol -f memdump.mem windows.info
Volatility 3 Framework 2.5.2
Progress:  100.00               PDB scanning finished
Variable        Value

Kernel Base     0xf8066161b000
DTB     0x1ad000
Symbols file:///home/analyst/volatility3-2.5.2/volatility3/symbols/windows/ntkrnlmp.p
db/4DBE144182FF4156845CD3BD8B654E56-1.json.xz
Is64Bit True
IsPAE   False
layer_name      0 WindowsIntel32e
memory_layer    1 FileLayer
KdVersionBlock  0xf8066222a400
Major/Minor     15.19041
MachineType     34404
KeNumberProcessors      2
SystemTime      2024-02-24 22:52:52
NtSystemRoot    C:\Windows
NtProductType   NtProductWinNt
NtMajorVersion  10
NtMinorVersion  0
PE MajorOperatingSystemVersion  10
PE MinorOperatingSystemVersion  0
PE Machine      34404
PE TimeDateStamp        Sat Jan 13 03:45:32 2085
analyst@ip-10-10-151-11:~$ 
```

Now that we have the information from the target we are working on let's try to identify any suspicious activity in the memory dump.

Using the plugin "windows.netscan" can you identify the IP address that establish a connection on port 80? Answer: 192.168.182.128

Using the plugin "windows.netscan," can you identify the program (owner) used to access through port 80? Answer: msedge.exe



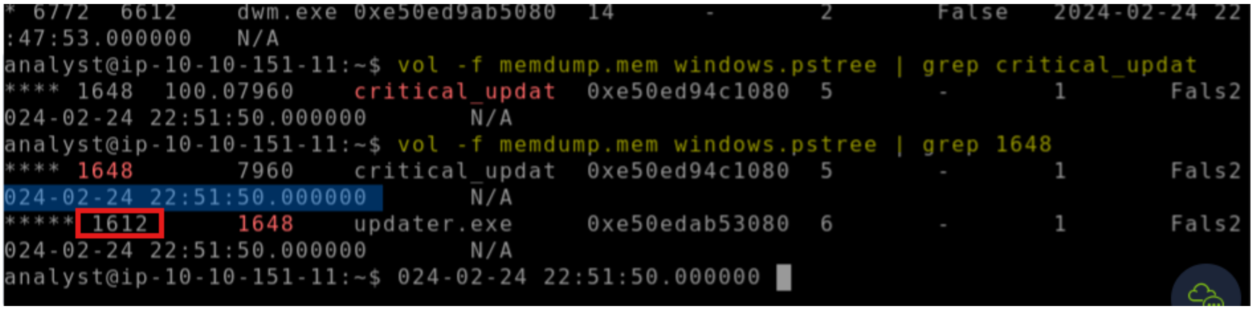
```
024-02-24 22:52:53.000000 ^[[B
analyst@ip-10-10-151-11:~$ vol -f memdump.mem windows.netscan | grep 80
0xe50ed60deb60.0TCPv4   192.168.182.139n49747fin13.107.42.254   443     CLOSED  4780S
earchApp.exe    2024-02-24 22:50:42.000000
0xe50ed7ef8180  UDPv4   192.168.182.139 137     *       0               4       Syste
m       2024-02-24 22:47:36.000000
0xe50ed8054060  UDPv4   0.0.0.0 0       *       0               1360    svchost.exe 2
024-02-24 22:47:36.000000
0xe50ed8054060  UDPv6   ::      0       *       0               1360    svchost.exe 2
024-02-24 22:47:36.000000
0xe50ed8054510  UDPv4   0.0.0.0 5353    *       0               1360    svchost.exe 2
024-02-24 22:47:36.000000
0xe50ed8054510  UDPv6   ::      5353    *       0               1360    svchost.exe 2
024-02-24 22:47:36.000000
0xe50ed8056c20  UDPv4   0.0.0.0 5353    *       0               1360    svchost.exe 2
024-02-24 22:47:36.000000
0xe50ed8083090  UDPv4   0.0.0.0 5355    *       0               1360    svchost.exe 2
024-02-24 22:47:36.000000
0xe50ed8083090  UDPv6   ::      5355    *       0               1360    svchost.exe 2
024-02-24 22:47:36.000000
0xe50ed8083860  UDPv4   0.0.0.0 5355    *       0               1360    svchost.exe 2
024-02-24 22:47:36.000000
0xe50ed818e0d0  TCPv4   0.0.0.0 7680    0.0.0.0 0       LISTENING       5572    svcho
st.exe  2024-02-24 22:47:44.000000
0xe50ed818e0d0  TCPv6   ::      7680    ::      0       LISTENING       5572    svcho
st.exe  2024-02-24 22:47:44.000000
0xe50ed83ea4d0  TCPv4   192.168.182.139 49743   23.222.237.203  443     CLOSE_WAIT  4
780     SearchApp.exe   2024-02-24 22:50:39.000000
0xe50ed86f80d0  TCPv4   0.0.0.0 5040    0.0.0.0 0       LISTENING       1220    svcho
st.exe  2024-02-24 22:47:39.000000
0xe50ed8aaf010  TCPv4   192.168.182.139 49748   204.79.197.222  443     CLOSED  4780S
earchApp.exe    2024-02-24 22:50:42.000000
0xe50ed8b7e480  UDPv6   ::1     1900    *       0               7544    svchost.exe 2
024-02-24 22:47:57.000000
0xe50ed8c52a20  TCPv4   192.168.182.139 49719   23.222.237.202  443     CLOSE_WAIT  4
780     SearchApp.exe   2024-02-24 22:48:47.000000
0xe50ed9087b40  TCPv4   192.168.182.139 49817   192.168.182.128 80      ESTABLISHED 8
300     msedge.exe      2024-02-24 22:52:53.000000
0xe50ed9170ac0  TCPv4   192.168.182.139 49723   192.16.49.85    80      CLOSE_WAIT  4
780     SearchApp.exe   2024-02-24 22:48:49.000000
0xe50ed91b68a0  TCPv4   192.168.182.139 49812   192.168.182.128 80      CLOSED  8300m
sedge.exe       2024-02-24 22:52:40.000000
0xe50ed9428a20  TCPv4   192.168.182.139 49746   13.107.128.254  443     CLOSED  4780S
earchApp.exe    2024-02-24 22:50:42.000000
0xe50ed9508a20  TCPv4   192.168.182.139 49744   23.222.237.203  443     CLOSE_WAIT  4
```

Analyzing the process present on the dump, what is the PID of the child process of critical_updat? Answer: 1612

What is the time stamp time for the process with the truncated name critical_updat?

Answer: 024-02-24 22:51:50.000000

First, get the PID of the critical_updat using grep critical_updat then use the PID to find the child process

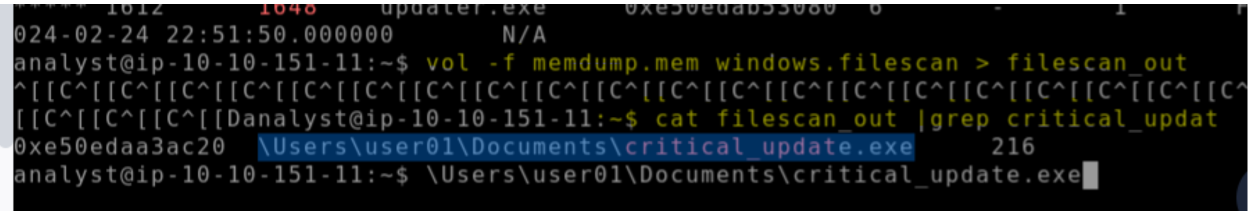


```
* 6772  6612     dwm.exe 0xe50ed9ab5080  14        -        2        False   2024-02-24 22
:47:53.000000   N/A
analyst@ip-10-10-151-11:~$ vol -f memdump.mem windows.pstree | grep critical_updat
**** 1648  100.07960    critical_updat  0xe50ed94c1080  5        -        1        Fals2
024-02-24 22:51:50.000000          N/A
analyst@ip-10-10-151-11:~$ vol -f memdump.mem windows.pstree | grep 1648
**** 1648          7960     critical_updat  0xe50ed94c1080  5        -        1        Fals2
024-02-24 22:51:50.000000          N/A
***** 1612         1648     updater.exe     0xe50edab53080  6        -        1        Fals2
024-02-24 22:51:50.000000          N/A
analyst@ip-10-10-151-11:~$ 024-02-24 22:51:50.000000
```

With the information we have collected, we can investigate the process critical_updat that we identified in our previous task, which has a child process called updater. Let's investigate the child process more in-depth. Let's start by looking at where on the disk it was saved; for that, we can use the plugin windows.filescan which will allow us to examine the files accessed that are stored in the memory dump. This output is quite big, so to access the data in a better way, we will use the > character in bash to redirect the output to a file, in this case, filescan_out.

Analyzing the "windows.filescan" output, what is the full path and name for critical_updat?

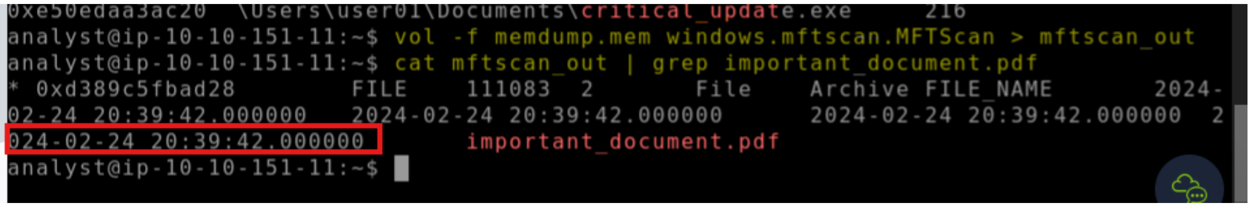
Answer: C:\Users\user01\Documents\critical_update.exe



```
***** 1612         1648     updater.exe     0xe50edab53080  6        -        1
024-02-24 22:51:50.000000          N/A
analyst@ip-10-10-151-11:~$ vol -f memdump.mem windows.filescan > filescan_out
^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^
[[C^[[C^[[C^[[Danalyst@ip-10-10-151-11:~$ cat filescan_out |grep critical_updat
0xe50edaa3ac20   \Users\user01\Documents\critical_update.exe      216
analyst@ip-10-10-151-11:~$ \Users\user01\Documents\critical_update.exe
```

Analyzing the "windows.mftscan.MFTScan" what is the Timestamp for the created date of important_document.pdf?

Answer: 2024-02-24 20:39:42.000000



```
0xe50edaa3ac20  \Users\user01\Documents\critical_update.exe     216
analyst@ip-10-10-151-11:~$ vol -f memdump.mem windows.mftscan.MFTScan > mftscan_out
analyst@ip-10-10-151-11:~$ cat mftscan_out | grep important_document.pdf
* 0xd389c5fbad28        FILE    111083  2       File    Archive FILE_NAME       2024-
02-24 20:39:42.000000   2024-02-24 20:39:42.000000      2024-02-24 20:39:42.000000  2
024-02-24 20:39:42.000000       important_document.pdf
analyst@ip-10-10-151-11:~$
```

Analyzing the updater.exe memory output, can you observe the HTTP request and determine the server used by the attacker?

Answer: Server: SimpleHTTP/0.6 Python/3.10.4